VR-Link 3.4
HLA Class Documentation

This file documents VR-Link classes. It is based on the header files and includes many of the comments from them. It provides an alternative to looking through the individual header files when you need information about constructors or other aspects of a class's functioning. We hope you find this format helpful.

Classes are listed alphabetically and in a class hierarchy. The hypertext links will help you quickly jump to related information.

- Table of Contents (alphabetical listing)
- Hierarchical List

Notes

- There are two separate versions of VR-Link class documentation. One is for DIS and the other is for HLA.
- If you find a difference between the information in this file and the original header file shipped with your version of VR-Link, rely on the header file.
- The top line in each class description lists the header file in which it is defined. However, the capitalization of the header file is incorrect. The actual header files used mixed capitalization, not all lowercase.
- Some links point to header files. If you click one of these links, you will get a message indicating that the file could not be found. The file name will be included in the message, so if you want to, you can look at the file in the VR-Link Reference Manual or the delivered files.

MÄK Technologies, Inc.

185 Alewife Brook Parkway

Cambridge, MA 02138

Phone: 617.876.8085

Fax: 617.876.9208

www.mak.com

support@mak.com

Copyright 1999 MÄK Technologies, Inc. All Rights Reserved

Document ID: VRL-3.4-5H-990428

VR-Link 3.4 HLA Class Documentation

Hierarchy of classes

Table of Contents

- Packages
- Classes
- Functions
- Variables
- Macros
- Enums, Unions, Structs
- Typedefs

Packages

- Services* ////////////////////////////

Classes

- DtAbsNotifyObserver *class DtAbsNotifyObserver: DtAbsNotifyObserver is an abstract base class*
- DtAbsoluteVCPdu
- DtAcknowledgeDecoder
- DtAcknowledgeEncoder
- DtAcknowledgeInteraction
- DtActionRequestDecoder
- DtActionRequestEncoder
- DtActionRequestInteraction
- DtActionRequestToObjDecoder
- DtActionRequestToObjEncoder
- DtActionRequestToObjInter
- DtActionResponseDecoder
- DtActionResponseEncoder
- DtActionResponseFromObjDecoder
- DtActionResponseFromObjEncoder
- DtActionResponseFromObjInter
- DtActionResponseInteraction
- DtAggregateEntityDecoder *class DtAggregateEntityDecoder: Instances of DtAggregateEntityDecoder are used to decode an aggregate state message into a state repository in an HLA (as opposed to DIS) environment*
- DtAggregateEntityEncoder *class DtAggregateEntityEncoder: Instances of DtAggregateEntityEncoder are used to*

encode an aggregate entity's state repository into a state message in an HLA environment

- DtAggregatePublisher *class DtAggregatePublisher: Instances of DtAggregatePublisher are used for local or source simulations of aggregate entities, (i*
- DtAggregateStateRepository *class DtAggregateStateRepository: Instances of DtAggregateStateRepository are used to maintain the state of an aggregate*
- DtAntennaPatternRecord *class DtAntennaPatternRecord: Instances of DtAntennaPatternRecord are*
- DtAppSpecificRadioSignalDecoder
- DtAppSpecificRadioSignalEncoder
- DtApproximator *class DtApproximator: DtApproximator is an abstract base class*
- DtArtPart *One Articulated Part*
- DtArtPartList *class DtArtPartList: Instances of DtArtPartList are used to manage articulated parts*
- DtArtPartThresholder *class DtArtPartThresholder: DtArtPartThresholder is used to store art-part-related thresholds, and to perform some art part thresholding*
- DtAttachedPartList *class DtAttachedPartList: Instances of DtAttachedPartList are lists of attached parts*
- DtAttributeChangeReqDec
- DtAttributeChangeReqEnc
- DtAttributeChangeReqInter
- DtAttributeChangeResDec
- DtAttributeChangeResEnc
- DtAttributeChangeResInter
- DtBaseEmbeddedSystemDecoder
- DtBaseEmbeddedSystemEncoder
- DtBaseEmitterBeamDecoder
- DtBaseEmitterBeamEncoder
- DtBaseEntityDecoder
- DtBaseEntityEncoder
- DtBaseEntityStateRepository *class DtBaseEntityStateRepository: Instances of DtBaseEntityStateRepository are used to maintain basic state information needed by all kinds of entities*
- DtBaseExerciseConn
- DtBaseHashKey *class DtBaseHashKey: DtBaseHashKey is an abstract base class for hashkeys*
- DtBaseMsg
- DtBaseReflectedAggregate
- DtBaseReflectedEntity
- DtBeamAntennaPatternRecord *class DtBeamAntennaPatternRecord: Instances of DtBeamAntennaPatternRecord are beam antenna pattern records used in conjunction with DtRadioTransmitterStateRepository (TSR)*
- DtBurstDescriptor
- DtCartesianView *class DtCartesianView: Instances of DtCartesianView are used to create a Cartesian coordinate view of the geocentric coordinates in an Entity State Repository*
- DtClientSocket
- DtCollisionDecoder
- DtCollisionEncoder

- DtCollisionInteraction
- DtCommentDecoder
- DtCommentEncoder
- DtCommentInteraction
- DtCompassVCPdu
- DtCoordTransform
- DtCoordTranslation
- DtCreateEntityDecoder
- DtCreateEntityEncoder
- DtCreateEntityInteraction
- DtCreateObjReqDecoder
- DtCreateObjReqEncoder
- DtCreateObjReqInter
- DtCreateObjectResultDecoder
- DtCreateObjectResultEncoder
- DtCreateObjectResultInter
- DtCsFilterTest
- DtDataDecoder
- DtDataEncoder
- DtDataInteraction
- DtDataQueryDecoder
- DtDataQueryEncoder
- DtDataQueryInteraction
- DtDatabaseIndexRadioSignalDecoder
- DtDatabaseIndexRadioSignalEncoder
- DtDatumRecordSet
- DtDcm
- DtDeadReckoner *class DtDeadReckoner: Instances of DtDeadReckoner are used to provide approximated position and orientation information for entities using various dead reckoning algorithms*
- DtDesignatorDecoder
- DtDesignatorEncoder
- DtDesignatorPublisher *class DtDesignatorPublisher: Instances of DtDesignatorPublisher are used for local or source simulations of designators, (i*
- DtDesignatorRepository *class DtDesignatorRepository: Instances of DtDesignatorRepository are*
- DtDetonationDecoder
- DtDetonationEncoder
- DtDetonationInteraction
- DtDirectRtiAmbassador *class DtDirectRtiAmbassador: Instances of DtDirectRtiAmbassador are for wrapping the calls to The RTI ambassador so that we may override them*
- DtDirectedRotRate *rotation rate given by single rate and axis of rotation*

- DtDiscoverObjectCbInfo
- DtEmbeddedSystemRepository *class DtEmbeddedSystemRepository: Instances of DtEmbeddedSystemRepository are*
- DtEmitterBeamPublisher *class DtEmitterBeamPublisher: Instances of DtEmitterBeamPublisher are used for local or source simulations of emitter beams, (i*
- DtEmitterBeamRepository *class DtEmitterBeamRepository: Instances of DtEmitterBeamRepository are*
- DtEmitterSystemDecoder
- DtEmitterSystemEncoder
- DtEmitterSystemPublisher *class DtEmitterSystemPublisher: Instances of DtEmitterSystemPublisher are used for local or source simulations of emitter systems, (i*
- DtEmitterSystemRepository *class DtEmitterSystemRepository: Instances of DtEmitterSystemRepository are*
- DtEmptyFomMapper *class DtEmptyFomMapper: An instances of DtEmptyFomMapper is a DtFomMapper that has no mappings registered between FOM classes and VR-Link classes*
- DtEncodedAudioRadioSignalDecoder
- DtEncodedAudioRadioSignalEncoder
- DtEntityIdentifier
- DtEntityPublisher *class DtEntityPublisher: Instances of DtEntityPublisher are used for local or source simulations of entities, (i*
- DtEntityStateRepository *class DtEntityStateRepository: Instances of DtEntityStateRepository are used to maintain entity state*
- DtEntityType
- DtEnvironmentEntityDecoder *class DtEnvironmentEntityDecoder: Instances of DtEnvironmentEntityDecoder are used to decode an environmental entity state message into a state repository in an HLA (as opposed to DIS) environment*
- DtEnvironmentEntityEncoder *class DtEnvironmentEntityEncoder: Instances of DtEnvironmentEntityEncoder are used to encode an environment entity's entity's state repository into a state message in an HLA environment*
- DtEventId
- DtEventReportDecoder
- DtEventReportEncoder
- DtEventReportInteraction
- DtExerciseConn *class DtExerciseConn: Instances of DtExerciseConn are used to provide an application's interface or connection to the network in an HLA environment (as opposed to DIS)*
- DtFedReader *class DtFedReader: Instances of DtFedReader are used to read a FED file, and fill in a DtFedTarget with the information read*
- DtFedTarget *class DtFedTarget: DtFedTarget is an abstract class representing some representation of FED file data that can be filled in by a DtFedReader*
- DtFederateId
- DtFilename *class DtFilename: Instances of DtFilename are DtStrings which convert filenames to the native format*
- DtFilterTest
- DtFireDecoder
- DtFireEncoder
- DtFireInteraction
- DtFom *class DtFom: An instances of DtFom is used to represent a particular FOM, as described by a FED file*

- DtFomMapper
- DtGenericObjectDecoder *The DtGenericObjectDecoder class:*
- DtGenericObjectEncoder *The DtGenericObjectEncoder class - the encoder used by DtGenericObjectPublisher*
- DtGenericObjectPublisher *class DtGenericObjectPublisher: Instances of DtGenericObjectPublisher are used for local or source simulations of objects for which a more specific publisher does not exist*
- DtGenericStateRepository *class DtGenericStateRepository: Instances of DtGenericStateRepository are repositories for the state of an HLA object of unknown type*
- DtGeodeticCoord *class DtGeodeticCoord: Instances of DtGeodeticCoord are used to explicitly represent Geodetic coordinates*
- DtGlobalIdHashlist *class DtGlobalIdHashlist: Instances of DtGlobalIdHashlist are hashlists, where you can store associations between DtGlobalObjectDesignators and arbitrary pieces of data*
- DtGlobalObjectDesignatorList
- DtGroupSocket
- DtGroupVCPdu
- DtHashItem *class DtHashItem: Instances of DtHashItem are used internally to a hashlist to associate a pointer to an item in a list with a key*
- DtHashlist
- DtHlaObject *class DtHlaObject: Instances of DtHlaObject are used to represent HLA objects (as opposed to interactions) which are communicated across the network*
- DtHlaObjectFactory *class DtHlaObjectFactory: Instances of DtHlaObjectFactory are used to create HLA objects*
- DtHlaObjectWithStateRep *class DtHlaObjectWithStateRep: Instances of DtHlaObjectWithStateRep are used to represent an HLA object via a state representation*
- DtHlaStateDecoder *class DtHlaStateDecoder: DtHlaStateDecoder is used to provide decoding of a state message into a state repository in an HLA (as opposed to DIS) environment*
- DtHlaStateEncoder *class DtHlaStateEncoder: DtHlaStateEncoder is used to provide encoding of a state repository into a state message in an HLA (as opposed to DIS) environment*
- DtIntHashKey *class DtIntHashKey: Instances of DtIntHashKey are used as a hash key made from an integer*
- DtInterClassDesc *class DtInterClassDesc: Instances of DtInterClassDesc are used to represent HLA Object and Interaction classes*
- DtInteraction *class DtInteraction: DtInteraction is an abstract base class for representing HLA interactions (or events) (as opposed to HLA objects)*
- DtInteractionDecoder *class DtInteractionDecoder: Instances of DtInteractionDecoder are used to decode interactions from the network*
- DtInteractionDecoderFactory *class DtInteractionDecoderFactory: Instances of DtInteractionDecoderFactory are used to create decoders which know how to decode an HLA interaction message (or network representation) into an interaction repository (or host representation)*
- DtInteractionEncoder *class DtInteractionEncoder: DtInteractionEncoder is used to provide encoding of an interaction into an RTI::ParameterHandleValuePairSet in an HLA (as opposed to DIS) environment*
- DtInteractionEncoderFactory *class DtInteractionEncoderFactory: Instances of DtInteractionEncoderFactory are used to create encoders which know how to encode an interaction repository (or host representation) into an HLA interaction message (or network representation)*
- DtInteractionFactory *class DtInteractionFactory: Instances of DtInteractionFactory are used to create derived DtInteractions based on Interaction Class*
- DtInteractionManager *class DtInteractionManager: Instances of DtInteractionManager are used to manage the creation and processing of HLA interactions*

- DtInteractionWithData *class DtInteractionWithData: Instances of DtInteractionWithData are used to represent RPR FOM Interactions that use fixed and var datum records*
- DtInteractionWithEncDec *class DtInteractionWithEncDec: DtInteractionWithEncDec is a DtInteraction that uses an encoder and decoder to implement the virtual functions phvps and setFromPhvps*
- DtIntrusiveList *An intrusive list is a list that assumes that the elements that you add are pointers to instances of classes derived from DtListedObject*
- DtJammerBeamDecoder
- DtJammerBeamEncoder
- DtLgrControlDecoder
- DtLgrControlEncoder
- DtLgrControlInteraction
- DtLgrControlPdu *class DtLgrControlPdu: Instances of DtLgrControlPdu are a family of messgae used to control a remote MaK-Logger over the network*
- DtLgrEndPdu *class DtLgrEndPdu: Instances of DtLgrEndPdu are to tell remote logger to skip to end of file during playback*
- DtLgrEofActionPdu *class DtLgrEofActionPdu: Instances of DtLgrEofActionPdu are to tell remote logger that it should pause, wrap, or stop at the end of playback*
- DtLgrHeartbeatPdu *class DtLgrHeartbeatPdu: Instances of DtLgrHeartbeatPdu are to tell remote logger to send packet for each playback entity at least every HEARTBEAT seconds*
- DtLgrPausePdu *class DtLgrPausePdu: Instances of DtLgrPausePdu are to tell remote logger to pause*
- DtLgrPbFilePdu *class DtLgrPbFilePdu: Instances of DtLgrPbFilePdu are used for switching to play-mode and defining which file to read from when sending traffic into the simulation*
- DtLgrPlayActionPdu *class DtLgrPlayActionPdu: Instances of DtLgrPlayActionPdu are to tell remote logger what action to take initially when set to playback mode*
- DtLgrPlayPdu *class DtLgrPlayPdu: Instances of DtLgrPlayPdu are used to tell remote logger to begin recording*
- DtLgrQuitPdu *class DtLgrQuitPdu: Instances of DtLgrQuitPdu are messages telling a remote logger to quit*
- DtLgrRecFilePdu *class DtLgrRecFilePdu: Instances of DtLgrRecFilePdu are used to switching remote logger to recording-mode and to set the file into which to store captured simulation traffic*
- DtLgrRecordActionPdu *class DtLgrRecordActionPdu: Instances of DtLgrRecordActionPdu are to tell remote logger to stop or begin recording when entering record-mode*
- DtLgrRecordPdu *class DtLgrRecordPdu: Instances of DtLgrRecordPdu are to tell remote logger to begin recording*
- DtLgrSetTimePdu *class DtLgrSetTimePdu: Instances of DtLgrSetTimePdu are to tell remote logger to go to set position to HH:MM:SS*
- DtLgrSkipTimePdu *class DtLgrSkipTimePdu: Instances of DtLgrSetTimePdu are to tell remote logger to forward or backward HH:MM:SS*
- DtLgrStartPdu *class DtLgrStartPdu: Instances of DtLgrStartPdu are used to tell remote logger to start recording*
- DtLgrStopPdu *class DtLgrStopPdu: Instances of DtLgrStopPdu are to tell remote logger to stop*
- DtLgrTimeScalePdu *class DtLgrTimeScalePdu: Instances of DtLgrTimeScalePdu are to tell remote logger to slow down or speed up playback*
- DtLgrTimeoutPdu *class DtLgrTimeoutPdu: Instances of DtLgrTimeoutPdu are to tell remote logger how long to generate interim packets for entity before assuming no more packets in file*
- DtList *A DtList is a list of elements of arbitrary type*
- DtListItem *DtListItem is an envelope class used by DtList*
- DtListedObj *This is a base class from which elements of a DtIntrusiveList must be derived*

- DtLocalHlaObjectManager *class DtLocalHlaObjectManager: Instances of DtLocalHlaObjectManager are used to manage local HLA objects (objects which are being published by this application)*
- DtMilitaryEntityDecoder *class DtMilitaryEntityDecoder: Instances of DtMilitaryEntityDecoder are used to decode a military entity state message into a state repository in an HLA (as opposed to DIS) environment*
- DtMilitaryEntityEncoder *class DtMilitaryEntityEncoder: Instances of DtMilitaryEntityEncoder are used to encode a military entity's state repository into a state message in an HLA environment*
- DtMilitaryPlatformEntityDecoder *class DtMilitaryPlatformEntityDecoder: Instances of DtMilitaryPlatformEntityDecoder are used to decode a military platform entity state message into a state repository in an HLA (as opposed to DIS) environment*
- DtMilitaryPlatformEntityEncoder *class DtMilitaryPlatformEntityEncoder: Instances of DtMilitaryPlatformEntityEncoder are used to encode a military platform entity's state repository into a state message in an HLA environment*
- DtMimicTrackVCPdu
- DtMimicVCPdu
- DtModelessVCPdu
- DtMunitionEntityDecoder *class DtMunitionEntityDecoder: Instances of DtMunitionEntityDecoder are used to decode a munition entity state message into a state repository in an HLA (as opposed to DIS) environment*
- DtMunitionEntityEncoder *class DtMunitionEntityEncoder: Instances of DtMunitionEntityEncoder are used to encode a munition entity's state repository into a state message in an HLA environment*
- DtNet32Vector
- DtNet64Vector
- DtNetBoolean
- DtNetBurstDescriptor
- DtNetEntityIdentifier
- DtNetEntityMarking
- DtNetEntityType
- DtNetEulerAngles
- DtNetEventId
- DtNetFederateId
- DtNetFloat32 *Network-order 32-bit IEEE-754 floating-point quantity*
- DtNetFloat64 *Network-order 64-bit IEEE-754 floating-point quantity*
- DtNetGlobalObjectDesignator
- DtNetInt16 *Network-order signed 16-bit quantity*
- DtNetInt32 *Network-order signed 32-bit quantity*
- DtNetRadioEntityType
- DtNetSimulationAddress
- DtNetSocket
- DtNetTimeStamp *class DtNetTimeStamp: Instances of DtNetTimeStamp are network representations of a DIS time stamp*
- DtNetU16 *Network-order unsigned 16-bit quantity*
- DtNetU32 *Network-order unsigned 32-bit quantity*
- DtNullFederateAmbassador

- DtObjClassDesc *class DtObjClassDesc: An instance of DtObjClassDesc is a "class descriptor" for an HLA Object class*
- DtObjectId
- DtObjectIdHashKey *class DtObjectIdHashKey: Instances of DtObjectIdHashKey are hash keys made from DtObjectIds*
- DtObjectIdHashlist *class DtObjectIdHashlist: Instances of DtObjectIdHashlist are hashlists, where you can store associations between DtObjectIds and arbitrary pieces of data*
- DtObjectPublisher *class DtObjectPublisher: Instances of DtObjectPublisher are used for local or source simulations of objects, (i*
- DtObjectReflectAttributeValuesCbInfo *class DtObjectReflectAttributeValuesCbInfo: Instances of DtReflectAttributeValuesCbInfo are used for providing a callback when attributes change for an object in the federation*
- DtOrbitVCPdu
- DtPdu
- DtPhysicalEntityDecoder *class DtPhysicalEntityDecoder: Instances of DtPhysicalEntityDecoder are used to decode a physical entity state message into a state repository in an HLA (as opposed to DIS) environment*
- DtPhysicalEntityEncoder *class DtPhysicalEntityEncoder: Instances of DtPhysicalEntityEncoder are used to encode a physical entity's state repository into a state message in an HLA environment*
- DtRadarBeamDecoder
- DtRadarBeamEncoder
- DtRadioEntityType
- DtRadioReceiverDecoder
- DtRadioReceiverEncoder
- DtRadioReceiverPublisher *class DtRadioReceiverPublisher: Instances of DtRadioReceiverPublisher are used for local or source simulations of radio receivers, (i*
- DtRadioReceiverRepository *class DtRadioReceiverRepository: Instances of DtRadioReceiverRepository are*
- DtRadioTransmitterDecoder
- DtRadioTransmitterEncoder
- DtRadioTransmitterPublisher *class DtRadioTransmitterPublisher: Instances of DtRadioTransmitterPublisher are used for local or source simulations of radio transmitters, (i*
- DtRadioTransmitterRepository *class DtRadioTransmitterRepository: Instances of DtRadioTransmitterRepository are*
- DtRawBinaryRadioSignalDecoder
- DtRawBinaryRadioSignalEncoder
- DtReceiveInteractionCbInfo
- DtReflectAttributeValuesCbInfo *class DtReflectAttributeValuesCbInfo: Instances of DtReflectAttributeValuesCbInfo are used for providing a callback when attributes change for an object in federation*
- DtReflectAttributeValuesCbList
- DtReflectedAggregate *class DtReflectedAggregate: Instances of DtReflectedAggregate are used to represent a simulation aggregate reflected across a network (i*
- DtReflectedAggregateList *class DtReflectedAggregateList: Instances of DtReflectedAggregateList are used to manage remote aggregates from the network*
- DtReflectedDesignator *class DtReflectedDesignator: Instances of DtReflectedDesignator are*

- DtReflectedDesignatorList *class DtReflectedDesignatorList: Instances of DtReflectedDesignatorList manage the list of of remote Designators in the exercise*
- DtReflectedEmitterBeam *class DtReflectedEmitterBeam: Instances of DtReflectedEmitterBeam represent EmitterBeam object simulated by remote applications*
- DtReflectedEmitterBeamList *class DtReflectedEmitterBeamList: Instances of DtReflectedEmitterBeamList manage the list of of remote EmitterBeams in the exercise*
- DtReflectedEmitterSystem *class DtReflectedEmitterSystem: Instances of DtReflectedEmitterSystem represent EmitterSystem object simulated by remote applications*
- DtReflectedEmitterSystemList *class DtReflectedEmitterSystemList: Instances of DtReflectedEmitterSystemList manage the list of of remote EmitterSystems in the exercise*
- DtReflectedEntity *class DtReflectedEntity: Instances of DtReflectedEntity are used to represent a simulation entity reflected across a network (i*
- DtReflectedEntityList *class DtReflectedEntityList: Instances of DtReflectedEntityList are used to manage remote entities from the network*
- DtReflectedGenericObject *class DtReflectedGenericObject: Instances of DtReflectedGenericObject are used to represent "GenericObject" objects simulated remotely, that we hear about through the RTI*
- DtReflectedGenericObjectList *class DtReflectedGenericObjectList: An instance of this class manages the list of Generic objects that are simulated remotely, and reflected to us by the RTI*
- DtReflectedHlaObjectManager *class DtReflectedHlaObjectManager: Instances of DtReflectedHlaObjectManager are used to manage the creation and destruction of HLA objects from the network*
- DtReflectedObject *class DtReflectedObject: Instances of DtReflectedObject are used to represent a simulation entity reflected across a network (i*
- DtReflectedObjectCallbackInfo *class DtReflectedObjectCallbackInfo: Instances of DtReflectedObjectCallbackInfo are used to contain the function and user data associated with a DtReflectedObject callback,*
- DtReflectedObjectList *class DtReflectedObjectList: Instances of DtReflectedObjectList are list of instances of classes derived from DtReflectedObject*
- DtReflectedRadioReceiver *class DtReflectedRadioReceiver: Instances of DtReflectedRadioReceiver represent RadioReceiver object simulated by remote applications*
- DtReflectedRadioReceiverList *class DtReflectedRadioReceiverList: Instances of DtReflectedRadioReceiverList manage the list of of remote RadioReceivers in the exercise*
- DtReflectedRadioTransmitter *class DtReflectedRadioTransmitter: Instances of DtReflectedRadioTransmitter represent RadioTransmitter object simulated by remote applications*
- DtReflectedRadioTransmitterList *class DtReflectedRadioTransmitterList: Instances of DtReflectedRadioTransmitterList manage the list of of remote RadioTransmitters in the exercise*
- DtRegControlCbInfo *class DtRegControlCbInfo: Instances of DtRegControlCbInfo are used for providing a callback when the RTI update control services are invoked*
- DtRegControlCbList
- DtRemoveEntityDecoder
- DtRemoveEntityEncoder
- DtRemoveEntityInteraction
- DtRemoveObjectCbInfo
- DtRemoveObjectRequestDecoder
- DtRemoveObjectRequestEncoder
- DtRemoveObjectRequestInter
- DtRemoveObjectResultDecoder

- DtRemoveObjectResultEncoder
- DtRemoveObjectResultInter
- DtRepairCompleteDecoder
- DtRepairCompleteEncoder
- DtRepairCompleteInteraction
- DtRepairResponseDecoder
- DtRepairResponseEncoder
- DtRepairResponseInteraction
- DtResupplyCancelDecoder
- DtResupplyCancelEncoder
- DtResupplyCancelInteraction
- DtResupplyOfferDecoder
- DtResupplyOfferEncoder
- DtResupplyOfferInteraction
- DtResupplyRecInteraction
- DtResupplyReceivedDecoder
- DtResupplyReceivedEncoder
- DtRprFomMapper
- DtRprHlaObjectFactory *class DtRprHlaObjectFactory: Instances of DtRprHlaObjectFactory are used to create RPR (Real-time Platform Reference) FOM HLA objects*
- DtRprInteractionDecoderFactory
- DtRprInteractionEncoderFactory
- DtRprInteractionFactory
- DtRprStateDecoderFactory
- DtRprStateEncoderFactory
- DtScript *A DtScript is a DtList of StealthInstructions*
- DtSelectParam
- DtServiceRequestDecoder
- DtServiceRequestEncoder
- DtServiceRequestInteraction
- DtSetDataDecoder
- DtSetDataEncoder
- DtSetDataInteraction
- DtSignalInteraction
- DtSilentAggregateList *class DtSilentAggregateList: DtSilentAggregateList is a class which contains the information needed to describe a list of silent aggregates for the aggregate state PDU*
- DtSilentEntityList *class DtSilentEntityList: DtSilentEntityList is a class which contains the information needed to describe a list of silent entities for the aggregate state PDU*
- DtSmoothOriWithDr *class DtSmoothOriWithDr: Instances of DtSmoothOriWithDr are*
- DtSmoothOrient *---------- ORIENTATION ----------*

- DtSmoothPos ---------- *POSITION* ----------
- DtSmoothPosWithDr *class DtSmoothPosWithDr: Instances of DtSmoothPosWithDr are*
- DtSmoother *class DtSmoother: Instances of DtSmoother are used to provide approximated position and orientation information for entities*
- DtSocket
- DtSocketPairSocket
- DtSoldierDecoder *class DtSoldierDecoder: Instances of DtSoldierDecoder are used to decode a soldier entity state message into a state repository in an HLA (as opposed to DIS) environment*
- DtSoldierEncoder *class DtSoldierEncoder: Instances of DtSoldierEncoder are used to encode a soldier entity's state repository into a state message in an HLA environment*
- DtStartResumeDecoder
- DtStartResumeEncoder
- DtStartResumeInteraction
- DtStartStopInterCbInfo *class DtStartStopInterCbInfo: Instances of DtStartStopInterCbInfo are used for providing a callback when the RTI's start or stopInteractionGeneration services are invoked*
- DtStartStopInterCbList
- DtStateDecoderFactory *class DtStateDecoderFactory: Instances of DtStateDecoderFactory are used to create decoders which know how to decode an HLA state message (or network representation) into a state repository (or host representation)*
- DtStateEncoderFactory *class DtStateEncoderFactory: Instances of DtStateEncoderFactory are used to create encoders which know how to encode a state repository (or host representation) into an HLA state message (or network representation)*
- DtStateMsg
- DtStateRepository *class DtStateRepository: DtStateRepository is an abstract base class for maintaining state for an object*
- DtStopFreezeDecoder
- DtStopFreezeEncoder
- DtStopFreezeInteraction
- DtString *class DtString: Instances of DtString are*
- DtStringHashKey *class DtStringHashKey: Instances of DtStringHashKey are used as a hash key made from a string*
- DtSystemHashKey *class DtSystemHashKey: Instances of DtSystemHashKey are used as hash keys for an emitter system*
- DtTaitBryan
- DtTcpForwarder
- DtTcpSocket
- DtTcpSocketMgr
- DtTetherTrackVCPdu
- DtTetherVCPdu
- DtThresholder *class DtThresholder: Instances of DtThresholder store thresholding information used to determine what attributes of an entity need to be updated*
- DtTopoView *class DtTopoView: Instances of DtTopoView are used to create a Topographic coordinate view of the geocentric coordinates in an Entity State Repository*

- DtTrackJamBeamDecoder
- DtTrackJamBeamEncoder
- DtTrackVCPdu
- DtUnknownHlaObject *class DtUnknownHlaObject: Instances of DtUnknownHlaObject*
- DtUnknownInteraction
- DtUnknownVCPdu
- DtUpdateControlCbInfo *class DtUpdateControlCbInfo: Instances of DtUpdateControlCbInfo are used for providing a callback when the RTI update control services are invoked*
- DtUpdateControlCbInfo *class DtUpdateControlCbInfo: Instances of DtUpdateControlCbInfo are used for providing a callback when the RTI update control services are invoked*
- DtUpdateControlCbList
- DtUpdateControlCbList
- DtUtmCoord *class DtUtmCoord: Instances of DtUtmCoord are used to represent UTM coordinates*
- DtUtmView *class DtUtmView: Instances of DtUtmView are used to create a UTM coordinate view of the geocentric coordinates in an Entity State Repository*
- DtVarDatumSpec *This class has public data members for backwards compatibility with pre-VR-Link3*
- DtVariableDatumRecord *class DtVariableDatumRecord: Instances of DtVariableDatumRecord are used to hold a single variable datum record and to provide access to its network representation*
- DtVector *class DtVector: No virtual functions, because we want the size of a DtVector to 3 * 8 bytes - so that you can make the same assuptions about it that we used to make about its size and layout*
- DtViewControlDecoder
- DtViewControlEncoder
- DtViewControlInteraction
- DtViewControlPdu
- DtVrlFedTime
- DtVrlFederateAmbassador *class DtVrlFederateAmbassador: Instances of DtVrlFederateAmbassador are used to manage the interface between the RTI and the VR-Link Federate*
- DtVrlRtiAmbassador *class DtVrlRtiAmbassador: DtVrlRtiAmbassador is a purely virtual class, designed to provide a foundation for different ways to send RTI messages*

Functions

- (*DtAcknowledgeInteractionCb) *Acknowledge interaction callback function signature*
- (*DtActionRequestInteractionCb) *ActionRequest interaction callback function signature*
- (*DtActionRequestToObjInterCb) *ActionRequestToObject interaction callback function signature*
- (*DtActionResponseFromObjInterCb) *ActionResponseFromObject interaction callback function signature*
- (*DtActionResponseInteractionCb) *ActionResponse interaction callback function signature*
- (*DtAggregateCallbackFunction)
- (*DtAttributeChangeReqInterCb) *AttributeChangeRequest interaction callback function signature*
- (*DtAttributeChangeResInterCb) *AttributeChangeResult interaction callback function signature*
- (*DtAttributeChecker) *Attribute checking function declaration*
- (*DtAttributeDecoder) *Attribute decoding function declaration*

- (*DtAttributeEncoder) *Attribute encoding function declaration*
- (*DtBaseStateRepCreator) *Function prototype for functions that create DtStateRepositories*
- (*DtCollisionInteractionCb) *Collision interaction callback function signature*
- (*DtCommentInteractionCb) *Comment interaction callback function signature*
- (*DtCreateEntityInteractionCb) *CreateEntity interaction callback function signature*
- (*DtCreateObjReqInterCb) *CreateObjectRequest interaction callback function signature*
- (*DtCreateObjectResultInterCb) *CreateObjectResult interaction callback function signature*
- (*DtDataInteractionCb) *Data interaction callback function signature*
- (*DtDataQueryInteractionCb) *DataQuery interaction callback function signature*
- (*DtDesignatorCallbackFunction)
- (*DtDetonationInteractionCb) *Detonation interaction callback function signature*
- (*DtDiscoverObjectCb) *class DtDiscoverObjectCbInfo: Instances of DtDiscoverObjectCbInfo are used for providing a callback when an object which meets the federate's subscription criteria is discovered*
- (*DtEmitterBeamCallbackFunction)
- (*DtEmitterSystemCallbackFunction)
- (*DtEntityCallbackFunction)
- (*DtEventReportInteractionCb) *EventReport interaction callback function signature*
- (*DtFedAmbCreator)
- (*DtFireInteractionCb) *Fire interaction callback function signature*
- (*DtFomMapperCreator)
- (*DtFomMapperDeletor)
- (*DtGenericObjectCallbackFunction)
- (*DtHlaObjectCreator) *HLA object creation function signature*
- (*DtInteractionClassChooser) *A function that chooses an interaction class from the FOM based on the contents of a derived DtInteraction object*
- (*DtInteractionCreator) *Interaction creation function signature*
- (*DtLgrControlInteractionCb) *LgrControl interaction callback function signature*
- (*DtObjectClassChooser) *A function that chooses an object class from the FOM based based on the name of a C++ class derived from DtObjectPublisher, and arbitary user data (e*
- (*DtObjectReflectAttributeValuesCb) *Object reflect attribute values callback signature*
- (*DtParameterDecoder) *Parameter decoder function signature*
- (*DtParameterEncoder) *Attribute encoding function declaration*
- (*DtPostDrainCb) *class DtBaseExerciseConn: DtBaseExerciseConn is an abstract base class*
- (*DtRadioReceiverCallbackFunction)
- (*DtRadioTransmitterCallbackFunction)
- (*DtReceiveInteractionCb) *class DtReceiveInteractionCbInfo: Instances of DtReceiveInteractionCbInfo are used for providing a callback when an interaction is received from another federate*
- (*DtReflectAttributeValuesCb)
- (*DtReflectedObjectCallbackFunction) *Callback function that takes a DtReflectedObject and user data*
- (*DtRegControlCb) *class DtRegControlCbList: Instances of DtRegControlCbList are used to provide a list of*

callbacks (DtRegControlCbInfo) which are invoked when a registration control advisory is called by RTI

- (*DtRemoveEntityInteractionCb) *RemoveEntity interaction callback function signature*
- (*DtRemoveObjectCb) *class DtRemoveObjectCbInfo: Instances of DtRemoveObjectCbInfo are used for providing a callback when an object has been removed from the federation or no longer meets the interest criteria of the federate*
- (*DtRemoveObjectRequestInterCb) *RemoveObjectRequest interaction callback function signature*
- (*DtRemoveObjectResultInterCb) *RemoveObjectResult interaction callback function signature*
- (*DtRepairCompleteInteractionCb) *RepairComplete interaction callback function signature*
- (*DtRepairResponseInteractionCb) *RepairResponse interaction callback function signature*
- (*DtResupplyCancelInteractionCb) *ResupplyCancel interaction callback function signature*
- (*DtResupplyOfferInteractionCb) *ResupplyOffer interaction callback function signature*
- (*DtResupplyRecInteractionCb) *ResupplyReceived interaction callback function signature*
- (*DtRtiAmbCreator)
- (*DtServiceRequestInteractionCb) *ServiceRequest interaction callback function signature*
- (*DtSetDataInteractionCb) *SetData interaction callback function signature*
- (*DtSignalInteractionCb) *Collision interaction callback function signature*
- (*DtStartResumeInteractionCb) *StartResume interaction callback function signature*
- (*DtStartStopInterCb) *class DtStartStopInterCbList: Instances of DtStartStopInterCbList are used to provide a list of callbacks (DtStartStopInterCbInfo) which are called when the RTI's start or stopInteractionGeneration services are invoked*
- (*DtStateRepCreator) *Function prototype for functions that create DtAggregateStateRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtDesignatorRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtBeamRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtEmitterSystemRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtEntityStateRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtRadioReceiverRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtRadioTransmitterRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtAggregateStateRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtDesignatorRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtEmitterBeamRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtEmitterSystemRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtEntityStateRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtRadioReceiverRepositories*
- (*DtStateRepCreator) *Function prototype for functions that create DtRadioTransmitterRepositories*
- (*DtStopFreezeInteractionCb) *StopFreeze interaction callback function signature*
- (*DtUnknownInteractionCb) *Unknown interaction callback function signature*
- (*DtUpdateControlCb) *class DtUpdateControlCbList: Instances of DtUpdateControlCbList are used to provide a list of callbacks (DtUpdateControlCbInfo) which are called when an object starts updating*
- (*DtUpdateControlCb) *class DtUpdateControlCbList: Instances of DtUpdateControlCbList are used to provide a list of callbacks (DtUpdateControlCbInfo) which are called when an object starts updating*
- (*DtViewControlInteractionCb) *ViewControl interaction callback function signature*

- (const interClass& inter, RTI::ParameterHandleValuePairSet* params, RTI::ParameterHandle paramHandle)
- DtAbort *DtAbort() exits the program*
- DtAbsNotifyObserver *copy constructor*
- DtAbsRealTime *Return current wallclock time - seconds since Jan 1, 1970*
- DtAbsTime2ClockTime
- DtAbsoluteVCPdu
- DtAcknowledgeDecoder
- DtAcknowledgeEncoder
- DtAcknowledgeInteraction *default constructor*
- DtActionRequestDecoder
- DtActionRequestEncoder
- DtActionRequestInteraction *default constructor*
- DtActionRequestToObjDecoder
- DtActionRequestToObjEncoder
- DtActionRequestToObjInter *copy constructor*
- DtActionResponseDecoder
- DtActionResponseEncoder
- DtActionResponseFromObjDecoder
- DtActionResponseFromObjEncoder
- DtActionResponseFromObjInter *copy constructor*
- DtActionResponseInteraction *copy constructor*
- DtAddDatumId2TypeMapping
- DtAddEnumString
- DtAggregateEntityDecoder *copy constructor*
- DtAggregateEntityEncoder *constructor*
- DtAggregatePublisher *copy constructor - not implemented*
- DtAggregateStateRepository *copy constructor*
- DtAntennaPatternRecord *copy constructor*
- DtAppSpecificRadioSignalDecoder
- DtAppSpecificRadioSignalEncoder
- DtApproximator *copy constructor*
- DtArtPart *constructor*
- DtArtPartList *constructor*
- DtArtPartThresholder *copy constructor*
- DtAttachedPartList *copy constructor*
- DtAttributeChangeReqDec
- DtAttributeChangeReqEnc
- DtAttributeChangeReqInter *default constructor*

- DtAttributeChangeResDec
- DtAttributeChangeResEnc
- DtAttributeChangeResInter *default constructor*
- DtBaseEmbeddedSystemDecoder
- DtBaseEmbeddedSystemEncoder
- DtBaseEmitterBeamDecoder
- DtBaseEmitterBeamEncoder
- DtBaseEntityDecoder
- DtBaseEntityEncoder
- DtBaseEntityStateRepository *default constructor*
- DtBaseExerciseConn *copy constructor*
- DtBaseHashKey *copy constructor*
- DtBaseMsg *Constructor Object is not fully constructed until initMsg is called*
- DtBaseReflectedAggregate *constructor*
- DtBaseReflectedEntity *copy constructor*
- DtBeamAntennaPatternRecord *Initialize from DtAntennaPatternRecord*
- DtBecomeNonBlocking
- DtBecomeNonBlocking
- DtBecomeNonBlocking
- DtBecomeNonBlocking
- DtBecomeNonBlocking *designate the socket as non-blocking*
- DtBodyToMat
- DtBodyToRef_to_Euler *Given the transformation matrix that converts from a body frame to a reference frame (or visa-versa), produce the Euler angles between the body and the reference frames*
- DtBurstDescriptor
- DtCartesianView *constructor*
- DtClientSocket
- DtClockTime2AbsTime
- DtClose
- DtCollisionDecoder
- DtCollisionEncoder
- DtCollisionInteraction *default constructor*
- DtCommentDecoder
- DtCommentEncoder
- DtCommentInteraction *copy constructor*
- DtCompassVCPdu
- DtCoordTransform
- DtCoordTranslation
- DtCreateCross

- DtCreateEntityDecoder
- DtCreateEntityEncoder
- DtCreateEntityInteraction *default constructor*
- DtCreateObjReqDecoder
- DtCreateObjReqEncoder
- DtCreateObjReqInter *copy constructor*
- DtCreateObjectResultDecoder
- DtCreateObjectResultEncoder
- DtCreateObjectResultInter *copy constructor*
- DtCsFilterTest
- DtCurrentMapDatum
- DtDECLARE_ACK_PARAM_DECODER
- DtDECLARE_ACK_PARAM_ENCODER
- DtDECLARE_ACTREQOBJ_PARAM_DECODER
- DtDECLARE_ACTREQOBJ_PARAM_ENCODER
- DtDECLARE_ACTREQ_PARAM_DECODER
- DtDECLARE_ACTREQ_PARAM_ENCODER
- DtDECLARE_ACTRESPOBJ_PARAM_DECODER
- DtDECLARE_ACTRESP_PARAM_DECODER
- DtDECLARE_ACTRESP_PARAM_ENCODER
- DtDECLARE_ACTRSPOBJ_PARAM_ENCODER
- DtDECLARE_AGGREGATE_ATTR_CHECKER
- DtDECLARE_AGGREGATE_ATTR_DECODER
- DtDECLARE_AGGREGATE_ATTR_ENCODER
- DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER
- DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER
- DtDECLARE_ATTCHGREQ_PARAM_DECODER
- DtDECLARE_ATTCHGREQ_PARAM_ENCODER
- DtDECLARE_ATTCHGRES_PARAM_DECODER
- DtDECLARE_ATTCHGRES_PARAM_ENCODER
- DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER
- DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER
- DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER
- DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER
- DtDECLARE_BASE_ENTITY_ATTR_CHECKER
- DtDECLARE_BASE_ENTITY_ATTR_DECODER
- DtDECLARE_BASE_ENTITY_ATTR_ENCODER
- DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER

- DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER
- DtDECLARE_COLLISION_PARAM_DECODER
- DtDECLARE_COLLISION_PARAM_ENCODER
- DtDECLARE_COMMENT_PARAM_DECODER
- DtDECLARE_COMMENT_PARAM_ENCODER
- DtDECLARE_CREATENT_PARAM_DECODER
- DtDECLARE_CREATENT_PARAM_ENCODER
- DtDECLARE_CREATOBJRES_PARAM_DECODER
- DtDECLARE_CRTOBJREQ_PARAM_DECODER
- DtDECLARE_CRTOBJREQ_PARAM_ENCODER
- DtDECLARE_CRTOBJRES_PARAM_ENCODER
- DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER
- DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER
- DtDECLARE_DATAQ_PARAM_DECODER
- DtDECLARE_DATAQ_PARAM_ENCODER
- DtDECLARE_DATA_PARAM_DECODER
- DtDECLARE_DATA_PARAM_ENCODER
- DtDECLARE_DESIGNATOR_ATTR_CHECKER
- DtDECLARE_DESIGNATOR_ATTR_DECODER
- DtDECLARE_DESIGNATOR_ATTR_ENCODER
- DtDECLARE_DETONATION_PARAM_DECODER
- DtDECLARE_DETONATION_PARAM_ENCODER
- DtDECLARE_EMBEDSYS_ATTR_DECODER
- DtDECLARE_EMITTERBEAM_ATTR_DECODER
- DtDECLARE_EMITTERSYS_ATTR_CHECKER
- DtDECLARE_EMITTERSYS_ATTR_DECODER
- DtDECLARE_EMITTERSYS_ATTR_ENCODER
- DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER
- DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER
- DtDECLARE_ENTITY_ATTR_CHECKER *static need<Attribute> member functions*
- DtDECLARE_ENTITY_ATTR_CHECKER
- DtDECLARE_ENTITY_ATTR_CHECKER *static need<Attribute> member functions*
- DtDECLARE_ENTITY_ATTR_CHECKER *static need<Attribute> member functions*
- DtDECLARE_ENTITY_ATTR_CHECKER
- DtDECLARE_ENTITY_ATTR_CHECKER
- DtDECLARE_ENTITY_ATTR_DECODER *static decode<AttributeName> member functions*
- DtDECLARE_ENTITY_ATTR_DECODER
- DtDECLARE_ENTITY_ATTR_DECODER

- DtDECLARE_ENTITY_ATTR_DECODER *static decode<AttributeName> member functions*
- DtDECLARE_ENTITY_ATTR_DECODER
- DtDECLARE_ENTITY_ATTR_DECODER *static decode<AttributeName> member functions*
- DtDECLARE_ENTITY_ATTR_ENCODER *static decode<Attribute> member functions*
- DtDECLARE_ENTITY_ATTR_ENCODER
- DtDECLARE_ENTITY_ATTR_ENCODER
- DtDECLARE_ENTITY_ATTR_ENCODER *static decode<Attribute> member functions*
- DtDECLARE_ENTITY_ATTR_ENCODER
- DtDECLARE_ENTITY_ATTR_ENCODER
- DtDECLARE_EVENTREP_PARAM_DECODER
- DtDECLARE_EVENTREP_PARAM_ENCODER
- DtDECLARE_FIRE_PARAM_DECODER
- DtDECLARE_FIRE_PARAM_ENCODER
- DtDECLARE_JAMMERBEAM_ATTR_CHECKER
- DtDECLARE_JAMMERBEAM_ATTR_DECODER
- DtDECLARE_JAMMERBEAM_ATTR_ENCODER
- DtDECLARE_LGR_CONTROL_PARAM_DECODER
- DtDECLARE_LGR_CONTROL_PARAM_ENCODER
- DtDECLARE_RADARBEAM_ATTR_CHECKER
- DtDECLARE_RADARBEAM_ATTR_DECODER
- DtDECLARE_RADARBEAM_ATTR_ENCODER
- DtDECLARE_RADIORECVR_ATTR_CHECKER
- DtDECLARE_RADIORECVR_ATTR_DECODER
- DtDECLARE_RADIORECVR_ATTR_ENCODER
- DtDECLARE_RADIOXMIT_ATTR_CHECKER
- DtDECLARE_RADIOXMIT_ATTR_DECODER
- DtDECLARE_RADIOXMIT_ATTR_ENCODER
- DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER
- DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER
- DtDECLARE_REMOBJREQ_PARAM_DECODER
- DtDECLARE_REMOBJREQ_PARAM_ENCODER
- DtDECLARE_REMOBJRES_PARAM_DECODER
- DtDECLARE_REMOBJRES_PARAM_ENCODER
- DtDECLARE_REMOVENT_PARAM_DECODER
- DtDECLARE_REMOVENT_PARAM_ENCODER
- DtDECLARE_REPCOMP_PARAM_DECODER
- DtDECLARE_REPCOMP_PARAM_ENCODER
- DtDECLARE_REPRESP_PARAM_DECODER

- DtDECLARE_REPRESP_PARAM_ENCODER
- DtDECLARE_RESCANC_PARAM_DECODER
- DtDECLARE_RESCANC_PARAM_ENCODER
- DtDECLARE_RESOFF_PARAM_DECODER
- DtDECLARE_RESOFF_PARAM_ENCODER
- DtDECLARE_RESREC_PARAM_DECODER
- DtDECLARE_RESREC_PARAM_ENCODER
- DtDECLARE_SERVREQ_PARAM_DECODER
- DtDECLARE_SERVREQ_PARAM_ENCODER
- DtDECLARE_SETDATA_PARAM_DECODER
- DtDECLARE_SETDATA_PARAM_ENCODER
- DtDECLARE_START_PARAM_DECODER
- DtDECLARE_START_PARAM_ENCODER
- DtDECLARE_STOP_PARAM_DECODER
- DtDECLARE_STOP_PARAM_ENCODER
- DtDECLARE_TRKJAMBEAM_ATTR_DECODER
- DtDECLARE_VIEW_CONTROL_PARAM_DECODER
- DtDECLARE_VIEW_CONTROL_PARAM_ENCODER
- DtDataDecoder
- DtDataEncoder
- DtDataInteraction *default constructor*
- DtDataQueryDecoder
- DtDataQueryEncoder
- DtDataQueryInteraction *default constructor*
- DtDatabaseIndexRadioSignalDecoder
- DtDatabaseIndexRadioSignalEncoder
- DtDatumId2Type
- DtDatumRecordSet *Constructor*
- DtDcmAdd
- DtDcmAdjugate
- DtDcmCheck
- DtDcmCopy *make orthonormal, pick most important*
- DtDcmDcmMul *result = m1 - m2*
- DtDcmDcmRotAngle *Direction Cosine Matrix routines*
- DtDcmDcmRotAxis
- DtDcmDcmRotTrace *Returns the trace of the rotation matrix, R, between between matricies A and B*
- DtDcmDeterminant
- DtDcm

- DtDcmDump
- DtDcmFixMatrix *Is this matrix orthonormal?*
- DtDcmForm
- DtDcmFromQuaternion
- DtDcmIdentInit *Initialize to all zeros*
- DtDcmInit
- DtDcmInverse
- DtDcmLevelInit
- DtDcmRotInit *Rotation is counterclockwise when viewed a positive axis*
- DtDcmRotInit2
- DtDcmScale
- DtDcmSub *result = m1 + m2*
- DtDcmToElr
- DtDcmTranspose *result = m1 x m2*
- DtDcmTrigInit

  ```
  $ only pos rots?
  ```
- DtDcmVecMul *Combined vector/matrix routines*
- DtDeadReckoner *default constructor*
- DtDeg2DmsString
- DtDesignatorDecoder
- DtDesignatorEncoder
- DtDesignatorPublisher *constructor*
- DtDesignatorRepository *default constructor*
- DtDetonationDecoder
- DtDetonationEncoder
- DtDetonationInteraction *default constructor*
- DtDirectRotRate *directed rotation rates*
- DtDirectRtiAmbassador *default constructor*
- DtDirectedRotRate
- DtDiscoverObjectCbInfo *constructor*
- DtDistSqr *||p1-p2||_{2}*
- DtDistance *additional Routines*
- DtDumpBoolean
- DtDumpEnum
- DtDumpFloat
- DtDumpHexBytes
- DtDumpHexWord
- DtDumpInteger
- DtDumpString

- DtEXTEND_ATTR_ENUM *Areal object specific appearance*
- DtEXTEND_ATTR_VAL_ENUM
- DtEXTEND_BIT_SHIFT_ENUM
- DtElrCopy
- DtElrElrCat
- DtElrForm
- DtElrIdent
- DtElrToMat
- DtElrTranspose
- DtEmbeddedSystemRepository *copy constructor*
- DtEmitterBeamPublisher *copy constructor*
- DtEmitterBeamRepository *default constructor*
- DtEmitterSystemDecoder
- DtEmitterSystemEncoder
- DtEmitterSystemPublisher *constructor*
- DtEmitterSystemRepository *copy constructor*
- DtEmptyFomMapper *copy constructor - not implemented*
- DtEncodedAudioRadioSignalDecoder
- DtEncodedAudioRadioSignalEncoder
- DtEntityIdentifier
- DtEntityPublisher *copy constructor*
- DtEntityStateRepository *default constructor*
- DtEntityType
- DtEnuToNedDcm
- DtEnumString
- DtEnumString
- DtEnumString
- DtEnvironmentEntityDecoder *constructor*
- DtEnvironmentEntityEncoder *constructor*
- DtEulerToEuler *Given Euler angles with respect to reference frame "in", and the transformation matrix between frame "in" and frame öut", produce the Euler angles with repect to reference frame öut"*
- DtEuler_to_BodyToRef *Given Euler angles between a body and a reference frame, produce the transformation matrix that converts from the body frame to the reference frame (or visa-versa)*
- DtEuler_to_RefToBody
- DtEventId *Construct a DtEventId from a name and an event number*
- DtEventReportDecoder
- DtEventReportEncoder
- DtEventReportInteraction *default constructor*
- DtExerciseConn *copy constructor*

- DtFatalError *These functions print messages using stdio depending on the notify level*
- DtFatalNetPerror
- DtFatalPerror
- DtFedReader *Constructor*
- DtFedTarget *Constructor*
- DtFederateId
- DtFilename *copy constructor*
- DtFilterTest
- DtFireDecoder
- DtFireEncoder
- DtFireInteraction *default constructor*
- DtFirstHostInetAddr
- DtFom *Constructor*
- DtFomMapper *Default constructor*
- DtGenericObjectDecoder *default constructor*
- DtGenericObjectEncoder *default constructor*
- DtGenericObjectPublisher *Constructor - configure base class with proper FOM class, state repositories, encoder and decoder*
- DtGenericStateRepository *copy constructor*
- **DtGeocToGeod :**
  - in duplicate *Conversions between geocentric and geodetic: Use of these functions is discouraged in favor of DtGeodeticCoord member functions which perform conversions*
  - in DtGeodeticCoord
- DtGeocToTopoTransform *This function can initialize a DtCoordTransform so that it can provide transformations between the geocentric coordinate system, and a particular topographic coordinate system defined by latitude and longitude*
- **DtGeocToUtm :**
  - in duplicate *Use of these functions is discouraged in favor of DtUtmCoord member functions which perform conversions*
  - in DtUtmCoord
- DtGeodToGeoc
- **DtGeodToUtm :**
  - in duplicate *Use of these functions is discouraged in favor of DtUtmCoord member functions which perform conversions*
  - in DtUtmCoord
- DtGeodeticCoord *constructor*
- DtGetApproximateElapsedRealTime *Returns the value of DtGetElapsedRealTime at the time that DtSetSimTime was lask invoked*
- DtGetBroadcastMask
- DtGetElapsedRealTime *Return real seconds elapsed since DtTimeInit was called*
- DtGetSimTime *Returns the last value passed to DtSetSimTime - the current value of VR-Link simulation time*

- DtGlobalIdHashlist *copy constructor*
- DtGlobalObjectDesignatorList *default constructor*
- DtGridDeclinationDcm
- DtGroupSocket *Constructor - uses default address for listening*
- DtGroupVCPdu
- DtGtPacketDequeue
- DtGtPacketEnqueue
- DtGtRecv
- DtGtSend *No, but Ethernet does, and that's the only thing the GT100 will ever work with*
- DtHashItem *default constructor*
- DtHashlist *Constructor for a DtHashlist that uses an external DtList constructed by the user*
- DtHlaObject
- DtHlaObjectFactory *constructor*
- DtHlaObjectWithStateRep *constructor*
- DtHlaStateDecoder *constructor*
- DtHlaStateEncoder *copy constructor*
- DtInetAddrOfDevice
- DtInetAddrString *converts ulong to internet dot notation (returns internal static buffer)*
- DtInetBroadcastOfDevice
- DtInfo
- DtInitNetSocket
- DtIntHashKey *default constructor*
- DtInterClassDesc *copy constructor*
- DtInteractionDecoder *copy constructor*
- DtInteractionDecoderFactory *constructor*
- DtInteraction *default constructor*
- DtInteractionEncoder *constructor*
- DtInteractionEncoderFactory *copy constructor*
- DtInteractionFactory *constructor*
- DtInteractionManager *constructor*
- DtInteractionWithData *default constructor*
- DtInteractionWithEncDec *copy constructor*
- DtIsInList *Returns -1 if num is not in lst*
- DtIsMulticastAddr
- DtIsSimTimeSet *Abort if DtSetSimTime was never called*
- DtJammerBeamDecoder
- DtJammerBeamEncoder
- DtLatLon_to_GeocToTopo *The matrix produced by DtLatLon_to_GeocToTopo can be used to rotate a geocentric vector to a topographic vector like this: DtDcmVecMul(geocToTopo, Vgeoc, Vtopo);*

- DtLatLon_to_TopoToGeoc *From latitude and longitude, we can produce the rotation matrices that rotate between geocentric and topographic-down orientations*
- DtLgrControlDecoder
- DtLgrControlEncoder
- DtLgrControlInteraction *default constructor*
- DtLgrControlPdu *default constructor*
- DtLgrEndPdu *copy constructor*
- DtLgrEofActionPdu *copy constructor*
- DtLgrHeartbeatPdu *construct and prepare for send*
- DtLgrPausePdu *default constructor*
- DtLgrPbFilePdu *copy constructor*
- DtLgrPlayActionPdu *construct from network representation*
- DtLgrPlayPdu *construct and prepare for send*
- DtLgrQuitPdu *construct from network representation*
- DtLgrRecFilePdu *construct and prepare for send*
- DtLgrRecordActionPdu *default constructor*
- DtLgrRecordPdu *construct from network representation*
- DtLgrSetTimePdu *construct and prepare for send*
- DtLgrSkipTimePdu *construct from network representation*
- DtLgrStartPdu *default constructor*
- DtLgrStopPdu *construct and prepare for send*
- DtLgrTimeScalePdu *copy constructor*
- DtLgrTimeoutPdu *default constructor*
- DtLimit
- DtList *Constructor*
- DtListItem *Constructor - called only by DtList*
- DtListedObj *Constructor*
- DtLocalHlaObjectManager *copy constructor*
- DtM
- DtMM_Mult
- DtM_Init
- DtMakeRotationDcm
- DtMax
- DtMilitaryEntityDecoder *copy constructor*
- DtMilitaryEntityEncoder *copy constructor*
- DtMilitaryPlatformEntityDecoder *copy constructor*
- DtMilitaryPlatformEntityEncoder *constructor*
- DtMimicTrackVCPdu
- DtMimicVCPdu

- DtMin
- DtModPer *For x with period ABS(incl-excl), return DtModPerLo(x, incl, excl) if incl < excl return DtModPerHi(x, excl, incl) if incl > excl*
- DtModPerHi *For x with period (max-min), return min < x <= max*
- DtModPerLo *For x with period (max-min), return min <= x < max*
- DtModPerZero *For x with period per, return 0 <= x < per*
- DtModelessVCPdu
- DtMunitionEntityDecoder *copy constructor*
- DtMunitionEntityEncoder *constructor*
- DtNedToEnuDcm
- DtNet32Vector
- DtNet64Vector
- DtNetBoolean *default constructor*
- DtNetEulerAngles
- DtNetEventId
- DtNetFloat32
- DtNetFloat64 *Least Significant Least*
- DtNetInt16
- DtNetInt32
- DtNetMaskOfDevice
- DtNetSocket
- DtNetTimeStamp *copy constructor*
- DtNetU16
- DtNetU32 *Least Significant Word*
- DtNewDfltSocket
- DtNewViewControlPdu
- DtNullFederateAmbassador
- DtObjClassDesc *Constructor*
- DtObjectId
- DtObjectIdHashKey *copy constructor*
- DtObjectIdHashlist *default constructor*
- DtObjectPublisher *copy constructor*
- DtObjectReflectAttributeValuesCbInfo *constructor*
- DtOrbitVCPdu
- DtPdu *Copy constructor*
- DtPhysicalEntityDecoder *constructor*
- DtPhysicalEntityEncoder *copy constructor*
- DtPollInputLine *polls line-buffered standard input returns NULL if no input returns internal buffer of line if have input exits on ^D*

- DtQuatQuatMul
- DtQuaternionDump *void DtQuatScale(DtQuaternion *qresult, DtQuaternion *q, DtReal delta_t);*
- DtQuaternionFromTaitBryan
- DtROUND_TO_MULT *Rounds num up to a multiple of mult*
- DtRadarBeamDecoder
- DtRadarBeamEncoder
- DtRadioEntityType
- DtRadioReceiverDecoder
- DtRadioReceiverEncoder
- DtRadioReceiverPublisher *constructor*
- DtRadioReceiverRepository *default constructor*
- DtRadioTransmitterDecoder
- DtRadioTransmitterEncoder
- DtRadioTransmitterPublisher *constructor*
- DtRadioTransmitterRepository *copy constructor*
- DtRawBinaryRadioSignalDecoder
- DtRawBinaryRadioSignalEncoder
- DtReal
- DtReceiveInteractionCbInfo *constructor*
- DtRecv
- DtRecv
- DtRecv
- DtRecv
- DtRecv
- DtRefToBody_to_Euler
- DtReflectAttributeValuesCbInfo *constructor*
- DtReflectAttributeValuesCbList *default constructor*
- DtReflectedAggregate *constructor*
- DtReflectedAggregateList *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*
- DtReflectedDesignator *constructor*
- DtReflectedDesignatorList *Constructor - obtains the set of object classes to subscribe to from the exConn's fomMapper*
- DtReflectedEmitterBeam *copy constructor*
- DtReflectedEmitterBeamList *copy constructor*
- DtReflectedEmitterSystem *Constructor used to construct a placeholder system when we discover a beam whose system hasn't been discovered yet*
- DtReflectedEmitterSystemList *Constructor If beamsUpdateSystems == DtTrue, then a system's post-update callbacks are called whenever one of the beams in the system is updated*
- DtReflectedEntity *copy constructor*
- DtReflectedEntityList *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*

- DtReflectedGenericObject *constructor*
- DtReflectedGenericObjectList *Constructor*
- DtReflectedHlaObjectManager *copy constructor*
- DtReflectedObjectCallbackInfo *default constructor*
- DtReflectedObject *Primary constructor*
- DtReflectedObjectList *Constructor*
- DtReflectedRadioReceiver *copy constructor*
- DtReflectedRadioReceiverList *copy constructor*
- DtReflectedRadioTransmitter *constructor*
- DtReflectedRadioTransmitterList *Constructor*
- DtRegControlCbInfo *constructor*
- DtRegControlCbList *default constructor*
- DtRemoveEntityDecoder
- DtRemoveEntityEncoder
- DtRemoveEntityInteraction *copy constructor*
- DtRemoveObjectCbInfo *copy constructor*
- DtRemoveObjectRequestDecoder
- DtRemoveObjectRequestEncoder
- DtRemoveObjectRequestInter *default constructor*
- DtRemoveObjectResultDecoder
- DtRemoveObjectResultEncoder
- DtRemoveObjectResultInter *copy constructor*
- DtRepairCompleteDecoder
- DtRepairCompleteEncoder
- DtRepairCompleteInteraction *copy constructor*
- DtRepairResponseDecoder
- DtRepairResponseEncoder
- DtRepairResponseInteraction *default constructor*
- DtReportPktsRcvd
- DtReportPktsRcvd
- DtResupplyCancelDecoder
- DtResupplyCancelEncoder
- DtResupplyCancelInteraction *copy constructor*
- DtResupplyOfferDecoder
- DtResupplyOfferEncoder
- DtResupplyOfferInteraction *copy constructor*
- DtResupplyRecInteraction *default constructor*
- DtResupplyReceivedDecoder

- DtResupplyReceivedEncoder
- DtRotate
- DtRprFomMapper *copy constructor*
- DtRprHlaObjectFactory *constructor*
- DtRprInteractionDecoderFactory *constructor*
- DtRprInteractionEncoderFactory *constructor Adds RPR FOM specific encoders*
- DtRprInteractionFactory *constructor Adds creators for the RPR FOM interactions*
- DtRprStateDecoderFactory *copy constructor*
- DtRprStateEncoderFactory *constructor Adds RPR FOM specific encoders*
- DtScript
- DtSelect
- DtSelectParam
- DtSend *Virtual function override - send to all remote sites*
- DtSend
- DtServiceRequestDecoder
- DtServiceRequestEncoder
- DtServiceRequestInteraction *copy constructor*
- DtSetDataDecoder
- DtSetDataEncoder
- DtSetDataInteraction *default constructor*
- DtSetSimTime *Set the current value of VR-Link simulation time*
- DtSetSym
- DtSignalInteraction *default constructor*
- DtSilentAggregateList *Constructors*
- DtSilentEntityList
- DtSleep
- DtSmoothOriWithDr *copy constructor*
- DtSmoothOrient *Constructors*
- DtSmoothPos
- DtSmoothPosWithDr *copy constructor*
- DtSmoother *copy constructor*
- DtSocket
- DtSocketPairSocket
- DtSoldierDecoder *copy constructor*
- DtSoldierEncoder *copy constructor*
- DtStartResumeDecoder
- DtStartResumeEncoder
- DtStartResumeInteraction *default constructor*

- DtStartStopInterCbInfo *copy constructor*
- DtStartStopInterCbList *copy constructor*
- DtStateDecoderFactory *copy constructor Copies across added decoder creation functions*
- DtStateEncoderFactory *copy constructor*
- DtStateMsg *copy constructor*
- DtStateRepository *copy constructor*
- DtStopFreezeDecoder
- DtStopFreezeEncoder
- DtStopFreezeInteraction *copy constructor*
- DtStrdup *Strdup wrapper*
- DtString *default constructor*
- DtStringHashKey *copy constructor*
- DtStringToInetAddr
- DtSum
- DtSystemHashKey *default constructor*
- DtTaitBryan
- DtTbToMat
- DtTcpForwarder
- DtTcpSocket *Client Constructor*
- DtTcpSocketMgr
- DtTetherTrackVCPdu
- DtTetherVCPdu
- DtThresholder *default constructor*
- DtTimeInit *Initialize all VR-Link time functionality This must be called before any other time-related functions*
- DtTimeNow *Obsolete*
- DtToLower *Convert str to all lowercase*
- DtToUpper *Convert str to all uppercase*
- DtTopoView *copy constructor*
- DtTrackJamBeamDecoder
- DtTrackJamBeamEncoder
- DtTrackVCPdu
- DtTypeMatchPattern
- DtUnknownHlaObject *copy constructor*
- DtUnknownInteraction *constructor*
- DtUnknownVCPdu
- DtUpdateControlCbInfo *copy constructor*
- DtUpdateControlCbInfo *constructor*
- DtUpdateControlCbInfo *constructor*

- DtUpdateControlCbList *default constructor*
- DtUpdateControlCbList *default constructor*
- DtUpdateControlCbList *copy constructor*
- DtUseMapDatum *Set and Get current map datum NOTE: map datum memory passed to DtUseMapDatum() must remain valid until next change of datum*
- DtUtmCoord *constructor, construct UTM from geodetic*
- DtUtmGridDeclination *Use of these functions is discouraged in favor of DtUtmCoord member functions which perform conversions*
- DtUtmInit *Sets reference point for all DtUtmCoords Vrlink presently supports only one UTM reference at a time, but the reference point may be reinitialized at any point*
- DtUtmToGeoc
- DtUtmToGeod
- DtUtmView *copy constructor*
- DtVV_Product
- DtVariableDatumRecord *default constructor*
- DtVecAdd *result = -vec*
- DtVecBoxProd *sweep v1 to v2*
- DtVecCheck
- DtVecCopy *Initialize to all zeros*
- DtVecCosProd *result = v1 - v2*
- DtVecCrossProd
- DtVecDcmMul
- DtVecDotProd *cos of angle between vecs*
- DtVecDump
- DtVecElrTransform
- DtVecInfNorm
- DtVecInit
- DtVecNeg
- DtVecNormalize
- DtVecScale *Is this vector normalized?*
- DtVecSub
- DtVector *default constructor*
- DtVerbose
- DtViewControlDecoder
- DtViewControlEncoder
- DtViewControlInteraction *default constructor*
- DtVrlFedTime ---------- *Constructors and Destructors* ----------
- DtVrlFederateAmbassador *default constructor*
- DtVrlRtiAmbassador *default constructor*

- DtWarn
- DtWarnNetPerror
- DtWarnPerror
- ERP
- PRF
- PrivateListenToMulticastGroup
- absolute *Returns DtTrue if the filename is an absolute filename, or a relative filename, respectively*
- acceleration
- acceleration
- acceleration
- acceleration
- acceleration
- acceleration
- accept
- acknowledgeFlag
- action
- action *Set/Get value for what to do when Logger reaches EOF*
- action
- action
- actionId
- actionResult
- addAfter *Add element to the list just after the indicated item*
- addAggregateAdditionCallback *Aggregate added callback registration/deregistration*
- addAggregateRemovalCallback *Aggregate removed callback registration/deregistration*
- addAttr *Add an attribute (Used when reading FED file)*
- addAttributeChecker *Same as above, but identify class and attr by name*
- addAttributeDecoder *Same as above, but identify class and attr by name*
- addAttribute *Define an attribute of a class*
- addAttribute *Define an attribute of a class*
- addAttributeEncoder *This function is used to add an attribute encoder function to several prototype encoders all at once - namely, those associated with the object class denoted by hand, as well as all of its subclasses*
- addBeam *add/remove a new emitter beam within this system if add a beam with duplicate ID, retcode will be set to DtDuplicateBeam, and beamSRById will return the first beamSR with the given ID*
- addBefore
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*

- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Register/Deregister callback function cb with conn, on all interaction class name (char*) contained in classes*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction with specified class handle occurs*
- addCallback *Add (register)/Remove (deregister) function to be called when interaction occurs*
- addChecker *Add/Remove a checker function for the attribute specified by name*
- addCreator *Add an object creation function to the factory*
- addCreator
- addDecoder *Add/Remove a decoding function for the attribute specified by handle*
- addDecoder *Add/Remove a decoding function for the parameter specified by handle*
- addDecoder *Add a decoder prototype to the factory*
- addDecoder *Add a decoder prototype to the factory*
- addDesignatorAdditionCallback *Designator added callback registration/deregistration*

- addDesignatorRemovalCallback *Designator removed callback registration/deregistration*
- addDiscoverObjectCallback *Add a discoverObject callback which is called with the user data when the specified interaction is received*
- addDiscoverObjectCallback *Add/remove a callback to be called when an object which meets the federate's subscription criteria is discovered*
- addDiscoverObjectCb *Add/Remove (register/deregister) function to be called when the RTI's discoverObject service is invoked*
- add *Add an association between key and data to the list*
- add *Add something to the list*
- add *Add element to the end of the list*
- add *Add an association between key and data to the list*
- add *Add/Remove a reflect attribute callback to/from the list*
- add *Add/Remove a start updates callback to/from the list*
- add *Add/Remove a start updates callback to/from the list*
- add *Add/Remove a start updates callback to/from the list*
- add *Add/Remove a start updates callback to/from the list*
- addEmitterBeamAdditionCallback *EmitterBeam added callback registration/deregistration*
- addEmitterBeamRemovalCallback *EmitterBeam removed callback registration/deregistration*
- addEmitterSystemAdditionCallback *EmitterSystem added callback registration/deregistration Called when an emitter system is added due to discovery of a system OR beam*
- addEmitterSystemRemovalCallback *EmitterSystem removed callback registration/deregistration Called when an emitter system is removed due to discovery of a system OR beam*
- addEncoder *Add/Remove an encoding function for the attribute specified by handle*
- addEncoder *Add a parameter encoder function by parameter handle*
- addEncoder *Add a encoder prototype to the factory*
- addEncoder *Add a encoder prototype to the factory*
- addEntityAdditionCallback *Entity added callback registration/deregistration*
- addEntityRemovalCallback *Entity removed callback registration/deregistration*
- addGenericAdditionCallback *Registration/deregistration of object addition callbacks*
- addGenericRemovalCallback *Registration/deregistration of object removal callbacks*
- addInstruction
- addInterClass *Define a class*
- addInterClass *Define a class*
- addInteractionCallbackByName *Add an interaction callback by class name, which is called with the user data when the specified interaction is received*
- addInteractionCallback *Add an interaction callback by class handle, which is called with the user data when the specified interaction is received*
- addInteractionCallback *Add/remove a callback to be called when an interaction is received from another federate*
- addMyDecoders *Add decoding functions for the attributes*
- addMyDecoders
- addMyDecoders

- addMyDecoders
- addMyDecoders
- addMyDecoders
- addMyDecoders *Add decoding functions for the attributes*
- addMyDecoders
- addMyDecoders *Add decoding functions for the attributes*
- addMyDecoders *Add decoding functions for the attributes*
- addMyDecoders *Add decoding functions for the attributes*
- addMyDecoders *Add decoding functions for the attributes*
- addMyDecoders
- addMyDecoders
- addMyDecoders
- addMyDecoders *Add decoding functions for the attributes*
- addMyDecoders
- addObjClass *Define a class*
- addObjClass *Define a class*
- addObjectAdditionCallback *Object added callback registration/deregistration*
- addObjectClassByName *Register interest in this FOM object class by className*
- addObjectClass *Register interest in this FOM object class*
- addObjectRemovalCallback *Object removed callback registration/deregistration*
- addParam *Add the specified parameter type to the part*
- addParam *Add a parameter (Used when reading FED file)*
- addParameterDecoder *This function is used to add an parameter decoder function to several prototype decoders all at once - namely, those associated with the interaction class denoted by hand, as well as all of its subclasses*
- addParameter *Define a parameter of a class*
- addParameter *Define an parameter of a class*
- addParameterEncoder *Same as above, but identify class and param by name*
- addPostDrainCallback *add/remove callbacks to be executed from within drainInput right before it returns*
- addPostReflectCb *Add (register) function to be called after each reflectAttrValues*
- addPostUpdateCallback *Aggregate state callback registration/deregistration*
- addPostUpdateCallback *designator state callback registration/deregistration*
- addPostUpdateCallback *beam state callback registration/deregistration*
- addPostUpdateCallback *emitterSystem state callback registration/deregistration*
- addPostUpdateCallback *Entity state callback registration/deregistration*
- addPostUpdateCallback *Registration/deregistration of postUpdate callbacks*
- addPostUpdateCallback *receiver state callback registration/deregistration*
- addPostUpdateCallback *transmitter state callback registration/deregistration*
- addRadioReceiverAdditionCallback *RadioReceiver added callback registration/deregistration*
- addRadioReceiverRemovalCallback *RadioReceiver removed callback registration/deregistration*

- addRadioTransmitterAdditionCallback *RadioTransmitter added callback registration/deregistration*
- addRadioTransmitterRemovalCallback *RadioTransmitter removed callback registration/deregistration*
- addRead
- addReceiveInteractionCb *Add/Remove (register/deregister) function to be called when the RTI's receiveInteraction service is invoked*
- addReflectAttributeValuesCb *Add/Remove (register/deregister) function to be called when the RTI's reflectAttributeValues service is invoked*
- addRemoteSite *Add/remove a remote site to send to*
- addRemoveObjectCallback *Add a removeObject callback which is called with the user data when the specified interaction is received*
- addRemoveObjectCallback *Add/remove a callback to be called when an object has been removed from the federation or no longer meets the interest criteria of the federate*
- addRemoveObjectCb *Add (register) function to be called when object is removed*
- addRemoveObjectCb *Add/Remove (register/deregister) function to be called when the RTI's removeObject service is invoked*
- addStartInteractionsCb *Add/Remove (register/deregister) function to be called when the RTI's startInteractionGeneration service is invoked*
- addStartRegCb *Add/Remove (register/deregister) function to be called when the RTI's startRegistration service is invoked*
- addStopInteractionsCb *Add/Remove (register/deregister) function to be called when the RTI's stopInteractionGeneration service is invoked*
- addStopRegCb *Add/Remove (register/deregister) function to be called when the RTI's stopReg service is invoked*
- addToEnd *Add element to the end of the list*
- **addToList :**
  - in DtIntrusiveList *Override of virtual function*
  - in DtList *Add item to the list after after*
- addToStart
- addUpdatesOffCb *Add/Remove (register/deregister) function to be called when the RTI's turnUpdatesOff service is invoked*
- addUpdatesOnCb *Add/Remove (register/deregister) function to be called when the RTI's turnUpdatesOn service is invoked*
- afterburnerOn
- aggDimensionThreshold
- aggregateAdded *Called by objectAdded (which is called from processDiscoverObject()) after a remote aggregate has been added to the list*
- aggregateID
- aggregateObjectId *Get the aggregate HLA object's id For backwards compatibility*
- aggregateState
- aggregateStateRep
- aggregateStateRep *Get the aggregate's state repository*
- aggregateStateRep *Returns a pointer to the AggregateStateRepository - through which you can inspect current state information*
- aggregateType

- aggregateType
- ahvps *Get the attribute handle value pair set*
- algorithm
- algorithm
- allHlaObjects *Get the list of all HLA objects*
- allHlaObjects *Get the list of all HLA objects*
- alt
- angles *Get the euler angles*
- angles
- announceSynchronizationPoint
- antennaPattern
- antennaPatternType
- appearanceAttribute
- appearance
- applicationId
- apply *State messages are for a specific object id*
- apply *Callbacks are for a particular object class handle*
- apply *Callbacks are for a particular inter class handle*
- apply *Callbacks are for a particular object class handle*
- apply *Callbacks are for a particular object class handle*
- applyPostReflectCallbacks *apply callbacks after a state msg is reflected*
- approxBodyToGeoc *Get the approximated data body to geocentric coordinate matrix*
- approxBodyToGeoc *Get the approximated (i*
- approxBodyToGeoc *Get the approximated data body to geocentric coordinate matrix*
- approxEuler *Get approximated orientation (euler angles) at the specified time*
- approxEuler
- approxEuler *Get approximated orientation (euler angles) at the specified time*
- approxLocation *Get the approximated location at the specified time*
- approxLocation *Get the approximated (i*
- approxLocation *Get the approximated location at the specified time*
- approxVal *Get the approximated value*
- approxVelocity *Get the approximated velocity at the specified time*
- approxVelocity *Get the approximated (i*
- approxVelocity *Get the approximated velocity at the specified time*
- approximating
- approximator
- artNotAttached *Returns true if param is an articulated parameter record, false if its an attached part record*
- artParamIsRate *Get whether the parameter type is actually a rate*

- artPartFromPartNum *Get the articulated part from the part number*
- artPartFromPartType *Get the first articulated part with the part type*
- artPartList
- artPartList
- artPartSpecs *return the list of DtArtPartSpecs describing the parts in the list*
- artPartSpecs
- artPartThreshold *Get the thresholding value (functional, translational, or rotational) for the parameter type*
- artPartThresholder
- artPartsExceedAggThreshold *Get whether or not an aggregate threshold (such as position, translation, rotation, etc*
- artPartsExceedIndivThreshold *Get whether or not an individual parameter threshold has been exceeded*
- asr *Synonym for aggregateStateRep()*
- asr *Synonym for aggregateStateRep()*
- associateRegionForUpdates *Calls down to the RTI function of the same name*
- associateRegionForUpdates
- attPartList
- **attachId :**
  - in DtCompassVCPdu
  - in DtMimicTrackVCPdu
  - in DtMimicVCPdu
  - in DtOrbitVCPdu
  - in DtTetherTrackVCPdu
  - in DtTetherVCPdu
- attackerId
- attackerId
- attrHandleSet
- attrHandleSet *Get the set of valid attribute handles for this class*
- attributeChangeResult
- attributeHandle
- attributeHandle
- attributeHandle
- attributeIsNotOwned
- attributeOwnedByRTI
- attributeOwnershipAcquisition *Calls down to the RTI function of the same name*
- attributeOwnershipAcquisition
- attributeOwnershipAcquisitionIfAvailable *Calls down to the RTI function of the same name*
- attributeOwnershipAcquisitionIfAvailable
- attributeOwnershipAcquisitionNotification
- attributeOwnershipDivestitureNotification

- attributeOwnershipReleaseResponse *Calls down to the RTI function of the same name*
- attributeOwnershipReleaseResponse
- attributeOwnershipUnavailable
- attributeValue
- attributeValue
- attributeValue
- attributeValueLength
- attributeValueLength
- attributeValueLength
- attributesInScope
- attributesNeededByFederation *returns the set of attributes that are currently needed by the federation, i*
- attributesNeededByFederation
- attributesOutOfScope
- avPairSet
- avPairSet
- avPairSet
- azimuthBeamwidth *Get the azimuth beam bandwidth*
- azimuthCenter
- azimuthRate
- azimuthSweep
- baseAddPostUpdateCallback *Object state callback registration/deregistration*
- baseClass *Return a pointer to the base class's descriptor, NULL if none*
- baseClass *Return a pointer to the base class's descriptor, NULL if none*
- baseRemovePostUpdateCallback
- beamFunction
- beamId
- beamList *Returns DtList of DtEmitterBeamStateRepositories*
- beamParameterIndex
- beamRep
- beamSRById *Return a pointer to the first BeamSR with specified beamID*
- beamType
- beamUpdated *Tell the emitter system that one of its beams has been updated*
- bodyHeading
- bodyOrientation
- bodyToGeoc *Get the body to geocentric coordinate matrix*
- bodyToLocal *Get the body to local coordinate matrix*
- bodyToLocal *Get the body to local coordinate matrix*
- bsr *Synonym for emitrBeamRep()*

- bsr
- bucketFromHashKey *Determine the bucket from the hashkey*
- bucketItemLookup *Lookup an item in a bucket*
- byteLengthOfNetworkRecord *Get the length of the network representation of a record*
- byteLengthOfNetworkRecord *Get the length of the network representation of a record*
- bzero
- callReflect *Calls reflect, and any postReflect callbacks Override reflect rather than this function to change the way an HLA object reflects a stateMsg*
- callbackFunction
- camouflageType
- canRead
- cancelAttributeOwnershipAcquisition *Calls down to the RTI function of the same name*
- cancelAttributeOwnershipAcquisition
- cancelNegotiatedAttributeOwnershipDivestiture *Calls down to the RTI function of the same name*
- cancelNegotiatedAttributeOwnershipDivestiture
- capabilities
- caseCompare *Case-insensitive string compare*
- category
- cbFunction *Get the callback function*
- cbFunction *Get the callback function*
- cbFunction
- cbFunction *Get the callback function*
- cbFunction *Get the callback function*
- cbFunction
- cbFunction
- cbFunction *Get the callback function*
- cbFunction
- cbFunction
- changeAttributeOrderType *Calls down to the RTI function of the same name*
- changeAttributeOrderType
- changeAttributeTransportationType *Calls down to the RTI function of the same name*
- changeAttributeTransportationType
- changeFixedSize *Change size of fixed or variable fields*
- changeInteractionOrderType *Calls down to the RTI function of the same name*
- changeInteractionOrderType
- changeInteractionTransportationType *Calls down to the RTI function of the same name*
- changeInteractionTransportationType
- changeSize *Changes the size of the message*
- changeSuppliesSize

- changeSuppliesSize
- changeSuppliesSize
- changeVariableSize
- checkGoAway
- checker *Get the attribute checker for the specified handle*
- chooseInteractionClass *Chooses a FOM interaction class to be by a particular derived DtInteraction class named vrlinkClassName*
- chooseObjectClass *Chooses an object class to represent an object managed by the C++ class named vrlinkClassName (which should be a class derived from DtObjectPublisher)*
- classDesc *Get the class descriptor for the object's class*
- classDesc *Get the current class descriptor we're using for this interaction*
- classNeededByFederation *returns DtTrue if the class is needed by the federation, i*
- classNeededByFederation
- classToUse *Returns the name of the FOM object class to use*
- classToUse *Returns the name of the FOM object class handle to use*
- classToUse *Returns the name of the FOM object class to use*
- classToUse *Returns the name of the FOM object class to use*
- classToUse *Returns the name of the FOM object class to use*
- cleanup
- cleanupFromReflecting *Do the deinitialization required to stop handling reflection of object*
- cleanupFromUpdating *Do the deinitialization required to stop publishing this object*
- clearAll *zeros out all bytes in the network representation*
- clearData *zeros out all bytes after the PDU header*
- clear *removes all parts and parameters from list Non-virtual since called from ctor/dtor*
- clear *Remove all callbacks from the list*
- clear *Remove all callbacks from the list*
- clear
- clear *Remove all callbacks from the list*
- clear *deletes the string, and sets the myString pointer to NULL non-virtual because only called by destructor, and non-virtual members myString is set to NULL, so it should be allocated some memory if operators are to be used on the string*
- clear *Remove all callbacks from the list*
- clear *Remove all callbacks from the list*
- clearForcedAttributes *Clear (empty) the set of forced attributes*
- clearList
- clearRemoveList *Virtual version of above*
- clearRequestedAttributes *clears the set of requested attributes*
- clone *return an empty object of the same type as this*
- clone *Returns a new'ed clone of the instance*
- clone *Create a copy of this object*

- clone
- clone *return an empty object of the same type as this*
- clone *make a copy of this hash key*
- clone *Returns a clone of the instance*
- clone *Create a copy of this object*
- clone *return an empty object of the same type as this*
- clone *return an empty object of the same type as this*
- clone *return an empty object of the same type as this*
- clone *return an empty object of the same type as this*
- clone *return an empty object of the same type as this*
- clone *return an empty object of the same type as this*
- clone *Returns a new-ed clone of this object*
- clone *Returns a new-ed clone of this object*
- clone *make a copy of this hash key*
- clone *Returns a new-ed clone of this object*
- clone *Returns a new-ed clone of this object*
- clone *make a copy of this hash key*
- clone *return an empty object of the same type as this*
- clone *return an empty object of the same type as this*
- clone *Create a copy of this object*
- clone *return an empty object of the same type as this*
- clone *make a copy of this hash key*
- clone *make a copy of this hash key*
- codeName
- collideId
- collisionType
- combine *Constructs a new filename by combining the path with filename, which is an absolute or relative path*
- comment
- computeHashValue *compute the hashed value for this key*
- computeHashValue *compute the hashed value for this key*
- computeHashValue *compute the hashed value for this key*
- computeHashValue *compute the hashed value for this key*
- computeHashValue *compute the hashed value for this key*
- computeParameter *Get the parameter type from the part and parameter value*
- computePartType *Get the part type from the part and parameter value*
- confirmAttributeOwnershipAcquisitionCancellation
- connectionEstablished *Determines if the tcp socket has connected successfully*
- constDatumValPtr *Get a const pointer to the datum value inside of the record*

- constNetStruct *Cast the buffer to a const DtNetVarDatum struct*
- **coordTrans :**
  - in DtCoordTransform
  - in DtCoordTranslation
- copyCbList *Copy callback list*
- copyRemoveList *Copy another HLA object's remove callback list into this object's remove callback list*
- count *Returns the number of elements in the list*
- count *Get number of objects*
- count *returns the number of occurrences of the character*
- country
- createDecoder *Create a decoder for the specified interaction class*
- createDecoder *Called by decoderToUse to obtain a new'ed decoder*
- createDecoder *Create a decoder for the specified object class*
- create *Create a DtAcknowledgeInteraction*
- create *Create a DtActionRequestInteraction*
- create *Create a DtActionRequestToObjInter*
- create *Create a DtActionResponseFromObjInter*
- create *Create a DtActionResponseInteraction*
- create *Create a DtAggregateEntityDecoder*
- create *Create a DtAggregateEntityEncoder*
- create
- create *Create a DtAttributeChangeReqInter*
- create *Create a DtAttributeChangeResInter*
- create
- create
- create
- create
- create
- create
- create *Create a DtCollisionInteraction*
- create *Create a DtCommentInteraction*
- create *Create a DtCreateEntityInteraction*
- create *Create a DtCreateObjReqInter*
- create *Create a DtCreateObjectResultInter*
- create *Create a DtDataInteraction*
- create *Create a DtDataQueryInteraction*
- create
- create
- create

- create *Create a DtDetonationInteraction*
- create *returns a new DtDirectRtiAmbassador*
- create
- create
- create
- create
- create
- create *A Creator function that returns a DtEmptyFomMapper configured with no mappings at all*
- create
- create *Create a DtEnvironmentEntityDecoder*
- create *Create a DtEnvironmentEntityEncoder*
- create *Create a DtEventReportInteraction*
- create *Create a DtFireInteraction*
- create
- create
- create
- create *Create a DtLgrControlInteraction*
- create *Create a PDU representation from a network packet*
- create *Create a DtMilitaryEntityDecoder*
- create *Create a DtMilitaryEntityEncoder*
- create *Create a DtMilitaryPlatformEntityDecoder*
- create *Create a DtMilitaryPlatformEntityEncoder*
- create *Create a DtMunitionEntityDecoder*
- create *Create a DtMunitionEntityEncoder*
- create *Create a DtPhysicalEntityDecoder*
- create *Create a DtPhysicalEntityEncoder*
- create
- create
- create
- create
- create
- create
- create
- create
- create *Create a DtRemoveEntityInteraction*
- create *Create a DtRemoveObjectRequestInter*
- create *Create a DtRemoveObjectResultInter*
- create *Create a DtRepairCompleteInteraction*

- create *Create a DtRepairResponseInteraction*
- create *Create a DtResupplyCancelInteraction*
- create *Create a DtResupplyOfferInteraction*
- create *Create a DtResupplyRecInteraction*
- create *Creator function - returns new instance of DtRprFomMapper*
- create *Create a DtServiceRequestInteraction*
- create *Create a DtSetDataInteraction*
- create *Create a DtSignalInteraction*
- create *Create a DtSoldierDecoder*
- create *Create a DtSoldierEncoder*
- create *Create a DtStartResumeInteraction*
- create *Create a DtStopFreezeInteraction*
- create
- create
- create *Create a DtUnknownInteraction*
- create *Create a DtViewControlInteraction*
- create
- create *returns a new DtVrlFedAmbassador*
- createEncoder *Create an encoder for the specified interaction class*
- createEncoder *Called by encoderToUse to obtain a new'ed encoder*
- createEncoder *Create an encoder for the specified object class*
- createFedAmb *Create the federate ambassador*
- createFedEx *Create the federation execution*
- createFederationExecution *Calls down to the RTI function of the same name*
- createFederationExecution
- createHlaObject *Create the specified object*
- createInteraction *Create the specified interaction*
- **createObjectResult :**
  - in duplicate
  - in DtCreateObjectResultInter
- createRegion *Calls down to the RTI function of the same name*
- createRegion
- createRtiAmb *Create the RTI ambassador*
- cryptoKeyId
- cryptoMode
- cryptoSystem
- ctorCopyList *Copy another callback list into this object's callback list*
- ctorCopyList *Copy another callback list into this object's callback list*
- ctorCopyList *Copy another callback list into this inter's callback list*

- ctorCopyList *Copy another callback list into this object's callback list*
- ctorCopyList *Copy another callback list into this object's callback list*
- currentHeaderMatchesFirstInBundle
- currentTimeForStamping *Return time to send with outgoing messages (default is real time if absolute timestamping, sim time otherwise*
- dRAlgorithm
- damageState
- dataByte *Set/Get the byteNum'th data byte*
- data
- data
- dataLeftInCurrentBundle
- databaseIndex
- datumBitLength
- datumByteLength *Get the datum length in bytes (padded to an 8 byte boundary)*
- datumByteSize *Compute the total datum size in bytes from the datum bit length*
- datumID
- datumId *Returns the datum param found in the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
- datumId *Returns the datum param found in the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
- datumRecSet *Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records*
- datumRecSet *Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records*
- datumRecSet *Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records*
- datumRecSet *Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records*
- datumSizeBits *Returns the number of bits of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
- datumSizeBits *Returns the number of bits of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
- datumSizeBytes *Returns the number of bytes of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
- datumSizeBytes *Returns the number of bytes of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
- datumValByteArray
- datumValByteArray
- datumValFloat32 ******* INSPECTING DATUM VALUES BY INDEX ********* If you know the data type of the value contained in a particular record, use one of the following to return it*
- datumValFloat32 ******* INSPECTING DATUM VALUES BY PARAM ID ********* If you know the data type of the value contained in a particular record, use one of the following to return it*
- datumValFloat32
- datumValFloat64
- datumValFloat64
- datumValFloat64

- datumValInt16
- datumValInt32
- datumValInt32
- datumValInt32
- datumValPtr *Returns a pointer to the index'th fixed or variable record in the network representation, depending on whether fixedOrVar is DtFixed or DtVar*
- datumValPtr *Get a pointer to the datum value inside of the record*
- datumValU16
- datumValU32
- datumValU8
- datumValUnsigned32
- datumValUnsigned32
- deadReckonBodyToGeoc *If the last dead reckoned body to geocentri coordinate matrix is not valid for the specified time, recalculate it and same the validity time*
- deadReckonEuler *If the last dead reckoned orientation (euler angles) is not valid for the specified time, recalculate it and same the validity time*
- deadReckonLoc *If the last dead reckoned position is not valid for the specified time, recalculate it and same the validity time*
- deadReckonOrientation *Calculate the orientation matrix*
- deadReckonPosition *Calculate the dead reckon position information*
- deadReckonedBodyToGeoc *Get the dead-reckoned body to geocentric coordinate matrix*
- deadReckonedEuler *Get dead-reckoned orientation (euler angles) at the specified time*
- deadReckonedLocation
- deadReckonedVelocity
- decodeArtParts *called by the decoder for the ArticulatedParametersArray attribute*
- decodeAttParts *called by the decoder for the ArticulatedParametersArray attribute*
- decodeAttribute *Virtual function override*
- decodeAttribute *Decode the attribute at the specified index in the attribute handle pair set*
- decode
- decode *Decode the state message into the state repository*
- decode *Decode the interaction*
- decodeParameter *Process (decode) the specified parameter*
- **decoder :**
  - in duplicate
  - in DtHlaStateDecoder *Get the attribute decoder for the specified handle*
- decoder *Get the parameter decoder for the specified handle*
- decoder *Returns the decoder prototype associated with handle by a previous call to addDecoder*
- decoder *Return the decoder we are using*
- decoder *Returns a pointer to the Decoder being used to decode state messages that we send back into the asSeenByRemote SR*

- decoder *Returns a pointer to the Decoder being used to decode state messages into our SR*
- decoder *Returns the decoder prototype associated with handle by a previous call to addDecoder*
- decoderToUse *Returns a decoder to use for decoding the interaction*
- decodingBuffer *Returns a pointer to a large static buffer that can be used to copy data obtained from a phvps*
- decodingBuffer
- delRead
- deleteBytes *Deletes numBytes bytes starting at the indicated address*
- deleteIfNoExternal *Delete the internal buffer if there is no external buffer*
- deleteObjectInstance *Calls down to the RTI function of the same name*
- deleteObjectInstance
- deleteRegion *Calls down to the RTI function of the same name*
- deleteRegion
- density
- designatedObject
- designatorCode
- designatorId
- designatorPower
- designatorRep
- designatorSpotLocation
- designatorStateRep *Returns a pointer to the DtDesignatorRepository - through which you can set or inspect current state information*
- designatorWavelength
- destroyFedExecFlag
- destroyFederationExecution *Calls down to the RTI function of the same name*
- destroyFederationExecution
- detachHlaObject *Detach HLA object - bringing the DtObjectPublisher back to the state it was in before setHlaObject was called*
- detachHlaObject *Ends managing of current HLA object*
- detailedModulationType
- dfltAction *The default action is to dead-reckon*
- **dfltAction :**
    - in DtSmoothOrient
    - in DtSmoothPos
- dfltAction *The default action is to dead-reckon*
- dfltAggDimensionThreshold
- dfltEmitterBeamClassChooser *The default emitter beam class chooser function*
- dfltEntityClassChooser *The default entity class chooser function*
- dfltFractionalThreshold
- dfltHandleList *Asks the FOM Mapper for the DtList of RTI::ObjectClassHandles to use*

- dfltHandleList *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*
- dfltHandleList *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*
- dfltHandleList *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*
- dfltHandleList *Asks the FOM Mapper for the DtList of RTI::ObjectClassHandles to use*
- dfltHandleList *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*
- dfltHandleList *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*
- dfltHlaObjectFactory *Returns the default HlaObject factory*
- dfltInteractionDecoderFactory *Returns the default interaction decoder factory*
- dfltInteractionEncoderFactory *Returns the default state interaction factory*
- dfltInteractionFactory *Returns the default Interaction factory*
- dfltRotationThreshold
- dfltRotationThreshold
- dfltRotationThreshold
- dfltRotationTraceThreshold *Returns the "RotationTrace" threshold*
- dfltSignalClassChooser *The default signal class chooser function*
- **dfltSmoothPeriod :**
  - in DtSmoothOrient
  - in DtSmoothPos
- dfltSmoothPeriod
- dfltStateDecoderFactory *Returns the default state decoder factory*
- dfltStateEncoderFactory *Returns the default state encoder factory*
- dfltThreshMode
- dfltTranslationThreshold
- dfltTranslationThreshold
- dfltTranslationThreshold
- dimensions
- disableAsynchronousDelivery *Calls down to the RTI function of the same name*
- disableAsynchronousDelivery
- disableAttributeRelevanceAdvisorySwitch *Calls down to the RTI function of the same name*
- disableAttributeRelevanceAdvisorySwitch
- disableAttributeScopeAdvisorySwitch *Calls down to the RTI function of the same name*
- disableAttributeScopeAdvisorySwitch
- disableClassRelevanceAdvisorySwitch *Calls down to the RTI function of the same name*
- disableClassRelevanceAdvisorySwitch
- disableInteractionRelevanceAdvisorySwitch *Calls down to the RTI function of the same name*

- disableInteractionRelevanceAdvisorySwitch
- disableTest
- disableTimeConstrained *Calls down to the RTI function of the same name*
- disableTimeConstrained
- disableTimeRegulation *Calls down to the RTI function of the same name*
- disableTimeRegulation
- disconnectFromServer
- discoverObjectCb *Cast the void* to a DtReflectedObjectList* and calls processDiscoverObject()*
- discoverObjectInstance *//////////////////////////*
- discoverObjectInstance
- doTimeouts *Remove any remote objects which have not been updated during the timeout period*
- doTimeoutsPostDrainCb *calls doTimeouts*
- domain
- drainInput *Read and process messages from the network until either no more messages, or for timeout seconds, whichever happpens first*
- drainInput *Read and process messages from the network until either no more messages, or for timeout seconds, whichever happpens first*
- drainNotifySocket
- dropMulticastGroup
- dropMulticastGroup
- dropMulticastGroup
- dsr
- dsr *Synonym for designatorRep()*
- dtorClear *Remove all callbacks from the list*
- dtorClear *Remove all callbacks from the list*
- dtorClear *Remove all callbacks from the list*
- dtorClear *Remove all callbacks from the list*
- dtorClear *Remove all callbacks from the list*
- duration
- east
- electricFieldX *Get electric field X*
- electricFieldZ *Get electric field Z*
- elevationBeamwidth *Get the elevation beam width*
- elevationCenter
- elevationSweep
- emitrBeamRep
- emitterBeamAdditionCb *Callback functions registered with myBeamList on beam additions, updates, and removals*
- emitterBeamClassChooserPre08 *The default emitter beam class chooser function*
- emitterBeamRemovalCb

- emitterBeamUpdateCb
- emitterFunction
- emitterName
- emitterNumber
- emitterSysRep *Get the emitter system's state repository*
- emitterSystemRep *Returns a pointer to the DtEmitterSystemRepository - through which you can set or inspect current state information*
- emittingSystemId
- emptyCbList *Empty callback list*
- empty *Remove all keys and data from the list*
- emptyLists *Empty the object list and the timeout list*
- enableAsynchronousDelivery *Calls down to the RTI function of the same name*
- enableAsynchronousDelivery
- enableAttributeRelevanceAdvisorySwitch *Calls down to the RTI function of the same name*
- enableAttributeRelevanceAdvisorySwitch
- enableAttributeScopeAdvisorySwitch *Calls down to the RTI function of the same name*
- enableAttributeScopeAdvisorySwitch
- enableClassRelevanceAdvisorySwitch *Calls down to the RTI function of the same name*
- enableClassRelevanceAdvisorySwitch
- enableInteractionRelevanceAdvisorySwitch *Calls down to the RTI function of the same name*
- enableInteractionRelevanceAdvisorySwitch
- enableTimeConstrained *Calls down to the RTI function of the same name*
- enableTimeConstrained
- enableTimeRegulation *Calls down to the RTI function of the same name*
- enableTimeRegulation
- encode *Encode necessary values from the state repository*
- encode *Encode necessary values from the state repository*
- encode *Encode necessary values from the protocol-independent interaction*
- encode
- encodedLength
- encoder *Get the attribute encoder for the specified handle*
- encoder
- encoder *Returns the encoder prototype associated with handle by a previous call to addEncoder*
- encoder *Return the encoder we are using*
- encoder *Returns a pointer to the Encoder being used to encode current state into an outgoing message*
- encoder *Returns the encoder prototype associated with handle by a previous call to addEncoder*
- encoderToUse *Returns an encoder to use for encoding the interaction*
- encodingClass *Set/Get encoding class*
- encodingType

- engineSmokeOn
- entities *Return the list of published entities in the aggregate*
- entityAdded *Called by objectAdded (which is called from processDiscoverObject()) after a remote entity has been added to the list*
- entityClassChooserPre08 *The entity class chooser function for older RPR FOM versions*
- entityCoords
- entityId
- entityId
- entityId
- entityLocation
- entityNum
- entityObjectId *For backwards compatibility*
- entityStateRep *Get the entity's state repository*
- entityStateRep *Returns a pointer to the EntityStateRepository - through which you can set or inspect current state information*
- entityStateRep
- **entityToAdd :**
  - in DtGroupVCPdu
  - in DtModelessVCPdu
- **entityToRemove :**
  - in DtGroupVCPdu
  - in DtModelessVCPdu
- entityTypeAtStation *Returns the entityType at a particular station*
- entityType *Returns the entityType for a part, by index*
- entityType
- entityType
- equivalent *Returns true if we contain the exact same information as other*
- esr *Synonym for emitterSystemRep()*
- esr *Synonym for entityStateRep()*
- esr
- esr *Synonym for entityStateRep()*
- eulerTrans
- evalTest *No-op*
- evalTest
- eventId
- eventId
- eventId
- eventId
- eventId

- eventNum
- eventType
- everPublished *Return true if we have ever published the class using DtInterClassDesc::publish, false otherwise*
- everPublished *Return true if we have ever published the class using DtObjClassDesc::publish, false otherwise*
- everSubscribed *Return true if we have ever subscribed to the class using DtInterClassDesc::subscribe, false otherwise*
- everSubscribed *Return true if we have ever subscribed to the class using DtObjClassDesc::subscribe, false otherwise*
- executePostDrainCbs *should be called by derived class' drainInputs*
- exerciseConn
- exerciseConn
- exerciseConn *Get our exercise connection*
- exerciseConn
- exerciseId
- exerciseId
- exerciseId
- extendTest
- extendTest
- extra
- falseChecker *A checker function that always returns 0, meaning that the attribute for which it is being used will never be deemed necessary to send*
- fedAmbCreator
- fedAmb *Get the federate ambassador*
- fedAmb *Get my federate ambassador*
- federateId
- federateRestoreComplete *Calls down to the RTI function of the same name*
- federateRestoreComplete
- federateRestoreNotComplete *Calls down to the RTI function of the same name*
- federateRestoreNotComplete
- federateSaveBegun *Calls down to the RTI function of the same name*
- federateSaveBegun
- federateSaveComplete *Calls down to the RTI function of the same name*
- federateSaveComplete
- federateSaveNotComplete *Calls down to the RTI function of the same name*
- federateSaveNotComplete
- federationNotRestored
- federationNotSaved
- federationRestoreBegun
- federationRestored

federation

- federationSynchronized
- feedSmoothing *provide inputs to the actual smoothing algorithm*
- filename
- filename
- fillNetParams *fills in netParams based on current values*
- fillSpecList
- findAlreadyDiscoveredObjects *When list is created, we call this function to go through the list of all discovered HLA objects to see if any should be managed by this list*
- findDiscoveredObjectsCb *Post-drain callback that calls findAlreadyDiscovedObjects()*
- findItem
- findNextPdu
- findNextValidParamType *Find the next valid parameter type*
- findString *Returns 1 after reading a line containing str Returns 0 if hit EOF without finding such a line*
- fireMissionIndex
- firePowerDisabled
- first
- first *Get the first/last entity in the list*
- first *Get the first/last designator in the list*
- first *Get the first/last beam in the list*
- first *Get the first/last emitter system in the list*
- first *Get the first/last entity in the list*
- first *Get the first/last object in the list*
- first *Get the first/last receiver in the list*
- first *Get the first/last transmitter in the list*
- firstInterClass *Returns the first, last and next interaction class in the fom*
- firstObjClass *Returns the first, last and next object classes in the fom*
- firstObj
- firstValidParamType *Get the first valid (added) parameter type*
- fixedDatumIDs
- fixedDatumId
- fixedDatumId *Returns the datum param found in the index'th fixed datum record*
- fixedDatumId *Returns the datum param found in the index'th fixed datum record*
- flamesPresent
- flush
- flushQueueRequest *Calls down to the RTI function of the same name*
- flushQueueRequest
- fom *Get the FOM*
- fomMapperCreator
- fomMapper *Get the Fom Mapper*

- forceId
- forceId
- forceUpdate
- forceUpdate
- formation
- fractionalThreshold
- frequencyBandwidth
- frequency
- frequency *Get either the upper or lower word of the 64 bit frequency*
- frequencyHopInUse
- frequencyRange
- frozen
- frozen
- fuseType
- fuseType
- gGuiseSameAsType
- generateUpdate *Called by update to actually generate an outgoing message*
- generateUpdate *Called by update() to actually generate an outgoing message*
- genericStateRep *Returns a pointer to the DtGenericStateRepository - through which you can set current state information*
- genericStateRep *Returns a pointer to the DtGenericStateRepository, through which you can inspect current state information*
- geocToUtmMatrix *Get the matrix which transforms form geocentric to UTM*
- geocentric *Get geocentric coordinate equivalent (convert geodetic to geocentric)*
- geocentric *Get geocentric coordinate equivalent (convert UTM to geocentric)*
- geodetic *Get geodetic coordinate equivalent (convert UTM to geodetic)*
- getAngles
- getAppearanceAttribute
- getAppearance
- getArtParamVal
- getArtParamVal
- getAttributeHandle *Calls down to the RTI function of the same name*
- getAttributeHandle
- getAttributeName *Calls down to the RTI function of the same name*
- getAttributeName
- getAttributeRoutingSpaceHandle *Calls down to the RTI function of the same name*
- getAttributeRoutingSpaceHandle
- getDatumValByteArray
- getDimensionHandle *Calls down to the RTI function of the same name*

- getDimensionHandle
- getDimensionName *Calls down to the RTI function of the same name*
- getDimensionName
- getEntityID
- getEntityTypeAtStation
- getEntityType
- getFuncsFromSharedLib *Not virtual since called by ctor*
- getGeocentric *Get geocentric coordinate equivalent (convert geodetic to geocentric)*
- getGeocentric *Get geocentric coordinate equivalent (convert UTM to geocentric)*
- getGeodetic *Get geodetic coordinate equivalent (convert UTM to geodetic)*
- getIndex *Looks through all fixed and variable records for one containing param as the parameter ID*
- getIndex *Looks through all fixed and variable records for one containing param as the parameter ID*
- getInteractionClassHandle *Calls down to the RTI function of the same name*
- getInteractionClassHandle
- getInteractionClassName *Calls down to the RTI function of the same name*
- getInteractionClassName
- getInteractionRoutingSpaceHandle *Calls down to the RTI function of the same name*
- getInteractionRoutingSpaceHandle
- getMarkingText
- getObjectClass *Calls down to the RTI function of the same name*
- getObjectClass
- getObjectClassHandle *Calls down to the RTI function of the same name*
- getObjectClassHandle
- getObjectClassName *Calls down to the RTI function of the same name*
- getObjectClassName
- getObjectInstanceHandle *Calls down to the RTI function of the same name*
- getObjectInstanceHandle
- getObjectInstanceName *Calls down to the RTI function of the same name*
- getObjectInstanceName
- **getOffset :**
  - in DtCoordTransform
  - in DtCoordTranslation
- getOrderingHandle *Calls down to the RTI function of the same name*
- getOrderingHandle
- getOrderingName *Calls down to the RTI function of the same name*
- getOrderingName
- getOrientation
- getParameterHandle *Calls down to the RTI function of the same name*
- getParameterHandle

- getParameterName *Calls down to the RTI function of the same name*
- getParameterName
- getPrintableLength
- getPrintableString
- getPsMsg
- getRegion *Calls down to the RTI function of the same name*
- getRegion
- getRegionToken *Calls down to the RTI function of the same name*
- getRegionToken
- getRoutingSpaceHandle *Calls down to the RTI function of the same name*
- getRoutingSpaceHandle
- getRoutingSpaceName *Calls down to the RTI function of the same name*
- getRoutingSpaceName
- getSilentAggregateList
- getSilentEntityList
- getStationNumber
- getSubaggregateID
- getTime ---------- *Implementation functions* ----------
- getTranslations
- getTransportationHandle *Calls down to the RTI function of the same name*
- getTransportationHandle
- getTransportationName *Calls down to the RTI function of the same name*
- getTransportationName
- getVal
- getVariableDatumRecord
- getopt
- globalId *Set/Get object designator component*
- globalId *Get the HLA object's global object designator - the object name in RTI1*
- globalId *Get the HLA object's global object designator - the object name in RTI1*
- globalId
- gridDeclination *Get the heading that northing is off of north*
- groupCmd
- gsr *Synonym for genericStateRep()*
- gsr *Synonym for genericStateRep()*
- guessTimeValid *Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past that hour*
- guessTimeValid *Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past*

that hour

- guessTimeValid *Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past that hour*
- guise
- guiseSameAsType *Returns a constant that can be passed to the constructor indicating that we should use the entity type as our guise*
- handle *Get the object's class handle*
- handle
- hasAmmunitionSupplyCap
- hasFuelSupplyCap
- hasRecoveryCap
- hasRepairCap
- hashKey *get the key for the hashed item*
- hashValue *get the hashed value for this key*
- hatchState
- **heading :**
  - in DtGroupVCPdu
  - in DtOrbitVCPdu
- **headingOffset :**
  - in DtCompassVCPdu
  - in DtGroupVCPdu
  - in DtOrbitVCPdu
- heartbeat
- highDensity
- hlaObject *Get the HLA object based on its identifier*
- hlaObject
- hlaObject *Get the HLA object based on its identifier*
- hlaObject
- hlaObjectFactory
- host
- hostId
- i *Get a (1,0,0) vector*
- id *Get the aggregate's identifier*
- id *Get the entity's identifier*
- id *Get the object's local identifier*
- id *Get the HLA object's id (local ID)*
- identity
- ignoreFlags
- immobilized

- indexInBounds
- indexInBounds
- indexInBounds
- informAttributeOwnership
- initArtParts *Set/Get ArticulatedParts*
- initArtParts *initialize articulated parts*
- initDataSection *Initialize to reflect the fixed parameters and variable parameters and sizes indicated*
- initDatumIds *Initialize fixed and variable datum ids*
- init *The init functions don't set values for the parameters, they merely set up the part and parameter types*
- init *Stations is a zero-terminated array of stations*
- init
- init
- init *Initialize to reflect the fixed parameters and variable parameters and sizes indicated*
- init *Initialize the instance, once an exercise connection is available*
- **init :**
  - in DtEntityIdentifier
  - in DtEntityType
- init *initialize*
- init *Not virtual since called by ctor*
- init
- init *Initialize the instance, once an exercise connection is available*
- init
- init *Register interest in a whole list of RTI::ObjectClassHandles all at once*
- init *Virtual function override*
- **init :**
  - in DtSmoothOrient
  - in DtSmoothPos
- init *Initialize the ambassador*
- initFom *Not virtual since called by ctor*
- initHeader *Initializes PDU header*
- initInterChoosers *Setup the interaction class choosers*
- initInterClasses *Set myInterClasses to contain default mappings*
- initMsg
- initObjChoosers *Setup the object class choosers*
- initObjClasses *Set myObjClassesClasses to contain default mappings*
- initPdu *Must be called after default ctor before PDU is completely constructed*
- init_iodev
- init_read
- init_write

- initiateFederateRestore
- initiateFederateSave
- **input :**
  - in DtSmoothOrient
  - in DtSmoothPos
- inputSource
- insertBytes *Inserts numBytes zeroed-out bytes at the indicated address*
- interClassByHandle *Returns the class descriptor for the handle*
- interClassByName *Returns the class descriptor for the name*
- interClass *Get the inter class handle*
- interClassHandle
- interactionClassChooser *Return the FOM-class-chooser function, if any, that has been associated with the name of a C++ class derived from DtInteraction using setInteractionClassChooser*
- interactionClass
- interactionClassHandle
- interactionClassHandle
- interactionClassName *Get the interaction class name*
- interactionClassToUse *Returns the FOM interaction class name to use*
- interactionClasses
- interactionDecoderFactory
- interactionEncoderFactory
- interactionFactory *Get the interaction factory*
- interactionFactory
- internalGetPduKind
- internalGetPduKind *Returns the PDU kind associated with the derived PDU class*
- internalGetPduKind
- internalSetFromPhvps *Decode into this object from parameter value list*
- internalSetMarkingText
- ipAddr
- ipAddr
- isAttributeOwnedByFederate *Calls down to the RTI function of the same name*
- isAttributeOwnedByFederate
- isBundling
- isConcealed
- isInternal
- isMember *Returns DtTrue if the give BSR is in this ESR's list of BSRs*
- isOfClass *Returns 1 if the class is of class hand or a subclass of hand, 0 otherwise*
- isOfClass *Returns 1 if the class is of class hand or a subclass of hand, 0 otherwise*
- isOk

- isOk
- isOk
- isOk
- isOk
- isPositiveInfinity
- isRotational *Returns 1 if the dead reckoning algorithm is a rotational one, 0 otherwise*
- isUnbundling
- isZero
- issueId
- itemLookup *Lookup something in the list, return DtListItem*
- itemLookup *Returns the DtListItem envelope object containing the indicated data element*
- j *Get a (0,1,0) vector*
- jammingModeSequence
- joinFedEx *Join the federation execution*
- joinFederationExecution *Calls down to the RTI function of the same name*
- joinFederationExecution
- k *Get a (0,0,1) vector*
- keys *Returns a DtList of DtHashKey*'s representing the set of keys with which data is currently associated*
- kind *int entityKind() const {return(DtEntityKind);}*
- kind
- kindToFamily *Maps PDU kind to ProtocolFamily*
- last
- last
- last
- last
- last
- last
- last
- last
- last
- lastInterClass
- lastObjClass
- lastObj
- lastSetAcceleration
- lastSetLocation *Return the value last passed to setLocation, regardless of whether we have an approximator*
- lastSetOrientation
- lastSetRotationalVelocity
- lastSetVal

- lastSetVelocity
- lastSimTimeUpdateReceived
- lastSimTimeUpdated *Returns the value of simulation time at the time an update was last received for the entity*
- lastSourceInetAddr *int isUnbundling(); is not currently implemented*
- lastSourceInetAddr
- lastSourceInetAddr
- lat
- launcherFlashPresent
- launcherRaised
- lcPdu
- length *Get size of message*
- length *return the length of the string*
- lgrCommand *get message-type for proper casting to a subclass*
- lifeformState
- lightState
- list *Get a DtList containing all of the void* data that you have listed, regardless of keys*
- list *Returns a DtList of void*'s which are the data that you have added to the hashlist (independent of hash keys)*
- list *Get a DtList containing all of the void* data that you have listed, regardless of keys*
- listItem *get the list item*
- listenToMulticastGroup
- listenToMulticastGroup *must be kept around for GT*
- listenToMulticastGroup
- loadNetList *Load a list from network data representation*
- loadNetRecord
- loadNetRecord
- localDeleteObjectInstance *Calls down to the RTI function of the same name*
- localDeleteObjectInstance
- localId *Get the object's local identifier*
- localId
- localId
- localObjectManager *Get the localObjectManager*
- location
- location
- location
- location
- lon
- lookup *Lookup the data associated with key*
- lookup *Lookup something in the list, return data pointer*

- lookup *Lookup the data associated with key*
- lookup *Look up aggregate by name*
- lookup *Look up designator by name*
- lookup *Look up beam by name*
- lookup *Look up emitter system by name*
- lookup *Look up entity by name*
- lookup *Look up object by identifier*
- lookup *Look up receiver by identifier*
- lookup *Look up transmitter by name*
- lookupObj *Look up object by identifier*
- majorModulationType
- manage *add the object to the list of local objects*
- manageObjectClass *Register callbacks with exercise connection so that we get notified when an object of this class is discovered*
- markingCharSet
- markingCharacterSet
- markingText
- markingText
- mass
- **masterSmoothEnable :**
    - in DtSmoothOrient
    - in DtSmoothPos
- masterSmoothEnable
- match *Check whether one callback info is equal to another*
- match *== returns 1 if they are equal, 0 if they are not equal*
- match *Check whether one callback info is equal to another*
- match *Check whether one callback info is equal to another*
- match *Check whether one callback info is equal to another*
- match *Check whether one callback info is equal to another*
- match *Check whether one callback info is equal to another*
- match *Check whether one callback info is equal to another*
- match *Check whether one callback info is equal to another*
- match *Check whether one callback info is equal to another*
- **matchPattern :**
    - in DtEntityIdentifier
    - in DtEntityType
- maxAttributes
- maxClasses
- maxInteractionClasses *Returns maximum supported number of interaction classes*

- maxObjectClasses
- maxParameters
- modifyLookahead *Calls down to the RTI function of the same name*
- modifyLookahead
- modulationParamsAddr
- modulationParamsLength
- modulationSystemType
- munitionId
- munitionId
- munitionType
- munitionType
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name
- name
- name
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Get the object's indentifier*
- name *Get the object's class name*
- name *Name of the interaction (Not nec*
- name *Name of the interaction*
- name
- name *Get the object's name*
- name *Get the HLA object name*
- name
- name
- name *Name of the interaction (Not nec*

- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name
- name
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction (Not nec*
- name *Name of the interaction*
- name *Name of the interaction*
- neededByFederation *Returns DtTrue if the interaction class is needed by the federation false otherwise*
- neededByFederation *Returns DtTrue if the interaction class is needed by the federation DtFalse otherwise*
- negotiatedAttributeOwnershipDivestiture *Calls down to the RTI function of the same name*
- negotiatedAttributeOwnershipDivestiture
- netBeam *Returns network representation*
- netDataRep *Returns a pointer to the network representation, as might appear in a data PDU*
- netHeader *Returns a pointer to the PDU's header*
- netRecordLength *Get the network data length for this list*
- netRep *Returns network representation, fills size with size of netRep*
- netRep *Get network representation*
- netRep *Returns network representation, fills size with size of netRep*
- netRep *Convert to network representation*
- netRep *Returns network representation, fills size with size of netRep*
- netSize *Returns size of network representation*
- netSize *Returns size of network representation*
- netSize *Returns size of network representation*
- netStruct *Cast the buffer to a DtNetVarDatum struct*
- networkByteLength *Get the network byte length of this record*
- networkByteLength *Get the network byte length of the entire datum record*
- networkDataBuffer *Get a pointer to the buffer which stores the network representation of the datum record*
- networkRepresentation
- newInternalBuffer *Sets myInternalBuff to point to a malloc'ed buffer of size bytes*
- newReflectedAggregate *Create a remote aggregate object for the HLA object*
- newReflectedEmitterSystem *Create a remote emitter system object for the HLA object*
- newReflectedEntity *Create a remote entity object for the HLA object*
- newReflectedObject *Override of virtual function*

- newReflectedObject *Override of virtual function*
- newReflectedObject *Override of virtual function*
- newReflectedObject *Override of virtual function*
- newReflectedObject *Override of virtual function*
- newReflectedObject *Override of virtual function that the base class uses to create the appropriate kind of DtReflectedObject*
- newReflectedObject *Create a remote object object for the HLA object*
- newReflectedObject *Override of virtual function*
- newReflectedObject *Override of virtual function*
- **next :**
    - in DtListItem
    - in DtListedObj
- next *Get the next remote aggregate*
- next *Get the next remote designator*
- next *Get the next remote beam*
- next *Get the next remote emitterSystem*
- next *Get the next remote entity*
- next *Functions to get the next and previous objects in the list*
- next *Get the next remote receiver*
- next *Get the next remote transmitter*
- nextEventId *Get a unique event ID*
- nextEventRequestAvailable *Calls down to the RTI function of the same name*
- nextEventRequestAvailable
- nextEventRequest *Calls down to the RTI function of the same name*
- nextEventRequest
- nextInterClass
- nextObjClass
- nextObj
- nextValidParamType *Get the next valid parameter type*
- nomenclature
- nomenclatureVersion
- north
- notifyAboutRegionModification *Calls down to the RTI function of the same name*
- notifyAboutRegionModification
- notifyFromObserved *Called by DtObservee when something has happened to the observed object*
- notifyMe
- notifyMe
- nthFixedDataRecord *Return a pointer to the index'th fixed data record in the network representation*
- nthObjectId

- nthObjectId
- nthObjectId
- nthSupply
- nthSupply
- nthSupply
- nthVarDataRecord *Return a pointer to the index'th variable data record in the network representation*
- nullId
- nullId *Return a null DtEventId*
- **nullId :**
  - in DtFederateId
  - in DtObjectId
- nullString
- **nullType :**
  - in DtEntityType
  - in DtRadioEntityType
- numArtParams
- numArtParams *Get number of articulation parameters*
- numArtParts *Get the number of articulated parts*
- numAttachedParts
- numAttributes
- numAttributes
- numBeams *Return number of beam publishers*
- numBeams *Returns number of beams*
- numDataBits
- numDataBytes
- **numEntitiesToAdd :**
  - in DtGroupVCPdu
  - in DtModelessVCPdu
- **numEntitiesToRemove :**
  - in DtGroupVCPdu
  - in DtModelessVCPdu
- numFixedFields
- numFixedFields *returns number of fixed fields*
- numFixedFields
- numIds
- numIds
- numIds
- numObjects
- numPairs

- numPairs
- numPairs
- numParameters *Get the number of parameters associated with this object*
- numParameters
- numParams
- numSamples
- numSupplyTypes
- numSupplyTypes
- numSupplyTypes
- numVarFields
- numVarFields *returns number of variable length fields*
- numVarFields *returns number of variable length fields*
- numberOfAggregates
- numberOfAppearanceRecords
- numberOfDIS_Entities
- numberOfDIS_Subaggregates
- numberOfDatumRecords *Get the number of variable datum records*
- numberOfEntities
- numberOfSilentEntities
- numberOfSilentSubaggregates
- objClassByHandle *Returns the class descriptor for the handle*
- objClassByName *Returns the class descriptor for the name*
- objectAdded *Called from processDiscoverObject() after a remote transmitter has been added to the list*
- objectAdded *Called from processDiscoverObject() after a remote designator has been added to the list*
- objectAdded *Called from processDiscoverObject() after a remote beam has been added to the list*
- objectAdded *Called from processDiscoverObject() after a remote transmitter has been added to the list*
- objectAdded *Called from processDiscoverObject() after a remote transmitter has been added to the list*
- objectAdded *Called from processDiscoverObject() after a remote object has been added to the list*
- objectAdded *Called from processDiscoverObject() after a remote receiver has been added to the list*
- objectAdded *Called from processDiscoverObject() after a remote transmitter has been added to the list*
- objectClassChooser *Return the FOM-class-chooser function, if any, that has been associated with the name of a C++ class derived from DtObjectPublisher using setObjectClassChooser*
- objectClass
- objectClass
- objectClass
- objectClass
- objectClassHandle
- objectClassHandle *Get the object's class handle*
- objectClassHandle

- objectClassHandles *Similar to objectClassNames, but returns a DtList of ObjectClassHandles instead*
- objectClassNameFromType *Non-virtual since called from constructor*
- objectClassNameFromType *Non-virtual since called from constructor*
- objectClassNames
- object *Get index'th object designator*
- objectFactory *Get the object factory*
- objectHandle *Get the object handle*
- objectIDs *Return the Object ID list*
- objectIDs
- objectIDs
- objectId
- objectId *Set/Get object designator component*
- objectId *Get the object's indentifier*
- objectId *Get the object's local identifier*
- objectId
- objectId *Get the HLA object's id*
- objectId *Get the object identifier*
- objectProcessReflect *Call DtHlaObject::callReflect()*
- objectProcessRemoveObject *Call DtHlaObject::processRemoveObject()*
- **offset :**
  - in DtMimicTrackVCPdu
  - in DtMimicVCPdu
  - in DtTetherTrackVCPdu
- ones *Get a vector set to all ones*
- openFile *Open the file fedFileName*
- **operator = :**
  - in DtNet32Vector
  - in DtNet64Vector
  - in DtNetBurstDescriptor
  - in DtNetEntityIdentifier
  - in DtNetEntityType
  - in DtNetEulerAngles
  - in DtNetFederateId
- **operator = :**
  - in DtNetFloat32
  - in DtNetFloat64
  - in DtNetInt16
  - in DtNetInt32
- **operator = :**

    - in DtNetRadioEntityType
    - in DtNetSimulationAddress
- **operator = :**
    - in DtNetU16
    - in DtNetU32
- operator DtBoolean *to host cast operator*
- operator DtFloat32
- operator DtFloat64
- operator DtGlobalObjectDesignator
- operator DtInt16
- operator DtInt32
- operator DtNetBurstDescriptor
- operator DtNetEntityIdentifier
- operator DtNetEntityType
- operator DtNetFederateId
- operator DtNetRadioEntityType
- operator DtTaitBryan
- operator DtU16
- operator DtU32 *Cast to a DtU32 representing the whole timestamp Useful mainly for backwards compatibility for applications that used to cast DtNetTimeStamp to DtU32 before accessing it*
- operator DtU32
- operator DtVector*
- **operator DtVector :**
    - in DtNet32Vector
    - in DtNet64Vector
- operator const DtVector*
- operator const char* *cast operator*
- operator const double*
- operator double*
- operator!= *inequality operator*
- operator!=
- operator!= *equality operator*
- **operator!= :**
    - in DtEntityIdentifier
    - in DtEntityType *!= returns 0 if they are equal, nonzero if they are not equal*
- operator!=
- operator!=
- operator!=
- operator!=

- operator!=
- operator!=
- operator!=
- operator!=
- operator!=
- **operator* :**
  - in duplicate
  - in DtVrlFedTime
- operator*=
- **operator+ :**
  - in duplicate ---------- *Global operators* ----------
  - in DtVrlFedTime
- operator+= ---------- *Overloaded operators from RTI::FedTime* ----------
- operator+=
- operator+ *concatenation operators*
- **operator- :**
  - in duplicate
  - in DtVrlFedTime
- operator-=
- **operator/ :**
  - in duplicate
  - in DtVrlFedTime
- operator/=
- operator<
- operator<< ---------- *Implementation friends* ----------
- operator<<
- operator<=
- operator<=
- operator<
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator= *assignment operator*
- operator= *copy constructor*

- operator= *copy constructor*
- operator= *copy constructor*
- operator=
- operator= *assignment operator*
- operator=
- operator== *equality operator*
- operator== *---------- Implementation operators ----------*
- operator== *equality operator*
- operator== *Check whether one callback info is equal to another*
- **operator== :**
  - in DtEntityIdentifier
  - in DtEntityType
- operator== *equivalence operators*
- operator==
- operator== *equivlance operators*
- operator== *Check whether one callback info is equal to another*
- operator== *Check whether one callback info is equal to another*
- operator== *Check whether one callback info is equal to another*
- operator== *equality operators*
- operator== *Check whether one callback info is equal to another*
- operator== *Check whether one callback info is equal to another*
- operator== *Comparison operators*
- operator== *Comparison operators*
- operator== *Check whether one callback info is equal to another*
- operator==
- operator==
- operator== *Check whether one callback info is equal to another*
- operator== *Check whether one callback info is equal to another*
- operator== *comparison operators*
- operator== *Get whether another vector equals this vector*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=

- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator - not implemented*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *Assignment*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=

- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *Assignment operator*
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator - not implemented*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*

- operator= *assignment operator - not implemented*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator - not implemented*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *Assignment operator - Does not replace the DtHashlist's list*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator Copies across added decoder creation functions*
- operator= *assignment operator*
- operator= *assignment operator*

- operator= *assignment operator*
- operator= *assignment operator Care needs to be taken since this copies the callbacks*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *Assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *from host assignment operator*
- operator=
- operator= *assignment operator*

- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *Assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*

- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*

- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator Copies across added decoder creation functions*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*
- operator= *assignment operator*

- operator=
- operator=
- operator= *assignment operator*
- operator= *assignment operator - not implemented*
- operator= *assignment operator*
- operator>
- operator>=
- operator>=
- operator>>
- operator>
- operator[]
- operator[]
- operator[]
- **operator[] :**
  - in DtNet32Vector
  - in DtNet64Vector
- operator[]
- operator[]
- operator[] *Get the value for the specified subscript*
- orientation
- orientation
- orientation
- orientation
- orientation
- orientation
- otherFd
- ownAttrHandles
- ownParamHandles
- packet *Older function that returns a pointer to the network representation*
- paramHandleSet *Get the set of valid parameter handles for this class*
- paramNum *Returns the index of the parameter in a multiple part list as is found in HLA messages and DIS PDUs*
- params *return correctly casted parameters pointer; see lgrControlP*
- params *return correctly casted parameters pointer; see lgrControlP*
- params *return correctly casted parameters pointer; see lgrControlP*
- params *return correctly casted parameters pointer; see lgrControlP*
- params *return correctly casted parameters pointer; see lgrControlP*
- params *return correctly casted parameters pointer; see lgrControlP*
- params *return correctly casted parameters pointer; see lgrControlP*
- params *return correctly casted parameters pointer; see lgrControlP*

- params *return correctly casted parameters pointer; see lgrControlP*
- params *return correctly casted parameters pointer; see lgrControlP*
- partAttachedTo *Get the part this part is attached to*
- partNum *Get the part number*
- partNumFromPartType *returns 0 if there is no such part, otherwise the part number*
- partNumInBounds *Returns whether the part number is within bounds*
- partType
- pduKind
- pduKind
- phase
- phi
- phvps *Get the parameter handle value pair set for the interaction*
- phvps *Get the parameter handle value pair set for the interaction*
- phvps *Get the parameter handle value pair set for the interaction*
- pingClients
- **pitch :**
  - in DtCompassVCPdu
  - in DtGroupVCPdu
  - in DtOrbitVCPdu
- **pitchOffset :**
  - in DtCompassVCPdu
  - in DtGroupVCPdu
  - in DtOrbitVCPdu
- position
- power
- powerPlantOn
- **prev :**
  - in DtListItem *Return previous (next) item*
  - in DtListedObj
- prev *Get the previous remote aggregate*
- prev *Get the previous remote designator*
- prev *Get the previous remote beam*
- prev *Get the previous remote emitterSystem*
- prev *Get the previous remote entity*
- prev
- prev *Get the previous remote receiver*
- prev *Get the previous remote transmitter*
- prevObj
- primary

- primaryWeaponState
- printAttrs *Print the attributes (with hex values if withHex is 1)*
- printData
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the state data*
- printData *prints the data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the state data*
- printData *print the class data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *prints the data*
- printData
- printData *Print the interaction's data*
- printData
- printData
- printData
- printData *Print the state data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the state data*
- printData
- printData
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *print contents*
- printData *diagnostic output*

- printData *diagnostic output*
- **printData :**
    - in DtMimicTrackVCPdu
    - in DtMimicVCPdu
    - in DtModelessVCPdu
    - in DtOrbitVCPdu
- printData *Print the PDU's data (not including header)*
- printData
- printData
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print the interaction's data*
- printData *Print state data*
- printData *Print the interaction's data*
- **printData :**
    - in DtTetherTrackVCPdu
    - in DtTetherVCPdu
    - in DtTrackVCPdu
- printData *Just prints the parameters*
- printData *Print the interaction's data*
- printData
- print *Print the part data*
- print *Print the articulated part list*
- print *Print the information contained in the record*
- print *Prints the data in the matrix*
- print *prints the contents of the FOM*
- print *Print data*
- print *Prints the data in the hashlist*
- print *Print the interaction's host rep data, including header*
- print *Print the contents of the PDU by calling printHeader and printData*

- print *Print the SRs of all objects*
- print *Print the record*
- print *Print the record*
- print *Uses the object to print host representation of data*
- print *Print the contents of the record*
- printHeader *Print the header info*
- printHeader *Print the PDU's header*
- printHeader *prints the header information*
- printMsg *Prints contents of a state msg*
- printMsg *Prints contents of a state msg*
- printParams *Print the header followed by the interaction's parameters*
- printTrackJam *Prints track/jam target information*
- privPvList *Returns a pointer to a private pvList that can be used as a return value by phvps*
- processConnections
- processControlMsg
- processDiscoverObject *Process the discovery of a remote emitter system on the network*
- processDiscoverObject *Process the discovery of an object*
- processDiscoverObject *Process the discovery of a remote object on the network*
- processEmitterBeamAddition *Called by similar callback functions registered with myBeamList on beam additions/removals*
- processEmitterBeamRemoval
- processEmitterBeamUpdate
- processInteraction *Create an interaction object, call the registered callbacks, then delete the interaction object*
- processObjectAdditionCallbacks *Process object addition/removal callbacks*
- processObjectRemovalCallbacks
- processPostUpdateCallbacksCb *callback from HlaObject to process post update callbacks*
- processPostUpdateCallbacks *process the post update callbacks*
- processProvideUpdate *Set yourself up so that on the next tick, you send an update for the appropriate attributes*
- processProvideUpdate *process RTI's provideUpdate service*
- processReflectAttributeValues *Process an update message*
- processRemoveObject *Process notification of removal of the object from the network*
- processRemoveObject *Process the removal of a remote emitter system from the network*
- processRemoveObject *Process the removal of an object*
- processRemoveObject *Process the removal of a remote object from the network*
- processStartInters *Callback for handling startInters RTI callback*
- processStartReg *Callback for handling startRegistration RTI callback*
- processStopInters *Callback for handling stopInters RTI callback*
- processStopReg *Callback for handling stopRegistration RTI callback*
- processTimedOutObjectReflect *Handle the receipt of attributes for a remote object which was timed out*

- processUpdatesOff *Callback for handling turnUpdatesOffForObjectInstance RTI call*
- processUpdatesOn *Callback for handling turnUpdatesOnForObjectInstance RTI call*
- protocolFamily
- protocolId
- protocolVersion
- protocolVersion
- protocolVersion
- provideAttributeValueUpdate
- provideAttributeValueUpdate
- psi
- psuedoNoiseSpectrumInUse
- publish *Publish the class, and register start and stopUpdates callbacks*
- publish *Publishes the class, along with the attributes specified by hSet (or all attributes if no hSet is given)*
- publishInteractionClass *Calls down to the RTI function of the same name*
- publishInteractionClass
- publishObjectClass *Calls down to the RTI function of the same name*
- publishObjectClass
- publishedAttrs *Returns the set of published attributes*
- published *Returns DtTrue if the class is published, DtFalse otherwise*
- publisherForBsr *Returns the publisher responsible for publishing the indicated BSR, NULL if there is none*
- pulseWidth
- purgeBeamList
- quantity
- quantity
- queryAttributeOwnership *Calls down to the RTI function of the same name*
- queryAttributeOwnership
- queryFederateTime *Calls down to the RTI function of the same name*
- queryFederateTime
- queryLBTS *Calls down to the RTI function of the same name*
- queryLBTS
- queryLookahead *Calls down to the RTI function of the same name*
- queryLookahead
- queryMinNextEventTime *Calls down to the RTI function of the same name*
- queryMinNextEventTime
- radialOffset
- radioEntityType
- radioId
- radioId

- radioId
- radioXmitRep
- **radius :**
  - in DtCompassVCPdu
  - in DtOrbitVCPdu
- rampDeployed
- range
- rate
- rate
- readAndForward
- readFd
- readFd
- readFd *not all DtSocket have associated file descriptors those that do not will have readFd()/writeFd() return -1*
- readFd
- readFd
- readFed *Read the FED file fedFileName, filling in intMgr and objMgr*
- readInterClass *Read the description of a single interaction class, filling in interMgr with data read*
- readInteractions *Starting from the line after "(interactions", read through the ")" that denotes the end of the interactions definitions, filling in interMgr as we go*
- readLine *Read a line from the file*
- readObjClass *Read the description of a single object class, filling in objMgr with data read*
- readObjects *Starting from the line after "(objects", read through the ")" that denotes the end of the objects definitions, filling in objMgr as we go*
- realWorldTime
- realWorldTime
- reason *Set/Get stopReason*
- receiveInteraction
- receiveInteraction
- receivedPower
- receiverId
- receiverId
- receiverId
- receiverId
- receiverId
- receiverId
- receiverId
- receiverId
- receiverId

- receiverId
- receiverId
- receiverId
- receiverRep *Returns a pointer to the DtRadioReceiverRepository - through which you can set or inspect current state information*
- receiverRep
- receiverState
- receivingId
- receivingId
- receivingId
- receivingId
- **receivingId :**
  - in duplicate
  - in DtRepairCompleteInteraction
- receivingId
- receivingId
- receivingId
- receivingId
- recordByteSize *Compute the total record size in bytes from the datum bit length*
- recvFromNet
- recvWithStatus
- recvWithStatus
- recvWithStatus
- recvWithUnbundling
- reduceTest
- reduceTest
- referenceSystem
- reflectAttributeValuesCb *Callback for handling state messages for the HLA object*
- reflectAttributeValues
- reflectAttributeValues
- reflect *Handle reflection of object (receipt of state message) from the network*
- reflect *Receive representation (remote reflection) from the network*
- reflect *Does nothing*
- reflectWhenFrozen *SetGet reflectWhenFrozen flag*
- reflectedObjectManager
- reflecting
- reflecting
- reflecting
- registerFederationSynchronizationPoint *Calls down to the RTI function of the same name*

- registerFederationSynchronizationPoint
- registerObjectInstance *Calls down to the RTI function of the same name*
- registerObjectInstance
- registerObjectInstanceWithRegion *Calls down to the RTI function of the same name*
- registerObjectInstanceWithRegion
- relativeAntennaLoc
- relative
- relativePosition
- relativeSpotLocation
- remoteSitesList *Return the remote sites list*
- removeAggregateAdditionCallback
- removeAggregateRemovalCallback
- removeAll *Calls remove for all elements in the list*
- removeAllRemoteSites *Clear the list of remote sites*
- removeAndDelete *Default calls down to base class version, but can be overridden*
- removeAndDelete *Virtual function override*
- removeAndDelete *Remove the remote object from the list if found and delete it*
- removeBeam
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback

- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeCallback
- removeChecker
- removeDecoder
- removeDesignatorAdditionCallback
- removeDesignatorRemovalCallback
- removeDiscoverObjectCallback *Remove a discoverObject callback*
- removeDiscoverObjectCallback
- removeDiscoverObjectCb
- remove *Remove the current association involving key from the list*
- remove *Remove something from the list*
- remove *Remove an element from the list*
- remove *Remove the current association involving key from the list*
- remove
- remove
- remove
- remove
- remove
- removeEmitterBeamAdditionCallback
- removeEmitterBeamRemovalCallback
- removeEmitterSystemAdditionCallback
- removeEmitterSystemRemovalCallback
- removeEncoder
- removeEntityAdditionCallback
- removeEntityRemovalCallback
- **removeFromList :**
    - in DtIntrusiveList *Override of virtual function*

    - in DtList *Remove item from the list*
- removeGenericAdditionCallback
- removeGenericRemovalCallback
- removeInteractionCallbackByName *Remove an interaction callback by class name Note: The user data is significant for identifying the callback and needs to be the same as when the callback was added*
- removeInteractionCallback *Remove an interaction callback by class handle*
- removeInteractionCallback
- removeObjectAdditionCallback
- removeObjectCb *Cast the void* to a DtReflectedObjectList* and calls processRemoveObject()*
- removeObject *Have the application act as though we had never discovered the object*
- removeObjectInstance
- removeObjectInstance
- removeObjectRemovalCallback
- **removeObjectResult :**
    - in duplicate
    - in DtRemoveObjectResultInter
- removePostDrainCallback
- removePostReflectCb *Remove (deregister) function to be called after each reflectAttrValues*
- removePostUpdateCallback
- removePostUpdateCallback
- removePostUpdateCallback
- removePostUpdateCallback
- removePostUpdateCallback
- removePostUpdateCallback
- removePostUpdateCallback
- removePostUpdateCallback
- removeRadioReceiverAdditionCallback
- removeRadioReceiverRemovalCallback
- removeRadioTransmitterAdditionCallback
- removeRadioTransmitterRemovalCallback
- removeReceiveInteractionCb
- removeReflectAttributeValuesCb
- removeRemoteSite
- removeRemoveObjectCallback *Remove a removeObject callback*
- removeRemoveObjectCallback
- removeRemoveObjectCb *Remove (deregister) function to be called when object is removed*
- removeRemoveObjectCb
- removeStartInteractionsCb
- removeStartRegCb

- removeStopInteractionsCb
- removeStopRegCb
- removeUpdatesOffCb
- removeUpdatesOnCb
- repairResult
- repairType
- repairingId
- **repairingId :**
  - in duplicate
  - in DtRepairCompleteInteraction
- repairingId
- replaceSlashes *Replaces slashes*
- requestAttributeOwnershipAssumption ////////////////////////////
- requestAttributeOwnershipRelease
- requestClassAttributeValueUpdate *Calls down to the RTI function of the same name*
- requestClassAttributeValueUpdate
- requestClassAttributeValueUpdateWithRegion *Calls down to the RTI function of the same name*
- requestClassAttributeValueUpdateWithRegion
- requestClassUpdateFlag
- requestFederationRestore *Calls down to the RTI function of the same name*
- requestFederationRestore
- requestFederationRestoreFailed
- requestFederationRestoreSucceeded
- requestFederationSave *Calls down to the RTI function of the same name*
- requestFederationSave
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId
- requestId

- requestObjectAttributeValueUpdate *Calls down to the RTI function of the same name*
- requestObjectAttributeValueUpdate
- requestObjectUpdateFlag
- requestRetraction
- requestServerCStates
- requestServerState
- requestStatus
- requestUpdate *Request an update for the object for the set of attributes indicated by attrs (or all subscribed attributes if attrs is NULL)*
- requestUpdate *Request that the owner of this entity send an attribute update*
- requestUpdatePostDrainCb *calls requestUpdate*
- requestUpdates *Request updates of all objects of this class, include the attributes specified by hSet (or all subscribed attributes if no hSet is given)*
- requestedAttributes
- **requestingId :**
    - in duplicate
    - in DtServiceRequestInteraction
- resignFederationExecution *Calls down to the RTI function of the same name*
- resignFederationExecution
- resolveOrphans *Check if any orphanBeams have non-NULL emittingSystemId*
- resolveOrphansPostDrainCb *Callback that calls resolveOrphans*
- responseFlag
- result
- retract *Calls down to the RTI function of the same name*
- retract
- **roll :**
    - in DtCompassVCPdu
    - in DtOrbitVCPdu
- rotationThreshold
- rotationThreshold
- rotationTraceThreshold *Returns the "RotationTrace" threshold*
- rotationTraceThreshold *Returns the "RotationTrace" threshold*
- rotationalVelocity
- rotationalVelocity
- rotationalVelocity
- rotationalVelocity
- rotationalVelocity
- rotationalVelocity
- rsr

- rsr
- rtiAmbCreator
- rtiAmb *Get the real RTI ambassador*
- rtiAmb *Get the RTI ambassador*
- rtiAmb *Returns our RTI ambassador*
- rtiAmb *Get my RTI ambassador*
- rtiAmb *Get my RTI ambassador*
- rtiAmb *Get the real RTI ambassador, if the implementation of the derived object permits*
- runClockWhenFrozen *SetGet runClockWhenFrozen flag*
- sameEntity
- sameHost
- sampleRate
- secondary
- secondaryWeaponState
- select
- self *Cast off const-ness for trivial internal change*
- self *Cast off const-ness for trivial internal change*
- self
- self *Cast off const-ness for trivial internal change*
- self
- self *Get rid of constness on the "this" pointer in order to access non-const functions from const member functions (such as simple accessors)*
- self
- self
- self *Casts away const-ness*
- self
- self *Cast off const-ness for trivial internal change*
- send *Send an interaction to the network*
- send *Send a state message to the network*
- sendInteraction *Calls down to the RTI function of the same name*
- sendInteraction
- sendInteractionWithRegion *Calls down to the RTI function of the same name*
- sendInteractionWithRegion
- sendStamped *Send an interaction to the network which has been time stamped with the current time*
- sendStamped *Send an interaction to the network which has been time stamped with the current time*
- sendToAll
- sendToAllRemoteSites *Send to all remote sites except dwSkipAddr*
- sendTo
- sendTo

- sendTo
- sendTo
- sendTo
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- senderId
- separator *returns the separator used in representing directories*
- serviceType
- **servicingId :**
  - in duplicate
  - in DtServiceRequestInteraction
- setAcceleration *Set/Get absolute acceleration*
- setAcceleration *Set/Get the acceleration*
- setAcceleration *Set/Get the acceleration*
- setAcceleration *Set/Get absolute acceleration*
- setAcceleration *Set/Get absolute acceleration*
- setAcceleration *Set/Get the acceleration*
- setAcknowledgeFlag *Set/Get acknowledge flag*
- setAction *Set/Get action enum*
- setAction *mutator functions*
- setAction *set/get action to take place when logger switched to playback mode*
- setAction *set/get action to take when switching to record-mode*
- setActionId *Set/Get action ID*
- **setActionResult :**
  - in duplicate
  - in DtActionResponseFromObjInter *Set/Get action enum*
- setAfterburnerOn *Set/Get whether entity has after burner on or not*
- setAggDimensionThreshold *Set/Get aggregate dimension threshold*

- setAggregateID *Set/Get the aggregate's identifier*
- setAggregateState *Set/Get the aggregate state*
- setAggregateType *Set/Get the aggregate type*
- setAggregateType *Set/Get the aggregate type*
- setAlgorithm *Set/Get the dead reckoning algorithm*
- setAlgorithm *Set/Get the dead reckoning algorithm used for generating the approximated values from the absolute values*
- setAlt *Set/Get the altitude*
- setAngles *Set/Get the beam direction*
- setAntennaPattern *Set/Get the antenna pattern*
- setAntennaPatternType *Set/Get the antenna pattern type*
- setAppearanceAttribute *Set/Get an appearance attribute*
- setAppearanceAttribute *Set/Get appearance by attribute*
- setAppearance
- setAppearance *Set/Get the appearance of an entity (as a 32 bit integer)*
- setApplicationId
- setApproximating *Set/Get whether or not to do approximating*
- setApproximator *Get and set the approximator used to provide approximated position and orientation information for the entity*
- setArtParamVal *Set/Get the specified part type's specified parameter type value*
- setArtParamVal *Set/Get articulation parameter values by part type and parameter type*
- **setAttachId :**
    - in DtCompassVCPdu
    - in DtMimicTrackVCPdu
    - in DtMimicVCPdu
    - in DtOrbitVCPdu
    - in DtTetherTrackVCPdu
    - in DtTetherVCPdu
- setAttackerId *Set/Get the attacker identifier*
- setAttackerId *Set/Get the attacker identifier*
- setAttributeChangeResult *Set/Get the AttributeChangeResult type*
- setAttributeHandle *Set/Get the attribute handle given an index*
- setAttributeHandle *Set/Get the attribute handle given an index*
- setAttributeHandle *Set/Get the attribute handle given an index*
- setAttributeValue
- setAttributeValue
- setAttributeValue
- setAttributeValueLength *Set/Get the attribute value length given and index*
- setAttributeValueLength *Set/Get the attribute value length given and index*

- setAttributeValueLength *Set/Get the attribute value length given and index*
- setAvPairSet
- setAvPairSet
- setAvPairSet
- setAzimuthCenter *Set/Get Azimuth Center*
- setAzimuthRate
- setAzimuthSweep *Set/Get Azimuth Sweep*
- setBeamFunction *Set/Get Beam Function*
- setBeamId *Set/Get Beam Id*
- setBeamParameterIndex *Set/Get Beam Parameter Index*
- setBeamType *Set/Get beam type - track, jam or neither*
- setBeamwidth *Set/Get the azimuth bandwidth and elevation*
- setBodyHeading
- setBodyOrientation
- setBsr *Set beam state repository*
- setBundling
- **setByInverse :**
  - in DtCoordTransform
  - in DtCoordTranslation
- setCallbackFunction *set/get the callback function*
- setCamouflageType *Set/Get camouflage type*
- setCapabilities *Set/Get the capabilities field*
- setCategory
- setClassDesc *Set the class descriptor of the encoder*
- setClassDesc
- setClassDesc
- setClassDesc
- setCodeName *Set/Get the code name*
- setCollideId *Set/Get the collide identifier*
- setCollisionType *Set/Get the collision type*
- setComment *Set/Get the comment; warning: inspector returns a readable text string*
- setCountry
- setCreateObjectResult
- setCryptoKeyId *Set/Get the crypto key id*
- setCryptoMode *Set/Get the crypto key id*
- setCryptoSystem *Set/Get the crypto system*
- setDRAlgorithm *Set/Get the dead reckoning algorithm*
- setDamageState *Set/Get the damage state*
- setDataByte

- setData *Set/Get signal data*
- setDatabaseIndex *Set/Get database index*
- setDatumBitLength *Set/Get the datum length in bits*
- setDatumID *Set/Get the variable datum ID*
- setDatumParam *Set the datum ID in the index'th fixed or variable datum record to param, depending on whether fixedOrVar is DtFixed or DtVar*
- setDatumParam *Set the datum ID in the index'th fixed or variable datum record to param, depending on whether fixedOrVar is DtFixed or DtVar*
- setDatumValByteArray
- setDatumValByteArray
- setDatumValByteArray *Set/Get datum value as a byte array*
- setDatumValFloat32
- setDatumValFloat32
- setDatumValFloat32 *Set/Get datum value as a DtFloat32 Set will set the datum bit length to 32*
- setDatumValFloat64 ******* SETTING DATA VALUES BY PARAM ID ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data*
- setDatumValFloat64 ******** SETTING DATA VALUES BY INDEX ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data*
- setDatumValFloat64 *Set/Get datum value as a DtFloat64 Set will set the datum bit length to 64*
- setDatumValInt16 *Set/Get datum value as a DtInt16 Set will set the datum bit length to 16*
- setDatumValInt32
- setDatumValInt32
- setDatumValInt32 *Set/Get datum value as a DtInt32 Set will set the datum bit length to 32*
- setDatumValU16 *Set/Get datum value as a DtU16 Set will set the datum bit length to 16*
- setDatumValU32 *Set/Get datum value as a DtU32 Set will set the datum bit length to 32*
- setDatumValU8 *Set/Get datum value as a DtU8 Set will set the datum bit length to 8*
- setDatumValUnsigned32
- setDatumValUnsigned32
- setDensity *Set/Get light state of entity*
- setDesignatedObject *Set/Get the designated object name*
- setDesignatorCode *Set/Get the designator code*
- setDesignatorId *Set/Get the designator object id*
- setDesignatorPower *Set/Get the designator power*
- setDesignatorSpotLocation *Set/Get the designator spot location*
- setDesignatorWavelength *Set/Get the designator wavelength*
- setDestroyFedExecFlag *Set/Get flag that determines whether or not the DtExerciseConn destructor will attempt to destroy the federation execution*
- setDetailedModulationType *Set/Get the detailed modulation*
- setDfltAggDimensionThreshold
- setDfltFractionalThreshold

- setDfltRotationThreshold *Set/Get rotational threshold value*
- setDfltRotationThreshold *These four functions are obsolete, but remain for backwards compatibility*
- setDfltRotationThreshold *Set/Get default rotation threshold in radians*
- **setDfltSmoothPeriod :**
    - in DtSmoothOrient *Set/Get default smooth period*
    - in DtSmoothPos *Set/Get default smooth period*
- setDfltSmoothPeriod *Set/Get the default smoothing period*
- setDfltThreshMode *Set/Get the theshold mode*
- setDfltTranslationThreshold *Set/Get translational threshold value*
- setDfltTranslationThreshold
- setDfltTranslationThreshold *Set/Get default translation threshold in radians*
- setDimensions *Set/Get the dimensions*
- setDomain
- setDuration *Set/Get duration*
- setERP *Set/Get ERP*
- setEast *Set/Get east component*
- setElectricField *Set electric field*
- setElevationCenter *Set/Get Elevation Center*
- setElevationSweep *Set/Get Elevation Sweep*
- setEmitterFunction *Set/Get the emitter function*
- setEmitterName *Set/Get the emitter name*
- setEmitterNumber *Set/Get the emitter number*
- setEmittingSystemId *Set/Get Emitting System Id*
- setEncodingClass
- setEncodingType *Set/Get encoding type*
- setEngineSmokeOn *Set/Get whether engine smoke is on or not*
- setEntityCoords *Set/Get the entity coordinates*
- setEntityID *Set/Get a DIS entity ID*
- setEntityId
- setEntityId *Set/Get the entity identifier*
- setEntityId *Set/Get Entity Id - not used after RPR 0*
- setEntityLocation *Set/Get the location of the detonation in entity coordinates*
- setEntityNum
- **setEntityToAdd :**
    - in DtGroupVCPdu
    - in DtModelessVCPdu
- **setEntityToRemove :**
    - in DtGroupVCPdu
    - in DtModelessVCPdu

- setEntityTypeAtStation *Set/Get the entity type for a part*
- setEntityType *Set/Get the entity type for a part*
- setEntityType *Set/Get the entity type*
- setEntityType *Set/Get the entity type*
- setEpsilon
- setEsrCreator *For backwards compatibility*
- setEventId *Set/Get the collision event id*
- setEventId *Set/Get the detonation event id*
- setEventId *Set/Get Event Id - not used until RPR 0*
- setEventId *Set/Get Event Id - not used until RPR 0*
- setEventId *Set/Get the event id*
- setEventNum *Set/Get event number component*
- setEventType *Set/Get event Type*
- setExConn *Sets the interaction's ExerciseConn and class descriptor*
- setExConn *Sets the interaction's ExerciseConn and class descriptor*
- setExConn *Sets the interaction's ExerciseConn and class descriptor*
- setExerciseId
- setExternalBuffer *Set the working buffer to an externally supplied buffer*
- setExtra
- setFedAmbCreator *Set/get the function to be used to create a federate ambassador*
- setFederateId *Set/Get the federate ID (part of entity ID)*
- setFileOutput
- setFilename *set/get file to open for playback*
- setFilename *set/get the name of file into which traffic will be recorded*
- setFireMissionIndex *Set/Get the fire mission index*
- setFirePowerDisabled *Set/Get whether entity is fire power disabled or not*
- setFixedDatumIDs *Set/Get the fixed fields memory block all at once*
- setFixedDatumId
- setFlamesPresent *Set/Get whether flames are present or not*
- setFomMapperCreator *Set/Get the default FomMapper creator - used to create a fomMapper when a NULL FOM Mapper or invalid FOM Mapper DSO name is passed to the constructor*
- setForceId *Set/Get the force type*
- setForceId
- setFormation *Set/Get the formation*
- setFractionalThreshold *Set/Get fractional threshold (fraction between 0 and 1)*
- setFrequencyBandwidth *Set/Get the frequency bandwidth*
- setFrequency *Set/Get Frequency*
- setFrequency
- setFrequencyHopInUse *Set/Get the frequency hop in use*

- setFrequencyRange *Set/Get Frequency Range*
- setFromNetBeam *Set from network representation*
- setFromNet *Set from network representation*
- setFromNet *initializes the part types and parameters, and sets values as well*
- setFromNet
- **setFromNet :**
    - in DtEntityIdentifier
    - in DtEntityType
- setFromNet *Set from network representation/*
- setFromNet *Set from network representation*
- setFromNet
- setFromNetFixed *Sets the fixed data from a network representation of a fixed datum record array*
- setFromNetPacket
- setFromNetPacket
- setFromNetVar *Sets the variable data from a network representation consisting of the variable length records concatentated together*
- setFromOther *Sets the contents of this hashlist to match those of other*
- setFromPhvps *Set contents from a parameter handle value pair set*
- setFromPhvps *Set contents from a parameter handle value pair set*
- setFromPhvps *Decode into this object from parameter value list*
- **setFromString :**
    - in DtAbsoluteVCPdu
    - in DtCompassVCPdu
    - in DtGroupVCPdu
    - in DtMimicTrackVCPdu
    - in DtMimicVCPdu
    - in DtOrbitVCPdu
    - in DtTetherTrackVCPdu
    - in DtTetherVCPdu
    - in DtTrackVCPdu
    - in DtViewControlPdu
- setFromVec
- setFrozen *Set/Get whether frozen or not*
- setFrozen *Set/Get whether frozen or not*
- setFuseType *Set/Get the detonator fuze type*
- setFuseType *Set/Get the detonator fuze type*
- setGeocentric *Set from geocentric coordinate (convert geocentric to geodetic)*
- setGeocentric *Set from geocentric coordinate (convert geocentric to UTM)*
- setGeodetic *Set from geodetic coordinate (convert geodetic to UTM)*

- setGlobalId
- setGlobalId *Get/Set global identifier, or name*
- setGroupCmd
- setGuise
- setHasAmmunitionSupplyCap *Set/Get whether entity has ammunition supply capability or not*
- setHasFuelSupplyCap *Set/Get whether entity has fuel supplying capability*
- setHasRecoveryCap *Set/Get whether entity has recovery capability*
- setHasRepairCap *Set/Get whether entity has repair capability*
- setHatchState *Set/Get the entity's hatch state*
- **setHeading :**
  - in DtGroupVCPdu
  - in DtOrbitVCPdu
- **setHeadingOffset :**
  - in DtCompassVCPdu
  - in DtGroupVCPdu
  - in DtOrbitVCPdu
- setHeartbeat *set/get the heartbeat*
- setHighDensity *Set/Get the high density track/jam flag*
- setHlaObject *Same as above, but you pass object class name rather than handle*
- setHlaObject *Start reflecting this object, using this decoder*
- setHlaObjectFactory *Set/Get the current factory that can be used to create DtHlaObjects*
- setHost
- setHostId *Set/Get the host identifier*
- setIgnoreFlags
- setImmobilized *Set/Get whether the entity is immobilized or not*
- setInputSource *Set/Get the input source*
- setInteractionClassChooser
- setInteractionClass *When there is only a single FOM class to be associated with a VR-Link interaction class, then there is no need to create a DtList*
- setInteractionClassToChoose *Tell the FOM Mapper how to choose a FOM interaction class to use to for a particular kind of derived DtInteraction*
- setInteractionClasses *Set/Get a list of names (char*) of FOM classes that can be decoded into the derived DtInteraction class whose name is name*
- setInteractionDecoderFactory *Set/Get the current factory that can be used to create DtInteractionDecoders*
- setInteractionEncoderFactory *Set/Get the current factory that can be used to create DtInteractionEncoders*
- setInteractionFactory *Set/Get the current factory that can be used to create DtInteractions*
- setIsConcealed *Set/Get whether entity is concealed or not*
- setIssueId *Set/Get the issuer identifier*
- setJammingModeSequence *Set/Get Jamming Mode Sequence*
- setKind *the funcs used to take longs, so for back-compat:*

- setLastSimTimeUpdateReceived *Set the time an update was last received or sent*
- setLat
- setLauncherFlashPresent *Set/Get whether launcher flash is present or not*
- setLauncherRaised *Set/Get whether entity has launcher raised or not*
- setLcPdu
- setLgrCommand *set is done by way of constructing the proper subclass*
- setLifeformState *Set/Get entity life form state*
- setLightState *Set/Get light state of entity*
- setLocalId *Get/Set local identifier, or handle*
- setLocation *Set absolute location*
- setLocation *Set/Get the location of the center of mass in world coordinates*
- setLocation *Set/Get the location in local coordinates*
- setLocation *Set absolute location*
- setLocation *Set/Get the location in world coordinates*
- setLocation *Set absolute location*
- setLocation *Set/Get the location in UTM coordinates*
- setLon
- setMajorModulationType *Set/Get the major modulation type*
- setMarkingCharSet *Set/Get the character set used for the marking*
- setMarkingCharacterSet *Set/Get marking character set*
- setMarkingText *Set/Get marking text (copies 31 bytes)*
- setMarkingText *Set/Get the marking text*
- setMass *Set/Get the mass involved*
- **setMasterSmoothEnable :**
  - in DtSmoothOrient *Set/Get the value of the "master" smoothing-enabled switch*
  - in DtSmoothPos *Set/Get the value of the "master" smoothing-enabled switch*
- setMasterSmoothEnable *Set/Get the value of the "master" smoothing-enabled switch*
- setMaxAttributes
- setMaxClasses *Set/Get max number of object or interaction classes*
- setMaxParameters
- setMcastTtl
- setMcastTtl
- setMcastTtl
- setModulationParams *Set/Get the modulation parameters' network representation*
- setModulationSystemType *Set/Get the modulation system*
- setMunitionId *Set/Get the munition identifier*
- setMunitionId *Set/Get the munition identifier*
- setMunitionType *Set/Get the munition type*
- setMunitionType *Set/Get the munition type*

- setNoHlaObject *Used to set up a proxy DtReflectedEntity - one that is not actually managing a real reflected HLA object*
- setNomenclature
- setNomenclatureVersion
- setNorth
- setNthObjectId *Set/Get the nth ObjectId*
- setNthObjectId *Set/Get the nth ObjectId*
- setNthObjectId *Set/Get the nth ObjectId*
- setNthSupply
- setNthSupply
- setNthSupply
- setNumAttachedParts *Set/Get the number of attached parts*
- setNumDataBits
- setNumDataBytes *Set/Get signal data length in bytes or bits*
- **setNumEntitiesToAdd :**
  - in DtGroupVCPdu
  - in DtModelessVCPdu
- **setNumEntitiesToRemove :**
  - in DtGroupVCPdu
  - in DtModelessVCPdu
- setNumFixedFields *Set/Get fixed fields*
- setNumFixedFields *Sets the number of fixed or variable fields to newNum*
- setNumFixedFields *Sets the number of fixed or variable fields to newNum*
- setNumIds *Set/Get number of IDs*
- setNumIds *Set/Get number of IDs*
- setNumIds *Set/Get number of IDs*
- setNumObjects *Set/Get number of objects*
- setNumPairs *Set/Get the number of attribute/value pairs*
- setNumPairs *Set/Get the number of attribute/value pairs*
- setNumPairs *Set/Get the number of attribute/value pairs*
- setNumSamples *Set/Get sample count*
- setNumSupplyTypes *Supporting functions to get/set the Supply Array*
- setNumSupplyTypes *Supporting functions to get/set the Supply Array*
- setNumSupplyTypes *Supporting functions to get/set the Supply Array*
- setNumVarFields *Set/Get variable fields*
- setNumVarFields
- setNumVarFields
- setNumberOfAggregates *Set/Get the number of aggregates*
- setNumberOfAppearanceRecords *Set/Get the number of appearance records*

- setNumberOfDIS_Entities *Set/Get the number of DIS entities*
- setNumberOfDIS_Subaggregates *Set/Get the number of DIS sub-aggregates*
- setNumberOfDatumRecords *Set the number of variable datum records*
- setNumberOfEntities *Set/Get the number of aggregates*
- setNumberOfSilentEntities *Set/Get the number of silent entities*
- setNumberOfSilentSubaggregates *Set/Get the number of silent aggregates*
- setObjectClassChooser
- setObjectClass
- setObjectClass *When there is only a single FOM class to be associated with a VR-Link DtReflectedObjectList class, then there is no need to create a DtList*
- setObjectClassNames *Set/Get a list of names (char*) of FOM object classes that should be subscribed to by a particular subclass of VR-Link's DtReflectedObjectList*
- setObjectClassToChoose *Tell the FOM Mapper how to choose a FOM object class to use to for objects being published by a particular kind of object publisher*
- setObject *Set index'th object designator*
- setObjectId *Set/Get the objectID*
- setObjectId
- setObjectIdForEventId *Set objectID to use in the next event ID*
- setOffset *set the origin of the local system in geocentric coords*
- **setOffset :**
  - in DtCoordTransform
  - in DtCoordTranslation
- **setOffset :**
  - in DtMimicTrackVCPdu
  - in DtMimicVCPdu
  - in DtTetherTrackVCPdu
- setOperator
- setOperator
- setOrientation
- setOrientation *Set absolute orientation*
- setOrientation *Set/Get the orientation*
- setOrientation *Set/Get the orientation*
- setOrientation *Set absolute orientation*
- setOrientation *Set absolute orientation*
- setOrientation
- setOrientation *Set/Get the orientation*
- setPRF *Set/Get PRF*
- setPacketServerExternal
- setPacketServerInternal
- setPacketServerUnbundling

- setParamNum *Set the parameter number of the specified parameter type*
- setPartNum *Set the part number*
- setPhase *Set/Get the phase*
- setPhi
- **setPitch :**
  - in DtCompassVCPdu
  - in DtGroupVCPdu
  - in DtOrbitVCPdu
- **setPitchOffset :**
  - in DtCompassVCPdu
  - in DtGroupVCPdu
  - in DtOrbitVCPdu
- setPositiveInfinity
- setPower *Set/Get the power*
- setPowerPlantOn *Set/Get whether entity's power plant is on or not*
- setPrimary
- setPrimaryWeaponState *Set/Get primary weapon deployment state*
- setProtocolId *Set/Get user protocol id*
- setPsMsgTimeout
- setPsi
- setPsuedoNoiseSpectrumInUse *Set/Get the Psuedo Noise Spectrum In Use*
- setPulseWidth *Set/Get Pulse Width*
- setQuantity *Set/Get the quantity*
- setQuantity *Set/Get the quantity*
- setRadialOffset
- setRadioEntityType *Set/Get the radio entity type*
- setRadioId *Set/Get the Receiver radio number*
- setRadioId *Set/Get the Transmitter radio number*
- setRadioId *Set/Get Host radio id*
- **setRadius :**
  - in DtCompassVCPdu
  - in DtOrbitVCPdu
- setRampDeployed *Set/Get whether entity has ramp deployed or not*
- setRange *Set/Get the range*
- setRate *Set/Get the rate*
- setRate *Set/Get the rate*
- setRealWorldTime *Set/Get real world time*
- setRealWorldTime *Set/Get real world time*
- setReason

- setReceivedPower *Set/Get the received power*
- setReceiverId *Set/Get receiver ID*
- setReceiverId *Set/Get receiver ID*
- setReceiverId *Set/Get receiver ID*
- setReceiverId *Set/Get receiver ID*
- setReceiverId
- setReceiverId *Set/Get receiver ID*
- setReceiverId *Set/Get receiver ID*
- setReceiverId *Set/Get receiver ID*
- setReceiverId *set/get Object Identifier of intended recipient (logger)*
- setReceiverId *Set/Get receiver ID*
- setReceiverId *Set/Get receiver ID*
- setReceiverId *Set/Get receiver ID*
- setReceiverId *Set/Get receiver ID*
- setReceiverState *Set/Get the receiver state*
- setReceivingId *Set/Get the receiving object*
- setReceivingId *Set/Get the receiving object*
- setReceivingId *Set/Get the receiving object*
- setReceivingId
- setReceivingId *Set/Get the receiving object*
- setReferenceSystem *Set/Get the beam reference system*
- setReflectWhenFrozen
- setReflecting *Set/Get whether to reflect back local objects and interactions, if supported by the derived class*
- setReflecting *Set/Get a flag that determines whether local objects and interactions will be reflected back to us*
- setReflecting *Set/Get whether to reflect back local objects and interactions, if supported by the derived class*
- setRelativeAntennaLoc *Set/Get the relative location (wrapper around EmbedSysSR member func, for compatibility with DIS transmitter PDU*
- setRelativePosition *Set/Get the relative position*
- setRelativeSpotLocation *Set/Get the relative spot location*
- setRemoveObjectResult
- setRepairResult
- setRepairType
- setRepairingId
- setRepairingId
- setRequestClassUpdateFlag *Set/Get flag that determines whether we request updates from all objects of the object classes we subscribe to*
- setRequestId *Set/Get request ID*
- setRequestId *Set/Get request ID*
- setRequestId *Set/Get request ID*

- setRequestId *Set/Get request ID*
- setRequestId
- setRequestId
- setRequestId *Set/Get request ID*
- setRequestId *Set/Get request ID*
- setRequestId
- setRequestId *Set/Get request ID*
- setRequestId *Set/Get request ID*
- setRequestId *Set/Get request ID*
- setRequestId *Set/Get request ID*
- setRequestId *Set/Get request ID*
- setRequestObjectUpdateFlag *Set/Get flag that determines whether we request updates from an object as we discover it*
- setRequestStatus *Set/Get the request status*
- setRequestingId *Set/Get the receiving object*
- setResponseFlag *Set/Get response flag*
- setResult *Set/Get the detonation result*
- **setRoll :**
    - in DtCompassVCPdu
    - in DtOrbitVCPdu
- setRotation *set the rotation matrix that converts from geocentric to local coords*
- setRotation
- setRotationThreshold *Set/Get rotation threshold in radians*
- setRotationThreshold *Set/Get rotation threshold in radians*
- setRotationalVelocity *Set/Get absolute rotational velocity*
- setRotationalVelocity *Set/Get the rotational velocity*
- setRotationalVelocity *Set/Get the rotational velocity*
- setRotationalVelocity *Set/Get absolute rotational velocity*
- setRotationalVelocity *Set/Get absolute rotational velocity*
- setRotationalVelocity *Set/Get the rotational velocity*
- setRtiAmbCreator *Set/get function to be used to create an RTI ambassador*
- setRunClockWhenFrozen
- setSampleRate *Set/Get data rate*
- setSecondary
- setSecondaryWeaponState *Set/Get secondary weapon deployment state*
- setSenderId *Set/Get sender ID*
- setSenderId `*Set/Get sender ID*
- setSenderId `*Set/Get sender ID*
- setSenderId `*Set/Get sender ID*

- setSenderId
- setSenderId *`Set/Get sender ID*
- setSenderId *`Set/Get sender ID*
- setSenderId *Set/Get sender ID*
- setSenderId *set/get Object Identifier of sender, if applicable*
- setSenderId *Set/Get sender ID*
- setSenderId *Set/Get sender ID*
- setSenderId *Set/Get sender ID*
- setSenderId *Set/Get sender ID*
- setSenderId
- setServiceType
- setServicingId *Set/Get the repairing object*
- setSilentAggregateList *Set/Get a silent aggregate list*
- setSilentEntityList *Set/Get a silent entity list*
- setSimulationTime *Set/Get simulation time*
- setSite
- setSiteId
- setSizeLength
- setSizeOffset
- setSmokePlumePresent *Set/Get whether a smoke plume is present or not*
- **setSmoothEnable :**
  - in DtSmoothOrient *Set/Get the individual smoothing switch*
  - in DtSmoothPos *Set/Get the individual smoothing switch*
- setSmoothEnable *Set/Get the individual smoothing switch*
- **setSmoothPeriod :**
  - in DtSmoothOrient *Set/Get smoothPeriod*
  - in DtSmoothPos *Set/Get smoothPeriod*
- setSmoothPeriod *Set/Get smoothing period - the amount of time over which smoothing takes place*
- setSpecific
- setSpotLinearAcceleration *Set/Get the spot linear acceleration*
- setStartOffset *Set/Get start offset*
- setStateDecoderFactory *Set/Get the current factory that can be used to create DtHlaStateDecoders*
- setStateEncoderFactory *Set/Get the current factory that can be used to create DtHlaStateEncoders*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*

- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStateRepCreator *Give us a function to be used for creating SRs*
- setStationNumber *Set/Get the station number for a part*
- setStatus *Sets the PDU's status to okStatus*
- setString *Set/Get for myString - char pointer that holds the string non-virtual because only called by constructor*
- setSubCategory
- setSubaggregateID *Set/Get a DIS sub-aggregate ID*
- setSupplies *Set/Get the supplies memory block all at once*
- setSupplies *Set/Get the supplies memory block all at once*
- setSupplies *Set/Get the supplies memory block all at once*
- setSupplyingId
- setSupplyingId *Set/Get the repairing object*
- setSupplyingId
- setSweepSync *Set/Get Sweep Sync*
- **setT1data :**
  - in DtSmoothOrient
  - in DtSmoothPos
- setTag *Set/Get the user-defined tag*
- setTag *Set/Get the user-defined tag*
- setTagIsAsciiTime *Set/Get flag that tells us whether we're using the RPR FOM convention that the UserDefinedTag in updates and interactions represents a DIS-style timestamp*
- setTagIsAsciiTime *Set/Get flag that tells us whether we're using the RPR FOM convention that the UserDefinedTag in updates and interactions represents a DIS-style timestamp*
- setTargetId *Set/Get the target identifier*
- setTargetId *Set/Get the target identifier*
- setTdlMessageCount *Set/Get TDL message count*
- setTdlType *Set/Get tactical data link*
- setTentDeployed *Set/Get whether a tent is deployed or not*
- setTest
- setTheta
- setThreshMode *Set/Get the theshold mode*
- setTime *set/get time to jump to within logger playback file*
- setTime *set/get amount of time to skip forward or back in playback file*

- setTime *Get the time contained in the timestamp to time % an hour*
- setTimeForSets
- setTimeHopInUse *Set/Get the Time Hop In Use*
- setTimeInterval *Set/Get the time interval*
- setTimeScale *set/get playback speed*
- setTimeStamp *Set the time of validity of the message, and whether this represents relative or absolute time*
- setTimeStamp
- setTimeStamp *Set the time of validity of the message, and whether this represents relative or absolute time*
- setTimeStampType *Set/Get flag indicating whether we're using absolute timestamps for dead reckoning*
- setTimeStampType *Set/Get time stamp type (DtTimeStampAbsolute or DtTimeStampRelative)*
- setTimeout *set/get timeout*
- setTimeoutInterval *Set/Get the remote object timeout period*
- setTimeoutProcessing *Set whether or not timeout processing occurs*
- setTopoUpOffset
- **setTrackId :**
  - in DtMimicTrackVCPdu
  - in DtTetherTrackVCPdu
  - in DtTrackVCPdu
- setTrailState *Set/Get trailing effect state*
- setTranslationThreshold *Set/Get translation threshold in meters*
- setTranslationThreshold *Set/Get translation threshold in meters*
- setTransmitRadioId *Set/Get the received transmitter*
- setTransmitState *Set/Get the transmit state*
- setType *Set/Get timestamp type*
- setUnbundling
- setUnbundling
- setUnbundling
- setUp
- setUpdateWhenFrozen
- setUserData *set/get the user data*
- setVal *Set/Get the value for the specified parameter type*
- setValsFromNet *sets values from net representation, but does not change the set of valid parts and parameters*
- setValue *Sets the value of an attribute by handle*
- setVarDataBytes *Allocates length bytes for the index'th variable datum record*
- setVarDataBytes *Allocates length bytes for the index'th variable datum record*
- setVarDatumId
- setVariableDatumIDs *Set/Get the variable fields memory block all at once*
- setVariableDatumRecord *Set/Get a variable datum record*
- setVcPdu

- setVelocity *Set absolute velocity*
- setVelocity *Set/Get the velocity*
- setVelocity *Set/Get the velocity*
- setVelocity *Set/Get the velocity*
- setVelocity *Set absolute velocity*
- setVelocity *Set/Get the velocity*
- setVelocity *Set/Get the velocity*
- setVelocity *Set absolute velocity*
- setVelocity *Set/Get the velocity*
- setVersion *Returns the version number of the FOM, as passed to setVersion*
- setVersion
- setViewerCommand
- setViewerId
- setWarheadType *Set/Get the warhead type*
- setWarheadType *Set/Get the warhead type*
- setWorldAntennaLoc *Set/Get the world location*
- setWorldLocation
- setWorldLocation *Set/Get the location of the detonation in world coordinates*
- setWorldLocation
- setZero *---------- Overloaded functions from RTI::FedTime ----------*
- setupClassForUpdating *If the class has never been published, publish it with all attributes using DtObjClassDesc::publish*
- setupReflecting *Do the initialization required to handle reflection of object*
- setupUpdating *Do the initialization required to begin publishing this object*
- shouldBeTimedOut *Returns whether or not the object should be timed out*
- shouldBeTimedOut *Returns whether or not the object should be timed out*
- simulationTime
- site
- siteId
- sizeChanged *Called by changeSize*
- sizeChanged *Really, should not be pure virtual in DtBaseMsg, but since it is, we need a null definition here*
- sizeChanged *Called by changeSize*
- sizeLength *Get/Set mySizeLength*
- sizeOffset *Get/Set mySizeOffset*
- smokePlumePresent
- **smoothEnable :**
    - in DtSmoothOrient
    - in DtSmoothPos

- smoothEnable
- **smoothPeriod :**
    - in DtSmoothOrient
    - in DtSmoothPos
- smoothPeriod
- snipBack
- snipBackToFirst
- snipFront *truncates everything before or after the character, including the character, respectively Normally, snipFront stops at the first occurrence of "ch", and snipBack stops at the last occurrence of "ch"*
- snipFrontToLast
- socketList
- socketMgr
- specific
- spotLinearAcceleration
- sr
- sr
- startIntersCb *Callback for handling startInters RTI callback*
- startOffset
- startRegCb *Callback for handling startRegistration RTI callback*
- startRegistrationForObjectClass *////////////////////////////*
- startRegistrationForObjectClass
- stateDecoderFactory
- stateEncoderFactory
- stateRep
- stateRep
- stationList
- stationNumber *Returns the station number for a part*
- status *Get PDU status*
- stopIntersCb *Callback for handling stopInters RTI callback*
- stopRegCb *Callback for handling stopRegistration RTI callback*
- stopRegistrationForObjectClass
- stopRegistrationForObjectClass
- **string :**
    - in DtAbsoluteVCPdu
    - in DtCompassVCPdu
- **string :**
    - in DtEntityIdentifier
    - in DtEntityType
- string *Convert to string*

- string
- **string :**
  - in DtGroupVCPdu
  - in DtMimicTrackVCPdu
  - in DtMimicVCPdu
- string
- string
- string
- string *returns the char* internal representation of the string*
- string
- **string :**
  - in DtTetherTrackVCPdu
  - in DtTetherVCPdu
  - in DtTrackVCPdu
- string *Get a string representation of the vector*
- string
- stringRep *get the string representation of this key*
- stringRep *get the string representation of this key*
- stringRep *get the string representation of this key*
- stringRep *get the string representation of this key*
- stringRep *get the string representation of this key*
- subAggregates
- subCategory
- subscribe *Subscribe to the class*
- subscribe *Subscribes to the class, along with the attributes specified by hSet (or all attributes is no hSet is given)*
- subscribeInteractionClass *Calls down to the RTI function of the same name*
- subscribeInteractionClass
- subscribeInteractionClassWithRegion *Calls down to the RTI function of the same name*
- subscribeInteractionClassWithRegion
- subscribeObjectClassAttributes *Calls down to the RTI function of the same name*
- subscribeObjectClassAttributes
- subscribeObjectClassAttributesWithRegion *Calls down to the RTI function of the same name*
- subscribeObjectClassAttributesWithRegion
- subscribedAttrs *Returns the set of subscribed attributes*
- subscribed *Returns DtTrue if the class is subscribed, DtFalse otherwise*
- supplies
- supplies
- supplies
- supplyQuantity

- supplyQuantity
- supplyQuantity
- supplyingId
- supplyingId
- supplyingId
- supplyingId
- supplyingId
- supplyingId
- sweepSync
- syncBeamList *Update myBeamPubs so that it has a publisher for all beamSRs in my systemSR, but no others*
- synchronizationPointAchieved *Calls down to the RTI function of the same name*
- synchronizationPointAchieved
- synchronizationPointRegistrationFailed
- synchronizationPointRegistrationSucceeded *//////////////////////////*
- tag
- tag
- tagIsAsciiTime
- tagIsAsciiTime
- targetId
- targetId
- tdlMessageCount
- tdlType
- tentDeployed
- terminatePacketServer
- testClients
- theta
- threshMode
- thresholder
- tick *Calls down to the RTI function of the same name*
- tick *syncBeamList and update beam objects as well as main system object*
- tick *Tick (update) this object*
- **tick :**
    - in DtSmoothOrient
    - in DtSmoothPos
- tick
- timeAdvanceGrant
- timeAdvanceRequestAvailable *Calls down to the RTI function of the same name*
- timeAdvanceRequestAvailable

- timeAdvanceRequest *Calls down to the RTI function of the same name*
- timeAdvanceRequest
- timeConstrainedEnabled
- time
- time
- time *Get the time contained in the timestamp (in seconds)*
- timeForSets
- timeHopInUse
- timeInterval
- timeOutObject *Times out object by removing it from the remote object list and deleting it*
- timeRegulationEnabled *//////////////////////*
- timeScale
- timeSet *Get the time at which the value was last set*
- timeStamp *Return the time of validity of the message (in seconds past an hour) as set by setTimeStamp*
- timeStamp
- timeStamp *Return the time of validity of the message (in seconds past an hour) as set by setTimeStamp*
- timeStampType
- timeStampType
- timeStampType *Return the type of the time stamp*
- timeStampType
- timeStampType *Return the type of the time stamp*
- timedOutObjectReflectCb *Cast the void* to a DtReflectedObjectList* and calls processTimedOutObjectReflect()*
- timedRecv
- timeout
- timeoutInterval
- topoUpOffset
- tpsiChanged
- **trackId :**
    - in DtMimicTrackVCPdu
    - in DtTetherTrackVCPdu
    - in DtTrackVCPdu
- trackJamTargets
- trailState
- translationThreshold
- translationThreshold
- translations *Get the translations*
- transmitRadioId
- transmitState
- transmitterRep

- trueChecker *A checker function that always returns 1, meaning that the attribute for which it is being used will be deemed necessary to send on every tick*
- tsr *Synonym for radioXmitRep()*
- tsr *Synonym for transmitterRep()*
- turnInteractionsOff
- turnInteractionsOff
- turnInteractionsOn
- turnInteractionsOn
- turnUpdatesOffForObjectInstance
- turnUpdatesOffForObjectInstance
- turnUpdatesOnForObjectInstance
- turnUpdatesOnForObjectInstance
- type
- unManage *remove the object from the list of local objects*
- unTimeOutObject *Manages callbacks and data used for reinstating a remote object which was timed out*
- unassociateRegionForUpdates *Calls down to the RTI function of the same name*
- unassociateRegionForUpdates
- unconditionalAttributeOwnershipDivestiture *Calls down to the RTI function of the same name*
- unconditionalAttributeOwnershipDivestiture
- unloadNetList *After this call, this attached part list will reflect the information contained in netParams*
- unloadNetRecord *Load/unload data from a packet record*
- unloadNetRecord *Load/unload data from a packet record*
- unpublishInteractionClass *Calls down to the RTI function of the same name*
- unpublishInteractionClass
- unpublishObjectClass *Calls down to the RTI function of the same name*
- unpublishObjectClass
- unsubscribe *Unsubscribe to the class*
- unsubscribe *Unsubscribe to the class*
- unsubscribeInteractionClass *Calls down to the RTI function of the same name*
- unsubscribeInteractionClass
- unsubscribeInteractionClassWithRegion *Calls down to the RTI function of the same name*
- unsubscribeInteractionClassWithRegion
- unsubscribeObjectClass *Calls down to the RTI function of the same name*
- unsubscribeObjectClass
- unsubscribeObjectClassWithRegion *Calls down to the RTI function of the same name*
- unsubscribeObjectClassWithRegion
- up
- updateAttributeValues *Calls down to the RTI function of the same name*
- updateAttributeValues

- update *Update (publish) this object on the network*
- update *Does nothing*
- updateWhenFrozen *SetGet updateWhenFrozen flag*
- updatesOffCb *Callback for handling turnUpdatesOffForObjectInstance RTI call*
- updatesOnCb *Callback for handling turnUpdatesOnForObjectInstance RTI call*
- useDeadReckoner *Create and use a new (private) dead-reckoner as our approximator*
- useSmoother *Create and use a new (private) smoother as our approximator*
- useThese *Specify an encoder, decoder, and interaction data repository to use in encoding and decoding*
- useTheseForReflecting *Set the state repository for holding a remote reflection and the decoder which is used to receive HLA state messages into the repository*
- useTheseForUpdating *Set the state repositories (local representation and remote reflection), and the encoder and decoder which are used to send and receive HLA state messages from/into the repository*
- userData
- userData *Get the user data*
- userData
- userData
- userData
- userData *Get the user data*
- userData *Get the user data*
- userData *Get the user data*
- userData *Get the user data*
- userData *Get the user data*
- usingDeadReckoner *If the ESR is using a dead reckoner, returns it*
- usingSmoother *If the ESR is using a smoother, returns it*
- utmToGeocMatrix *Get the matrix which transforms from UTM to geocentric*
- val *Get the value for the specified parameter type*
- valueDiffers
- value *Returns the value of the attribute associated with handle, or NULL if none exists*
- valueLength *Returns the length of the value of the attribute associated with handle, or - if none exists*
- varDatumId
- varDatumId *Returns the datum param found in the index'th variable datum record*
- varDatumId *Returns the datum param found in the index'th variable datum record*
- variableDatumIDs
- variableDatumRecordConstRef *Get a constant reference to a variable datum record*
- variableDatumRecordRef *Get a reference to a variable datum record*
- vcPdu
- vecTrans
- velocity
- velocity

- velocity
- velocity
- velocity
- velocity
- velocity
- version
- viewerCommand
- viewerId
- vrlinkVersion *Returns a string indicating the VR-Link version number*
- warheadType
- warheadType
- wipeReflectList *Remove all callbacks from the reflect callback list*
- wipeRemoveList *Remove all callbacks from the remove callback list*
- worldAntennaLoc
- worldLocation
- worldLocation
- worldLocation
- wrapNext *Return next (previous) elements on the list, but wrap to the beginning (end) of the list when you hit the end (beginning)*
- wrapNext *Get the next remote aggregate*
- wrapNext *Get the next remote designator*
- wrapNext *Get the next remote beam*
- wrapNext *Get the next remote emitterSystem*
- wrapNext *Get the next remote entity*
- wrapNext
- wrapNext *Get the next remote receiver*
- wrapNext *Get the next remote transmitter*
- wrapNextObj
- wrapPrev
- wrapPrev *Get the previous remote aggregate*
- wrapPrev *Get the previous remote designator*
- wrapPrev *Get the previous remote beam*
- wrapPrev *Get the previous remote emitterSystem*
- wrapPrev *Get the previous remote entity*
- wrapPrev
- wrapPrev *Get the previous remote receiver*
- wrapPrev *Get the previous remote transmitter*
- wrapPrevObj
- writeFd

- writeFd
- writeFd
- writeFd
- writeFd
- zero
- zero *Get a zero'ed vector*
- ~DtAbsNotifyObserver *destructor*
- ~DtAcknowledgeDecoder
- ~DtAcknowledgeEncoder
- ~DtAcknowledgeInteraction *destructor*
- ~DtActionRequestDecoder
- ~DtActionRequestEncoder
- ~DtActionRequestInteraction *destructor*
- ~DtActionRequestToObjDecoder
- ~DtActionRequestToObjEncoder
- ~DtActionRequestToObjInter *destructor*
- ~DtActionResponseDecoder
- ~DtActionResponseEncoder
- ~DtActionResponseFromObjDecoder
- ~DtActionResponseFromObjEncoder
- ~DtActionResponseFromObjInter *destructor*
- ~DtActionResponseInteraction *destructor*
- ~DtAggregateEntityDecoder *destructor*
- ~DtAggregateEntityEncoder *destructor*
- ~DtAggregatePublisher *destructor*
- ~DtAggregateStateRepository *destructor*
- ~DtAntennaPatternRecord *destructor*
- ~DtAppSpecificRadioSignalDecoder
- ~DtAppSpecificRadioSignalEncoder
- ~DtApproximator *destructor*
- ~DtArtPart *destructor*
- ~DtArtPartList *destructor*
- ~DtArtPartThresholder *Destructor*
- ~DtAttachedPartList *destructor*
- ~DtAttributeChangeReqDec
- ~DtAttributeChangeReqEnc
- ~DtAttributeChangeReqInter *destructor*
- ~DtAttributeChangeResDec

- ~DtAttributeChangeResEnc
- ~DtAttributeChangeResInter *destructor*
- ~DtBaseEmbeddedSystemDecoder
- ~DtBaseEmbeddedSystemEncoder
- ~DtBaseEmitterBeamDecoder
- ~DtBaseEmitterBeamEncoder
- ~DtBaseEntityDecoder
- ~DtBaseEntityEncoder
- ~DtBaseEntityStateRepository *destructor*
- ~DtBaseExerciseConn *destructor*
- ~DtBaseHashKey *destructor*
- ~DtBaseMsg *Destructor*
- ~DtBaseReflectedAggregate *destructor*
- ~DtBaseReflectedEntity *destructor*
- ~DtBeamAntennaPatternRecord *destructor*
- ~DtCartesianView *destructor*
- ~DtClientSocket
- ~DtCollisionDecoder
- ~DtCollisionEncoder
- ~DtCollisionInteraction *destructor*
- ~DtCommentDecoder
- ~DtCommentEncoder
- ~DtCommentInteraction *destructor*
- ~DtCreateEntityDecoder
- ~DtCreateEntityEncoder
- ~DtCreateEntityInteraction *destructor*
- ~DtCreateObjReqDecoder
- ~DtCreateObjReqEncoder
- ~DtCreateObjReqInter *destructor*
- ~DtCreateObjectResultDecoder
- ~DtCreateObjectResultEncoder
- ~DtCreateObjectResultInter *destructor*
- ~DtCsFilterTest
- ~DtDataDecoder
- ~DtDataEncoder
- ~DtDataInteraction *destructor*
- ~DtDataQueryDecoder
- ~DtDataQueryEncoder

- ~DtDataQueryInteraction *destructor*
- ~DtDatabaseIndexRadioSignalDecoder
- ~DtDatabaseIndexRadioSignalEncoder
- ~DtDatumRecordSet *Destructor*
- ~DtDeadReckoner *destructor*
- ~DtDesignatorDecoder
- ~DtDesignatorEncoder
- ~DtDesignatorPublisher *destructor*
- ~DtDesignatorRepository *destructor*
- ~DtDetonationDecoder
- ~DtDetonationEncoder
- ~DtDetonationInteraction *destructor*
- ~DtDirectRtiAmbassador *destructor*
- ~DtDiscoverObjectCbInfo *destructor*
- ~DtEmbeddedSystemRepository *destructor*
- ~DtEmitterBeamPublisher *destructor*
- ~DtEmitterBeamRepository *destructor*
- ~DtEmitterSystemDecoder
- ~DtEmitterSystemEncoder
- ~DtEmitterSystemPublisher *destructor*
- ~DtEmitterSystemRepository *destructor*
- ~DtEmptyFomMapper *destructor*
- ~DtEncodedAudioRadioSignalDecoder
- ~DtEncodedAudioRadioSignalEncoder
- ~DtEntityPublisher *destructor*
- ~DtEntityStateRepository *destructor*
- ~DtEnvironmentEntityDecoder *destructor*
- ~DtEnvironmentEntityEncoder *destructor*
- ~DtEventReportDecoder
- ~DtEventReportEncoder
- ~DtEventReportInteraction *destructor*
- ~DtExerciseConn *destructor*
- ~DtFedReader *Destructor*
- ~DtFedTarget *Destructor*
- ~DtFilename *destructor*
- ~DtFilterTest
- ~DtFireDecoder
- ~DtFireEncoder

- ~DtFireInteraction *destructor*
- ~DtFom *Destructor*
- ~DtFomMapper *destructor*
- ~DtGenericObjectDecoder *destructor*
- ~DtGenericObjectEncoder *destructor*
- ~DtGenericObjectPublisher *destructor*
- ~DtGenericStateRepository *destructor*
- ~DtGeodeticCoord *destructor*
- ~DtGlobalIdHashlist *destructor*
- ~DtGlobalObjectDesignatorList *destructor*
- ~DtGroupSocket *Destructor*
- ~DtHashItem *destructor*
- ~DtHashlist *Destructor*
- ~DtHlaObject *destructor*
- ~DtHlaObjectFactory *destructor*
- ~DtHlaObjectWithStateRep *destructor*
- ~DtHlaStateDecoder *destructor*
- ~DtHlaStateEncoder *destructor*
- ~DtIntHashKey *destructor*
- ~DtInterClassDesc *destructor*
- ~DtInteractionDecoder *destructor*
- ~DtInteractionDecoderFactory *destructor*
- ~DtInteraction *destructor*
- ~DtInteractionEncoder *destructor*
- ~DtInteractionEncoderFactory *destructor*
- ~DtInteractionFactory *destructor*
- ~DtInteractionManager *destructor*
- ~DtInteractionWithData *destructor*
- ~DtInteractionWithEncDec *destructor*
- ~DtJammerBeamDecoder
- ~DtJammerBeamEncoder
- ~DtLgrControlDecoder
- ~DtLgrControlEncoder
- ~DtLgrControlInteraction *destructor*
- ~DtLgrControlPdu *destructor*
- ~DtLgrEndPdu *destructor*
- ~DtLgrEofActionPdu *destructor*
- ~DtLgrHeartbeatPdu *destructor*

- ~DtLgrPausePdu *destructor*
- ~DtLgrPbFilePdu *destructor*
- ~DtLgrPlayActionPdu *destructor*
- ~DtLgrPlayPdu *destructor*
- ~DtLgrQuitPdu *destructor*
- ~DtLgrRecFilePdu *destructor*
- ~DtLgrRecordActionPdu *destructor*
- ~DtLgrRecordPdu *destructor*
- ~DtLgrSetTimePdu *destructor*
- ~DtLgrSkipTimePdu *destructor*
- ~DtLgrStartPdu *destructor*
- ~DtLgrStopPdu *destructor*
- ~DtLgrTimeScalePdu *destructor*
- ~DtLgrTimeoutPdu *destructor*
- ~DtList *Destructor - does NOT free memory of data associated with the items*
- ~DtListItem *Destructor*
- ~DtListedObj *Destructor*
- ~DtLocalHlaObjectManager *destructor*
- ~DtMilitaryEntityDecoder *destructor*
- ~DtMilitaryEntityEncoder *destructor*
- ~DtMilitaryPlatformEntityDecoder *destructor*
- ~DtMilitaryPlatformEntityEncoder *destructor*
- ~DtMunitionEntityDecoder *destructor*
- ~DtMunitionEntityEncoder *destructor*
- ~DtNetBoolean *destructor*
- ~DtNetSocket
- ~DtNetTimeStamp *destructor*
- ~DtNullFederateAmbassador
- ~DtObjClassDesc *destructor*
- ~DtObjectIdHashKey *destructor*
- ~DtObjectIdHashlist *destructor*
- ~DtObjectPublisher *Destructor*
- ~DtObjectReflectAttributeValuesCbInfo *destructor*
- ~DtPdu *Destructor*
- ~DtPhysicalEntityDecoder *destructor*
- ~DtPhysicalEntityEncoder *destructor*
- ~DtRadarBeamDecoder
- ~DtRadarBeamEncoder

- ~DtRadioReceiverDecoder
- ~DtRadioReceiverEncoder
- ~DtRadioReceiverPublisher *destructor*
- ~DtRadioReceiverRepository *destructor*
- ~DtRadioTransmitterDecoder
- ~DtRadioTransmitterEncoder
- ~DtRadioTransmitterPublisher *destructor*
- ~DtRadioTransmitterRepository *destructor*
- ~DtRawBinaryRadioSignalDecoder
- ~DtRawBinaryRadioSignalEncoder
- ~DtReceiveInteractionCbInfo *destructor*
- ~DtReflectAttributeValuesCbInfo *destructor*
- ~DtReflectAttributeValuesCbList *destructor*
- ~DtReflectedAggregate *destructor*
- ~DtReflectedAggregateList *destructor*
- ~DtReflectedDesignator *destructor*
- ~DtReflectedDesignatorList *destructor*
- ~DtReflectedEmitterBeam *destructor*
- ~DtReflectedEmitterBeamList *destructor*
- ~DtReflectedEmitterSystem *destructor*
- ~DtReflectedEmitterSystemList *destructor*
- ~DtReflectedEntity *destructor*
- ~DtReflectedEntityList *destructor*
- ~DtReflectedGenericObject *destructor*
- ~DtReflectedGenericObjectList *destructor*
- ~DtReflectedHlaObjectManager *destructor*
- ~DtReflectedObjectCallbackInfo *destructor*
- ~DtReflectedObject *destructor*
- ~DtReflectedObjectList *destructor*
- ~DtReflectedRadioReceiver *destructor*
- ~DtReflectedRadioReceiverList *destructor*
- ~DtReflectedRadioTransmitter *destructor*
- ~DtReflectedRadioTransmitterList *destructor*
- ~DtRegControlCbInfo *destructor*
- ~DtRegControlCbList *destructor*
- ~DtRemoveEntityDecoder
- ~DtRemoveEntityEncoder
- ~DtRemoveEntityInteraction *destructor*

- ~DtRemoveObjectCbInfo *destructor*
- ~DtRemoveObjectRequestDecoder
- ~DtRemoveObjectRequestEncoder
- ~DtRemoveObjectRequestInter *destructor*
- ~DtRemoveObjectResultDecoder
- ~DtRemoveObjectResultEncoder
- ~DtRemoveObjectResultInter *destructor*
- ~DtRepairCompleteDecoder
- ~DtRepairCompleteEncoder
- ~DtRepairCompleteInteraction *destructor*
- ~DtRepairResponseDecoder
- ~DtRepairResponseEncoder
- ~DtRepairResponseInteraction *destructor*
- ~DtResupplyCancelDecoder
- ~DtResupplyCancelEncoder
- ~DtResupplyCancelInteraction *destructor*
- ~DtResupplyOfferDecoder
- ~DtResupplyOfferEncoder
- ~DtResupplyOfferInteraction *destructor*
- ~DtResupplyRecInteraction *destructor*
- ~DtResupplyReceivedDecoder
- ~DtResupplyReceivedEncoder
- ~DtRprFomMapper *destructor*
- ~DtRprHlaObjectFactory *destructor*
- ~DtRprInteractionDecoderFactory *destructor*
- ~DtRprInteractionEncoderFactory *destructor*
- ~DtRprInteractionFactory *destructor*
- ~DtRprStateDecoderFactory *destructor*
- ~DtRprStateEncoderFactory *destructor*
- ~DtScript
- ~DtServiceRequestDecoder
- ~DtServiceRequestEncoder
- ~DtServiceRequestInteraction *destructor*
- ~DtSetDataDecoder
- ~DtSetDataEncoder
- ~DtSetDataInteraction *destructor*
- ~DtSignalInteraction *destructor*
- ~DtSilentAggregateList *Destructor*

- ~DtSilentEntityList *Destructor*
- ~DtSmoothOriWithDr *destructor*
- ~DtSmoothPos
- ~DtSmoothPosWithDr *destructor*
- ~DtSmoother *destructor*
- ~DtSocket
- ~DtSoldierDecoder *destructor*
- ~DtSoldierEncoder *destructor*
- ~DtStartResumeDecoder
- ~DtStartResumeEncoder
- ~DtStartResumeInteraction *destructor*
- ~DtStartStopInterCbInfo *destructor*
- ~DtStartStopInterCbList *destructor*
- ~DtStateDecoderFactory *destructor*
- ~DtStateEncoderFactory *destructor*
- ~DtStateMsg *destructor*
- ~DtStateRepository *destructor*
- ~DtStopFreezeDecoder
- ~DtStopFreezeEncoder
- ~DtStopFreezeInteraction *destructor*
- ~DtString *destructor*
- ~DtStringHashKey *destructor*
- ~DtSystemHashKey *destructor*
- ~DtTcpForwarder
- ~DtTcpSocket *Destructor*
- ~DtTcpSocketMgr
- ~DtThresholder *destructor*
- ~DtTopoView *destructor*
- ~DtTrackJamBeamDecoder
- ~DtTrackJamBeamEncoder
- ~DtUnknownHlaObject *destructor*
- ~DtUnknownInteraction *destructor*
- ~DtUpdateControlCbInfo *destructor*
- ~DtUpdateControlCbInfo *destructor*
- ~DtUpdateControlCbList *destructor*
- ~DtUpdateControlCbList *destructor*
- ~DtUtmCoord *destructor*
- ~DtUtmView *destructor*

- ~DtVariableDatumRecord *destructor*
- ~DtVector *destructor*
- ~DtViewControlDecoder
- ~DtViewControlEncoder
- ~DtViewControlInteraction *destructor*
- ~DtVrlFedTime
- ~DtVrlFederateAmbassador *destructor*
- ~DtVrlRtiAmbassador *destructor*

Variables

- BeamFuncJammer
- DetailedModulationUnmodulated
- Dt105m
- Dt120mm
- Dt25mm
- Dt7_62mm
- Dt7_62mmUK
- Dt7_62mmUSSR
- DtA10
- DtA2ATD_CAC2
- DtA2ATD_SINCGARS_ERF
- DtAA3Anan
- DtAA9Amos
- DtADPCM_per_CCITT_G721
- DtAGM88HARM
- DtAGTS
- DtAH1W
- DtAH64
- DtAMRAAM
- DtAN_PSS_11
- DtAN_PSS_12
- DtASCII_CharacterSet
- DtASTAMIDS
- DtASTAMIDS_I
- DtASTAMIDS_II
- DtAbbreviated_Command_And_Control
- DtAckCreate
- DtAckRemove

- DtAckStart
- DtAckStop
- DtAcknowledgePduKind
- DtAcknowledgeRPduKind
- DtAcknowledged
- DtActionAcceptSubscription
- DtActionChangeAggregateState
- DtActionDamage
- DtActionDataQuery
- DtActionDismount
- DtActionFireDisabled
- DtActionInitializeInternalParameters
- DtActionInitiateHitchFollow
- DtActionInitiateHitchLead
- DtActionInitiateServiceStationResupply
- DtActionInitiateTailgateResupply
- DtActionInitiateTetherFollow
- DtActionInitiateTetherLead
- DtActionJoinExercise
- DtActionKilledInAction
- DtActionLocalStorage
- DtActionLockSiteConfiguration
- DtActionMobilityDisabled
- DtActionMount
- DtActionMountIntent
- DtActionOther2
- DtActionOther
- DtActionOutOfAmmo
- DtActionOutOfFuel
- DtActionQuerySiteConfiguration
- DtActionRecallCheckpointData
- DtActionRecallInitialParameters
- DtActionReconstitute
- DtActionRequestPduKind
- DtActionRequestRPduKind
- DtActionRequestStartPDU
- DtActionResignExercise
- DtActionResponsePduKind

- DtActionResponseRPduKind
- DtActionRun
- DtActionSaveInternalParameters
- DtActionSendObjectStateData
- DtActionSendPlanData
- DtActionSimulateMalfunction
- DtActionStartDRC
- DtActionStatusRequest
- DtActionStopDRC
- DtActionSynchronizeInternalClocks
- DtActionTeleportEntity
- DtActionTetheringInformation
- DtActionTimeAdvance
- DtActionUnhitch
- DtActionUnlockSiteConfiguration
- DtActionUnsubscribe
- DtActionUntether
- DtActionUpdateSiteConfiguration
- DtActionWakeUpGetReadyForInitialization
- DtActuatorCatapults
- DtActuatorCoolingSys
- DtActuatorCranes
- DtActuatorLaunchers
- DtActuatorWaterSupply
- DtActuatorWinches
- DtAfsk
- DtAftTorpedoTube
- DtAggregateStatePduKind
- DtAggregated
- DtAircraftWingstation
- DtAm
- DtAmmunitionShip
- DtAmmunitionSupplyCap
- DtAmphibiousAssaultShip
- DtAmphibiousCargoShip
- DtAmphibiousTransportDock
- DtAmplitudeAndAngle
- DtAmplitudeAnglePulse

- DtAmplitude
- DtAngle
- DtAntiAir
- DtAntiArmor
- DtAntiArmorGuided
- DtAntiArmorUnguided
- DtAntiGuidedMunition
- DtAntiHandlingDevice
- DtAntiPersonnel
- DtAntiPersonnelGuided
- DtAntiPersonnelUnguided
- DtAntiRadar
- DtAntiRadarGuided
- DtAntiRadarUnguided
- DtAntiSatellite
- DtAntiShip
- DtAntiShipGuided
- DtAntiShipUnguided
- DtAntiSubmarine
- DtAntiSubmarineGuided
- DtAntiSubmarinePatrol
- DtAntiSubmarineUnguided
- DtApAzimuth
- DtApAzimuthRate
- DtApElevation
- DtApElevationRate
- DtApExtension
- DtApExtensionRate
- DtApPosition
- DtApPositionRate
- DtApRotation
- DtApRotationRate
- DtApX
- DtApXRate
- DtApY
- DtApYRate
- DtApZ
- DtApZRate

- DtAppDamageBitShift
- DtAppDamage
- DtAppDummyBitShift
- DtAppDummy
- DtAppFirePowerKillBitShift
- DtAppFirePowerKill
- DtAppFlamingBitShift
- DtAppFlaming
- DtAppFrozenBitShift
- DtAppFrozen
- DtAppHatchBitShift
- DtAppHatch
- DtAppLightsBitShift
- DtAppLights
- DtAppMobilityKillBitShift
- DtAppMobilityKill
- DtAppPaintSchemeBitShift
- DtAppPaintScheme
- DtAppPowerPlantStatusBitShift
- DtAppPowerPlantStatus
- DtAppReqTransferEntity
- DtAppReqTransferEnvProc
- DtAppSmokeBitShift
- DtAppSmoke
- DtAppTrailingEffectsBitShift
- DtAppTrailingEffects
- DtAppearInvalid
- DtApplicationSpecific
- DtArealObjectStatePduKind
- DtArmoredFightingVehicle
- DtArmoredUtilityVehicle
- DtArmy
- DtArtPartAggregateThresh
- DtArtPartDfltThreshMode
- DtArtPartIndividualThresh
- DtArtPartSpec
- DtArticulatedParts
- DtAssemblyFormation

- DtAttachedPartSuccess
- DtAttachedTo
- DtAttachedToBase
- DtAttack
- DtAttackHelicopter
- DtAttributeChangeResultOther
- DtAuxCraftEjectionSeat
- DtAuxCraftLandingCraft
- DtAuxCraftLifeBoat
- DtBGM109Tomahawk
- DtBGM71TOW
- DtBMP1
- DtBMP2
- DtBRDM1
- DtBadBeamId
- DtBasebandEncryption
- DtBasicF16
- DtBasicFixedWingAircraft
- DtBasicGroundCombatSoldier
- DtBasicGroundCombatVehicle
- DtBasicRotorWingAircraft
- DtBasicT72
- DtBattalion
- DtBattle_Command
- DtBattlefield105mm
- DtBattlefield105mmUK
- DtBattlefield120mm
- DtBattlefield20mm
- DtBattlefield30mm
- DtBattlefieldSupport
- DtBattlefieldSupportGuided
- DtBattlefieldSupportUnguided
- DtBattleship
- DtBeam
- DtBeamFuncAcquisitionAndTracking
- DtBeamFuncAcquisition
- DtBeamFuncActiveRadarMissileSeeker
- DtBeamFuncCommandGuidance

- DtBeamFuncHeightFinder
- DtBeamFuncIllumination
- DtBeamFuncMissileBeacon
- DtBeamFuncMissileFuze
- DtBeamFuncOther
- DtBeamFuncRangeOnlyRadar
- DtBeamFuncSearch
- DtBeamFuncTracking
- DtBelleauWood
- DtBest_Effort
- DtBinoculars
- DtBlackPaintScheme
- DtBluePaintScheme
- DtBodyAirframe
- DtBodyHull
- DtBodyRates *Angular rotation about body X, Y, Z axes*
- DtBombBay
- DtBomber
- DtBridgeConcreteTwoLane
- DtBrownPaintScheme
- DtBuildingFourStory
- DtBuildingOneStory
- DtBuildingThreeStory
- DtBuildingTwoStory
- DtC130
- DtCCSIL
- DtCCTT_SINCGARS_DATA
- DtCCTT_SINCGARS_ERF_HOPSET
- DtCCTT_SINCGARS_ERF_LOCKOUT
- DtCCTT_SINCGARS_OTAR
- DtCOBRA3_5
- DtCOBRA8_12
- DtCONUSNAD27
- DtCV59
- DtCVSD_per_MIL_STD_188_113
- DtCamouflageDesertPaintScheme
- DtCamouflageGravelPaintScheme
- DtCamouflageJunglePaintScheme

- DtCamouflagePavementPaintScheme
- DtCamouflageSandPaintScheme
- DtCamouflageSnowPaintScheme
- DtCargoBay
- DtCargo
- DtCarrier
- DtCategory
- DtCategory
- DtCategory
- DtChallengerMBT
- DtChange
- DtChangeDataUpdate
- DtCharacterSet
- DtCivilianCar
- DtClearPaintScheme
- DtCloud
- DtCloudWithRainFalling
- DtCloudWithSnowFalling
- DtCoPilot
- DtCollisionElasticPduKind
- DtCollisionPduKind
- DtCollisionTypeOther
- DtColumnFormation
- DtCombatStoresShip
- DtCombination
- DtCommandFuse
- DtCommandShip
- DtCommander
- DtCommentPduKind
- DtCommentRPduKind
- DtCompany
- DtComputerGeneratedForcesCarriedAsCargo
- DtConstDcm
- DtConstVector
- DtContinuous
- DtCorps
- DtCouldNotOpenFed
- DtCountry

- DtCountryCodeOther
- DtCountry
- DtCountry
- DtCreateEntityPduKind
- DtCreateEntityRPduKind
- DtCreateObjectResultOther
- DtCtlReqTransferEntity
- DtCtlReqTransferEnvProc
- DtCulturalFeatureDomainAir
- DtCulturalFeatureDomainLand
- DtCulturalFeatureDomainOther
- DtCulturalFeatureDomainSpace
- DtCulturalFeatureDomainSubsurface
- DtCulturalFeatureDomainSurface
- DtCulturalFeature
- DtCw
- DtD20
- DtDamageDestroyed
- DtDamageModerate
- DtDamageNone
- DtDamageSlight
- DtDataPduKind
- DtDataQueryPduKind
- DtDataQueryRPduKind
- DtDataRPduKind
- DtDatumAltCategory
- DtDatumAltCountry
- DtDatumAltDomain
- DtDatumAltExtra
- DtDatumAltKind
- DtDatumAltSpecific
- DtDatumAltSubCategory
- DtDatumAmmoQuantity
- DtDatumAmountAmmoSupplies
- DtDatumAppearance
- DtDatumArtParamId
- DtDatumArtParamType
- DtDatumArtParamValue

- DtDatumBadId
- DtDatumBadIndex
- DtDatumBadLength
- DtDatumBodyPartDamagedRatio
- DtDatumCapability
- DtDatumCategory
- DtDatumCountry
- DtDatumDomain
- DtDatumDrAlgorithm
- DtDatumExtra
- DtDatumForceId
- DtDatumFuelQuantity
- DtDatumKind
- DtDatumLoggerFile
- DtDatumMarkingCS
- DtDatumMarking
- DtDatumMass
- DtDatumMovingRadioTime
- DtDatumNumArtParam
- DtDatumPhi
- DtDatumPsi
- DtDatumRadarSysStatus
- DtDatumRadioCommSysStatus
- DtDatumSpecific
- DtDatumStationaryRadioTime
- DtDatumSubCategory
- DtDatumTerrainDatabase
- DtDatumTheta
- DtDatumValid
- DtDatumXaccel
- DtDatumXangvel
- DtDatumXpos
- DtDatumXvel
- DtDatumYaccel
- DtDatumYangvel
- DtDatumYpos
- DtDatumYvel
- DtDatumZaccel

- DtDatumZangvel
- DtDatumZpos
- DtDatumZvel
- DtDcmRef
- DtDeferedEntityId
- DtDegMinSec
- DtDensityClear
- DtDensityDense
- DtDensityHazy
- DtDensityOpaque
- DtDensityVeryDense
- DtDesertCamouflage
- DtDesignatorCodeOther
- DtDesignatorNameOther
- DtDesignatorNameTBD
- DtDesignatorPduKind
- DtDestroyer
- DtDestroyerTender
- DtDetResAirBurst
- DtDetResAirHit
- DtDetResArmorPiercingHit
- DtDetResBuildingHitLarge
- DtDetResBuildingHitMedium
- DtDetResBuildingHitSmall
- DtDetResDetonation
- DtDetResDirtBlastLarge
- DtDetResDirtBlastMedium
- DtDetResDirtBlastSmall
- DtDetResEntityImpact
- DtDetResEntityProximate
- DtDetResEnvironmentObjectImpact
- DtDetResEnvironmentObjectProximate
- DtDetResGroundImpact
- DtDetResGroundProximate
- DtDetResHE_HitLarge
- DtDetResHE_HitMedium
- DtDetResHE_HitSmall
- DtDetResMineClearingLineCharge

- DtDetResNone
- DtDetResOther
- DtDetResWaterBlastLarge
- DtDetResWaterBlastMedium
- DtDetResWaterBlastSmall
- DtDetResWaterImpact
- DtDetailedAmplitudeAndAngle
- DtDetailedModulationDummy
- DtDetonationPduKind
- DtDigitalDataDevice
- DtDiphaseEncryption
- DtDisaggregated
- DtDismountedInfantry
- DtDivision
- DtDockLandingShip
- DtDomain
- DtDomain *this conflicts with enum of same name, so must be before funcs that use it*
- DtDomain
- DtDrDrmFpb
- DtDrDrmFpw
- DtDrDrmFvb
- DtDrDrmFvw
- DtDrDrmRpb
- DtDrDrmRpw
- DtDrDrmRvb
- DtDrDrmRvw
- DtDrOther
- DtDrStatic
- DtDragonsTeeth
- DtDriveTrainAlternator
- DtDriveTrainBattery
- DtDriveTrainBrakes
- DtDriveTrainEngineCoolantLeak
- DtDriveTrainEngineOilLeak
- DtDriveTrainFilters
- DtDriveTrainFuelFilter
- DtDriveTrainGenerator
- DtDriveTrainMotor

- DtDriveTrainPumps
- DtDriveTrainStarter
- DtDriveTrainTranmissionOilLeak
- DtDriveTrainTransmission
- DtDriver
- DtDsb
- DtDumpFile
- DtDuplicateBeamId
- DtED50
- DtEParm *Euler parameter routines*
- DtEast
- DtElastic
- DtElectromagneticEmissionPduKind
- DtElectronicCoders
- DtElectronicComputers
- DtElectronicDecoders
- DtElectronicDetectionSys
- DtElectronicIntercoms
- DtElectronicLasers
- DtElectronicRadios
- DtElectronicRangeFinder
- DtElectronicSys
- DtElectronicWarfare
- DtElectronicWarfareSys
- DtEmissionFuncAcquisitionDetection
- DtEmissionFuncDecoy
- DtEmissionFuncEarlyWarningSurveillance
- DtEmissionFuncFireControl
- DtEmissionFuncFiringPointLaunchPointLocation
- DtEmissionFuncGuidanceIllumination
- DtEmissionFuncHeightFinding
- DtEmissionFuncIdentificationClassification
- DtEmissionFuncImaging
- DtEmissionFuncInstrumentation
- DtEmissionFuncJammingDeception
- DtEmissionFuncJammingNoise
- DtEmissionFuncMotionDetection
- DtEmissionFuncMultiFunction

- DtEmissionFuncNavigation
- DtEmissionFuncOther
- DtEmissionFuncRadarAltimeter
- DtEmissionFuncRanging
- DtEmissionFuncTracking
- DtEmissionFuncWeaponLethal
- DtEmissionFuncWeaponNonLethal
- DtEmissionFuncWeather
- DtEmitterMountedOnHost
- DtEmitterSuccess
- DtEmitterSysOther
- DtEncodedVoice
- DtEncodingTypeOther
- DtEncoding_16_bit_linear_PCM
- DtEncoding_8_bit_linear_PCM
- DtEncoding_8_bit_mu_law
- DtEnhancedFixedWingAircraft
- DtEnhancedGroundCombatSoldier
- DtEnhancedGroundCombatVehicle
- DtEnhancedRotorWingAircraft
- DtEntity
- DtEntityBase
- DtEntity
- DtEntity
- DtEntityKind
- DtEntityKind
- DtEntityKindOther
- DtEntityStatePduKind
- DtEntityStateUpdatePduKind
- DtEntityStationedAt
- DtEnvModelInvalid
- DtEnvironmentalDomainAir
- DtEnvironmentalDomainAirOther
- DtEnvironmentalDomainLand
- DtEnvironmentalDomainOther
- DtEnvironmentalDomainSpace
- DtEnvironmentalDomainSubsurface
- DtEnvironmentalDomainSurface

- DtEnvironmental
- DtEnvironmentalProcessPduKind
- DtEnvironmentalStatusLast
- DtEnvironmentalStatusOn
- DtErrorReadingFed
- DtEsrCreator *For backwards compatibility*
- DtEvent
- DtEventDamage
- DtEventID
- DtEventKilledInAction
- DtEventOther
- DtEventOutOfAmmo
- DtEventOutOfFuel
- DtEventReportPduKind
- DtEventReptRPduKind
- DtExerciseId
- DtExtra
- DtExtra
- DtExtra
- DtF117
- DtF15
- DtF15E
- DtF16C
- DtF16
- DtF18
- DtFFG7
- DtFLIR_SensorType
- DtFalse
- DtFamilyDistributedEmissionRegeneration
- DtFamilyEntityInteraction
- DtFamilyEntityManagement
- DtFamilyLiveEntity
- DtFamilyLogistics
- DtFamilyMinefield
- DtFamilyNonRealTime
- DtFamilyOther
- DtFamilyRadioCommunication
- DtFamilySimulationManagement

- DtFamilySimulationManagementWithReliability
- DtFamilySyntheticEnvironment
- DtFamilyWarfare
- DtFedReadOK
- DtFighter
- DtFirePduKind
- DtFireTeam
- DtFirstOfficer
- DtFixed
- DtFloat32
- DtFloat64
- DtFm
- DtFog
- DtForceFriendly
- DtForceNeutral
- DtForceOpposing
- DtForceOther
- DtForestCamouflage
- DtForwardTorpedoTube
- DtFrance
- DtFrequencyHoppingTimeHopping
- DtFrequncyHopping
- DtFrigate
- DtFsk
- DtFtInetAddr
- DtFtNetU8
- DtFuelCarriedAsCarge
- DtFuelGauges
- DtFuelLines
- DtFuelSupplyCap
- DtFuelTransferPump
- DtFullyDisaggregated
- DtFuze
- DtFuzeAcoustic
- DtFuzeAltitude
- DtFuzeCommand
- DtFuzeContactDelayed
- DtFuzeContact

- DtFuzeContactInstant
- DtFuzeDepth
- DtFuze
- DtFuze
- DtFuzeOther
- DtFuzeProximity
- DtFuzeTimed
- DtGAZ66
- DtGSTAMIDS3_5
- DtGSTAMIDS8_12
- DtGSTAMIDS
- DtGSTAMIDS_I
- DtGSTAMIDS_II
- DtGazelle
- DtGeneric3_5
- DtGeneric8_12
- DtGenericAntenna
- DtGeneric
- DtGenericGPR
- DtGenericPduKind
- DtGenericProbe
- DtGlobalIdHashList
- DtGlobalObjectDesignator
- DtGreenPaintScheme
- DtGridAxisFixed
- DtGridAxisVariable
- DtGridDataRecType0
- DtGridDataRecType1
- DtGridDataRecType2
- DtGriddedDataPduKind
- DtGroundLogisticsVehicle
- DtGuidedMissleCruiser
- DtGuidedMissleDestroyer
- DtGuidedMissleFrigate
- DtGunmountAttachedToHost
- DtGunner
- DtHMMWV
- DtHMMWV_OccActivelySearching

- DtHMMWV_OccNotActivelySearching
- DtHSTAMIDS3_5
- DtHSTAMIDS8_12
- DtHSTAMIDS_I
- DtHSTAMIDS_II
- DtHashList
- DtHatchClosed
- DtHatchNA
- DtHatchOpenAndPerson
- DtHatchOpen
- DtHatchPoppedAndPerson
- DtHatchPopped
- DtHawk
- DtHeartbeatMode
- DtHeavyAlgae
- DtHeavyCruiser
- DtHellfire
- DtHost
- DtHost
- DtHost
- DtHostFireableMunition
- DtHour2Sec
- DtHq
- DtHqIIA
- DtHqII
- DtIffAtcNavaidsATCRBS
- DtIffAtcNavaidsAltParm4Invalid
- DtIffAtcNavaidsAltParm4NoResponse
- DtIffAtcNavaidsAltParm4Other
- DtIffAtcNavaidsAltParm4Valid
- DtIffAtcNavaidsAlternateMode4
- DtIffAtcNavaidsAlternateMode5C
- DtIffAtcNavaidsAppModeInvalid
- DtIffAtcNavaidsChangeIndicator
- DtIffAtcNavaidsInterrogator
- DtIffAtcNavaidsLayerInfoInvalid
- DtIffAtcNavaidsMarkX
- DtIffAtcNavaidsMarkXII

- DtIffAtcNavaidsMarkX_XII_ATCRBS
- DtIffAtcNavaidsMarkX_XII_ATCRBS_ModeS
- DtIffAtcNavaidsModeInvalid
- DtIffAtcNavaidsModeS
- DtIffAtcNavaidsModifierInvalid
- DtIffAtcNavaidsOperationalStatus
- DtIffAtcNavaidsOtherSysName
- DtIffAtcNavaidsOtherSysType
- DtIffAtcNavaidsParameter1
- DtIffAtcNavaidsParameter2
- DtIffAtcNavaidsParameter3
- DtIffAtcNavaidsParameter4
- DtIffAtcNavaidsParameter5
- DtIffAtcNavaidsParameter6
- DtIffAtcNavaidsParameterDamage
- DtIffAtcNavaidsParameterMalfunction
- DtIffAtcNavaidsParameterStatus
- DtIffAtcNavaidsParameterTCAS
- DtIffAtcNavaidsParm1Invalid
- DtIffAtcNavaidsParm2Invalid
- DtIffAtcNavaidsPduKind
- DtIffAtcNavaidsSoviet
- DtIffAtcNavaidsSovietInterrogator
- DtIffAtcNavaidsSovietTransponder
- DtIffAtcNavaidsSystemOnOff
- DtIffAtcNavaidsTransponder
- DtImageIntensifier
- DtInelastic
- DtInetAddr ---------- *inet utils* ----------
- DtInetAddrAny
- DtInetBroadcastMask
- DtInitErrorFatal
- DtInitErrorIgnore
- DtInitErrorWarn
- DtInsideOf
- DtInt16
- DtInt32
- DtInt8

- DtIntercomAck
- DtIntercomCmdStatus
- DtIntercomConnect
- DtIntercomControlPduKind
- DtIntercomCtlStatus
- DtIntercomDisconnect
- DtIntercom
- DtIntercomEntDestRec
- DtIntercomFDX
- DtIntercomGrpAsgmtRec
- DtIntercomGrpDestRec
- DtIntercomHDX
- DtIntercomHDX_RecOnly
- DtIntercomHDX_TransOnly
- DtIntercomLS_NoCmd
- DtIntercomNack
- DtIntercomNoCmd
- DtIntercomNotApplicable
- DtIntercomNotTransmitting
- DtIntercomOff
- DtIntercomOn
- DtIntercomReqAck
- DtIntercomReqNoAck
- DtIntercomReset
- DtIntercomRetLocLS_Ctl
- DtIntercomSetLS_NotTrans
- DtIntercomSetLS_Trans
- DtIntercomSignalPduKind
- DtIntercomSimCommChnl
- DtIntercomSimCommSupChnl
- DtIntercomTransmitting
- DtInterfaceFilters
- DtInterfacePropeller
- DtInternalArtParam
- DtIraq
- DtIsGroupOfPduKind
- DtIsPartOfPduKind
- DtIsb

- DtJamBeam
- DtKC130
- DtLE_AppearanceAlternativeType
- DtLE_AppearanceAudio
- DtLE_AppearanceCapabilities
- DtLE_AppearanceEM
- DtLE_AppearanceFlagField2
- DtLE_AppearanceForceID
- DtLE_AppearanceIR
- DtLE_AppearanceMarking
- DtLE_AppearanceType
- DtLE_AppearanceVisual
- DtLE_DetonBurstQuantityRate
- DtLE_DetonBurstWarheadFuse
- DtLE_DetonEventNumber
- DtLE_DetonEventSiteApp
- DtLE_DetonField2
- DtLE_DetonLocationEntityCoords
- DtLE_DetonMunitionID
- DtLE_DetonMunitionOrientation
- DtLE_DetonMunitionSiteApp
- DtLE_DetonTargetID
- DtLE_FireBurstQuantityRate
- DtLE_FireBurstWarheadFuse
- DtLE_FireEventSiteApp
- DtLE_FireLocation
- DtLE_FireMunitionID
- DtLE_FireMunitionSiteApp
- DtLE_FireTargetID
- DtLE_TSPI_DeadReckoning
- DtLE_TSPI_EntityLinearVelocity
- DtLE_TSPI_EntityOrientation
- DtLE_TSPI_MeasuredSpeed
- DtLE_TSPI_OrientationError
- DtLE_TSPI_PositionError
- DtLE_TSPI_Reserved
- DtLE_TSPI_SystemSpecific
- DtLGR_ALL

- DtLGR_BACKWARDS
- DtLGR_EMPTY
- DtLGR_END
- DtLGR_FORWARDS
- DtLGR_LOOP
- DtLGR_MIDDLE
- DtLGR_MINE
- DtLGR_NOFILE
- DtLGR_NONE
- DtLGR_NOPKT
- DtLGR_PAUSE
- DtLGR_PLAY
- DtLGR_PRISTINE
- DtLGR_RECORD
- DtLGR_RVA
- DtLGR_START
- DtLGR_STOP
- DtLGR_STOPPED
- DtLandingCraft
- DtLandingGear
- DtLarge
- DtLargeTrackedUtilityVehicle
- DtLargeWheeledUtilityVehicle
- DtLaserGeneric
- DtLaserSensorType
- DtLeAppearancePduKind
- DtLeArticulatedPartsPduKind
- DtLeDetonationPduKind
- DtLeFirePduKind
- DtLeTspiPduKind
- DtLeftAileron
- DtLeftFlap
- DtLeftWeaponBayDoor
- DtLevi_Civita
- DtLgrControlPduKind
- DtLgrEnd
- DtLgrEofAction
- DtLgrHeartbeat

- DtLgrPause
- DtLgrPbFile
- DtLgrPlayAction
- DtLgrPlay
- DtLgrQuit
- DtLgrRecFile
- DtLgrRecordAction
- DtLgrRecord
- DtLgrSetTime
- DtLgrSkipTime
- DtLgrStart
- DtLgrStop
- DtLgrTimeScale
- DtLgrTimeout
- DtLifeFormDomainAir
- DtLifeFormDomainLand
- DtLifeFormDomainOther
- DtLifeFormDomainSpace
- DtLifeFormDomainSubsurface
- DtLifeFormDomainSurface
- DtLifeForm
- DtLifeSupportAirSupply
- DtLifeSupportCBRProtection
- DtLifeSupportDecontaminationSys
- DtLifeSupportFilters
- DtLifeSupportRefrigerationSys
- DtLifeSupportWaterSupply
- DtLifeSupportWaterWashSys
- DtLifeformCrawling
- DtLifeformCrouching
- DtLifeformJumping
- DtLifeformKneeling
- DtLifeformNA
- DtLifeformParachuting
- DtLifeformProne
- DtLifeformRunning
- DtLifeformSitting
- DtLifeformSquatting

- DtLifeformStanding
- DtLifeformSwimming
- DtLifeformWading
- DtLifeformWalking
- DtLightAlgae
- DtLightsFormation
- DtLightsNavigation
- DtLightsNone
- DtLightsRunning
- DtLineFormation
- DtLinearObjectStatePduKind
- DtLoader
- DtLowerWord
- DtLsb
- DtM109
- DtM113
- DtM1
- DtM2
- DtM3
- DtM88
- DtMAX_PARAMS_PER_PART
- DtMH53
- DtMH60
- DtMI24
- DtMI28
- DtMOPMigratorPduKind
- DtMTLB
- DtMagneticFuse
- DtMagneticGeneric
- DtMagneticSensorType
- DtMapDatum
- DtMaverick
- DtMaximumSocketNumber *the following is used as the first argument to the PC version of select*
- DtMedium
- DtMessagePduKind
- DtMiG23
- DtMiG29
- DtMineCounterMeasureShip

- DtMineEmplacementAge
- DtMineFusing
- DtMineGroundBurialDepthOffset
- DtMineOrientation
- DtMinePaintScheme
- DtMineReflectance
- DtMineScalarDetectionCoefficient
- DtMineSnowBurialDepthOffset
- DtMineThermalContrast
- DtMineTripDetonationWireLocation
- DtMineWaterBurialDepthOffset
- DtMinefieldDataPduKind
- DtMinefieldQueryPduKind
- DtMinefieldResponseNackPduKind
- DtMinefieldStatePduKind
- DtMk48AntiSubmarine
- DtMobileCommandAndControlEntityCarriedAboardHost
- DtModLocation
- DtModOrientation
- DtModSAF_Artillery_Fire_Control
- DtModSAF_FWA_Forward_Air_Controller
- DtModSAF_Text_Radio
- DtModSAF_Threat_ADA_C3
- DtModerateAlgae
- DtMortar
- DtMultispectralGeneric
- DtMultispectralSensorType
- DtMunitionCarriedAsCargo
- DtMunitionDomainOther
- DtMunition
- DtMunition
- DtMunition
- DtMunitionGuided
- DtMunitionUnguided
- DtNASANAD27
- DtNASBNAD27
- DtNPCollisionPduKind
- DtNPFieldPduKind

- DtNPSurfaceContactPduKind
- DtNassau
- DtNaturalWoodPaintScheme
- DtNet16Bits
- DtNet32Bits
- DtNet32Vector
- DtNet64Bits
- DtNet64Vector
- DtNet8Bits
- DtNetAngularVelocity
- DtNetArtParamRecord
- DtNetArtPartRecord
- DtNetAttachedPart
- DtNetBoolean *class DtNetBoolean: Instances of DtNetBoolean are network representations of booleans*
- DtNetBoolean
- DtNetBurstDescriptor
- DtNetClockTime
- DtNetEntityCoordinates
- DtNetEntityIdentifier
- DtNetEntityType
- DtNetEulerAngles
- DtNetEventId
- DtNetFloat32
- DtNetFloat64
- DtNetGlobalObjectDesignator
- DtNetInt16
- DtNetInt32
- DtNetInt8
- DtNetLinearVelocity
- DtNetObjectId
- DtNetRadioEntityType
- DtNetSimulationAddress
- DtNetTopoUpCoord
- DtNetU16
- DtNetU32
- DtNetU8
- DtNetWorldCoordinates
- DtNoAlgae

- DtNoAntiHandlingDevice
- DtNoAttachedParts *This constant can be passed to a VR-Link function that expects a list of attached part stations to indicate an empty list*
- DtNoFuse
- DtNoRepairsPerformed
- DtNoSuchAttachStation
- DtNoWay
- DtNoWeapon
- DtNorth
- DtNotTrackingAllTargets
- DtNotifyLevel
- DtOH58
- DtObjectCulturalFeature
- DtObjectIdHashList
- DtObstacleBreach
- DtObstacle
- DtObstacleMarker
- DtOff
- DtOlivePaintScheme
- DtOmniDirectional
- DtOn
- DtOnStation_RNG_BRG
- DtOnStation_XYZ
- DtOnTopOf
- DtOpticalSensorType
- DtOtherAggregateFormation
- DtOtherAggregateState
- DtOther
- DtOtherFuse
- DtOtherNature
- DtOtherObjectKind
- DtOtherPaintScheme
- DtOtherPduKind
- DtOtherPosition
- DtOtherSensorTypeCategory
- DtOtherStationName
- DtOtherTransfer
- DtPartAttached
- DtPartNotAttached

- DtPartiallyDisaggregated
- DtPassageway
- DtPatriot
- DtPatrolCraft
- DtPeleliu
- DtPeriscope
- DtPhoenix
- DtPhysicalSensorType
- DtPilot
- DtPlatformDomainAir
- DtPlatformDomainLand
- DtPlatformDomainOther
- DtPlatformDomainSpace
- DtPlatformDomainSubsurface
- DtPlatformDomainSurface
- DtPlatform
- DtPlatoon
- DtPm
- DtPointObjectStatePduKind
- DtPreRecordedVoice
- DtPreparedPosition
- DtPressureFuse
- DtPrimaryDefenseSys10
- DtPrimaryDefenseSys1
- DtPrimaryDefenseSys2
- DtPrimaryDefenseSys3
- DtPrimaryDefenseSys4
- DtPrimaryDefenseSys5
- DtPrimaryDefenseSys6
- DtPrimaryDefenseSys7
- DtPrimaryDefenseSys8
- DtPrimaryDefenseSys9
- DtPrimaryGun10
- DtPrimaryGun1
- DtPrimaryGun2
- DtPrimaryGun3
- DtPrimaryGun4
- DtPrimaryGun5

- DtPrimaryGun6
- DtPrimaryGun7
- DtPrimaryGun8
- DtPrimaryGun9
- DtPrimaryLauncher10
- DtPrimaryLauncher1
- DtPrimaryLauncher2
- DtPrimaryLauncher3
- DtPrimaryLauncher4
- DtPrimaryLauncher5
- DtPrimaryLauncher6
- DtPrimaryLauncher7
- DtPrimaryLauncher8
- DtPrimaryLauncher9
- DtPrimaryRadar10
- DtPrimaryRadar1
- DtPrimaryRadar2
- DtPrimaryRadar3
- DtPrimaryRadar4
- DtPrimaryRadar5
- DtPrimaryRadar6
- DtPrimaryRadar7
- DtPrimaryRadar8
- DtPrimaryRadar9
- DtPrimaryTurret10
- DtPrimaryTurret1
- DtPrimaryTurret2
- DtPrimaryTurret3
- DtPrimaryTurret4
- DtPrimaryTurret5
- DtPrimaryTurret6
- DtPrimaryTurret7
- DtPrimaryTurret8
- DtPrimaryTurret9
- DtProbeMetalContent
- DtProbeNoMetalContent
- DtProtocolVersionToRecvMax
- DtProtocolVersionToRecvMin

- DtProtocolVersionToSend
- DtPseudoDisaggregated
- DtPseudoNoise
- DtPseudoNoiseFrequencyHopping
- DtPseudoNoiseFrequencyHoppingTimeHopping
- DtPseudoNoiseTimeHopping
- DtPulse
- DtQ1
- DtQ2
- DtQ3
- DtQRP_Mode
- DtQuantity
- DtQuantity
- DtQuantity
- DtQuaternion
- DtRADAR_Generic
- DtRADAR_SensorType
- DtRAH66
- DtRadarBeam
- DtRadioCryptoSysKY_28
- DtRadioCryptoSysKY_8
- DtRadioCryptoSysNarrowSpectrumSecureVoice_NSVE
- DtRadioCryptoSysOther
- DtRadioCryptoSysSINCGARS_ICOM
- DtRadioCryptoSysWideSpectrumSecureVoice_WSVE
- DtRadioDetailModCombo
- DtRadioDetailModOther
- DtRadioDetailModUnmod
- DtRadioDetailedModulation
- DtRadioInputOther
- DtRadioMajorModOther
- DtRadioModAmpAngOther
- DtRadioModeAngleOther
- DtRadioOff
- DtRadioOn
- DtRadioOnNotReceiving
- DtRadioOnNotTrasmitting
- DtRadioSignalEncoding

- DtRadioSpreadOther
- DtRadioSystemOther
- DtRate
- DtRate
- DtRate
- DtRawBinary
- DtReal
- DtReceive
- DtReceiverPduKind
- DtReconnaisance
- DtRecordPduKind
- DtRecordQueryPduKind
- DtRecoveryCap
- DtRedPaintScheme
- DtReferenceSysEntityCoordinates
- DtReferenceSysWorldCoordinates
- DtRegiment
- DtRegularBeam
- DtReliable
- DtRemoveEntityPduKind
- DtRemoveEntityRPduKind
- DtRemoveObjectResultOther
- DtRepairCap
- DtRepairCompletePduKind
- DtRepairEverything
- DtRepairResponsePduKind
- DtRepairResultCancelled
- DtRepairResultEnded
- DtRepairResultInterrupted
- DtRepairResultInvalid
- DtRepairResultOther
- DtReqExchangeEntity
- DtReqExchangeEnvProc
- DtRequestStatusComplete
- DtRequestStatusExecuting
- DtRequestStatusInvalidTimeParameters
- DtRequestStatusOther
- DtRequestStatusPartial

- DtRequestStatusPending
- DtRequestStatusRejected
- DtRequestStatusRequestDone
- DtRequestStatusRetransmitLater
- DtRequestStatusRetransmitNow
- DtRequestStatusSimulationTimeExceeded
- DtResupplyCancelPduKind
- DtResupplyOfferPduKind
- DtResupplyReceivedPduKind
- DtRightAileron
- DtRightFlap
- DtRightWeaponBayDoor
- DtRoland
- DtRudder
- DtSA10FlapLidRadar
- DtSA10Grumble
- DtSA2Guideline
- DtSA3Goa
- DtSA4Ganef
- DtSA6Gainful
- DtSA6Launcher
- DtSA7Grail
- DtSA8Gecko
- DtSA8LauncherRadar
- DtSA9Gaskin
- DtSAMSite
- DtSBBNOhio
- DtSONAR_Generic
- DtSONAR_SensorType
- DtSS1ScudMissileLauncher
- DtSU25
- DtSaipan
- DtSeaGoingHelicopter
- DtSeaSparrow
- DtSec2Hour
- DtSecondaryDefenseSys10
- DtSecondaryDefenseSys1
- DtSecondaryDefenseSys2

- DtSecondaryDefenseSys3
- DtSecondaryDefenseSys4
- DtSecondaryDefenseSys5
- DtSecondaryDefenseSys6
- DtSecondaryDefenseSys7
- DtSecondaryDefenseSys8
- DtSecondaryDefenseSys9
- DtSecondaryGun10
- DtSecondaryGun1
- DtSecondaryGun2
- DtSecondaryGun3
- DtSecondaryGun4
- DtSecondaryGun5
- DtSecondaryGun6
- DtSecondaryGun7
- DtSecondaryGun8
- DtSecondaryGun9
- DtSecondaryLauncher10
- DtSecondaryLauncher1
- DtSecondaryLauncher2
- DtSecondaryLauncher3
- DtSecondaryLauncher4
- DtSecondaryLauncher5
- DtSecondaryLauncher6
- DtSecondaryLauncher7
- DtSecondaryLauncher8
- DtSecondaryLauncher9
- DtSecondaryRadar10
- DtSecondaryRadar1
- DtSecondaryRadar2
- DtSecondaryRadar3
- DtSecondaryRadar4
- DtSecondaryRadar5
- DtSecondaryRadar6
- DtSecondaryRadar7
- DtSecondaryRadar8
- DtSecondaryRadar9
- DtSecondaryTurret10

- DtSecondaryTurret1
- DtSecondaryTurret2
- DtSecondaryTurret3
- DtSecondaryTurret4
- DtSecondaryTurret5
- DtSecondaryTurret6
- DtSecondaryTurret7
- DtSecondaryTurret8
- DtSecondaryTurret9
- DtSelfPropelledArtillery
- DtServiceRequestPduKind
- DtServiceTypeOther
- DtServiceTypeRepair
- DtServiceTypeResupply
- DtSetDataPduKind
- DtSetDataRPduKind
- DtSetRecordPduKind
- DtShipsAftGunmountCenterline
- DtShipsAftGunmountPort
- DtShipsAftGunmountStarboard
- DtShipsForwardGunmountCenterline
- DtShipsForwardGunmountPort
- DtShipsForwardGunmountStarboard
- DtSidewinder
- DtSignalClassApplicationSpecific
- DtSignalClassEncodedVoice
- DtSignalClassPreRecorded
- DtSignalClassRawBinary
- DtSignalPduKind
- DtSimulationAddress
- DtSimulator
- DtSimulator
- DtSincgards
- DtSite
- DtSmall
- DtSmallTrackedUtilityVehicle
- DtSmallWheeledUtilityVehicle
- DtSmoke

- DtSnorkel
- DtSoldier1HourSleep
- DtSoldier2HourSleep
- DtSoldier3HourSleep
- DtSoldier4HourSleep
- DtSoldier5HourSleep
- DtSoldier6HourSleep
- DtSoldier7HourSleep
- DtSoldierFullyRested
- DtSoldierNotRested
- DtSouth
- DtSparrow
- DtSpecialOperations
- DtSpecific
- DtSpecificAH1W
- DtSpecificCV59
- DtSpecific
- DtSpecific
- DtSpecificFFG7
- DtSpecificSA10
- DtSpecificSA8
- DtSpeedBrake
- DtSphericalHarmonic
- DtSpheroid
- DtSquad
- DtSsbFull
- DtSsbReduc
- DtStandardPaintScheme
- DtStartRegCb
- DtStartRegCbList
- DtStartRegsCbInfo
- DtStartResumePduKind
- DtStartResumeRPduKind
- DtStartUpdatesCb
- DtStartUpdatesCbInfo
- DtStartUpdatesCbList
- DtStateUpdate
- DtStinger

- DtStopFreezePduKind
- DtStopFreezeRPduKind
- DtStopReasonEntityReconstitution
- DtStopReasonOther
- DtStopReasonRecess
- DtStopReasonSecurityViolation
- DtStopReasonSystemFailure
- DtStopReasonTermination
- DtStopRegCb
- DtStopRegCbList
- DtStopRegsCbInfo
- DtStopUpdatesCb
- DtStopUpdatesCbInfo
- DtStopUpdatesCbList
- DtStrategic
- DtSubCategory
- DtSubCategory
- DtSubCategory
- DtSubProtocolInvalid
- DtSuppEmisEntStatePduKind
- DtSupply
- DtSupplyFood
- DtSupplyFuels
- DtSupplyLubricants
- DtSupplyOils
- DtSupplyOther
- DtSupplyPersonnel
- DtSupplySpareParts
- DtSureThing
- DtT64
- DtT72
- DtTU26
- DtTU95
- DtTacticalDataLinkOther
- DtTacticalSmoke
- DtTailHook
- DtTanPaintScheme
- DtTankAPCHatch

- DtTank
- DtTankLandingShip
- DtTarawaClass
- DtTarawa
- DtTdlGC3
- DtTeamMemberInFormationWith
- DtText
- DtThomasSGates
- DtThreeIndex *Unobvious Constants*
- DtTiconderogaClass
- DtTiconderoga
- DtTiltRodFuse
- DtTime
- DtTimeHopping
- DtTimeStamp *Host representation of a timeStamp*
- DtTimeStampAbsolute
- DtTimeStampRelative
- DtTimestamp
- DtTowedArtillery
- DtTrackBeam
- DtTrackJamDesignator
- DtTrackJamDesignatorList
- DtTrackedVehicleOccClosedHatchActSearching
- DtTrackedVehicleOccClosedHatchNotActSearching
- DtTrackedVehicleOccOpenHatchActSearching
- DtTrackedVehicleOccOpenHatchNotActSearching
- DtTrackingAllTargets
- DtTrailerBed
- DtTrailingEffectsLarge
- DtTrailingEffectsMedium
- DtTrailingEffectsNone
- DtTrailingEffectsSmall
- DtTransferControlRequestPduKind
- DtTransmitterPduKind
- DtTripWireFuse
- DtTruckBed
- DtTruckOccActivelySearching
- DtTruckOccNotActivelySearching

- DtTrue
- DtType
- DtU16
- DtU32
- DtU8
- DtUAV
- DtUSCivilianCar
- DtUnacknowledged
- DtUnaidedEyeActivelySearching
- DtUnaidedEyeNotActivelySearching
- DtUndefinedGroupEntityCategory
- DtUnderwaterAcousticALFS
- DtUnderwaterAcousticAN_AQS_901
- DtUnderwaterAcousticAN_AQS_902
- DtUnderwaterAcousticAN_BQQ_5
- DtUnderwaterAcousticAN_SQS_23
- DtUnderwaterAcousticAN_SQS_26
- DtUnderwaterAcousticAN_SQS_53
- DtUnderwaterAcousticAN_SSQ_62
- DtUnderwaterAcousticActiveOtherIndex
- DtUnderwaterAcousticBattery
- DtUnderwaterAcousticChangedDataUpdate
- DtUnderwaterAcousticConical
- DtUnderwaterAcousticContinuousSearch
- DtUnderwaterAcousticDiesel
- DtUnderwaterAcousticDiesel_electric
- DtUnderwaterAcousticGasTurbine
- DtUnderwaterAcousticGrahamsMKV_CoffeeMaker
- DtUnderwaterAcousticHelical
- DtUnderwaterAcousticHullMountedMaskerOff
- DtUnderwaterAcousticHullMountedMaskerOn
- DtUnderwaterAcousticLFA
- DtUnderwaterAcousticMineHunting
- DtUnderwaterAcousticNavigation
- DtUnderwaterAcousticNuclear
- DtUnderwaterAcousticOtherAdditionalIndex
- DtUnderwaterAcousticOtherConfig
- DtUnderwaterAcousticOtherFunction

- DtUnderwaterAcousticOtherPassiveIndex
- DtUnderwaterAcousticOtherSystem
- DtUnderwaterAcousticPduKind
- DtUnderwaterAcousticPlatformSearch_detect_track
- DtUnderwaterAcousticRaster
- DtUnderwaterAcousticScanPatternNotUsed
- DtUnderwaterAcousticSectorSearch
- DtUnderwaterAcousticStateUpdate
- DtUnderwaterAcousticSteam
- DtUnderwaterAcousticTurbineReduction
- DtUnderwaterAcousticWeaponSearch_detect_track_detect
- DtUnionOfSovietSocialistRepublics
- DtUnitedKingdom
- DtUnitedStates
- DtUnmannedAirVehicle
- DtUnmodulated
- DtUnusedCharacterSet
- DtUpdatesOffCb
- DtUpdatesOffCbList
- DtUpdatesOnCb
- DtUpdatesOnCbInfo
- DtUpdatesOnCbList
- DtUpperWord
- DtUral375
- DtUsb
- DtUseSimTimeForApproxRealTime
- DtUserGC3
- DtUserProtocolOther
- DtUtilityHelicopter
- DtUtmLatRef1_1 *Origin of the IITSEC '92 terrain database*
- DtUtmLatRefZone10 *The standard UTM reference for a zone is the point along the zone's central meridian at the equator*
- DtUtmLatRef_IITSEC_93 *Origin of the IITSEC '94 & '95 terrain database*
- DtUtmLonRef1_1
- DtUtmLonRefZone10
- DtUtmLonRef_IITSEC_93
- DtVQ_Vector_Quantization
- DtValleyForge
- DtValue

- DtVar
- DtVectorRef
- DtVeeFormation
- DtVehicleCarriedAsCargo
- DtVeryLarge
- DtVerySmall
- DtViewControlPduKind
- DtViewerAbsolute
- DtViewerCompass
- DtViewerGroup
- DtViewerMimic
- DtViewerMimicTrack
- DtViewerModeless
- DtViewerOrbit
- DtViewerTether
- DtViewerTetherTrack
- DtViewerTrack
- DtViewerTurret
- DtVincennes
- DtVsb
- DtWGS84
- DtWNCP_data
- DtWarhead
- DtWarheadBiologicalGeneral
- DtWarheadChemicalBlisterAgent
- DtWarheadChemicalBloodAgent
- DtWarheadChemicalGeneral
- DtWarheadChemicalNerveAgent
- DtWarhead
- DtWarhead
- DtWarheadHighExplosiveAntiTank
- DtWarheadHighExplosiveBomblets
- DtWarheadHighExplosive
- DtWarheadHighExplosiveFragmentation
- DtWarheadHighExplosiveIncendiary
- DtWarheadHighExplosivePlastic
- DtWarheadHighExplosiveShapedCharge
- DtWarheadIllumination

- DtWarheadKinetic
- DtWarheadNuclear
- DtWarheadOther
- DtWarheadPractice
- DtWarheadSmoke
- DtWarheadUnused
- DtWeaponCommandersGPSExtension
- DtWeaponDeployed
- DtWeaponGunElevationDrive
- DtWeaponGunStabilizationSys
- DtWeaponGunnersPrimarySight
- DtWeaponInFirePosition
- DtWeaponLoadingMecahnism
- DtWeaponStowed
- DtWedgeFormation
- DtWellDeck
- DtWest
- DtWhitePaintScheme
- DtWingSweep
- DtWinterCamouflage
- DtX
- DtY
- DtY
- DtYellowPaintScheme
- DtYorktown
- DtZ
- DtZ
- DtZIL157
- DtZSU23_4
- DtkObjectIsDis
- DtkObjectIsHla
- DtkObjectIsUnknown
- Dto_EQ
- Dto_NEQ
- GtQueueElement
- North
- PacketQueueIn
- PacketQueueOut

- RW_flags
- StealthInstruction
- TRANSPOSE_M1_AND_M2
- TRANSPOSE_M1
- TRANSPOSE_M2
- TRANSPOSE_NONE
- Up
- _epsilon
- _fedTime ---------- *Implementation member variables* ----------
- _positiveInfinity
- _zero
- a1
- a2
- addr
- addr
- ap
- app
- attachedType
- bits
- blocking
- boundToDestAddr
- buffer
- bundling
- caddr_t
- **category :**
  - in duplicate
  - in DtNetRadioEntityType
- change
- clientId
- clientSocketPath
- controlMsg
- **country :**
  - in duplicate
  - in DtNetRadioEntityType
- data
- datalen
- datumShift
- destAddr

- destInetAddr
- dfltNumBuckets *Not static member since g++ doesn't like to use static members as default arguments*
- direction
- **domain :**
  - in duplicate
  - in DtNetRadioEntityType
- dp
- dv
- eventNum
- exercise
- exercise
- filterMsg
- firstExId
- firstInBundle
- firstProtocol
- from_addr
- **frozen :**
  - in DtSmoothOrient
  - in DtSmoothPos
- hourFrac
- initErrorAction
- internal
- kind
- kind
- kind
- **lastT :**
  - in DtSmoothOrient
  - in DtSmoothPos
- length
- loopbackAddr
- **lsbyte :**
  - in DtNetInt16
  - in DtNetU16 *Most Significant Byte*
- lslong *Most Significant Long*
- **lsword :**
  - in DtNetInt32 *Most Significant Word*
  - in DtNetU32 *Most Significant Word*
- m_bExternalSocket
- m_devAddr

- maxBundleSize
- maxfd
- min
- **msbyte :**
  - in DtNetInt16
  - in DtNetU16
- mslong
- **msword :**
  - in DtNetInt32
  - in DtNetU32
- myAccel
- myAcceleration
- myAckFlag
- myAction
- myAction *parameter*
- myAction *parameter*
- myAction *parameter*
- myActionId
- myActionResult
- myAggDimThresh
- myAggregateState
- myAlgorithm
- myAlgorithm
- myAngles
- myAntPat
- myAntennaPatternType
- myAppId
- myAppearance
- myApproxFlag
- myApproximator *holds what's currently being used as our approximator*
- myArtPartList
- myArtPartThresh
- myAsSeenByRemote
- myAsSeenByRemote
- myAttPartList
- myAttachedTo
- myAttackerId
- myAttackerId
- myAttributeChangeResult

- myAttrs
- myAttrs
- myAvList
- myAzimuthBandwidth
- myAzimuthCenter *Common to emitter beams and track jam beams*
- myAzimuthSweep
- myBase
- myBase
- myBeamDirection
- myBeamFunction
- myBeamId
- myBeamList *list of beams in this system*
- myBeamList
- myBeamParameterIndex
- myBeamPubs *List of DtEmitterBeamPubs*
- myBeamType
- myBeamsUpdateSystem
- myBeamsUpdateSystems
- myBeginItem
- myBeginSocket
- myBodyToGeocAtTimeOfValidity
- myBodyToGeocDirty
- myBodyToGeoc
- myBodyToLocal
- myBodyToLocal
- myBuckets *Array of buckets (DtLists of DtHashItems)*
- myCallbackFunction
- myCapabilities
- myCbList *myCbList is an array of DtInteraction::maxInteractionClasses() items*
- myChange
- myCheckerList
- myClassDesc
- myClassDesc
- myClassDesc
- myClassDesc
- myClassDesc
- myClassDesc
- myClass

- myClass
- myClass
- myClass
- myClass
- myClassHandle
- myClassNeeded
- myCodeName
- myCollideId
- myCollisionType
- myConn
- myConn
- myConstructedBsr
- myCount
- myCreateObjectResult
- myCreatedDecoder
- myCreatedEncoder
- myCreators *myCreators is an array of DtHlaObject::theMaxObjectClasses items*
- myCreators *myCreators is an array of DtInteraction::theMaxInteractionClasses items*
- myCryptoKeyId
- myCryptoMode
- myCryptoSystem
- myCurrentEventId
- myDRAlgorithm
- myDRBodyToGeoc
- myDRBodyToGeocTime
- myDREuler
- myDREulerTime
- myDRLoc
- myDRPosTime
- myDRVel
- myData
- myData
- myDatabaseIndex
- myDeadReckoner *holds what was most recently passed to useDeadReckoner*
- myDecFact *Current factories and other objects*
- myDecoder
- myDecoder
- myDecoder

- myDecoder
- myDecoderList *myDecoderList is an array of theMaxAttributesPerObject items*
- myDesHashKey
- myDesignatedObject
- myDesignatorCode
- myDesignatorId
- myDesignatorPower
- myDesignatorSpotLocation
- myDesignatorWavelength
- myDestroyFedExFlag
- myDetailedModulation
- myDfltDeadReckoner *Created and deleted by this ESR if needed*
- myDfltDecFact *Default factories and other objects*
- myDfltEncFact
- myDfltInterDecFact
- myDfltInterEncFact
- myDfltInterFact
- myDfltObjFact
- myDfltSmoother
- myDimensions
- myDirectedRotRate
- myDiscoverCbList *myDiscoverCbList is an array of DtHlaObject::theMaxObjectClasses items*
- myDsoCreator
- myDsoDeletor
- myDumRec
- myDuration
- myERP
- myESR
- myESR
- myElevationBeamwidth
- myElevationCenter
- myElevationSweep
- myEmitterFunction
- myEmitterName
- myEmitterNumber
- myEmittingSystemId
- myEmptyList *Just an empty list to return when no class list has been registered for a particular VR-Link class*
- **myEnable :**
    - in DtSmoothOrient

    - in DtSmoothPos
- myEncFact
- myEncoder
- myEncoder
- myEncoder
- myEncoderList *myEncoderList and myCheckerList are arrays of theMaxParametersPerInteraction items*
- myEncodingClass
- myEncodingType
- myEntities
- myEntityAppearanceList
- myEntityCoords
- myEntityId
- myEntityId *embedded system class attributes*
- myEntityId
- myEntityLocation
- myEntityType
- myEntityType
- myEntityType
- myEventId
- myEventId
- myEventId
- myEventId
- myEventId
- myEventNum
- myEventType
- myEverPublished
- myEverPublished
- myEverSubscribed
- myEverSubscribed
- myExConn
- myExConn
- myExConn
- myExConn
- myExConn
- myExConn
- myExConn
- myExConn
- myExConn

- myExConn
- myExConn
- myExConn
- myExConn
- myExConn
- myExConn
- myExConn
- myExConn
- myEx
- myExecName
- myExternalBuffer
- myEz
- myFactory
- myFactory
- myFd
- myFedAmb
- myFedAmb
- myFedFileName
- myFedHandle
- myFedName
- myFederateId
- myFilename *parameter*
- myFilename *parameter*
- myFireMissionIndex
- myFirst
- myFirstParam
- myFixedDatumIDs
- myFom
- myFomMapper
- myForceID
- myForceId
- myForcedAttributes *Requested by a call to setForcedUpdate*
- myFormation
- myFracThresh
- myFreqHi
- myFreqLo
- myFrequencyBandwidth
- myFrequency

- myFrequencyHopInUse
- myFrequencyRange
- myFrozen
- myFunc
- myFunc
- myFunc
- myFunc
- myFunc
- myFunc
- myFunc
- myFunc
- myFunc
- myFuseType
- myFuseType
- myGeocLocForGeocToUtm
- myGeocLocForUtmToGeoc
- myGeocToLocal
- myGeocToUtmMatrix
- myGlobalId
- myGuise
- myHand
- myHandle
- myHandle
- myHandle
- myHandle
- myHandle
- myHandle
- myHandleList
- myHashValue
- myHashValueIsComputed
- myHashlist
- myHashlist
- myHeartbeat *parameter*
- myHighDensity
- myHighestInter
- myHighestObj
- myHlaObj
- myHlaObj

- myHostId
- myHostRadioId
- myHtau
- myId
- myId
- myId
- myId
- myId
- myInputSource
- myIntBuffSize
- myIntMapList *Hashlist that maps names of C++ classes derived from DtInteraction to lists of names of FOM classes*
- myInterChooserList *Hashlist that maps names of C++ classes derived from DtInteraction to DtInteractionClassChoosers - functions that can choose an interaction class name from the FOM to use for the interaction*
- myInterClasses
- myInterDecFact
- myInterEncFact
- myInterFact
- myInterRootName
- myInteractionManager
- myInteractionManager
- myInternalBuff
- myIssueId
- myItem
- myItem
- myJammingModeSequence *For Track Jam Beams*
- myKey
- myKeys *DtList of keys (DtBaseHashKeys)*
- myLast
- myLastParam
- myLastSimTimeUpdateReceived
- myLcPdu
- myList
- myList *Regular DtList containing all of the items added to the hashlist, without regard to key information*
- myList *myList is an array of theMaxParametersPerInteraction items*
- myList *myList is an array of theMaxParametersPerInteraction items*
- myList
- myList
- myList

- myList
- myList
- myList
- myList
- myListIsMine
- myLocAtTimeOfValidity
- myLocalAccel
- myLocalAccel
- myLocalHtau
- myLocalId
- myLocalLoc *used as return values by inspectors*
- myLocalLoc *used as return values by inspectors*
- myLocalObjectManager
- myLocalObjectManager
- myLocalOrient
- myLocalOrient
- **myLocalTau :**
    - in DtSmoothOrient
    - in DtSmoothPos
- myLocalTauInv
- myLocalToGeoc
- myLocalVel
- myLocalVel
- myLocation
- myLocation
- myLowestInter
- myLowestObj
- myMajorModulationType
- myMarkingCharSet
- myMarkingCharSet
- myMarkingText
- myMarkingText *myMarkingText is an array of theMaxMarkingSize*
- myMass
- myModulationParamsAddr
- myModulationParamsLength
- myModulationSystem
- myMsgSize
- myMunitionId
- myMunitionId

- myMunitionType
- myMunitionType
- myName
- myName
- myName
- myName
- myName
- myNeededAttrs *Needed attributes is a per-object concept in RTI1*
- myNeededFlag
- myNetMsg
- myNetRep
- myNetSize
- myNext
- myNext
- myNumArtParams
- myNumArtParts
- myNumBuckets
- myNum
- myNum
- myNumFixed
- myNumParams
- myNumParts
- myNumSupplyTypes
- myNumSupplyTypes
- myNumSupplyTypes
- myNumVar
- myNumberOfAggregates *Data members:*
- myNumberOfAppearanceRecords
- myNumberOfEntities *Data members:*
- myNumberOfSilentAggregateTypes
- myNumberOfSilentEntityTypes
- myNumberOfVariableDatumRecords
- myOKStatus
- myObjChooserList *Hashlist that maps names of C++ classes derived from DtObjectPublisher to DtObjectClassChoosers - functions that can choose an object class name from the FOM to use for the object*
- myObjClasses
- myObjFact
- myObjHandleList *The object returned by objectClassHandles*
- myObjMapList *Hashlist that maps names of C++ classes derived from DtReflectedObjectList to lists of names of*

FOM classes

- myObjName
- myObjRootName
- myObjectAddedCallbacks
- myObjectClass
- myObjectId
- myObjectId
- myObjectId
- myObjectId
- myObjectId
- myObjectList
- myObjectList
- myObjectList
- myObjectList
- myObjectList
- myObjectList
- myObjectRemovedCallbacks
- myObjs
- myOk
- myOk
- myOrientation
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrigEnt
- myOrphanBeams
- myOwnAttrs
- myOwnBsr
- myOwnParams
- myPRF
- myParamNum

- myParams
- myParams
- myPartAttachedTo
- myPartList
- myPartList
- myPartNum
- myPartType
- myPhase
- myPhi *about intermediate y*
- myPktBuff
- myPostDrainCbs
- myPower
- myPrev
- myProtos *myProtos is an array of DtInteraction::maxInteractionClasses() items*
- myProtos *myProtos is an array of DtInteraction::maxInteractionClasses() items*
- myProtos *myProtos is an array of DtHlaObject::maxObjectClasses() items*
- myProtos *myProtos is an array of DtHlaObject::maxObjectClasses() items*
- myPsi
- myPsuedoNoiseSpectrumInUse
- myPubAttrs
- myPublished
- myPulseWidth
- myQuantity
- myQuantity
- myRadioEntityType
- myRange
- myRate
- myRate
- myRavCbList
- myRealTime
- myRealTime
- myReason
- myRecSet
- myReceiveEnt
- myReceiveEnt
- myReceiveEnt
- myReceiveEnt
- myReceiveEnt

- myReceiveEnt
- myReceiveEnt
- myReceiveEnt
- myReceiveEnt
- myReceiveEnt
- myReceiveEnt
- myReceiveEnt
- myReceivedPower
- myReceiverId
- myReceiverRadioNumber
- myReceiverState
- myReceivingObject
- myReceivingObject
- myReceivingObject
- myReceivingObject
- myReceivingObject
- myRecordBuffer
- myRecordSet
- myRecordSet
- myRecordSet
- myReferenceSystem
- myReflectCbList
- myReflect
- myReflectedObjectManager
- myReflectedObjectManager
- myReflecting
- myReflecting
- myRelOrAbs
- myRelativePosition
- myRelativeSpotLocation
- myRemoteSites *DtList of DtInetAddrs (which do not need to be deleted)*
- myRemoveCbList
- myRemoveObjectResult
- myRep
- myRep
- myRep
- myRep
- myRepairResult

- myRepairType
- myRepairingObject
- myRepairingObject
- myRequestCbRegistered
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestId
- myRequestStatus
- myRequestedAttributes *Requested by other federates*
- myRequestingObject
- myRequiredAttributes *Union of forced and requested - not computed until needed*
- myRespFlag
- myResult
- myRotThresh
- myRotThresh
- myRotVel
- myRotationalVelocity
- myRtiAmb
- myRtiAmb
- myRtiAmb
- myRtiAmb
- myRtiAmb
- myRtiAmb
- myRunClock
- mySampleCount
- mySampleRate
- myScale *parameter*

- mySenderId *parameter*
- myServiceType
- myServicingObject
- mySignalData *bits*
- mySignalDataLength
- mySilentAggregates
- mySilentEntities
- mySimTime
- mySimTimeOrientDataValid
- mySimTimePosDataValid
- mySimpleInterChooserList *Hashlist that maps names of C++ classes derived from DtInteraction to single FOM object class names*
- mySimpleObjChooserList *Hashlist that maps names of C++ classes derived from DtObjectPublisher to single FOM object class names*
- mySiteId
- mySizeLength
- mySizeOffset
- mySmoothCapable
- mySmoothCapable
- mySmoother *holds what was most recently passed to useSmoother*
- mySockList
- mySockList
- mySocketMgr
- mySockfd
- mySorient
- mySpecList
- mySpos
- mySpotLinearAcceleration
- myStartIntCbList
- myStartOffset
- myStartRegCbList
- myStateRep
- myStateRep
- myStateRep
- myStateUpdateCallbacks
- myStation
- myStationList *zero terminated list that should always be of length myNumParts + 1*
- myStopIntCbList
- myStopRegCbList

- myStr
- myStr
- myStr
- myString
- myString
- mySubAggs
- mySubAttrs
- mySubscribed
- mySupplies
- mySupplies
- mySupplies
- mySupplyingObject
- mySupplyingObject
- mySupplyingObject
- mySweepSync
- mySystemId
- myTDLMessageCount
- myTacticalDataLink
- myTag
- myTag
- myTargetId
- myTargetId
- **myTau :**
    - in DtSmoothOrient
    - in DtSmoothPos
- myTauInv
- myTheta *about reference z*
- myThreshMode
- myThresholder
- myTime
- myTime *parameter*
- myTime *parameter*
- myTime
- myTimeForSets
- myTimeHopInUse
- myTimeInterval
- myTimeSet
- myTimeStampType
- myTimeType

- myTimeType
- myTimedOutObjectList
- myTimeout *parameter*
- myTimeoutInterval
- myTimeoutProcessingFlag
- myTraceRotThresh
- myTraceRotThresh
- myTrackJamTargets
- myTransThresh
- myTransThresh
- myTransmitId
- myTransmitState
- myTransmitterRadioNumber
- myType
- myTypeList *NULL if myNumArtParts = 0, otherwise has length = myNumParts*
- myUpdate
- myUpdatesOffCbList
- myUpdatesOnCbList
- myUpdating
- myUseFomMapperDeletor
- myUserData
- myUserProtocolId
- myUsingDfltAggDim
- myUsingDfltFrac
- myUsingDfltMode
- myUsingDfltRot
- myUsingDfltRot
- myUsingDfltTrans
- myUsingDfltTrans
- myUsr
- myUsr
- myUsr
- myUsr
- myUsr
- myUsr
- myUsr
- myUsr
- myUsr

- myUtmToGeocMatrix
- myVals *myVals is an array of DtMAX_PARAMS_PER_PART + 1 items which are themselves pointers*
- myVariableDatumIDs
- myVariableDatumRecords
- myVcPdu
- myVector
- myVelAtTimeOfValidity
- myVelocity
- myVelocity
- myVelocity
- myVelocity
- myVersion
- myWarheadType
- myWarheadType
- myWorldLocation
- myWorldLocation
- n *radians per sec*
- needT1data
- needT2data
- netSock
- nextPdu
- **nomenclature :**
  - in duplicate
  - in DtNetRadioEntityType
- **nomenclatureVersion :**
  - in duplicate
  - in DtNetRadioEntityType
- notifyPending
- o0
- objectId
- ofd
- **offset :**
  - in DtCoordTransform *these were protected, but it is often more efficient to set these by passing them to another function*
  - in DtCoordTranslation
- ok
- ok
- optarg
- opterr

- optind
- os
- outBuf
- outBuff
- outSize
- p0
- p1
- param
- paramType
- params
- partAttachedTo
- pdu
- **phi :**
    - in duplicate
    - in DtNetEulerAngles *Counter clockwise about y*
- pktBuff8
- pktBuff8
- pkts_rcvd
- ps
- psMsgTimeout
- psi
- read0
- read
- reader
- recvFd
- rep
- **rep :**
    - in DtNet32Vector
    - in DtNet64Vector
- retId
- rotation *"to" origin in base coords*
- sec
- semiMinor
- sendFd
- serverPort
- serverSockfd
- shmAddr
- site
- sizeLeft

- skt_num
- sourceAddr
- **t0 :**
    - in DtSmoothOrient
    - in DtSmoothPos
- t1
- **t2 :**
    - in DtSmoothOrient
    - in DtSmoothPos
- testOp
- **theAngles :**
    - in DtAbsoluteVCPdu
    - in DtMimicVCPdu
    - in DtTetherVCPdu
- theAnyInstancesFlag
- theClassName
- theClassName
- theClassName
- theClassName
- theClassName
- theClassName
- theClassName
- theClassName
- theClassName
- theClassName
- theClassName
- theDecBuffer
- theDecBuffer
- theDfltAggDimThresh
- theDfltFracThresh
- theDfltHtau
- theDfltRotThresh
- theDfltRotThresh
- theDfltSmoothPeriod
- **theDfltTau :**
    - in DtSmoothOrient
    - in DtSmoothPos
- theDfltTauInv
- theDfltThreshMode

- theDfltTraceRotThresh
- theDfltTraceRotThresh
- theDfltTransThresh
- theDfltTransThresh
- theFedAmbCreator
- theFomMapInitData
- theFomMapperCreator
- theHourFracPerHour
- theHourPerHourFrac
- **theId :**
    - in DtCompassVCPdu
    - in DtGroupVCPdu
    - in DtMimicTrackVCPdu
    - in DtMimicVCPdu
    - in DtModelessVCPdu
    - in DtOrbitVCPdu
    - in DtTetherTrackVCPdu
    - in DtTetherVCPdu
    - in DtTrackVCPdu
- **theMasterSmoothEnable :**
    - in DtSmoothOrient
    - in DtSmoothPos
- theMaxAttributesPerObject
- theMaxAttributesPerObject
- theMaxAttrs
- theMaxClasses
- theMaxInsertionStringSize
- theMaxInteractionClasses
- theMaxMarkingSize
- theMaxObjectClasses
- theMaxParametersPerInteraction
- theMaxParametersPerInteraction
- theMaxParams
- theNumBucketsDefault
- thePvList
- thePvListMaxSize
- theRequestClassUpdateFlag
- theRequestObjectUpdateFlag
- theRtiAmbCreator

- theSeparator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theStateRepCreator
- theTagIsAsciiTime
- theTagIsAsciiTime
- **theVec :**
    - in DtAbsoluteVCPdu
    - in DtMimicTrackVCPdu
    - in DtMimicVCPdu
    - in DtTetherTrackVCPdu
    - in DtTetherVCPdu
    - in DtTrackVCPdu
- theta
- thetaDot
- theta *Counter clockwise about z*
- udpPort
- unbundleBuff8
- unbundling
- **usr :**
    - in DtSmoothOrient
    - in DtSmoothPos
- v0
- v1
- value
- values
- version
- version

- vs
- ws
- wy
- wz

Macros

- ACK_INTERACTION_H_
- ACTION_REQUEST_INTERACTION_H_
- ACTION_RESPONSE_H_
- ADD *Generic Vector Sum :Ans=A+s*B*
- AGGREGATE_PUB_H_
- AGGREGATE_SR_MARKING_TEXT_LENGTH
- COLLISION_H_
- COMMENT_H_
- CREATEENTINTER_H_
- DATA_H_
- DATA_QUERY_H_
- DETONATION_INTERACTION_H_
- DT_CANT_CAST_TO_CONST_2DARRAY
- DT_CANT_HAVE_NET_UNIONS
- DT_FD_ZERO_USES_MEMSET
- DT_FUNCTION_MUST_RETURN
- DT_HAVE_AINT
- DT_HAVE_ARPA_INET_H
- DT_HAVE_ATT_SELECT *Both of these next two should NOT be true (but both MAY be false)*
- DT_HAVE_BSD_SELECT
- DT_HAVE_BZERO_PROTO_IN_BSTRING_H
- DT_HAVE_BZERO_PROTO_IN_CC_LIBC_H
- DT_HAVE_BZERO_PROTO_IN_MEMORY_H
- DT_HAVE_COPYSIGN
- DT_HAVE_CPP_STRINGIZE
- DT_HAVE_DAEMON
- DT_HAVE_DRAND_48
- DT_HAVE_FILE_H
- DT_HAVE_GETHOSTID
- DT_HAVE_GETHOSTID_PROTO
- DT_HAVE_GETOPT_H
- DT_HAVE_GETTIMEOFDAY

- DT_HAVE_GETTIMEOFDAY_PROTO
- DT_HAVE_MESSAGE_SOCKETS
- DT_HAVE_NET_IF_H
- DT_HAVE_NUMERIC_CONSTANTS
- DT_HAVE_PARAM_H
- DT_HAVE_PROCESS_H
- DT_HAVE_SELECT_PROTO
- DT_HAVE_SEMUN
- DT_HAVE_SOCKET_PROTOS
- DT_HAVE_SOCKIO_H
- DT_HAVE_STD_H
- DT_HAVE_STRTOUL
- DT_HAVE_SYSV_IPC
- DT_HAVE_SYSV_IPC_PROTOS
- DT_HAVE_SYS_UIO_H
- DT_HAVE_UNISTD_H *these were added while doing the MSVC Dos port*
- DT_HAVE_WRITEV_PROTO
- DT_LITTLE_ENDIAN
- DT_NEED_DLL_DECLARATIONS *If building a DLL MachineTypes*
- DT_NO_CONSTPTRPTR
- DT_NO_MULTICAST
- DT_NO_RAND
- DT_NO_RANDOM
- DtABS
- DtADD_ATTR_CHECKER
- DtADD_ATTR_CHECKER_QUIET
- DtADD_ATTR_DECODER *This macro is used to add attribute decoder functions (functions that decode individual attributes) to a DtHlaStateDecoder*
- DtADD_ATTR_DECODER_QUIET *Same as above, but don't complain if name is not found*
- DtADD_ATTR_ENCODER *This macro is used to add attribute encoder functions (functions that encode individual attributes) to a DtHlaStateEncoder*
- DtADD_ATTR_ENCODER_QUIET
- DtADD_ENUM_STRING
- DtADD_PARAM_DECODER *This can be used to add a parameter decoder function from a routine*
- DtATOI
- DtATOI *some versions of strtol returns INT_MAX for input larger than 0x7fffffff*
- DtAbs *Returns the absolute value of x*
- DtAbsNotifyObserver_H_
- DtAcknowledgeDecoder_H_

- DtAcknowledgeEncoder_H_
- DtAcknowledgeInteraction_H_
- DtAcknowledgePdu_H_ *This file provides protocol independence for Acknowledge Interactions*
- DtAcknowledgeRPdu_H_
- DtActionRequestDecoder_H_
- DtActionRequestEncoder_H_
- DtActionRequestInteraction_H_
- DtActionRequestPdu_H_
- DtActionRequestRPdu_H_ *This file provides protocol independence for ActionRequest Interactions*
- DtActionRequestToObjDecoder_H_
- DtActionRequestToObjEncoder_H_
- DtActionRequestToObjInter_H_
- DtActionResponseDecoder_H_
- DtActionResponseEncoder_H_
- DtActionResponseFromObjDecoder_H_
- DtActionResponseFromObjEncoder_H_
- DtActionResponseFromObjInter_H_
- DtActionResponseInteraction_H_
- DtActionResponseRPdu_H_ *This file provides protocol independence for ActionResponse Interactions*
- DtAggregateEntityDecoder_H_
- DtAggregateEntityEncoder_H_
- DtAggregatePublisher_H_
- DtAggregateStateDecoder_H_
- DtAggregateStateEncoder_H_
- DtAggregateStatePdu_H_
- DtAggregateStateRepository_H_ *This file provides protocol independence for local aggregates*
- DtAntennaPatternRecord_H_
- DtAny
- DtAppSpecificRadioSignalDecoder_H_
- DtAppSpecificRadioSignalEncoder_H_
- DtApproximator_H_
- DtArealObjectStatePdu_H_
- DtArtPartList_H_
- DtArtPartSpec_H_
- DtArtPartThresholder_H_
- DtArtPart_H_
- DtAttachedPartList_H_
- DtAttributeChangeReqDec_H_

- DtAttributeChangeReqEnc_H_
- DtAttributeChangeReqInter_H_
- DtAttributeChangeResDec_H_
- DtAttributeChangeResEnc_H_
- DtAttributeChangeResInter_H_
- DtBaseEmbeddedSystemDecoder_H_
- DtBaseEmbeddedSystemEncoder_H_
- DtBaseEmitterBeamDecoder_H_
- DtBaseEmitterBeamEncoder_H_
- DtBaseEntityDecoder_H_
- DtBaseEntityEncoder_H_
- DtBaseEntityStateRepository_H_
- DtBaseExerciseConn_H_
- DtBaseHashKey_H_
- DtBaseMsg_H_
- DtBaseReflectedAggregate_H_
- DtBaseReflectedEntity_H_
- DtBasicFixedWingAircraftRecord_H_
- DtBasicGroundCombatSoldierRecord_H_
- DtBasicGroundCombatVehicleRecord_H_
- DtBasicRotorWingAircraftRecord_H_
- DtBeamAntennaPatternRecord_H_
- DtBindToDestAddr *setting this bit in the flags constructor argument will cause the NetSocket to ignore all traffic (incl*
- DtCartesianView_H_
- DtClark1866Spheroid
- DtCollisionDecoder_H_
- DtCollisionElasticPdu_H_
- DtCollisionEncoder_H_
- DtCollisionInteraction_H_
- DtCommentDecoder_H_
- DtCommentEncoder_H_
- DtCommentInteraction_H_
- DtCommentPdu_H_ *This file provides protocol independence for Comment Interactions*
- DtCommentRPdu_H_
- DtCreateEntityDecoder_H_
- DtCreateEntityEncoder_H_
- DtCreateEntityInteraction_H_
- DtCreateEntityPdu_H_ *This file provides protocol independence for CreateEntity Interactions*

- DtCreateEntityRPdu_H_
- DtCreateObjReqDecoder_H_
- DtCreateObjReqEncoder_H_
- DtCreateObjReqInter_H_
- DtCreateObjectResultDecoder_H_
- DtCreateObjectResultEncoder_H_
- DtCreateObjectResultInter_H_
- DtCsFlagsDefault
- DtCsQDefaultSize
- DtDATUM_BAD_ID
- DtDATUM_BAD_INDEX
- DtDATUM_BAD_LENGTH
- DtDATUM_VALID
- DtDECLARE_ACK_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_ACK_PARAM_ENCODER
- DtDECLARE_ACTREQOBJ_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_ACTREQOBJ_PARAM_ENCODER
- DtDECLARE_ACTREQ_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_ACTREQ_PARAM_ENCODER
- DtDECLARE_ACTRESPOBJ_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_ACTRESP_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_ACTRESP_PARAM_ENCODER
- DtDECLARE_ACTRSPOBJ_PARAM_ENCODER
- DtDECLARE_AGGREGATE_ATTR_CHECKER
- DtDECLARE_AGGREGATE_ATTR_DECODER
- DtDECLARE_AGGREGATE_ATTR_ENCODER
- DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER
- DtDECLARE_ATTCHGREQ_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_ATTCHGREQ_PARAM_ENCODER
- DtDECLARE_ATTCHGRES_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_ATTCHGRES_PARAM_ENCODER
- DtDECLARE_ATTR_CHECKER

- DtDECLARE_ATTR_DECODER *This macro is used to declare attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes) to a DtHlaStateDecoder*
- DtDECLARE_ATTR_ENCODER
- DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER
- DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER
- DtDECLARE_BASE_ENTITY_ATTR_CHECKER
- DtDECLARE_BASE_ENTITY_ATTR_DECODER
- DtDECLARE_BASE_ENTITY_ATTR_ENCODER
- DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER
- DtDECLARE_COLLISION_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_COLLISION_PARAM_ENCODER
- DtDECLARE_COMMENT_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_COMMENT_PARAM_ENCODER
- DtDECLARE_CREATENT_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_CREATENT_PARAM_ENCODER
- DtDECLARE_CREATOBJRES_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_CRTOBJREQ_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_CRTOBJREQ_PARAM_ENCODER
- DtDECLARE_CRTOBJRES_PARAM_ENCODER
- DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER
- DtDECLARE_DATAQ_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_DATAQ_PARAM_ENCODER
- DtDECLARE_DATA_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_DATA_PARAM_ENCODER
- DtDECLARE_DESIGNATOR_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_DESIGNATOR_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_DESIGNATOR_ATTR_ENCODER

- DtDECLARE_DETONATION_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_DETONATION_PARAM_ENCODER
- DtDECLARE_EMBEDSYS_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_EMITTERBEAM_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_EMITTERSYS_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_EMITTERSYS_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_EMITTERSYS_ATTR_ENCODER
- DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER
- DtDECLARE_ENTITY_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_ENTITY_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_ENTITY_ATTR_ENCODER
- DtDECLARE_EVENTREP_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_EVENTREP_PARAM_ENCODER
- DtDECLARE_FIRE_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_FIRE_PARAM_ENCODER
- DtDECLARE_JAMMERBEAM_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_JAMMERBEAM_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_JAMMERBEAM_ATTR_ENCODER
- DtDECLARE_LGR_CONTROL_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_LGR_CONTROL_PARAM_ENCODER
- DtDECLARE_PARAM_DECODER
- DtDECLARE_RADARBEAM_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_RADARBEAM_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_RADARBEAM_ATTR_ENCODER
- DtDECLARE_RADIORECVR_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_RADIORECVR_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_RADIORECVR_ATTR_ENCODER

- DtDECLARE_RADIOXMIT_ATTR_CHECKER *This is used to declare static member functions for determining whether or not an attribute needs to be encoded*
- DtDECLARE_RADIOXMIT_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_RADIOXMIT_ATTR_ENCODER
- DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER
- DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_REMOBJREQ_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_REMOBJREQ_PARAM_ENCODER
- DtDECLARE_REMOBJRES_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_REMOBJRES_PARAM_ENCODER
- DtDECLARE_REMOVENT_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_REMOVENT_PARAM_ENCODER
- DtDECLARE_REPCOMP_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_REPCOMP_PARAM_ENCODER
- DtDECLARE_REPRESP_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_REPRESP_PARAM_ENCODER
- DtDECLARE_RESCANC_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_RESCANC_PARAM_ENCODER
- DtDECLARE_RESOFF_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_RESOFF_PARAM_ENCODER
- DtDECLARE_RESREC_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_RESREC_PARAM_ENCODER
- DtDECLARE_SERVREQ_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_SERVREQ_PARAM_ENCODER
- DtDECLARE_SETDATA_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_SETDATA_PARAM_ENCODER
- DtDECLARE_START_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_START_PARAM_ENCODER
- DtDECLARE_STOP_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_STOP_PARAM_ENCODER

- DtDECLARE_TRKJAMBEAM_ATTR_DECODER *This is used to declare attribute decoding functions which are members of the class*
- DtDECLARE_VIEW_CONTROL_PARAM_DECODER *This can be used to declare parameter processor functions which are static members of a class*
- DtDECLARE_VIEW_CONTROL_PARAM_ENCODER
- DtDEFAULT_PORT
- DtDEFINE_SIMPLE_ACK_PARAM_DECODER
- DtDEFINE_SIMPLE_ACK_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_ACK_PARAM_ENCODER
- DtDEFINE_SIMPLE_ACTREQOBJ_PARAM_DECODER
- DtDEFINE_SIMPLE_ACTREQOBJ_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_ACTREQOBJ_PARAM_ENCODER
- DtDEFINE_SIMPLE_ACTREQ_PARAM_DECODER
- DtDEFINE_SIMPLE_ACTREQ_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_ACTREQ_PARAM_ENCODER
- DtDEFINE_SIMPLE_ACTRESPOBJ_PARAM_DECODER
- DtDEFINE_SIMPLE_ACTRESPOBJ_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_ACTRESP_PARAM_DECODER
- DtDEFINE_SIMPLE_ACTRESP_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_ACTRESP_PARAM_ENCODER
- DtDEFINE_SIMPLE_ACTRSPOBJ_PARAM_ENCODER
- DtDEFINE_SIMPLE_AGGREGATE_ATTR_DECODER
- DtDEFINE_SIMPLE_AGGREGATE_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_AGG_ENTITY_ATTR_CHECKER
- DtDEFINE_SIMPLE_AGG_ENTITY_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER
- DtDEFINE_SIMPLE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER
- DtDEFINE_SIMPLE_ATTCHGREQ_PARAM_DECODER
- DtDEFINE_SIMPLE_ATTCHGREQ_PARAM_ENCODER
- DtDEFINE_SIMPLE_ATTCHGRES_PARAM_DECODER
- DtDEFINE_SIMPLE_ATTCHGRES_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_ATTCHGRES_PARAM_ENCODER
- DtDEFINE_SIMPLE_ATTR_CHECKER *This macro is used to define simple attribute checker functions (members of a derived DtHlaStateEncoder that check whether a particular attribute is out of date*
- DtDEFINE_SIMPLE_ATTR_DECODER
- DtDEFINE_SIMPLE_ATTR_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_ATTR_ENCODER

- DtDEFINE_SIMPLE_BASE_EMBEDSYS_ATTR_CHECKER
- DtDEFINE_SIMPLE_BASE_EMBEDSYS_ATTR_DECODER
- DtDEFINE_SIMPLE_BASE_EMBEDSYS_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_BASE_EMBEDSYS_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_BASE_EMITTERBEAM_ATTR_CHECKER
- DtDEFINE_SIMPLE_BASE_EMITTERBEAM_ATTR_DECODER
- DtDEFINE_SIMPLE_BASE_EMITTERBEAM_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_BASE_EMITTERBEAM_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_BASE_ENTITY_ATTR_CHECKER
- DtDEFINE_SIMPLE_BASE_ENTITY_ATTR_DECODER
- DtDEFINE_SIMPLE_BASE_ENTITY_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_BASE_ENTITY_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_BASE_JAMMERBEAM_ATTR_DECODER
- DtDEFINE_SIMPLE_BASE_JAMMERBEAM_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_BASE_TRKJAMBEAM_ATTR_CHECKER
- DtDEFINE_SIMPLE_BASE_TRKJAMBEAM_ATTR_DECODER
- DtDEFINE_SIMPLE_BASE_TRKJAMBEAM_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_BASE_TRKJAMBEAM_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_COLLISION_PARAM_DECODER
- DtDEFINE_SIMPLE_COLLISION_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_COLLISION_PARAM_ENCODER
- DtDEFINE_SIMPLE_COMMENT_PARAM_DECODER
- DtDEFINE_SIMPLE_COMMENT_PARAM_ENCODER
- DtDEFINE_SIMPLE_CREATENT_PARAM_DECODER
- DtDEFINE_SIMPLE_CREATENT_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_CREATENT_PARAM_ENCODER
- DtDEFINE_SIMPLE_CREATOBJRES_PARAM_DECODER
- DtDEFINE_SIMPLE_CREATOBJRES_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_CRTOBJREQ_PARAM_DECODER
- DtDEFINE_SIMPLE_CRTOBJREQ_PARAM_ENCODER
- DtDEFINE_SIMPLE_CRTOBJRES_PARAM_ENCODER
- DtDEFINE_SIMPLE_DATABASE_RADIO_SIGNAL_PARAM_DECODER
- DtDEFINE_SIMPLE_DATABASE_RADIO_SIGNAL_PARAM_DECODER_WITH_CAST

- DtDEFINE_SIMPLE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER
- DtDEFINE_SIMPLE_DATAQ_PARAM_DECODER
- DtDEFINE_SIMPLE_DATAQ_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_DATAQ_PARAM_ENCODER
- DtDEFINE_SIMPLE_DATA_PARAM_DECODER
- DtDEFINE_SIMPLE_DATA_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_DATA_PARAM_ENCODER
- DtDEFINE_SIMPLE_DESIGNATOR_ATTR_CHECKER
- DtDEFINE_SIMPLE_DESIGNATOR_ATTR_DECODER
- DtDEFINE_SIMPLE_DESIGNATOR_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_DESIGNATOR_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_DETONATION_PARAM_DECODER
- DtDEFINE_SIMPLE_DETONATION_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_DETONATION_PARAM_ENCODER
- DtDEFINE_SIMPLE_EMITTERSYS_ATTR_CHECKER
- DtDEFINE_SIMPLE_EMITTERSYS_ATTR_DECODER
- DtDEFINE_SIMPLE_EMITTERSYS_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_EMITTERSYS_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER
- DtDEFINE_SIMPLE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER
- DtDEFINE_SIMPLE_ENVENT_ATTR_CHECKER
- DtDEFINE_SIMPLE_ENVENT_ATTR_DECODER
- DtDEFINE_SIMPLE_ENVENT_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_ENVENT_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_EVENTREP_PARAM_DECODER
- DtDEFINE_SIMPLE_EVENTREP_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_EVENTREP_PARAM_ENCODER
- DtDEFINE_SIMPLE_FIRE_PARAM_DECODER
- DtDEFINE_SIMPLE_FIRE_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_FIRE_PARAM_ENCODER
- DtDEFINE_SIMPLE_JAMMERBEAM_ATTR_CHECKER
- DtDEFINE_SIMPLE_JAMMERBEAM_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_LGR_CONTROL_PARAM_ENCODER

- DtDEFINE_SIMPLE_MIL_ENTITY_ATTR_CHECKER
- DtDEFINE_SIMPLE_MIL_ENTITY_ATTR_DECODER
- DtDEFINE_SIMPLE_MIL_ENTITY_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_MIL_ENTITY_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_MIL_PLAT_ENTITY_ATTR_CHECKER
- DtDEFINE_SIMPLE_MIL_PLAT_ENTITY_ATTR_DECODER
- DtDEFINE_SIMPLE_MIL_PLAT_ENTITY_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_MIL_PLAT_ENTITY_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_MUN_ENTITY_ATTR_CHECKER
- DtDEFINE_SIMPLE_MUN_ENTITY_ATTR_DECODER
- DtDEFINE_SIMPLE_MUN_ENTITY_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_MUN_ENTITY_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_PARAM_DECODER
- DtDEFINE_SIMPLE_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_PHYS_ENTITY_ATTR_CHECKER
- DtDEFINE_SIMPLE_PHYS_ENTITY_ATTR_DECODER
- DtDEFINE_SIMPLE_PHYS_ENTITY_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_PHYS_ENTITY_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_RADARBEAM_ATTR_CHECKER
- DtDEFINE_SIMPLE_RADARBEAM_ATTR_DECODER
- DtDEFINE_SIMPLE_RADARBEAM_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_RADARBEAM_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_RADIORECVR_ATTR_CHECKER
- DtDEFINE_SIMPLE_RADIORECVR_ATTR_DECODER
- DtDEFINE_SIMPLE_RADIORECVR_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_RADIORECVR_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*
- DtDEFINE_SIMPLE_RADIOXMIT_ATTR_CHECKER
- DtDEFINE_SIMPLE_RADIOXMIT_ATTR_DECODER
- DtDEFINE_SIMPLE_RADIOXMIT_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_RADIOXMIT_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*

- DtDEFINE_SIMPLE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER
- DtDEFINE_SIMPLE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER
- DtDEFINE_SIMPLE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_REMOBJREQ_PARAM_DECODER
- DtDEFINE_SIMPLE_REMOBJREQ_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_REMOBJREQ_PARAM_ENCODER
- DtDEFINE_SIMPLE_REMOBJRES_PARAM_DECODER
- DtDEFINE_SIMPLE_REMOBJRES_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_REMOBJRES_PARAM_ENCODER
- DtDEFINE_SIMPLE_REMOVENT_PARAM_DECODER
- DtDEFINE_SIMPLE_REMOVENT_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_REMOVENT_PARAM_ENCODER
- DtDEFINE_SIMPLE_REPCOMP_PARAM_DECODER
- DtDEFINE_SIMPLE_REPCOMP_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_REPCOMP_PARAM_ENCODER
- DtDEFINE_SIMPLE_REPRESP_PARAM_DECODER
- DtDEFINE_SIMPLE_REPRESP_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_REPRESP_PARAM_ENCODER
- DtDEFINE_SIMPLE_RESCANC_PARAM_DECODER
- DtDEFINE_SIMPLE_RESCANC_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_RESCANC_PARAM_ENCODER
- DtDEFINE_SIMPLE_RESOFF_PARAM_DECODER
- DtDEFINE_SIMPLE_RESOFF_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_RESOFF_PARAM_ENCODER
- DtDEFINE_SIMPLE_RESREC_PARAM_DECODER
- DtDEFINE_SIMPLE_RESREC_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_RESREC_PARAM_ENCODER
- DtDEFINE_SIMPLE_SERVREQ_PARAM_DECODER
- DtDEFINE_SIMPLE_SERVREQ_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_SERVREQ_PARAM_ENCODER
- DtDEFINE_SIMPLE_SETDATA_PARAM_DECODER
- DtDEFINE_SIMPLE_SETDATA_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_SETDATA_PARAM_ENCODER
- DtDEFINE_SIMPLE_SOLDIER_ATTR_CHECKER
- DtDEFINE_SIMPLE_SOLDIER_ATTR_DECODER
- DtDEFINE_SIMPLE_SOLDIER_ATTR_DECODER_WITH_CAST *These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)*
- DtDEFINE_SIMPLE_SOLDIER_ATTR_ENCODER *This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)*

- DtDEFINE_SIMPLE_START_PARAM_DECODER
- DtDEFINE_SIMPLE_START_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_START_PARAM_ENCODER
- DtDEFINE_SIMPLE_STOP_PARAM_DECODER
- DtDEFINE_SIMPLE_STOP_PARAM_DECODER_WITH_CAST
- DtDEFINE_SIMPLE_STOP_PARAM_ENCODER
- DtDEFINE_SIMPLE_VIEW_CONTROL_PARAM_ENCODER
- DtDELETE
- DtDInteraction_H_
- DtDataDecoder_H_
- DtDataEncoder_H_
- DtDataInteraction_H_
- DtDataPdu_H_ *This file provides protocol independence for Comment Interactions*
- DtDataQueryDecoder_H_
- DtDataQueryEncoder_H_
- DtDataQueryInteraction_H_
- DtDataQueryPdu_H_
- DtDataQueryRPdu_H_ *This file provides protocol independence for ActionResponse Interactions*
- DtDataRPdu_H_
- DtDatabaseIndexRadioSignalDecoder_H_
- DtDatabaseIndexRadioSignalEncoder_H_
- DtDatumMap_H_
- DtDatumRecordSet_H_
- DtDcm_H_
- DtDeadReckoner_H_
- DtDeg2Rad
- DtDesignatorDecoder_H_
- DtDesignatorEncoder_H_
- DtDesignatorPdu_H_
- DtDesignatorPublisher_H_
- DtDesignatorRepository_H_
- DtDetonationDecoder_H_
- DtDetonationEncoder_H_
- DtDetonationInteraction_H_
- DtDirectRtiAmbassador_H_
- DtDirectedRotRate_H_
- DtDisAggregatePublisher_H_
- DtDisEntityPublisher_H_

- DtDisExerciseConn_H_
- DtDisPduFactory_H_
- DtDiscoverObjectCbInfo_H_
- DtECenti
- DtEDeca
- DtEDeci
- DtEGiga
- DtEHalf
- DtEHecto
- DtEKilo
- DtEMega
- DtEMicro
- DtEMilli
- DtENano
- DtENineTenths
- DtEOne
- DtEPico *A selection of constants useful as tolerance and threshold levels, etc*
- DtEXTEND_ATTR_ENUM *This macro creates definitions that look like this: const DtAppearAttribute DtAppDensity = DtAppearAttribute(16 << DtAppDensityBitShift);*
- DtEXTEND_ATTR_VAL_ENUM
- DtEXTEND_BIT_SHIFT_ENUM *This macro creates definitions that look like this: const DtAppearAttributeBitShift DtAppDensityBitShift = DtAppearAttributeBitShift(16);*
- DtEXTEND_ENUM *This file should contain only enumeration definitions*
- DtElectromagneticEmissionPdu_H_
- DtEmbeddedSystemRepository_H_
- DtEmissionPdu_H_
- DtEmitterBeamPublisher_H_
- DtEmitterBeamRepository_H_
- DtEmitterSystemDecoder_H_
- DtEmitterSystemEncoder_H_
- DtEmitterSystemPublisher_H_
- DtEmitterSystemRepository_H_
- DtEmptyFomMapper_H_
- DtEncodedAudioRadioSignalDecoder_H_
- DtEncodedAudioRadioSignalEncoder_H_
- DtEnhancedFixedWingAircraftRecord_H_
- DtEnhancedGroundCombatSoldierRecord_H_
- DtEnhancedGroundCombatVehicleRecord_H_
- DtEnhancedRotorWingAircraftRecord_H_

- DtEntityEncoder_H_
- DtEntityIsAntiPersonnel
- DtEntityPublisher_H_
- DtEntityStateDecoder_H_
- DtEntityStatePdu_H_
- DtEntityStateRepository_H_ *This file provides protocol independence for Local Entities*
- DtEntityStateUpdatePdu_H_
- DtEnvironmentEntityDecoder_H_
- DtEnvironmentEntityEncoder_H_
- DtEnvironmentTypeRecord_H_
- DtEnvironmentalProcessPdu_H_
- DtEq *A macro to make use of these tolerances*
- DtEventId_H_
- DtEventReportDecoder_H_
- DtEventReportEncoder_H_
- DtEventReportInteraction_H_
- DtEventReportPdu_H_ *This file provides protocol independence for EventReport Interactions*
- DtEventReptRPdu_H_
- DtExerciseConn_H_
- DtFIXED *For backwards compatibility:*
- DtFedReader_H_
- DtFedTarget_H_
- DtFilename_H_
- DtFireDecoder_H_
- DtFireEncoder_H_
- DtFireInteraction_H_
- DtFirePdu_H_ *This file provides protocol independence for Fire Interactions*
- DtFomMapper_H_
- DtFom_H_
- DtGenericObjectDecoder_H_
- DtGenericObjectEncoder_H_
- DtGenericObjectPublisher_H_
- DtGenericStateRepository_H_
- DtGeodeticCoord_H_
- DtGlobalIdHashlist_H_
- DtGlobalObjectDesignatorList_H_
- DtGlobalObjectDesignator_H_
- DtGridAxisRecordFixed_H_

- DtGridAxisRecordVariable_H_
- DtGridDataRecordType0_H_
- DtGridDataRecordType1_H_
- DtGridDataRecordType2_H_
- DtGridDataRecord_H_
- DtGriddedDataPdu_H_
- DtGroundLogisticsVehicleRecord_H_
- DtGroupSocket_H_
- DtHInteraction_H_
- DtHashItem_H_
- DtHashlist_H_
- DtHlaObjectFactory_H_
- DtHlaObjectWithStateRep_H_
- DtHlaObject_H_
- DtHlaStateDecoder_H_
- DtHlaStateEncoder_H_
- DtIGNORE_AZIMUTH_RATE
- DtIGNORE_BODY_HEADING
- DtIGNORE_BODY_PITCH
- DtIGNORE_BODY_ROLL
- DtIGNORE_EAST
- DtIGNORE_EAST_OFFSET
- DtIGNORE_FORWARD
- DtIGNORE_GROUP_IDS
- DtIGNORE_HEADING
- DtIGNORE_HEADING_OFFSET
- DtIGNORE_HEIGHT
- DtIGNORE_HEIGHT_OFFSET
- DtIGNORE_NORTH
- DtIGNORE_NORTH_OFFSET
- DtIGNORE_PITCH
- DtIGNORE_PITCH_OFFSET
- DtIGNORE_PRIMARY_ID
- DtIGNORE_RADIAL_OFFSET
- DtIGNORE_RADIUS
- DtIGNORE_RIGHT
- DtIGNORE_ROLL
- DtIGNORE_SECONDARY_ID

- DtIGNORE_UP
- DtISOFF
- DtISON
- DtIffAtcNavaidsLayerHeader_H_
- DtIffAtcNavaidsOperational_H_
- DtIffAtcNavaidsParameter_H_
- DtIffAtcNavaidsPdu_H_
- DtIffAtcNavaidsSecondOperational_H_
- DtIffAtcNavaidsSystemID_H_
- DtIntHashKey_H_ *This file is used as a convenience to group interaction concepts together*
- DtInterClassDesc_H_
- DtInteractionDecoderFactory_H_
- DtInteractionDecoder_H_
- DtInteractionEncoderFactory_H_
- DtInteractionEncoder_H_
- DtInteractionFactory_H_
- DtInteractionManager_H_
- DtInteractionWithData_H_
- DtInteractionWithEncDec_H_
- DtIntercomControlPdu_H_
- DtIntercomEntityDestRec_H_
- DtIntercomGroupAsgmtRec_H_
- DtIntercomGroupDestRec_H_
- DtIntercomParmRec_H_
- DtIntercomSignalPdu_H_
- DtInternationalSpheroid
- DtIsAirPlatform
- DtIsFixedWing
- DtIsGroupOfPdu_H_
- DtIsLandLifeForm
- DtIsLandPlatform
- DtIsPartOfPdu_H_
- DtIsRotaryWing
- DtIsTank
- DtJammerBeamDecoder_H_
- DtJammerBeamEncoder_H_
- DtLIMIT
- DtLaserPdu_H_

- DtLeAppearancePdu_H_
- DtLeArticulatedPartsPdu_H_
- DtLeDeadReckoningParameters_H_
- DtLeDetonationPdu_H_
- DtLeEntityID_H_
- DtLeEntityLinearVelocity_H_
- DtLeEntityLocationEntityCoordinates_H_
- DtLeEntityLocationWorldCoordinates_H_
- DtLeEntityOrientation_H_
- DtLeFirePdu_H_
- DtLeOrientationError_H_
- DtLePositionError_H_
- DtLeTspiPdu_H_
- DtLgrControlDecoder_H_
- DtLgrControlEncoder_H_
- DtLgrControlInteraction_H_
- DtLgrControlPdu_H_ *This file provides protocol independence for LgrControl Interactions*
- DtLgrEndPdu_H_
- DtLgrEofActionPdu_H_
- DtLgrHeartbeatPdu_H_
- DtLgrPausePdu_H_
- DtLgrPbFilePdu_H_
- DtLgrPlayActionPdu_H_
- DtLgrPlayPdu_H_
- DtLgrQuitPdu_H_
- DtLgrRecFilePdu_H_
- DtLgrRecordActionPdu_H_
- DtLgrRecordPdu_H_
- DtLgrSetTimePdu_H_
- DtLgrSkipTimePdu_H_
- DtLgrStartPdu_H_
- DtLgrStopPdu_H_
- DtLgrTimeScalePdu_H_
- DtLgrTimeoutPdu_H_
- DtLinearObjectStatePdu_H_
- DtLinearSegmentParameter_H_
- DtList_H_
- DtLocalHlaObjectManager_H_

- DtMAX
- DtMAX_OBJECT_NAME_LENGTH *This class is basically a hack so that the simple encoders and decoders will work for DtGlobalObjectDesignators*
- DtMIN
- DtMaxEntityMarkingLength
- DtMaxPacketSize *IP + UDP headers are 20 + 8*
- DtMessagePdu_H_
- DtMilitaryEntityDecoder_H_
- DtMilitaryEntityEncoder_H_
- DtMilitaryPlatformEntityDecoder_H_
- DtMilitaryPlatformEntityEncoder_H_
- DtMinefieldDataPdu_H_
- DtMinefieldPerimeterRecord_H_
- DtMinefieldQueryPdu_H_
- DtMinefieldResponseNackPdu_H_
- DtMinefieldStatePdu_H_
- DtMunitionEntityDecoder_H_
- DtMunitionEntityEncoder_H_
- DtNUMBER
- DtNeg *Returns x if x is less than zero*
- DtNetBoolean_H_
- DtNetTimeStamp_H_ *example device names are "ec0" and "le0"*
- DtNlFatal *Notification Levels*
- DtNlInfo
- DtNlVerbose
- DtNlWarn
- DtNoArtParts
- DtNullArtPartSpec
- DtNullDatumParam
- DtNullFederateAmbassador_H_ *Adapted from the DMSO RTI 1*
- DtNullVarDatumSpec
- DtOFFSET
- DtObjClassDesc_H_
- DtObjectIdHashKey_H_
- DtObjectIdHashlist_H_
- DtObjectPublisher_H_
- DtObjectReflectAttributeValuesCbInfo_H_
- DtObjectType_H_
- DtPduCallbackInfo_H_

- DtPduCallbackManager_H_
- DtPduDatumRecordSet_H_
- DtPduFactory_H_
- DtPduWithData_H_
- DtPdu_H_
- DtPhysicalEntityDecoder_H_
- DtPhysicalEntityEncoder_H_
- DtPointObjectStatePdu_H_
- DtPos *Returns x if greater than zero*
- DtPropulsionSystem_H_
- DtQuiet
- DtRad2Deg
- DtRadarBeamDecoder_H_
- DtRadarBeamEncoder_H_
- DtRadioOther
- DtRadioReceiverDecoder_H_
- DtRadioReceiverEncoder_H_
- DtRadioReceiverPublisher_H_
- DtRadioReceiverRepository_H_
- DtRadioTransmitterDecoder_H_
- DtRadioTransmitterEncoder_H_
- DtRadioTransmitterPublisher_H_
- DtRadioTransmitterRepository_H_
- DtRawBinaryRadioSignalDecoder_H_
- DtRawBinaryRadioSignalEncoder_H_
- DtRead
- DtReadWrite
- DtReceiveInteractionCbInfo_H_
- DtReceiverPdu_H_
- DtRecordPdu_H_
- DtRecordQueryPdu_H_
- DtRecordSetPdu_H_
- DtRecordSet_H_
- DtReflectAttributeValuesCbInfo_H_
- DtReflectAttributeValuesCbList_H_
- DtReflectedAggregateList_H_
- DtReflectedAggregate_H_
- DtReflectedAggregate_H_

- DtReflectedDesignatorList_H_
- DtReflectedDesignator_H_
- DtReflectedEmitterBeamList_H_
- DtReflectedEmitterBeam_H_
- DtReflectedEmitterSystemList_H_
- DtReflectedEmitterSystem_H_
- DtReflectedEmitterSystem_H_
- DtReflectedEntityList_H_
- DtReflectedEntityList_H_
- DtReflectedEntity_H_
- DtReflectedEntity_H_
- DtReflectedGenericObjectList_H_
- DtReflectedGenericObject_H_ *This file provides protocol independence for reflected emitter system lists*
- DtReflectedHlaObjectManager_H_ *This file provides protocol independence for Reflected Object Lists*
- DtReflectedObjectCallbackInfo_H_
- DtReflectedObjectList_H_
- DtReflectedObjectList_H_
- DtReflectedObject_H_
- DtReflectedObject_H_
- DtReflectedRadioReceiverList_H_
- DtReflectedRadioReceiver_H_
- DtReflectedRadioTransmitterList_H_
- DtReflectedRadioTransmitter_H_
- DtRegControlCbInfo_H_
- DtRegControlCbList_H_
- DtRemoveEntityDecoder_H_
- DtRemoveEntityEncoder_H_
- DtRemoveEntityInteraction_H_
- DtRemoveEntityPdu_H_
- DtRemoveEntityRPdu_H_
- DtRemoveObjectCbInfo_H_ *This file provides protocol independence for RemoveEntity Interactions*
- DtRemoveObjectRequestDecoder_H_
- DtRemoveObjectRequestEncoder_H_
- DtRemoveObjectRequestInter_H_
- DtRemoveObjectResultDecoder_H_
- DtRemoveObjectResultEncoder_H_
- DtRemoveObjectResultInter_H_
- DtRepairCompleteDecoder_H_

- DtRepairCompleteEncoder_H_
- DtRepairCompleteInteraction_H_
- DtRepairResponseDecoder_H_
- DtRepairResponseEncoder_H_
- DtRepairResponseInteraction_H_
- DtRepairResponsePdu_H_
- DtResupplyCancelDecoder_H_
- DtResupplyCancelEncoder_H_
- DtResupplyCancelInteraction_H_
- DtResupplyCancelPdu_H_ *This file provides protocol independence for the Resupply Interactions: ResupplyCancel, ResupplyOffer, ResupplyReceived*
- DtResupplyOfferDecoder_H_
- DtResupplyOfferEncoder_H_
- DtResupplyOfferInteraction_H_
- DtResupplyRecInteraction_H_
- DtResupplyReceivedDecoder_H_
- DtResupplyReceivedEncoder_H_
- DtRprFomMapper *If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleFomMapper - a FOM Mapper that contains mappings for only the subset of RPR FOM objects/interactions that are used by most MAK tools*
- DtRprFomMapper_H_
- DtRprFomMapper_H_WAS_DEFINED
- DtRprHlaObjectFactory_H_
- DtRprInteractionDecoderFactory *If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprInteractionDecoderFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools*
- DtRprInteractionDecoderFactory_H_
- DtRprInteractionEncoderFactory *If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprInteractionEncoderFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools*
- DtRprInteractionEncoderFactory_H_
- DtRprInteractionFactory *If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprInteractionFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools*
- DtRprInteractionFactory_H_
- DtRprStateDecoderFactory *If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprStateDecoderFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools*
- DtRprStateDecoderFactory_H_
- DtRprStateEncoderFactory *If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprStateEncoderFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools*
- DtRprStateEncoderFactory_H_

- DtSIMPLE_RPR *This header file provides a definition for the class DtSimpleRprFomMapper, a FOM Mapper that includes mappings for only the subset of the RPR FOM (with MAK extensions) that is used by most MAK applications*
- DtSIMPLE_RPR_WAS_DEFINED
- DtSIZEOF
- DtSOCKET_FILTERED_PACKET
- DtSOCKET_NO_PACKET
- DtSOCKET_PACKET
- DtSOCKET_SERVER_MSG
- DtSQR
- DtSTATUS_NOT_OK
- DtSTATUS_OK
- DtSUPPLY_BAD_INDEX
- DtSUPPLY_VALID
- DtSameEntityKind
- DtServiceRequestDecoder_H_
- DtServiceRequestEncoder_H_
- DtServiceRequestInteraction_H_
- DtSetDataDecoder_H_
- DtSetDataEncoder_H_
- DtSetDataInteraction_H_
- DtSetDataPdu_H_ *This file provides protocol independence for ActionResponse Interactions*
- DtSetDataRPdu_H_
- DtSetRecordPdu_H_
- DtSetVec *Vector routines*
- DtSgn *Returns the sign of x*
- DtSignalInteraction_H_
- DtSignalPdu_H_
- DtSilentAggregateList_H_
- DtSilentEntityList_H_
- DtSimanPdu_H_
- DtSimpleRprFomMapper_H_
- DtSmoothOriWithDr_H_
- DtSmoothPosWithDr_H_
- DtSmoothPos_H_
- DtSmoother_H_ *This file provides protocol independence for Signal Interactions*
- DtSoldierDecoder_H_
- DtSoldierEncoder_H_
- DtSquare *Returns the square of x*

- DtStartResumeDecoder_H_
- DtStartResumeEncoder_H_
- DtStartResumeInteraction_H_
- DtStartResumePdu_H_ *This file provides protocol independence for StartResume Interactions*
- DtStartResumeRPdu_H_
- DtStartStopInterCbInfo_H_
- DtStartStopInterCbList_H_
- DtStateDecoderFactory_H_
- DtStateEncoderFactory_H_
- DtStateRepository_H_ *This file provides protocol independence for state messages*
- DtStopFreezeDecoder_H_
- DtStopFreezeEncoder_H_
- DtStopFreezeInteraction_H_
- DtStopFrzRPdu_H_
- DtStopPdu_H_ *This file provides protocol independence for StopFreeze Interactions*
- DtStringHashKey_H_
- DtString_H_
- DtSuppEmisEntStatePdu_H_
- DtSystemHashKey_H_
- DtTURNOFF
- DtTURNON
- DtTYPE_BYTE_ARRAY
- DtTYPE_FLOAT32
- DtTYPE_FLOAT64
- DtTYPE_INT32
- DtTYPE_NOT_DEFINED
- DtTYPE_STRING
- DtTYPE_UNSIGNED32
- DtTaitBryan_H_
- DtTcpForwarder_H_
- DtTcpSocketMgr_H_
- DtTcpSocket_H_
- DtThresholder_H_
- DtTimeStampType_H_
- DtTopoView_H_
- DtTrackJamBeamDecoder_H_
- DtTrackJamBeamEncoder_H_
- DtTrackJamDesignatorList_H_

- DtTrackJamDesignator_H_
- DtTransferControlRequestPdu_H_
- DtTransmitterPdu_H_
- DtUSE_DEFAULT_IP_ADDR
- DtUSE_INTERNAL_BUFFER
- DtUSE_INTERNAL_BUFFER_REP
- DtUnderwaterAcousticAddtnlPassiveAct_H_
- DtUnderwaterAcousticBeam_H_
- DtUnderwaterAcousticEmitterSystem_H_
- DtUnderwaterAcousticPdu_H_
- DtUnderwaterAcousticShaftSpeed_H_
- DtUnknownHlaObject_H_
- DtUnknownInteraction_H_
- DtUnknownPdu_H_
- DtUpdateControlCbInfo103_H_
- DtUpdateControlCbInfo_H_
- DtUpdateControlCbList103_H_
- DtUpdateControlCbList_H_
- DtUtmCoord_H_
- DtUtmView_H_
- DtVAR
- DtVariableDatumRecord_H_
- DtVector_H_
- DtVectoringNozzleSystem_H_
- DtViewControlDecoder_H_
- DtViewControlEncoder_H_
- DtViewControlInteraction_H_
- DtViewControlKind *backwards compatible*
- DtViewControlPdu_H_ *This file provides protocol independence for ViewControl Interactions*
- DtVrlFedTime_H_ *This file is based on MyFedTime*
- DtVrlFederateAmbassador_H_
- DtVrlRtiAmbassador_H_
- DtWGS84Spheroid *Here are a few of the most important spheroids; we should put all 13 in*
- DtWrite
- ENTITY_PUB_H_
- EVENTREPORT_H_
- EXERCISE_CONNECTION_H_
- Euler_H_

- FIRE_INTERACTION_H_ *This file provides protocol independence for exercise connections*
- IDENTITY
- INTERACTION_H_
- LGR_CONTROL_INTERACTION_H_
- LIMIT
- MAX
- MIN
- M_1_PI
- M_2_PI
- M_2_SQRTPI
- M_E
- M_LN10
- M_LN2
- M_LOG10E
- M_LOG2E
- M_PI
- M_PI_2
- M_PI_4
- M_SQRT2
- MathTypes_H_
- NULL
- ObjectHandle
- PDUS_H_
- REFLECTED_EMITTER_H *This file provides protocol independence for Reflected Objects*
- REFLECTED_EMITTER_LIST_H
- REMOTE_AGGREGATE_H_ *This file provides protocol independence for remote aggregate lists*
- REMOTE_AGGREGATE_LIST_H_
- REMOTE_ENTITY_H_ *This file provides protocol independence for reflected aggregates*
- REMOTE_ENTITY_LIST_H_ *This file provides protocol independence for reflected emitter systems*
- REMOTE_OBJ_H_ *This file provides protocol independence for Reflected Entities*
- REMOTE_OBJ_LIST_H_ *This file provides protocol independence for Remote Entity Lists*
- REMOVE_ENTITY_H_
- REPAIR_COMPLETE_H_
- REPAIR_RESPONSE_H_ *This file provides protocol independence for ActionResponse Interactions*
- RESUPPLY_H_ *This file provides protocol independence for ActionResponse Interactions*
- REUSE_ALLOWED
- SCRIPT_H_
- SERVICE_REQUEST_H_

- SET_DATA_H_ *This file provides protocol independence for ActionResponse Interactions*
- SIGNAL_INTER_H_
- SQR
- START_INTERACTION_H_
- STATE_MESSAGE_H_
- STOP_INTERACTION_H_
- SUBTRACT
- TopoCoord_H_
- VIEW_CONTROL_INTERACTION_H_
- VLCONFIG_INCLUDED *File - libsrc/md/c-NT*
- VL_TIME_H_
- ZERO *generic initialize matrix routines*
- _ACKPRED_H_
- _CLIENTSOCKET_H_
- _COLLPDU_H_
- _DATUMIDTYPE_H_ *This file provides protocol independence for Collision Interactions*
- _DISENUMS_H_
- _ENTITYTYPES_H_
- _ENUM2STRING_H_
- _HOSTSTRUCTS_H_
- _LibMatrix_h
- _MachineTypes_h
- _NETSTRUCTS_H_
- _NetSocket_h *class NetSocket NetSocket provides several overloaded constructors that allow the user to specify UDP port, device, and destination address in any combination except port and destination address together*
- _NetTypes
- _PAIRSOCK_H_
- _PRINT_UTIL_H_
- _ProcessControl_h
- _Socket_h *DtSocket is a misnomer This provides net-level Packet IO Sometimes, it is implemented with sockets, sometimes not*
- _VLPRINT_H
- _VLUTIL_H_
- __declspec
- _detonationPdu_h *This file provides protocol independence for Detonation Interactions*
- _parseEnum_h_
- antiPersonell
- dDtStateMsg_H_
- define

- gGuiseSameAsType
- hDtStateMsg_H_
- looplist *Macro to loop through the list*
- looptlist
- looptlist2
- netPdu
- what *Macro definition so that exception*

Enums, Unions, Structs

- DtAH1WSpecific
- DtAcknowledgeFlag
- DtActionId
- DtAegisCruiserSpecific
- DtAggregateFormation
- DtAggregateState *Aggregate State PDU enums*
- DtAntennaPatternType
- DtAppearAttrVal *Create an enumeration type, but since many of the names will have the same value, define them all as consts below*
- DtAppearAttribute *if you put spaces between the "<<" and the surrounding strings, enumstr gets confused and thinks the shift is another value*
- DtAppearAttributeBitShift
- DtArtParamType *Articulated parts*
- DtArtPartThresholdType
- DtArtPartType
- DtAttachedPartStatus
- DtAttributeChangeResult
- DtBeamFunc
- DtBeamType
- DtBoolean *For convenience*
- DtCV59Specific
- DtCamouflageType
- DtCapability
- DtCharacterSet
- DtCollisionType
- DtCountryCode
- DtCreateObjectResult
- DtCryptoMode
- DtCulturalFeatureDomain
- DtCulturalFeatureLand

- DtDamageState *This is for the tank ditch and concertina wire breach appearance Areal object specific appearance*
- DtDatumParam
- DtDatumRecordType
- DtDatumRetCode
- DtDeadReckonTypes *Section 7*
- DtDensityType
- DtDesignatorCode
- DtDesignatorCodeName
- DtDetailedModulationAmplitudeAngle
- DtDetailedModulationAngle
- DtDetailedModulationCombination
- DtDetailedModulationType
- DtDetailedModulationUnmodulated
- DtDetonationResult
- DtDetonatorFuze
- DtEmissionFunc
- DtEmissionPduType
- DtEmitterSysName
- DtEmitterSysRetCode *error codes for looking up individual beams*
- DtEncodingClass
- DtEncodingType
- DtEntityKind
- DtEnvironmentalAir
- DtEnvironmentalDomain
- DtEnvironmentalModelType *Environmental Process PDU enums*
- DtEnvironmentalStatusBit
- DtEventType
- DtF16
- DtFFG7Specific
- DtFLIR_SensorTypeSubcategory
- DtFedReadStatus
- DtFilterType
- DtForceType *Section 4*
- DtGridAxisType
- DtGridDataRecType *Gridded data PDU enums*
- DtGroupedEntityCategory *IsGroupOf PDU enums*
- DtHatchState
- DtIffAtcNavaidsAltParm4

- DtIffAtcNavaidsApplicableModes
- DtIffAtcNavaidsChangeOptionsBit
- DtIffAtcNavaidsLayerSpecificInfo
- DtIffAtcNavaidsModifier
- DtIffAtcNavaidsParameterBit
- DtIffAtcNavaidsParm1
- DtIffAtcNavaidsParm2
- DtIffAtcNavaidsStatusBit
- DtIffAtcNavaidsSysMode
- DtIffAtcNavaidsSysName *IFF/ATC/NAVAIDS PDU enums*
- DtIffAtcNavaidsSysType
- DtInitErrorAction
- DtIntercomCmdType
- DtIntercomCommClass
- DtIntercomCommType
- DtIntercomCtlRecType
- DtIntercomCtlType *Intercom Control PDU enums*
- DtIntercomDestLineStateCmd
- DtIntercomTransLineState
- DtLE_AppearanceField1Flag
- DtLE_AppearanceField2Flag
- DtLE_DetonField2Flag
- DtLE_DetonFieldFlag
- DtLE_FireFieldFlag
- DtLE_TSPI_FieldFlag
- DtLHASpecific
- DtLaserSensorTypeSubcategory
- DtLgrCommand
- DtLgrDirection *Play forwards or backwards when in LGR_PLAY mode*
- DtLgrFilePosition *Location status*
- DtLgrMode *General operation modes*
- DtLgrProcess *Receiving application may filter lcPdu based on this start-up flag*
- DtLgrStyle
- DtLifeFormDomain
- DtLifeFormLand
- DtLifeformState
- DtLightState
- DtLongWordPortion

- DtMagneticSensorTypeSubcategory
- DtMinefieldAlgae
- DtMinefieldAntiHandlingDevice
- DtMinefieldDataFilterField *Minefield PDU enums*
- DtMinefieldFusing
- DtMinefieldPaintScheme
- DtMinefieldProtocol
- DtModificationField
- DtMultispectralSensorTypeSubcategory
- DtMunitionAntiAir
- DtMunitionAntiArmor
- DtMunitionAntiPersonnel
- DtMunitionAntiRadar
- DtMunitionAntiShip
- DtMunitionAntiSubmarine
- DtMunitionBattlefieldSupport
- DtMunitionDomain
- DtMunitionUKAntiPersonnelUnguided *UK*
- DtMunitionUKBattlefieldSupportUnguided
- DtMunitionUSAntiAirGuided *US*
- DtMunitionUSAntiArmorGuided
- DtMunitionUSAntiArmorUnguided
- DtMunitionUSAntiPersonnelUnguided
- DtMunitionUSAntiRadarGuided
- DtMunitionUSAntiSubmarineGuided
- DtMunitionUSBattlefieldSupportGuided
- DtMunitionUSBattlefieldSupportUnguided
- DtMunitionUSSRAntiAirGuided *USSR*
- DtMunitionUSSRAntiPersonnelUnguided
- DtNature *IsPartOf PDU enums*
- DtNumberOfIndividualLifeForms
- DtObjectKind *Environmental process PDU enums*
- DtOpticalSensorTypeSubcategory
- DtOrderType
- DtPduKind
- DtPhysicalSensorTypeSubcategory
- DtPlatformAir
- DtPlatformAirFranceUtilityHelicopter *France*

- DtPlatformAirUSAttack
- DtPlatformAirUSAttackHelicopter
- DtPlatformAirUSCargo
- DtPlatformAirUSFighter
- DtPlatformAirUSObservationHelicopter
- DtPlatformAirUSSRAttack
- DtPlatformAirUSSRAttackHelicopter
- DtPlatformAirUSSRBomber
- DtPlatformAirUSSRFighter
- DtPlatformAirUSSpecialOperations
- DtPlatformAirUSUnmannedAirVehicle
- DtPlatformDomain
- DtPlatformLand
- DtPlatformLandAnySmallWheeledUtilityVehicle *Any Country*
- DtPlatformLandUKTank *UK*
- DtPlatformLandUSArmoredFightingVehicle
- DtPlatformLandUSArmoredUtilityVehicle
- DtPlatformLandUSSRArmoredFightingVehicle
- DtPlatformLandUSSRLargeWheeledUtilityVehicle
- DtPlatformLandUSSRSelfPropelledArtillery
- DtPlatformLandUSSRSelfTowedArtillery
- DtPlatformLandUSSRTank *USSR*
- DtPlatformLandUSSelfPropelledArtillery
- DtPlatformLandUSSmallWheeledUtilityVehicle
- DtPlatformLandUSTank *US*
- DtPlatformSubsurfaceUSStrategicMissleSubmarine
- DtPlatformSurface
- DtPlatformSurfaceUSAmphibiousAssaultShip
- DtPlatformSurfaceUSCarrier
- DtPlatformSurfaceUSGuidedMissleCruiser
- DtPlatformSurfaceUSGuidedMissleFrigate
- DtPolarDirection
- DtPosition
- DtProtocolFamily
- DtRADAR_SensorTypeSubcategory
- DtRadioCryptoSystem
- DtRadioDetailedModulationAmplitude
- DtRadioInputSource

- DtRadioMajorModulation
- DtRadioReceiverState
- DtRadioSpreadSpectrum
- DtRadioSystem
- DtRadioTransmitState
- DtReferenceSys
- DtRemoveObjectResult
- DtRepairResult
- DtRepairType *Section 2*
- DtRequestStatus
- DtRequiredReliabilityService *Reliable service PDU enums*
- DtResponseFlag
- DtRestStatus
- DtSA10Specific
- DtSA8Specific
- DtSONARSensorTypeSubcategory
- DtSensorTypeCategory
- DtServiceType
- DtSizeQualifier
- DtStationName
- DtStopReason
- DtSubProtocol *Live entity protocol PDU enums*
- DtSupplyCategories
- DtT72Specific
- DtTacticalDataLinkType
- DtTestOperator
- DtTimeStampType
- DtTrackingJamming
- DtTrailState
- DtTransferType *Transfer Control Request PDU enums*
- DtTransportType
- DtUnderwaterAcousticActiveEmissionParmIndex
- DtUnderwaterAcousticAdditionalPassiveActivityParmIndex
- DtUnderwaterAcousticFunction
- DtUnderwaterAcousticHullMountedMaskerStatus
- DtUnderwaterAcousticPassiveParmIndex
- DtUnderwaterAcousticPropulsionPlantConfig
- DtUnderwaterAcousticScanPattern

- DtUnderwaterAcousticSystemName
- DtUnderwaterAcousticUpdate *Underwater accoustic PDU enums*
- DtUserProtocol
- DtVectorOffset
- DtViewerCommand
- DtWarheadType
- DtWeaponState
- generic_mm_multflag *Generic DcmDcm Multiply routine*

Typedefs

- DtBufferPtr
- DtEntityKind
- DtMunition
- DtQ0 *Quaternion routines*
- DtSimulator
- DtSite *for backwards compatibility*
- DtSite
- DtX
- East
- artOrAttached
- deg
- hour
- kind
- myArtOrAttached
- myValue *Used only for internal representation of Art Param information*
- partType
- partType
- psi
- semiMajor *Spheroid*
- spheroid
- t
- wx

Hierarchy of classes

VR-Link 3.4 HLA Class Documentation

Alphabetic index

Hierarchy of Classes

- DtAbsNotifyObserver
- DtAntennaPatternRecord
    - DtBeamAntennaPatternRecord
- DtApproximator
    - DtDeadReckoner
        - DtSmoother
- DtArtPart
- DtArtPartList
- DtArtPartThresholder
- DtAttachedPartList
- DtBaseExerciseConn
    - DtExerciseConn
- DtBaseHashKey
    - DtSystemHashKey
    - DtStringHashKey
    - DtObjectIdHashKey
    - DtIntHashKey
- DtBaseMsg
    - DtPdu
        - DtViewControlPdu
            - DtUnknownVCPdu
            - DtTrackVCPdu
            - DtTetherVCPdu
            - DtTetherTrackVCPdu
            - DtOrbitVCPdu
            - DtModelessVCPdu
            - DtMimicVCPdu
            - DtMimicTrackVCPdu
            - DtGroupVCPdu
            - DtCompassVCPdu
            - DtAbsoluteVCPdu
        - DtLgrControlPdu

            - DtLgrTimeoutPdu
            - DtLgrTimeScalePdu
            - DtLgrStopPdu
            - DtLgrStartPdu
            - DtLgrSkipTimePdu
            - DtLgrSetTimePdu
            - DtLgrRecordPdu
            - DtLgrRecordActionPdu
            - DtLgrRecFilePdu
            - DtLgrQuitPdu
            - DtLgrPlayPdu
            - DtLgrPlayActionPdu
            - DtLgrPbFilePdu
            - DtLgrPausePdu
            - DtLgrHeartbeatPdu
            - DtLgrEofActionPdu
            - DtLgrEndPdu
        - DtDatumRecordSet
- DtBurstDescriptor
- DtCartesianView
    - DtTopoView
- DtCoordTransform
- DtCoordTranslation
- DtDcm
- DtDirectedRotRate
- DtDiscoverObjectCbInfo
- DtEntityIdentifier
- DtEntityType
    - DtRadioEntityType
- DtEventId
- DtFedReader
- DtFedTarget
    - DtFom
- DtFederateId
- DtFilterTest
    - DtCsFilterTest
- DtFomMapper
    - DtEmptyFomMapper

        - DtRprFomMapper
- DtGlobalIdHashlist
- DtGlobalObjectDesignatorList
- DtHashItem
- DtHashlist
- DtHlaObject
    - DtUnknownHlaObject
    - DtHlaObjectWithStateRep
- DtHlaObjectFactory
    - DtRprHlaObjectFactory
- DtHlaStateDecoder
    - DtGenericObjectDecoder
    - DtBaseEntityDecoder
        - DtPhysicalEntityDecoder
            - DtMilitaryEntityDecoder
                - DtSoldierDecoder
                - DtMunitionEntityDecoder
                - DtMilitaryPlatformEntityDecoder
        - DtEnvironmentEntityDecoder
        - DtAggregateEntityDecoder
    - DtBaseEmitterBeamDecoder
        - DtTrackJamBeamDecoder
        - DtRadarBeamDecoder
        - DtJammerBeamDecoder
    - DtBaseEmbeddedSystemDecoder
        - DtRadioTransmitterDecoder
        - DtRadioReceiverDecoder
        - DtEmitterSystemDecoder
        - DtDesignatorDecoder
- DtHlaStateEncoder
    - DtGenericObjectEncoder
    - DtBaseEntityEncoder
        - DtPhysicalEntityEncoder
            - DtMilitaryEntityEncoder
                - DtSoldierEncoder
                - DtMunitionEntityEncoder
                - DtMilitaryPlatformEntityEncoder
        - DtEnvironmentEntityEncoder

        - DtAggregateEntityEncoder
    - DtBaseEmitterBeamEncoder
        - DtTrackJamBeamEncoder
        - DtRadarBeamEncoder
        - DtJammerBeamEncoder
    - DtBaseEmbeddedSystemEncoder
        - DtRadioTransmitterEncoder
        - DtRadioReceiverEncoder
        - DtEmitterSystemEncoder
        - DtDesignatorEncoder
- DtInterClassDesc
- DtInteraction
    - DtUnknownInteraction
    - DtInteractionWithEncDec
        - DtViewControlInteraction
        - DtStopFreezeInteraction
        - DtStartResumeInteraction
        - DtSignalInteraction
        - DtServiceRequestInteraction
        - DtResupplyRecInteraction
        - DtResupplyOfferInteraction
        - DtResupplyCancelInteraction
        - DtRepairResponseInteraction
        - DtRepairCompleteInteraction
        - DtRemoveObjectResultInter
        - DtRemoveObjectRequestInter
        - DtRemoveEntityInteraction
        - DtLgrControlInteraction
        - DtInteractionWithData
            - DtSetDataInteraction
            - DtEventReportInteraction
            - DtDataInteraction
            - DtCommentInteraction
            - DtActionResponseInteraction
            - DtActionRequestInteraction
        - DtFireInteraction
        - DtDetonationInteraction
        - DtDataQueryInteraction

        - DtCreateObjectResultInter
        - DtCreateObjReqInter
        - DtCreateEntityInteraction
        - DtCollisionInteraction
        - DtAttributeChangeResInter
        - DtAttributeChangeReqInter
        - DtActionResponseFromObjInter
        - DtActionRequestToObjInter
        - DtAcknowledgeInteraction
- DtInteractionDecoder
    - DtViewControlDecoder
    - DtStopFreezeDecoder
    - DtStartResumeDecoder
    - DtSetDataDecoder
    - DtServiceRequestDecoder
    - DtResupplyReceivedDecoder
    - DtResupplyOfferDecoder
    - DtResupplyCancelDecoder
    - DtRepairResponseDecoder
    - DtRepairCompleteDecoder
    - DtRemoveObjectResultDecoder
    - DtRemoveObjectRequestDecoder
    - DtRemoveEntityDecoder
    - DtRawBinaryRadioSignalDecoder
    - DtLgrControlDecoder
    - DtFireDecoder
    - DtEventReportDecoder
    - DtEncodedAudioRadioSignalDecoder
    - DtDetonationDecoder
    - DtDatabaseIndexRadioSignalDecoder
    - DtDataQueryDecoder
    - DtDataDecoder
    - DtCreateObjectResultDecoder
    - DtCreateObjReqDecoder
    - DtCreateEntityDecoder
    - DtCommentDecoder
    - DtCollisionDecoder
    - DtAttributeChangeResDec

    - DtAttributeChangeReqDec
    - DtAppSpecificRadioSignalDecoder
    - DtActionResponseFromObjDecoder
    - DtActionResponseDecoder
    - DtActionRequestToObjDecoder
    - DtActionRequestDecoder
    - DtAcknowledgeDecoder
- DtInteractionDecoderFactory
    - DtRprInteractionDecoderFactory
- DtInteractionEncoder
    - DtViewControlEncoder
    - DtStopFreezeEncoder
    - DtStartResumeEncoder
    - DtSetDataEncoder
    - DtServiceRequestEncoder
    - DtResupplyReceivedEncoder
    - DtResupplyOfferEncoder
    - DtResupplyCancelEncoder
    - DtRepairResponseEncoder
    - DtRepairCompleteEncoder
    - DtRemoveObjectResultEncoder
    - DtRemoveObjectRequestEncoder
    - DtRemoveEntityEncoder
    - DtRawBinaryRadioSignalEncoder
    - DtLgrControlEncoder
    - DtFireEncoder
    - DtEventReportEncoder
    - DtEncodedAudioRadioSignalEncoder
    - DtDetonationEncoder
    - DtDatabaseIndexRadioSignalEncoder
    - DtDataQueryEncoder
    - DtDataEncoder
    - DtCreateObjectResultEncoder
    - DtCreateObjReqEncoder
    - DtCreateEntityEncoder
    - DtCommentEncoder
    - DtCollisionEncoder
    - DtAttributeChangeResEnc

        - DtAttributeChangeReqEnc
        - DtAppSpecificRadioSignalEncoder
        - DtActionResponseFromObjEncoder
        - DtActionResponseEncoder
        - DtActionRequestToObjEncoder
        - DtActionRequestEncoder
        - DtAcknowledgeEncoder
- DtInteractionEncoderFactory
    - DtRprInteractionEncoderFactory
- DtInteractionFactory
    - DtRprInteractionFactory
- DtInteractionManager
- DtList
    - DtScript
    - DtIntrusiveList
- DtListItem
- DtListedObj
    - DtReflectedObjectCallbackInfo
    - DtReflectedObject
        - DtReflectedRadioTransmitter
        - DtReflectedRadioReceiver
        - DtReflectedGenericObject
        - DtReflectedEntity
        - DtReflectedEmitterSystem
        - DtReflectedEmitterBeam
        - DtReflectedDesignator
        - DtReflectedAggregate
    - DtBaseReflectedEntity
    - DtBaseReflectedAggregate
- DtLocalHlaObjectManager
- DtNet32Vector
- DtNet64Vector
- DtNetBoolean
- DtNetBurstDescriptor
- DtNetEntityIdentifier
- DtNetEntityMarking
- DtNetEntityType
- DtNetEulerAngles

- DtNetEventId
- DtNetFederateId
- DtNetFloat32
- DtNetFloat64
- DtNetGlobalObjectDesignator
- DtNetInt16
- DtNetInt32
- DtNetRadioEntityType
- DtNetSimulationAddress
- DtNetTimeStamp
- DtNetU16
- DtNetU32
- DtNullFederateAmbassador
    - DtVrlFederateAmbassador
- DtObjClassDesc
- DtObjectId
- DtObjectIdHashlist
- DtObjectPublisher
    - DtRadioTransmitterPublisher
    - DtRadioReceiverPublisher
    - DtGenericObjectPublisher
    - DtEntityPublisher
    - DtEmitterSystemPublisher
    - DtEmitterBeamPublisher
    - DtDesignatorPublisher
    - DtAggregatePublisher
- DtObjectReflectAttributeValuesCbInfo
- DtReceiveInteractionCbInfo
- DtReflectAttributeValuesCbInfo
- DtReflectAttributeValuesCbList
- DtReflectedHlaObjectManager
- DtReflectedObjectList
    - DtReflectedRadioTransmitterList
    - DtReflectedRadioReceiverList
    - DtReflectedGenericObjectList
    - DtReflectedEntityList
    - DtReflectedEmitterSystemList
    - DtReflectedEmitterBeamList

        - DtReflectedDesignatorList
        - DtReflectedAggregateList
- DtRegControlCbInfo
- DtRegControlCbList
- DtRemoveObjectCbInfo
- DtSelectParam
- DtSilentAggregateList
- DtSilentEntityList
- DtSmoothOrient
    - DtSmoothOriWithDr
- DtSmoothPos
    - DtSmoothPosWithDr
- DtSocket
    - DtTcpSocket
    - DtSocketPairSocket
    - DtNetSocket
        - DtGroupSocket
    - DtClientSocket
- DtStartStopInterCbInfo
- DtStartStopInterCbList
- DtStateDecoderFactory
    - DtRprStateDecoderFactory
- DtStateEncoderFactory
    - DtRprStateEncoderFactory
- DtStateMsg
- DtStateRepository
    - DtGenericStateRepository
    - DtEmitterBeamRepository
    - DtEmbeddedSystemRepository
        - DtRadioTransmitterRepository
        - DtRadioReceiverRepository
        - DtEmitterSystemRepository
        - DtDesignatorRepository
    - DtBaseEntityStateRepository
        - DtEntityStateRepository
        - DtAggregateStateRepository
- DtString
    - DtFilename

- DtTaitBryan
- DtTcpForwarder
- DtTcpSocketMgr
- DtThresholder
- DtUpdateControlCbInfo
- DtUpdateControlCbInfo
- DtUpdateControlCbList
- DtUpdateControlCbList
- DtUtmView
- DtVarDatumSpec
- DtVariableDatumRecord
- DtVector
    - DtUtmCoord
    - DtGeodeticCoord
- DtVrlFedTime
- DtVrlRtiAmbassador
    - DtDirectRtiAmbassador

Alphabetic index

In file absnotifyobs.h:

class DtAbsNotifyObserver

class DtAbsNotifyObserver: DtAbsNotifyObserver is an abstract base class

---

Public Methods

- virtual **~DtAbsNotifyObserver** ()
  destructor
- DtAbsNotifyObserver& **operator=** (const DtAbsNotifyObserver& orig)
  assignment operator
- virtual DtBoolean **notifyFromObserved** (DtObservee* observed, const DtObservee::DtWhyNotifyObserver& why)
  Called by DtObservee when something has happened to the observed object
- **DtAbsNotifyObserver** ()
  default constructor

Protected Methods

- **DtAbsNotifyObserver** (const DtAbsNotifyObserver& orig)
  copy constructor

Protected

- *Constructors are protected since this is an abstract base class*

Documentation

class DtAbsNotifyObserver:

DtAbsNotifyObserver is an abstract base class. Instances of DtAbsNotifyObserver are used by DtObservee to notify an observer object that something has happened to the observed object. Observers should subclass from DtAbsNotifyObserver to receive notification for their class.

`virtual ~DtAbsNotifyObserver()`

destructor

`DtAbsNotifyObserver& operator=(const DtAbsNotifyObserver& orig)`

assignment operator

`virtual DtBoolean notifyFromObserved(DtObservee* observed, const DtObservee::DtWhyNotifyObserver& why)`

Called by DtObservee when something has happened to the observed object. Default does nothing. This should be overridden.

- Constructors are protected since this is an abstract base class

- **`DtAbsNotifyObserver()`**
  default constructor

- **`DtAbsNotifyObserver(const DtAbsNotifyObserver& orig)`**
  copy constructor

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file machinetypes.h:

enum DtBoolean

For convenience

- **DtFalse**
- **DtTrue**

Documentation

For convenience

- `DtFalse`
- `DtTrue`

In file antpatrec.h:

class DtAntennaPatternRecord

class DtAntennaPatternRecord: Instances of DtAntennaPatternRecord are

---

Inheritance:

DtAntennaPatternRecord

Public Methods

	DtAntennaPatternRecord () *default constructor*
	DtAntennaPatternRecord (const DtAntennaPatternRecord& orig) *copy constructor*
virtual	**~DtAntennaPatternRecord** () *destructor*
DtAntennaPatternRecord&	**operator=** (const DtAntennaPatternRecord& orig) *assignment operator*
DtBoolean	**operator==** (const DtAntennaPatternRecord& orig) const *equality operator*
DtBoolean	**operator!=** (const DtAntennaPatternRecord& orig) const *inequality operator*
virtual void	**setFromNet** (const void* net, int size) *Set from network representation*
virtual const void*	**netRep** (int* size = NULL) const *Returns network representation, fills size with size of netRep*
virtual int	**netSize** () const *Returns size of network representation*
virtual DtAntennaPatternRecord*	**clone** () const *Returns a new'ed clone of the instance*
virtual void	**printData** () const *prints the data*

Protected Fields

char*	**myNetRep**
int	**myNetSize**

Documentation

class DtAntennaPatternRecord: Instances of DtAntennaPatternRecord are

- **`DtAntennaPatternRecord()`**
  default constructor
- **`DtAntennaPatternRecord(const DtAntennaPatternRecord& orig)`**
  copy constructor
- **`virtual ~DtAntennaPatternRecord()`**
  destructor
- **`DtAntennaPatternRecord& operator= (const DtAntennaPatternRecord& orig)`**
  assignment operator
- **`DtBoolean operator== (const DtAntennaPatternRecord& orig) const`**
  equality operator
- **`DtBoolean operator!= (const DtAntennaPatternRecord& orig) const`**
  inequality operator
- **`virtual void setFromNet(const void* net, int size)`**
  Set from network representation
- **`virtual const void* netRep(int* size = NULL) const`**
  Returns network representation, fills size with size of netRep. Return value is pointer to internal object with same lifetime as this. Do not delete.
- **`virtual int netSize() const`**
  Returns size of network representation
- **`virtual DtAntennaPatternRecord* clone() const`**
  Returns a new'ed clone of the instance. Caller deletes.
- **`virtual void printData() const`**
  prints the data
- **`char* myNetRep`**
- **`int myNetSize`**

Direct child classes:

DtBeamAntennaPatternRecord

Alphabetic index *Hierarchy of classes*

In file beamantrec.h:

class DtBeamAntennaPatternRecord: public DtAntennaPatternRecord

class DtBeamAntennaPatternRecord: Instances of DtBeamAntennaPatternRecord are beam antenna pattern records used in conjunction with DtRadioTransmitterStateRepository (TSR)

Inheritance:

DtBeamAntennaPatternRecord - DtAntennaPatternRecord

Public Methods

	DtBeamAntennaPatternRecord () *default constructor*
	DtBeamAntennaPatternRecord (const DtBeamAntennaPatternRecord& orig) *copy constructor*
virtual	**~DtBeamAntennaPatternRecord** () *destructor*
DtBeamAntennaPatternRecord&	**operator=** (const DtBeamAntennaPatternRecord& orig) *assignment operator*
	DtBeamAntennaPatternRecord (const DtAntennaPatternRecord& orig) *Initialize from DtAntennaPatternRecord*
virtual void	**setAngles** (const DtTaitBryan &angles) *Set/Get the beam direction*
virtual const DtTaitBryan&	**angles** () const
virtual void	**setBeamwidth** (DtFloat32 azimuth, DtFloat32 elevation) *Set/Get the azimuth bandwidth and elevation*
virtual DtFloat32	**azimuthBeamwidth** () const *Get the azimuth beam bandwidth*
virtual DtFloat32	**elevationBeamwidth** () const *Get the elevation beam width*
virtual void	**setReferenceSystem** (const DtReferenceSys sys) *Set/Get the beam reference system*
virtual DtReferenceSys	**referenceSystem** () const
virtual void	**setElectricField** (DtFloat32 z, DtFloat32 x) *Set electric field*
virtual DtFloat32	**electricFieldZ** () const *Get electric field Z*

virtual DtFloat32	**electricFieldX** () const *Get electric field X*
virtual void	**setPhase** (DtFloat32 phase) *Set/Get the phase*
virtual DtFloat32	**phase** () const
virtual void	**printData** () const *print the class data*
virtual DtAntennaPatternRecord*	**clone** () const *Returns a clone of the instance*
virtual void	**setFromNet** (const void* net, int size) *Set from network representation*
virtual const void*	**netRep** (int* size = NULL) const *Returns network representation, fills size with size of netRep*
virtual int	**netSize** () const *Returns size of network representation*
virtual void	**setFromNetBeam** (const DtNetBeamAntennaPatternRecord& net) *Set from network representation*
virtual void	**setFromNet** (const DtNetBeamAntennaPatternRecord& net) *Also sets from network representation*
virtual const DtNetBeamAntennaPatternRecord&	**netBeam** () const *Returns network representation*
virtual const DtNetBeamAntennaPatternRecord&	**networkRepresentation** () const *Also returns network representation*
inline const DtNetBeamAntennaPatternRecord&	**networkRepresentation** () const
inline void	**setFromNet** (const DtNetBeamAntennaPatternRecord& net)

Protected Fields

DtTaitBryan	**myBeamDirection**
DtFloat32	**myAzimuthBandwidth**
DtFloat32	**myElevationBeamwidth**
DtReferenceSys	**myReferenceSystem**
DtFloat32	**myEz**
DtFloat32	**myEx**
DtFloat32	**myPhase**

Inherited from DtAntennaPatternRecord:

Public Methods

- DtAntennaPatternRecord& **operator=** (const DtAntennaPatternRecord& orig)
- DtBoolean **operator==** (const DtAntennaPatternRecord& orig) const
- DtBoolean **operator!=** (const DtAntennaPatternRecord& orig) const

Protected Fields

- char* **myNetRep**
- int **myNetSize**

Documentation

class DtBeamAntennaPatternRecord:

Instances of DtBeamAntennaPatternRecord are beam antenna pattern records used in conjunction with DtRadioTransmitterStateRepository (TSR). A DtAntennaPatternRecord obtained using the TSR's antennaPattern() inspector can be used to initialize a DtBeamAntennaPatternRecord, whose inspectors you can then use to view the data (assuming the AntennaPattern Record really contained BeamAntennaPattern data. Similarly, a DtBeamAntennaPatternRecord can be filled out and passed to TSR's setAntennaPattern.

- **`DtBeamAntennaPatternRecord()`**
  default constructor
- **`DtBeamAntennaPatternRecord(const DtBeamAntennaPatternRecord& orig)`**
  copy constructor
- **`virtual ~DtBeamAntennaPatternRecord()`**
  destructor
- **`DtBeamAntennaPatternRecord& operator=( const DtBeamAntennaPatternRecord& orig)`**
  assignment operator
- **`DtBeamAntennaPatternRecord(const DtAntennaPatternRecord& orig)`**
  Initialize from DtAntennaPatternRecord
- **`virtual void setAngles(const DtTaitBryan &angles)`**
  Set/Get the beam direction
- **`virtual const DtTaitBryan& angles() const`**
- **`virtual void setBeamwidth(DtFloat32 azimuth, DtFloat32 elevation)`**
  Set/Get the azimuth bandwidth and elevation
- **`virtual DtFloat32 azimuthBeamwidth() const`**
  Get the azimuth beam bandwidth
- **`virtual DtFloat32 elevationBeamwidth() const`**
  Get the elevation beam width
- **`virtual void setReferenceSystem(const DtReferenceSys sys)`**
  Set/Get the beam reference system
- **`virtual DtReferenceSys referenceSystem() const`**

- **virtual void setElectricField(DtFloat32 z, DtFloat32 x)**
  Set electric field
- **virtual DtFloat32 electricFieldZ() const**
  Get electric field Z
- **virtual DtFloat32 electricFieldX() const**
  Get electric field X
- **virtual void setPhase(DtFloat32 phase)**
  Set/Get the phase
- **virtual DtFloat32 phase() const**
- **virtual void printData() const**
  print the class data
- **virtual DtAntennaPatternRecord* clone() const**
  Returns a clone of the instance
- **virtual void setFromNet(const void* net, int size)**
  Set from network representation
- **virtual const void* netRep(int* size = NULL) const**
  Returns network representation, fills size with size of netRep
- **virtual int netSize() const**
  Returns size of network representation
- **virtual void setFromNetBeam(const DtNetBeamAntennaPatternRecord& net)**
  Set from network representation
- **virtual void setFromNet(const DtNetBeamAntennaPatternRecord& net)**
  Also sets from network representation. This function is here for backwards compatibility with pre-3.3 releases.
- **virtual const DtNetBeamAntennaPatternRecord& netBeam() const**
  Returns network representation
- **virtual const DtNetBeamAntennaPatternRecord& networkRepresentation() const**
  Also returns network representation. This function is here for backwards compatibility with pre-3.3 releases.
- **DtTaitBryan myBeamDirection**
- **DtFloat32 myAzimuthBandwidth**
- **DtFloat32 myElevationBeamwidth**
- **DtReferenceSys myReferenceSystem**
- **DtFloat32 myEz**
- **DtFloat32 myEx**
- **DtFloat32 myPhase**
- **inline const DtNetBeamAntennaPatternRecord& networkRepresentation() const**

`inline void setFromNet( const DtNetBeamAntennaPatternRecord& net)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file taitbryan.h:

class DtTaitBryan

Public Methods

	DtTaitBryan ()
	DtTaitBryan (double a, double b, double c)
	DtTaitBryan (const DtTaitBryan &tait)
DtTaitBryan&	**operator=** (const DtTaitBryan &tait)
int	**operator==** (const DtTaitBryan& other) const *Get whether another tait bryan equals or does not equal this one*
int	**operator!=** (const DtTaitBryan& other) const
const char*	**string** () const
DtFloat64	**psi** () const
DtFloat64	**theta** () const
DtFloat64	**phi** () const
void	**setPsi** (DtFloat64 psi)
void	**setTheta** (DtFloat64 theta)
void	**setPhi** (DtFloat64 phi)
inline	**DtTaitBryan** () *Inline functions*
inline	**DtTaitBryan** (double a, double b, double c)
inline	**DtTaitBryan** (const DtTaitBryan &tait)
inline int	**operator!=** (const DtTaitBryan& other) const
inline DtTaitBryan&	**operator=** (const DtTaitBryan &tait)
inline int	**operator==** (const DtTaitBryan& other) const
inline DtFloat64	**phi** () const
inline DtFloat64	**psi** () const
inline void	**setPhi** (DtFloat64 phi)
inline void	**setPsi** (DtFloat64 psi)
inline void	**setTheta** (DtFloat64 theta)
inline const char*	**string** () const
inline DtFloat64	**theta** () const

Protected Fields

- DtFloat64 **myPsi**
- DtFloat64 **myTheta**
  about reference z
- DtFloat64 **myPhi**
  about intermediate y
-
  about body x

Documentation

- **`DtTaitBryan()`**
- **`DtTaitBryan(double a, double b, double c)`**
- **`DtTaitBryan(const DtTaitBryan &tait)`**
- **`DtTaitBryan& operator=(const DtTaitBryan &tait)`**
- **`int operator==(const DtTaitBryan& other) const`**
  Get whether another tait bryan equals or does not equal this one
- **`int operator!=(const DtTaitBryan& other) const`**
- **`const char* string() const`**
- **`DtFloat64 psi() const`**
- **`DtFloat64 theta() const`**
- **`DtFloat64 phi() const`**
- **`void setPsi(DtFloat64 psi)`**
- **`void setTheta(DtFloat64 theta)`**
- **`void setPhi(DtFloat64 phi)`**
- **`DtFloat64 myPsi`**
- **`DtFloat64 myTheta`**
  about reference z
- **`DtFloat64 myPhi`**
  about intermediate y
-
  about body x
- **`inline DtTaitBryan()`**
  Inline functions
- **`inline DtTaitBryan(double a, double b, double c)`**

- `inline DtTaitBryan(const DtTaitBryan &tait)`
- `inline int operator!=( const DtTaitBryan& other) const`
- `inline DtTaitBryan& operator=(const DtTaitBryan &tait)`
- `inline int operator==( const DtTaitBryan& other) const`
- `inline DtFloat64 phi() const`
- `inline DtFloat64 psi() const`
- `inline void setPhi(DtFloat64 phi)`
- `inline void setPsi(DtFloat64 psi)`
- `inline void setTheta(DtFloat64 theta)`
- `inline const char* string() const`
- `inline DtFloat64 theta() const`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtReferenceSys

- **DtReferenceSysWorldCoordinates**
- **DtReferenceSysEntityCoordinates**

Documentation

- `DtReferenceSysWorldCoordinates`
- `DtReferenceSysEntityCoordinates`

In file approximator.h:

class DtApproximator

class DtApproximator: DtApproximator is an abstract base class

---

Inheritance:

DtApproximator

Public Methods

virtual	**~DtApproximator** () *destructor*
DtApproximator&	**operator=** (const DtApproximator& orig) *assignment operator*
virtual DtApproximator*	**clone** () const *Create a copy of this object*
virtual void	**setLocation** (DtConstVector loc, DtTime t = DtGetSimTime()) *Set absolute location*
virtual DtConstVector	**approxLocation** (DtTime t = DtGetSimTime()) const *Get the approximated location at the specified time*
virtual void	**setVelocity** (DtConstVector vel, DtTime t = DtGetSimTime()) *Set absolute velocity*
virtual DtConstVector	**approxVelocity** (DtTime t = DtGetSimTime()) const *Get the approximated velocity at the specified time*
virtual void	**setOrientation** (const DtTaitBryan& ori, DtTime t = DtGetSimTime()) *Set absolute orientation*
virtual const DtTaitBryan&	**approxEuler** (DtTime t = DtGetSimTime()) const *Get approximated orientation (euler angles) at the specified time*
virtual void	**setAcceleration** (DtConstVector acc, DtTime t = DtGetSimTime()) *Set/Get absolute acceleration*
virtual DtConstVector	**acceleration** (DtTime t = DtGetSimTime()) const
virtual void	**setRotationalVelocity** (DtConstVector vel, DtTime t = DtGetSimTime()) *Set/Get absolute rotational velocity*
virtual DtConstVector	**rotationalVelocity** (DtTime t = DtGetSimTime()) const
virtual DtConstDcm	**approxBodyToGeoc** (DtTime t = DtGetSimTime()) const *Get the approximated data body to geocentric coordinate matrix*
	DtApproximator () *default constructor*

Protected Methods

- **DtApproximator** (const DtApproximator& orig)
  copy constructor

Protected

- *Constructors are protected since this is an abstract base class*

Documentation

class DtApproximator:

DtApproximator is an abstract base class. It is used to provide approximated position and orientation information for entities. Position and orientation information is approximated over time from the last set absolute (non-approximated) information.

- **`virtual ~DtApproximator()`**

  destructor

- **`DtApproximator& operator=(const DtApproximator& orig)`**

  assignment operator

- **`virtual DtApproximator* clone() const`**

  Create a copy of this object. Memory is allocated and it is the responsibility of the caller to free the memory via delete.

- **`virtual void setLocation(DtConstVector loc, DtTime t = DtGetSimTime())`**

  Set absolute location

- **`virtual DtConstVector approxLocation( DtTime t = DtGetSimTime()) const`**

  Get the approximated location at the specified time

- **`virtual void setVelocity(DtConstVector vel, DtTime t = DtGetSimTime())`**

  Set absolute velocity

- **`virtual DtConstVector approxVelocity( DtTime t = DtGetSimTime()) const`**

  Get the approximated velocity at the specified time

- **`virtual void setOrientation(const DtTaitBryan& ori, DtTime t = DtGetSimTime())`**

  Set absolute orientation

- **`virtual const DtTaitBryan& approxEuler( DtTime t = DtGetSimTime()) const`**

  Get approximated orientation (euler angles) at the specified time

- **`virtual void setAcceleration(DtConstVector acc, DtTime t = DtGetSimTime())`**

  Set/Get absolute acceleration

- **`virtual DtConstVector acceleration( DtTime t = DtGetSimTime()) const`**

- **`virtual void setRotationalVelocity(DtConstVector vel, DtTime t = DtGetSimTime())`**

  Set/Get absolute rotational velocity

- **`virtual DtConstVector rotationalVelocity( DtTime t = DtGetSimTime()) const`**

- **`virtual DtConstDcm approxBodyToGeoc(DtTime t = DtGetSimTime()) const`**

  Get the approximated data body to geocentric coordinate matrix

-

  Constructors are protected since this is an abstract base class

- **`DtApproximator()`**

  default constructor

- **`DtApproximator(const DtApproximator& orig)`**

  copy constructor

Direct child classes:

DtDeadReckoner

Alphabetic index *Hierarchy of classes*

In file deadreckoner.h:

class DtDeadReckoner: public DtApproximator

class DtDeadReckoner: Instances of DtDeadReckoner are used to provide approximated position and orientation information for entities using various dead reckoning algorithms

---

Inheritance:

DtDeadReckoner - DtApproximator

Public Methods

	DtDeadReckoner () *default constructor*
virtual	**~DtDeadReckoner** () *destructor*
	DtDeadReckoner (const DtDeadReckoner& orig) *copy constructor*
DtDeadReckoner&	**operator=** (const DtDeadReckoner& orig) *assignment operator*
virtual DtApproximator*	**clone** () const *Create a copy of this object*
virtual void	**setLocation** (DtConstVector loc, DtTime curTime = DtGetSimTime()) *Set absolute location*
virtual DtConstVector	**approxLocation** (DtTime curTime = DtGetSimTime()) const *Get the approximated (i*
virtual DtConstVector	**deadReckonedLocation** (DtTime curTime = DtGetSimTime()) const *Get the dead-reckoned location at the specified time*
virtual void	**setVelocity** (DtConstVector vel, DtTime curTime = DtGetSimTime()) *Set absolute velocity*
virtual DtConstVector	**approxVelocity** (DtTime curTime = DtGetSimTime()) const *Get the approximated (i*
virtual DtConstVector	**deadReckonedVelocity** (DtTime curTime = DtGetSimTime()) const *Get the dead-reckoned velocity at the specified time*
virtual void	**setOrientation** (const DtTaitBryan& ori, DtTime curTime = DtGetSimTime()) *Set absolute orientation*
virtual const DtTaitBryan&	**approxEuler** (DtTime curTime = DtGetSimTime()) const *Get the approximated (i*
virtual const DtTaitBryan&	**deadReckonedEuler** (DtTime curTime = DtGetSimTime()) const *Get dead-reckoned orientation (euler angles) at the specified time*
virtual void	**setAcceleration** (DtConstVector acc, DtTime curTime = DtGetSimTime()) *Set/Get absolute acceleration*
virtual DtConstVector	**acceleration** (DtTime curTime = DtGetSimTime()) const

virtual void	**setRotationalVelocity** (DtConstVector vel, DtTime curTime = DtGetSimTime()) *Set/Get absolute rotational velocity*
virtual DtConstVector	**rotationalVelocity** (DtTime curTime = DtGetSimTime()) const
virtual DtConstDcm	**approxBodyToGeoc** (DtTime curTime = DtGetSimTime()) const *Get the approximated (i*
virtual DtConstDcm	**deadReckonedBodyToGeoc** (DtTime curTime = DtGetSimTime()) const *Get the dead-reckoned body to geocentric coordinate matrix*
virtual void	**setAlgorithm** (DtDeadReckonTypes alg) *Set/Get the dead reckoning algorithm used for generating the approximated values from the absolute values*
virtual DtDeadReckonTypes	**algorithm** () const
static void	**deadReckonPosition** (DtTime dt, DtDeadReckonTypes algorithm, DtConstVector oldPos, DtConstVector oldVel, DtConstVector oldAccel, DtVectorRef newVel, DtVectorRef newPos) *Calculate the dead reckon position information*
static void	**deadReckonOrientation** (DtTime dt, DtDeadReckonTypes algorithm, DtConstDcm old_b2w, const DtDirectedRotRate& drr, DtDcmRef new_b2w) *Calculate the orientation matrix*
inline DtConstVector	**acceleration** (DtTime curTime) const
inline DtDeadReckonTypes	**algorithm** () const
inline DtConstDcm	**approxBodyToGeoc** (DtTime curTime) const
inline const DtTaitBryan&	**approxEuler** (DtTime curTime) const
inline DtConstVector	**approxLocation** (DtTime curTime) const
inline DtConstVector	**approxVelocity** (DtTime curTime) const
inline DtConstDcm	**deadReckonedBodyToGeoc** (DtTime curTime) const
inline const DtTaitBryan&	**deadReckonedEuler** (DtTime curTime) const
inline DtConstVector	**deadReckonedLocation** (DtTime curTime) const
inline DtConstVector	**deadReckonedVelocity** (DtTime curTime) const
inline DtConstVector	**rotationalVelocity** (DtTime curTime) const
inline DtDeadReckoner*	**self** () const
inline void	**setAlgorithm** (DtDeadReckonTypes alg)

Protected Fields

DtVector	**myLocAtTimeOfValidity**
DtVector	**myVelAtTimeOfValidity**
DtTime	**mySimTimePosDataValid**
DtDcm	**myBodyToGeocAtTimeOfValidity**
DtTime	**mySimTimeOrientDataValid**
DtVector	**myDRLoc**

DtVector	**myDRVel**
DtTime	**myDRPosTime**
DtDcm	**myDRBodyToGeoc**
DtTime	**myDRBodyToGeocTime**
DtTaitBryan	**myDREuler**
DtTime	**myDREulerTime**
DtDirectedRotRate	**myDirectedRotRate**
DtDeadReckonTypes	**myAlgorithm**
DtVector	**myAccel**
DtBodyRates	**myRotVel**

Protected Methods

virtual void	**deadReckonLoc** (DtTime curTime) const *If the last dead reckoned position is not valid for the specified time, recalculate it and same the validity time*
virtual void	**deadReckonBodyToGeoc** (DtTime curTime) const *If the last dead reckoned body to geocentri coordinate matrix is not valid for the specified time, recalculate it and same the validity time*
virtual void	**deadReckonEuler** (DtTime curTime) const *If the last dead reckoned orientation (euler angles) is not valid for the specified time, recalculate it and same the validity time*
DtDeadReckoner*	**self** () const *Get rid of constness on the "this" pointer in order to access non-const functions from const member functions (such as simple accessors)*
static int	**isRotational** (DtDeadReckonTypes algorithm) *Returns 1 if the dead reckoning algorithm is a rotational one, 0 otherwise*

Inherited from DtApproximator:

Protected

Documentation

class DtDeadReckoner:

Instances of DtDeadReckoner are used to provide approximated position and orientation information for entities using various dead reckoning algorithms. Position and orientation information is approximated over time from the last set absolute (non-approximated) information. Functions that start with deadReckoned return dead-reckoned values. Functions that start with approx return approximated values - which default to the dead-reckoned values, but this might be overridden in classes derived from DtDeadReckoner.

- **`DtDeadReckoner()`**
  default constructor
- **`virtual ~DtDeadReckoner()`**
  destructor
- **`DtDeadReckoner(const DtDeadReckoner& orig)`**
  copy constructor
- **`DtDeadReckoner& operator=(const DtDeadReckoner& orig)`**
  assignment operator
- **`virtual DtApproximator* clone() const`**
  Create a copy of this object. Memory is allocated and it is the responsibility of the caller to free the memory via delete.
- **`virtual void setLocation(DtConstVector loc, DtTime curTime = DtGetSimTime())`**
  Set absolute location
- **`virtual DtConstVector approxLocation( DtTime curTime = DtGetSimTime()) const`**
  Get the approximated (i.e. dead-reckoned for base DtDeadReckoner) location at the specified time.
- **`virtual DtConstVector deadReckonedLocation( DtTime curTime = DtGetSimTime()) const`**
  Get the dead-reckoned location at the specified time
- **`virtual void setVelocity(DtConstVector vel, DtTime curTime = DtGetSimTime())`**
  Set absolute velocity
- **`virtual DtConstVector approxVelocity( DtTime curTime = DtGetSimTime()) const`**
  Get the approximated (i.e. dead-reckoned for base DtDeadReckoner) velocity at the specified time.
- **`virtual DtConstVector deadReckonedVelocity( DtTime curTime = DtGetSimTime()) const`**
  Get the dead-reckoned velocity at the specified time
- **`virtual void setOrientation(const DtTaitBryan& ori, DtTime curTime = DtGetSimTime())`**
  Set absolute orientation
- **`virtual const DtTaitBryan& approxEuler( DtTime curTime = DtGetSimTime()) const`**
  Get the approximated (i.e. dead-reckoned for base DtDeadReckoner) orientation at the specified time.
- **`virtual const DtTaitBryan& deadReckonedEuler( DtTime curTime = DtGetSimTime()) const`**
  Get dead-reckoned orientation (euler angles) at the specified time
- **`virtual void setAcceleration(DtConstVector acc, DtTime curTime = DtGetSimTime())`**
  Set/Get absolute acceleration
- **`virtual DtConstVector acceleration( DtTime curTime = DtGetSimTime()) const`**
- **`virtual void setRotationalVelocity(DtConstVector vel, DtTime curTime = DtGetSimTime())`**
  Set/Get absolute rotational velocity
- **`virtual DtConstVector rotationalVelocity( DtTime curTime = DtGetSimTime()) const`**
- **`virtual DtConstDcm approxBodyToGeoc( DtTime curTime = DtGetSimTime()) const`**

Get the approximated (i.e. dead-reckoned for base DtDeadReckoner) body to geocentric coordinate matrix.

- **virtual DtConstDcm deadReckonedBodyToGeoc(DtTime curTime = DtGetSimTime()) const**

  Get the dead-reckoned body to geocentric coordinate matrix

- **virtual void setAlgorithm(DtDeadReckonTypes alg)**

  Set/Get the dead reckoning algorithm used for generating the approximated values from the absolute values

- **virtual DtDeadReckonTypes algorithm() const**

- **static void deadReckonPosition(DtTime dt, DtDeadReckonTypes algorithm, DtConstVector oldPos, DtConstVector oldVel, DtConstVector oldAccel, DtVectorRef newVel, DtVectorRef newPos)**

  Calculate the dead reckon position information

- **static void deadReckonOrientation(DtTime dt, DtDeadReckonTypes algorithm, DtConstDcm old_b2w, const DtDirectedRotRate& drr, DtDcmRef new_b2w)**

  Calculate the orientation matrix

- **virtual void deadReckonLoc(DtTime curTime) const**

  If the last dead reckoned position is not valid for the specified time, recalculate it and same the validity time

- **virtual void deadReckonBodyToGeoc(DtTime curTime) const**

  If the last dead reckoned body to geocentri coordinate matrix is not valid for the specified time, recalculate it and same the validity time

- **virtual void deadReckonEuler(DtTime curTime) const**

  If the last dead reckoned orientation (euler angles) is not valid for the specified time, recalculate it and same the validity time

- **DtDeadReckoner* self() const**

  Get rid of constness on the "this" pointer in order to access non-const functions from const member functions (such as simple accessors). This should be used sparingly since it bypasses the const protection. Note: This is essentially like an operator and is non-virtual since can't override based on return type. Allows derived classes to also define this member.

- **static int isRotational(DtDeadReckonTypes algorithm)**

  Returns 1 if the dead reckoning algorithm is a rotational one, 0 otherwise

- **DtVector myLocAtTimeOfValidity**

- **DtVector myVelAtTimeOfValidity**

- **DtTime mySimTimePosDataValid**

- **DtDcm myBodyToGeocAtTimeOfValidity**

- **DtTime mySimTimeOrientDataValid**

- **DtVector myDRLoc**

- **DtVector myDRVel**

- **DtTime myDRPosTime**

- **DtDcm myDRBodyToGeoc**

- DtTime myDRBodyToGeocTime
- DtTaitBryan myDREuler
- DtTime myDREulerTime
- DtDirectedRotRate myDirectedRotRate
- DtDeadReckonTypes myAlgorithm
- DtVector myAccel
- DtBodyRates myRotVel
- inline DtConstVector acceleration(DtTime curTime) const
- inline DtDeadReckonTypes algorithm() const
- inline DtConstDcm approxBodyToGeoc(DtTime curTime) const
- inline const DtTaitBryan& approxEuler(DtTime curTime) const
- inline DtConstVector approxLocation(DtTime curTime) const
- inline DtConstVector approxVelocity(DtTime curTime) const
- inline DtConstDcm deadReckonedBodyToGeoc(DtTime curTime) const
- inline const DtTaitBryan& deadReckonedEuler(DtTime curTime) const
- inline DtConstVector deadReckonedLocation(DtTime curTime) const
- inline DtConstVector deadReckonedVelocity(DtTime curTime) const
- inline DtConstVector rotationalVelocity(DtTime curTime) const
- inline DtDeadReckoner* self() const
- inline void setAlgorithm(DtDeadReckonTypes alg)

Direct child classes:

DtSmoother

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtDeadReckonTypes

Section 7

- **DtDrOther**
- **DtDrStatic**
- **DtDrDrmFpw**
- **DtDrDrmRpw**
- **DtDrDrmRvw**
- **DtDrDrmFvw**
- **DtDrDrmFpb**
- **DtDrDrmRpb**
- **DtDrDrmRvb**
- **DtDrDrmFvb**

Documentation

Section 7

- `DtDrOther`
- `DtDrStatic`
- `DtDrDrmFpw`
- `DtDrDrmRpw`
- `DtDrDrmRvw`
- `DtDrDrmFvw`
- `DtDrDrmFpb`
- `DtDrDrmRpb`
- `DtDrDrmRvb`
- `DtDrDrmFvb`

In file directrotrate.h:

class DtDirectedRotRate

rotation rate given by single rate and axis of rotation

Public Fields

- double **thetaDot**
- double **n** [3]
  radians per sec
- *axis unit vector*

Public Methods

- **DtDirectedRotRate** ()
- **DtDirectedRotRate** (DtConstVector pqr)
- void **setFromVec** (DtConstVector pqr)
- inline **DtDirectedRotRate** ()
  Inline functions
- inline **DtDirectedRotRate** (DtConstVector pqr)
- inline void **setFromVec** (DtConstVector pqr)

Documentation

rotation rate given by single rate and axis of rotation

- `DtDirectedRotRate()`
- `DtDirectedRotRate(DtConstVector pqr)`
- `void setFromVec(DtConstVector pqr)`
- `double thetaDot`
- `double n[3]`
  radians per sec
-
  axis unit vector
- `inline DtDirectedRotRate()`
  Inline functions

- `inline DtDirectedRotRate(DtConstVector pqr)`
- `inline void setFromVec(DtConstVector pqr)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vlvector.h:

class DtVector

class DtVector: No virtual functions, because we want the size of a DtVector to 3 * 8 bytes - so that you can make the same assuptions about it that we used to make about its size and layout

---

Inheritance:

DtVector

Public Methods

	DtVector () *default constructor*
	DtVector (double x, double y, double z) *construct from three doubles*
	~DtVector () *destructor*
	DtVector (const DtVector& vec) *copy constructor*
DtVector&	**operator=** (const DtVector& vec) *assignment operator*
	operator double* () *Convert the vector to a pointer to double*
	operator const double* () const
double&	**operator[]** (int ind) *Get the value for the specified subscript*
double	**operator[]** (int ind) const
double&	**operator[]** (DtVectorOffset ind) *Get the value for the specified subscript*
double	**operator[]** (DtVectorOffset ind) const
int	**operator==** (const DtVector& vec) const *Get whether another vector equals this vector*
const char*	**string** () const *Get a string representation of the vector*
static DtConstVector	**zero** () *Get a zero'ed vector*
static DtConstVector	**i** () *Get a (1,0,0) vector*
static DtConstVector	**j** () *Get a (0,1,0) vector*

- static DtConstVector **k** ()
  Get a (0,0,1) vector
- static DtConstVector **ones** ()
  Get a vector set to all ones
- inline **DtVector** ()
- inline **DtVector** (double a, double b, double c)
- inline **DtVector** (const DtVector& vec)
- inline **operator const double*** () const
- inline **operator double*** ()
- inline DtVector& **operator=** (const DtVector& vec)
- inline int **operator==** (const DtVector& vec) const
- inline double& **operator[]** (DtVectorOffset ind)
  Inline DtVector functions
- inline double **operator[]** (DtVectorOffset ind) const
- inline double& **operator[]** (int ind)
- inline double **operator[]** (int ind) const
- inline **~DtVector** ()

Protected Fields

- double **myRep** [3]

Documentation

class DtVector:

No virtual functions, because we want the size of a DtVector to 3 * 8 bytes - so that you can make the same assuptions about it that we used to make about its size and layout. This is especially important in DtDcm, where we use an array of 3 DtVectors as the myRep. If DtVector had any virtual functions, there would be 4 bytes between vector reps.

- **`DtVector()`**
  default constructor
- **`DtVector(double x, double y, double z)`**
  construct from three doubles
- **`~DtVector()`**
  destructor
- **`DtVector(const DtVector& vec)`**
  copy constructor
- **`DtVector& operator=(const DtVector& vec)`**
  assignment operator

- **operator double* ()**
  Convert the vector to a pointer to double
- **operator const double* () const**
- **double& operator[](int ind)**
  Get the value for the specified subscript. (int version)
- **double operator[](int ind) const**
- **double& operator[](DtVectorOffset ind)**
  Get the value for the specified subscript. (enum version)
- **double operator[](DtVectorOffset ind) const**
- **int operator==(const DtVector& vec) const**
  Get whether another vector equals this vector
- **const char* string() const**
  Get a string representation of the vector
- **static DtConstVector zero()**
  Get a zero'ed vector
- **static DtConstVector i()**
  Get a (1,0,0) vector
- **static DtConstVector j()**
  Get a (0,1,0) vector
- **static DtConstVector k()**
  Get a (0,0,1) vector
- **static DtConstVector ones()**
  Get a vector set to all ones
- **double myRep[3]**
- **inline DtVector()**
- **inline DtVector(double a, double b, double c)**
- **inline DtVector(const DtVector& vec)**
- **inline operator const double* () const**
- **inline operator double* ()**
- **inline DtVector& operator=(const DtVector& vec)**
- **inline int operator==(const DtVector& vec) const**
- **inline double& operator[](DtVectorOffset ind)**
  Inline DtVector functions
- **inline double operator[](DtVectorOffset ind) const**

- `inline double& operator[](int ind)`
- `inline double operator[](int ind) const`
- `inline ~DtVector()`

Direct child classes:

DtUtmCoord
DtGeodeticCoord

Alphabetic index *Hierarchy of classes*

In file dcm.h:

class DtDcm

Public Methods

	DtDcm ()
	DtDcm (double a, double b, double c, double d, double e, double f, double g, double h, double i)
	DtDcm (DtConstVector a, DtConstVector b, DtConstVector c)
	DtDcm (const DtDcm &mat)
DtDcm&	**operator=** (const DtDcm &mat)
	operator DtVector* ()
	operator const DtVector* () const
DtVector&	**operator[]** (int i)
const DtVector&	**operator[]** (int i) const
void	**print** () const *Prints the data in the matrix*
static DtConstDcm	**zero** ()
static DtConstDcm	**identity** ()
inline	**DtDcm** ()
inline	**DtDcm** (double a, double b, double c, double d, double e, double f, double g, double h, double i)
inline	**DtDcm** (DtConstVector a, DtConstVector b, DtConstVector c)
inline	**DtDcm** (const DtDcm &mat)
inline	**operator DtVector*** ()
inline	**operator const DtVector*** () const
inline DtDcm&	**operator=** (const DtDcm &mat)
inline DtVector&	**operator[]** (int i) *Inline functions*
inline const DtVector&	**operator[]** (int i) const

Protected Fields

- DtVector **rep** [3]

Documentation

- `DtDcm()`
- `DtDcm(double a, double b, double c, double d, double e, double f, double g, double h, double i)`
- `DtDcm(DtConstVector a, DtConstVector b, DtConstVector c)`
- `DtDcm(const DtDcm &mat)`
- `DtDcm& operator=(const DtDcm &mat)`
- `operator DtVector* ()`
- `operator const DtVector* () const`
- `DtVector& operator[](int i)`
- `const DtVector& operator[](int i) const`
- `void print() const`

  Prints the data in the matrix
- `static DtConstDcm zero()`
- `static DtConstDcm identity()`
- `DtVector rep[3]`
- `inline DtDcm()`
- `inline DtDcm(double a, double b, double c, double d, double e, double f, double g, double h, double i)`
- `inline DtDcm(DtConstVector a, DtConstVector b, DtConstVector c)`
- `inline DtDcm(const DtDcm &mat)`
- `inline operator DtVector* ()`
- `inline operator const DtVector* () const`
- `inline DtDcm& operator=(const DtDcm &mat)`
- `inline DtVector& operator[](int i)`

  Inline functions
- `inline const DtVector& operator[](int i) const`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file smoother.h:

class DtSmoother: public DtDeadReckoner

class DtSmoother: Instances of DtSmoother are used to provide approximated position and orientation information for entities

---

Inheritance:

DtSmoother - DtDeadReckoner - DtApproximator

Public Fields

- static DtTime **theDfltSmoothPeriod**

Public Methods

	DtSmoother () *constructor*
virtual	**~DtSmoother** () *destructor*
	DtSmoother (const DtSmoother& orig) *copy constructor*
DtSmoother&	**operator=** (const DtSmoother& orig) *assignment operator*
virtual DtApproximator*	**clone** () const *Create a copy of this object*
virtual void	**setLocation** (DtConstVector loc, DtTime t = DtGetSimTime()) *Set absolute location*
virtual DtConstVector	**approxLocation** (DtTime t = DtGetSimTime()) const *Get the approximated location at the specified time*
virtual void	**setVelocity** (DtConstVector vel, DtTime t = DtGetSimTime()) *Set absolute velocity*
virtual DtConstVector	**approxVelocity** (DtTime t = DtGetSimTime()) const *Get the approximated velocity at the specified time*
virtual void	**setOrientation** (const DtTaitBryan& ori, DtTime t = DtGetSimTime()) *Set absolute orientation*
virtual const DtTaitBryan&	**approxEuler** (DtTime t = DtGetSimTime()) const *Get approximated orientation (euler angles) at the specified time*
virtual void	**setAcceleration** (DtConstVector acc, DtTime t = DtGetSimTime()) *Set/Get absolute acceleration*

- virtual DtConstVector **acceleration** (DtTime t = DtGetSimTime()) const
- virtual void **setRotationalVelocity** (DtConstVector vel, DtTime t = DtGetSimTime())
  Set/Get absolute rotational velocity
- virtual DtConstVector **rotationalVelocity** (DtTime t = DtGetSimTime()) const
- virtual DtConstDcm **approxBodyToGeoc** (DtTime t = DtGetSimTime()) const
  Get the approximated data body to geocentric coordinate matrix
- virtual void **setSmoothEnable** (DtBoolean on)
  Set/Get the individual smoothing switch
- virtual DtBoolean **smoothEnable** () const
- virtual void **setSmoothPeriod** (DtTime smoothPeriod)
  Set/Get smoothing period - the amount of time over which smoothing takes place
- virtual DtTime **smoothPeriod** () const
- static void **setDfltSmoothPeriod** (DtTime smoothPeriod)
  Set/Get the default smoothing period
- static DtTime **dfltSmoothPeriod** ()
- static void **setMasterSmoothEnable** (DtBoolean onOff)
  Set/Get the value of the "master" smoothing-enabled switch
- static DtBoolean **masterSmoothEnable** ()
- inline DtSmoother* **self** () const

Protected Fields

- DtSmoothPosWithDr **mySpos**
- DtSmoothOriWithDr **mySorient**

Protected Methods

- virtual void **feedSmoothing** ()
  provide inputs to the actual smoothing algorithm
- DtSmoother* **self** () const
  Get rid of constness on the "this" pointer in order to access non-const functions from const member functions (such as simple accessors)

Inherited from DtDeadReckoner:

Public Methods

- virtual DtConstVector **deadReckonedLocation**(DtTime curTime = DtGetSimTime()) const
- virtual DtConstVector **deadReckonedVelocity**(DtTime curTime = DtGetSimTime()) const
- virtual const DtTaitBryan& **deadReckonedEuler**(DtTime curTime = DtGetSimTime()) const
- virtual DtConstDcm **deadReckonedBodyToGeoc**(DtTime curTime = DtGetSimTime()) const

- virtual void **setAlgorithm**(DtDeadReckonTypes alg)
- virtual DtDeadReckonTypes **algorithm**() const
- static void **deadReckonPosition**(DtTime dt, DtDeadReckonTypes algorithm, DtConstVector oldPos, DtConstVector oldVel, DtConstVector oldAccel, DtVectorRef newVel, DtVectorRef newPos)
- static void **deadReckonOrientation**(DtTime dt, DtDeadReckonTypes algorithm, DtConstDcm old_b2w, const DtDirectedRotRate& drr, DtDcmRef new_b2w)
- inline DtDeadReckonTypes **algorithm**() const
- inline DtConstDcm **deadReckonedBodyToGeoc**(DtTime curTime) const
- inline const DtTaitBryan& **deadReckonedEuler**(DtTime curTime) const
- inline DtConstVector **deadReckonedLocation**(DtTime curTime) const
- inline DtConstVector **deadReckonedVelocity**(DtTime curTime) const
- inline void **setAlgorithm**(DtDeadReckonTypes alg)

Protected Fields

- DtVector **myLocAtTimeOfValidity**
- DtVector **myVelAtTimeOfValidity**
- DtTime **mySimTimePosDataValid**
- DtDcm **myBodyToGeocAtTimeOfValidity**
- DtTime **mySimTimeOrientDataValid**
- DtVector **myDRLoc**
- DtVector **myDRVel**
- DtTime **myDRPosTime**
- DtDcm **myDRBodyToGeoc**
- DtTime **myDRBodyToGeocTime**
- DtTaitBryan **myDREuler**
- DtTime **myDREulerTime**
- DtDirectedRotRate **myDirectedRotRate**
- DtDeadReckonTypes **myAlgorithm**
- DtVector **myAccel**
- DtBodyRates **myRotVel**

Protected Methods

- virtual void **deadReckonLoc**(DtTime curTime) const
- virtual void **deadReckonBodyToGeoc**(DtTime curTime) const
- virtual void **deadReckonEuler**(DtTime curTime) const
- static int **isRotational**(DtDeadReckonTypes algorithm)

Inherited from DtApproximator:

Protected

-

Documentation

class DtSmoother:

Instances of DtSmoother are used to provide approximated position and orientation information for entities. Position and orientation information is approximated over time from the last set absolute (non-approximated) information. When a new position is set, the current position moves smoothly to reflect the new information, rather than jumping to the new absolute position.

- **`DtSmoother()`**
  constructor
- **`virtual ~DtSmoother()`**
  destructor
- **`DtSmoother(const DtSmoother& orig)`**
  copy constructor
- **`DtSmoother& operator=(const DtSmoother& orig)`**
  assignment operator
- **`virtual DtApproximator* clone() const`**
  Create a copy of this object. Memory is allocated and it is the responsibility of the caller to free the memory via delete.
- **`virtual void setLocation(DtConstVector loc, DtTime t = DtGetSimTime())`**
  Set absolute location
- **`virtual DtConstVector approxLocation( DtTime t = DtGetSimTime()) const`**
  Get the approximated location at the specified time
- **`virtual void setVelocity(DtConstVector vel, DtTime t = DtGetSimTime())`**
  Set absolute velocity

- **virtual DtConstVector approxVelocity(DtTime t = DtGetSimTime()) const**

  Get the approximated velocity at the specified time

- **virtual void setOrientation(const DtTaitBryan& ori, DtTime t = DtGetSimTime())**

  Set absolute orientation

- **virtual const DtTaitBryan& approxEuler(DtTime t = DtGetSimTime()) const**

  Get approximated orientation (euler angles) at the specified time

- **virtual void setAcceleration(DtConstVector acc, DtTime t = DtGetSimTime())**

  Set/Get absolute acceleration

- **virtual DtConstVector acceleration(DtTime t = DtGetSimTime()) const**

- **virtual void setRotationalVelocity(DtConstVector vel, DtTime t = DtGetSimTime())**

  Set/Get absolute rotational velocity

- **virtual DtConstVector rotationalVelocity(DtTime t = DtGetSimTime()) const**

- **virtual DtConstDcm approxBodyToGeoc(DtTime t = DtGetSimTime()) const**

  Get the approximated data body to geocentric coordinate matrix

- **virtual void setSmoothEnable(DtBoolean on)**

  Set/Get the individual smoothing switch. In order for smoothing to be enabled, the "master" smoothing-enabled switch must be on, AND the individual smoothing switch (accessed through these functions), must be on. Inidivdual smoothing switch defaults to on.

- **virtual DtBoolean smoothEnable() const**

- **virtual void setSmoothPeriod(DtTime smoothPeriod)**

  Set/Get smoothing period - the amount of time over which smoothing takes place. Pass a value of -1 to indicate use of the default smoothing period.

- **virtual DtTime smoothPeriod() const**

- **static void setDfltSmoothPeriod(DtTime smoothPeriod)**

  Set/Get the default smoothing period

- **static DtTime dfltSmoothPeriod()**

- **static void setMasterSmoothEnable(DtBoolean onOff)**

  Set/Get the value of the "master" smoothing-enabled switch. Master smoothing switch defaults to on.

- **static DtBoolean masterSmoothEnable()**

- **static DtTime theDfltSmoothPeriod**

- **virtual void feedSmoothing()**

  provide inputs to the actual smoothing algorithm

- **DtSmoother* self() const**

  Get rid of constness on the "this" pointer in order to access non-const functions from const member functions (such as simple accessors). This should be used sparingly since it bypasses the const protection. Note: This is essentially like an operator and is non-virtual since can't override based on return type. Allows derived classes to also define this member.

- `DtSmoothPosWithDr mySpos`
- `DtSmoothOriWithDr mySorient`
- `inline DtSmoother* self() const`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vlvector.h:

enum DtVectorOffset

- **DtX**
- **DtY**
- **DtZ**

Documentation

- `DtX`
- `DtY`
- `DtZ`

In file utmcoord.h:

class DtUtmCoord: public DtVector

class DtUtmCoord: Instances of DtUtmCoord are used to represent UTM coordinates

---

Inheritance:

DtUtmCoord - DtVector

Public Methods

	DtUtmCoord () *default constructor*
	DtUtmCoord (double east, double north, double up) *constructor*
	DtUtmCoord (const DtGeodeticCoord& geodetic) *constructor, construct UTM from geodetic*
virtual	**~DtUtmCoord** () *destructor*
	DtUtmCoord (const DtUtmCoord& orig) *copy constructor*
DtUtmCoord&	**operator=** (const DtUtmCoord& orig) *assignment operator*
DtUtmCoord&	**operator=** (const DtGeodeticCoord& geodetic) *assignment operator, convert geodetic to UTM*
virtual DtUtmCoord&	**setGeodetic** (const DtGeodeticCoord& geodetic) *Set from geodetic coordinate (convert geodetic to UTM)*
virtual void	**getGeodetic** (DtGeodeticCoord& geodetic) const *Get geodetic coordinate equivalent (convert UTM to geodetic)*
virtual DtGeodeticCoord	**geodetic** () const *Get geodetic coordinate equivalent (convert UTM to geodetic)*
virtual DtUtmCoord&	**setGeocentric** (DtConstVector geocentric) *Set from geocentric coordinate (convert geocentric to UTM)*
virtual void	**getGeocentric** (DtVectorRef geocentric) const *Get geocentric coordinate equivalent (convert UTM to geocentric)*
virtual DtVector	**geocentric** () const *Get geocentric coordinate equivalent (convert UTM to geocentric)*
virtual void	**setEast** (DtFloat64 east) *Set/Get east component*
virtual DtFloat64	**east** () const
virtual void	**setNorth** (DtFloat64 north) *Set/Get north component*

- virtual DtFloat64 **north** () const
- virtual void **setUp** (DtFloat64 up)
  Set/Get up component
- virtual DtFloat64 **up** () const
- virtual double **gridDeclination** () const
  Get the heading that northing is off of north
- inline DtFloat64 **east** () const
- inline DtFloat64 **north** () const
- inline void **setEast** (DtFloat64 east)
  inline functions
- inline void **setNorth** (DtFloat64 north)
- inline void **setUp** (DtFloat64 up)
- inline DtFloat64 **up** () const

Public

- *Note: For coordinate conversions, DtUtmInit must be called first*

Private Methods

- friend void **DtGeodToUtm** (const DtGeodeticCoord& gd, DtUtmCoord& utm)
- friend void **DtGeocToUtm** (DtConstVector gd, DtUtmCoord& utm)

Inherited from DtVector:

Public Methods

- **operator double*** ()
- **operator const double*** () const
- double& **operator[]**(int ind)
- double **operator[]**(int ind) const
- double& **operator[]**(DtVectorOffset ind)
- double **operator[]**(DtVectorOffset ind) const
- int **operator==**(const DtVector& vec) const
- const char* **string**() const
- static DtConstVector **zero**()

- static DtConstVector **i**()
- static DtConstVector **j**()
- static DtConstVector **k**()
- static DtConstVector **ones**()
- inline **operator const double*** () const
- inline **operator double*** ()
- inline int **operator==**(const DtVector& vec) const
- inline double& **operator[]**(DtVectorOffset ind)
- inline double **operator[]**(DtVectorOffset ind) const
- inline double& **operator[]**(int ind)
- inline double **operator[]**(int ind) const

Protected Fields

- double **myRep**[3]

Documentation

class DtUtmCoord:

Instances of DtUtmCoord are used to represent UTM coordinates. Universal Transverse Mercator projection coordinate with respect to reference set by DtUtmInit.

- **`friend void DtGeodToUtm(const DtGeodeticCoord& gd, DtUtmCoord& utm)`**
- **`friend void DtGeocToUtm(DtConstVector gd, DtUtmCoord& utm)`**
- Note: For coordinate conversions, DtUtmInit must be called first
- **`DtUtmCoord()`**
  default constructor
- **`DtUtmCoord(double east, double north, double up)`**
  constructor
- **`DtUtmCoord(const DtGeodeticCoord& geodetic)`**
  constructor, construct UTM from geodetic
- **`virtual ~DtUtmCoord()`**
  destructor
- **`DtUtmCoord(const DtUtmCoord& orig)`**
  copy constructor

- **DtUtmCoord& operator=(const DtUtmCoord& orig)**
  assignment operator
- **DtUtmCoord& operator=(const DtGeodeticCoord& geodetic)**
  assignment operator, convert geodetic to UTM
- **virtual DtUtmCoord& setGeodetic(const DtGeodeticCoord& geodetic)**
  Set from geodetic coordinate (convert geodetic to UTM). Can also use assignment operator.
- **virtual void getGeodetic(DtGeodeticCoord& geodetic) const**
  Get geodetic coordinate equivalent (convert UTM to geodetic)
- **virtual DtGeodeticCoord geodetic() const**
  Get geodetic coordinate equivalent (convert UTM to geodetic). Note that this is not returning a reference.
- **virtual DtUtmCoord& setGeocentric(DtConstVector geocentric)**
  Set from geocentric coordinate (convert geocentric to UTM). Can also use assignment operator.
- **virtual void getGeocentric(DtVectorRef geocentric) const**
  Get geocentric coordinate equivalent (convert UTM to geocentric)
- **virtual DtVector geocentric() const**
  Get geocentric coordinate equivalent (convert UTM to geocentric). Note that this is not returning a reference.
- **virtual void setEast(DtFloat64 east)**
  Set/Get east component
- **virtual DtFloat64 east() const**
- **virtual void setNorth(DtFloat64 north)**
  Set/Get north component
- **virtual DtFloat64 north() const**
- **virtual void setUp(DtFloat64 up)**
  Set/Get up component
- **virtual DtFloat64 up() const**
- **virtual double gridDeclination() const**
  Get the heading that northing is off of north
- **inline DtFloat64 east() const**
- **inline DtFloat64 north() const**
- **inline void setEast(DtFloat64 east)**
  inline functions
- **inline void setNorth(DtFloat64 north)**
- **inline void setUp(DtFloat64 up)**
- **inline DtFloat64 up() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file geodcoord.h:

class DtGeodeticCoord: public DtVector

class DtGeodeticCoord: Instances of DtGeodeticCoord are used to explicitly represent Geodetic coordinates

---

Inheritance:

DtGeodeticCoord - DtVector

Public Methods

	DtGeodeticCoord () *default constructor*
	DtGeodeticCoord (double lat, double lon, double alt) *constructor*
virtual	**~DtGeodeticCoord** () *destructor*
	DtGeodeticCoord (const DtGeodeticCoord& orig) *copy constructor*
DtGeodeticCoord&	**operator=** (const DtGeodeticCoord& orig) *assignment operator*
virtual DtGeodeticCoord&	**setGeocentric** (DtConstVector geocentric) *Set from geocentric coordinate (convert geocentric to geodetic)*
virtual void	**getGeocentric** (DtVectorRef geocentric) const *Get geocentric coordinate equivalent (convert geodetic to geocentric)*
virtual DtVector	**geocentric** () const *Get geocentric coordinate equivalent (convert geodetic to geocentric)*
virtual void	**setLat** (DtFloat64 lat) *Set/Get the latitude*
virtual DtFloat64	**lat** () const
virtual void	**setLon** (DtFloat64 lon) *Set/Get the longitude*
virtual DtFloat64	**lon** () const
virtual void	**setAlt** (DtFloat64 alt) *Set/Get the altitude*
virtual DtFloat64	**alt** () const
inline DtFloat64	**alt** () const
inline DtFloat64	**lat** () const *inline functions*
inline DtFloat64	**lon** () const

- inline void **setAlt** (DtFloat64 alt)
- inline void **setLat** (DtFloat64 lat)
- inline void **setLon** (DtFloat64 lon)

Private Methods

- friend void **DtGeocToGeod** (DtConstVector gc, DtGeodeticCoord& gd)

Inherited from DtVector:

Public Methods

- **operator double*** ()
- **operator const double*** () const
- double& **operator[]**(int ind)
- double **operator[]**(int ind) const
- double& **operator[]**(DtVectorOffset ind)
- double **operator[]**(DtVectorOffset ind) const
- int **operator==**(const DtVector& vec) const
- const char* **string**() const
- static DtConstVector **zero**()
- static DtConstVector **i**()
- static DtConstVector **j**()
- static DtConstVector **k**()
- static DtConstVector **ones**()
- inline **operator const double*** () const
- inline **operator double*** ()
- inline int **operator==**(const DtVector& vec) const
- inline double& **operator[]**(DtVectorOffset ind)
- inline double **operator[]**(DtVectorOffset ind) const
- inline double& **operator[]**(int ind)
- inline double **operator[]**(int ind) const

Protected Fields

- double **myRep**[3]

Documentation

class DtGeodeticCoord: Instances of DtGeodeticCoord are used to explicitly represent Geodetic coordinates

- **`friend void DtGeocToGeod(DtConstVector gc, DtGeodeticCoord& gd)`**
- **`DtGeodeticCoord()`**
  default constructor
- **`DtGeodeticCoord(double lat, double lon, double alt)`**
  constructor
- **`virtual ~DtGeodeticCoord()`**
  destructor
- **`DtGeodeticCoord(const DtGeodeticCoord& orig)`**
  copy constructor
- **`DtGeodeticCoord& operator=(const DtGeodeticCoord& orig)`**
  assignment operator
- **`virtual DtGeodeticCoord& setGeocentric(DtConstVector geocentric)`**
  Set from geocentric coordinate (convert geocentric to geodetic). Can also use assignment operator.
- **`virtual void getGeocentric(DtVectorRef geocentric) const`**
  Get geocentric coordinate equivalent (convert geodetic to geocentric)
- **`virtual DtVector geocentric() const`**
  Get geocentric coordinate equivalent (convert geodetic to geocentric). Note that this is not returning a reference.
- **`virtual void setLat(DtFloat64 lat)`**
  Set/Get the latitude
- **`virtual DtFloat64 lat() const`**
- **`virtual void setLon(DtFloat64 lon)`**
  Set/Get the longitude
- **`virtual DtFloat64 lon() const`**
- **`virtual void setAlt(DtFloat64 alt)`**
  Set/Get the altitude
- **`virtual DtFloat64 alt() const`**
- **`inline DtFloat64 alt() const`**
- **`inline DtFloat64 lat() const`**

inline functions

- `inline DtFloat64 lon() const`
- `inline void setAlt(DtFloat64 alt)`
- `inline void setLat(DtFloat64 lat)`
- `inline void setLon(DtFloat64 lon)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file smoothposwdr.h:

class DtSmoothPosWithDr: public DtSmoothPos

class DtSmoothPosWithDr: Instances of DtSmoothPosWithDr are

Inheritance:

DtSmoothPosWithDr - DtSmoothPos

Public Methods

- **DtSmoothPosWithDr** (DtDeadReckoner* dr, DtTime smoothT)
  default constructor
- virtual **~DtSmoothPosWithDr** ()
  destructor

Protected Methods

- **DtSmoothPosWithDr** (const DtSmoothPosWithDr& orig)
  copy constructor
- DtSmoothPosWithDr& **operator=** (const DtSmoothPosWithDr& orig)
  assignment operator
- virtual void **dfltAction** (DtTime simTime, void* usr, DtVectorRef vs, DtVectorRef ps)
  The default action is to dead-reckon

Inherited from DtSmoothPos:

Public Methods

- void **input**(DtConstVector pos, DtConstVector vel, DtConstVector acc, DtTime simTime, int freeze=0)
- void **tick**(DtTime simTime)
- DtConstVector **position**() const
- DtConstVector **velocity**() const
- virtual void **setSmoothEnable**(int on)
- virtual int **smoothEnable**() const

- virtual void **setSmoothPeriod**(DtTime t)
- virtual DtTime **smoothPeriod**() const
- static void **setDfltSmoothPeriod**(DtTime t)
- static DtTime **dfltSmoothPeriod**()
- static void **setMasterSmoothEnable**(DtBoolean onOff)
- static DtBoolean **masterSmoothEnable**()
- inline DtTime **smoothPeriod**() const

Protected Fields

- int **frozen**
- DtTime **t0**
- DtTime **t1**
- DtTime **t2**
- DtTime **lastT**
- DtTime **myLocalTau**
- DtTime **myLocalHtau**
- DtTime **myLocalTauInv**
- DtTime* **myTau**
- DtTime* **myHtau**
- DtTime* **myTauInv**
- DtVector **ps**
- DtVector **vs**
- DtVector **p0**
- DtVector **v0**
- DtVector **dp**
- DtVector **dv**
- DtVector **p1**
- DtVector **v1**
- DtVector **a1**
- DtVector **a2**
- DtVector **ap**

- int **myEnable**
- int **needT1data**
- int **needT2data**
- void* **usr**
- static DtTime **theDfltTau**
- static DtTime **theDfltHtau**
- static DtTime **theDfltTauInv**
- static DtBoolean **theMasterSmoothEnable**

Protected Methods

- void **init**(void *u, DtTime t)
- void **setT1data**()

Documentation

class DtSmoothPosWithDr: Instances of DtSmoothPosWithDr are

- **`DtSmoothPosWithDr(DtDeadReckoner* dr, DtTime smoothT)`**
  default constructor
- **`virtual ~DtSmoothPosWithDr()`**
  destructor
- **`DtSmoothPosWithDr(const DtSmoothPosWithDr& orig)`**
  copy constructor
- **`DtSmoothPosWithDr& operator=(const DtSmoothPosWithDr& orig)`**
  assignment operator
- **`virtual void dfltAction(DtTime simTime, void* usr, DtVectorRef vs, DtVectorRef ps)`**
  The default action is to dead-reckon

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file smoothoriwdr.h:

class DtSmoothOriWithDr: public DtSmoothOrient

class DtSmoothOriWithDr: Instances of DtSmoothOriWithDr are

Inheritance:

DtSmoothOriWithDr - DtSmoothOrient

Public Methods

- **DtSmoothOriWithDr** (DtDeadReckoner* dr, DtTime smoothT)
  default constructor
- virtual **~DtSmoothOriWithDr** ()
  destructor

Protected Methods

- **DtSmoothOriWithDr** (const DtSmoothOriWithDr& orig)
  copy constructor
- DtSmoothOriWithDr& **operator=** (const DtSmoothOriWithDr& orig)
  assignment operator
- virtual void **dfltAction** (DtTime simTime, void *usr, DtDcmRef os)
  The default action is to dead-reckon

Inherited from DtSmoothOrient:

Public Methods

- void **input**(DtConstDcm orient, const DtDirectedRotRate &drr, DtTime simTime, int freeze=0)
- void **tick**(DtTime simTime)
- DtConstDcm **orientation**() const
- virtual void **setSmoothEnable**(int on)
- virtual int **smoothEnable**() const
- virtual void **setSmoothPeriod**(DtTime t)

- virtual DtTime **smoothPeriod**() const
- static void **setDfltSmoothPeriod**(DtTime t)
- static DtTime **dfltSmoothPeriod**()
- static void **setMasterSmoothEnable**(DtBoolean onOff)
- static DtBoolean **masterSmoothEnable**()
- inline DtTime **smoothPeriod**() const

Protected Fields

- int **frozen**
- DtTime **t0**
- DtTime **t2**
- DtTime **lastT**
- DtTime* **myTau**
- DtTime **myLocalTau**
- DtDcm **os**
- DtDcm **o0**
- DtVector **ws**
- int **myEnable**
- void* **usr**
- static DtTime **theDfltTau**
- static DtBoolean **theMasterSmoothEnable**

Protected Methods

- void **init**(void *u, DtTime t)
- void **setT1data**()

Documentation

class DtSmoothOriWithDr: Instances of DtSmoothOriWithDr are

- **`DtSmoothOriWithDr(DtDeadReckoner* dr, DtTime smoothT)`**

  default constructor

- **`virtual ~DtSmoothOriWithDr()`**

destructor

- **`DtSmoothOriWithDr(const DtSmoothOriWithDr& orig)`**

  copy constructor

- **`DtSmoothOriWithDr& operator=(const DtSmoothOriWithDr& orig)`**

  assignment operator

- **`virtual void dfltAction(DtTime simTime, void *usr, DtDcmRef os)`**

  The default action is to dead-reckon

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file smoothutil.h:

class DtSmoothPos

---------- POSITION ----------

Inheritance:

DtSmoothPos

Public Methods

	DtSmoothPos (void *usr, DtTime smoothT, DtConstVector pos, DtConstVector vel, DtConstVector acc) *Constructors*
	DtSmoothPos (void *usr, DtTime smoothT) *WARNING: not fully constructed until input() & tick() are called*
virtual	**~DtSmoothPos** ()
void	**input** (DtConstVector pos, DtConstVector vel, DtConstVector acc, DtTime simTime, int freeze=0)
void	**tick** (DtTime simTime)
DtConstVector	**position** () const
DtConstVector	**velocity** () const
virtual void	**setSmoothEnable** (int on) *Set/Get the individual smoothing switch*
virtual int	**smoothEnable** () const
virtual void	**setSmoothPeriod** (DtTime t) *Set/Get smoothPeriod*
virtual DtTime	**smoothPeriod** () const
static void	**setDfltSmoothPeriod** (DtTime t) *Set/Get default smooth period*
static DtTime	**dfltSmoothPeriod** ()
static void	**setMasterSmoothEnable** (DtBoolean onOff) *Set/Get the value of the "master" smoothing-enabled switch*
static DtBoolean	**masterSmoothEnable** ()
inline	**DtSmoothPos** (const DtSmoothPos&)
inline DtTime	**smoothPeriod** () const

Protected Fields

int	**frozen**
DtTime	**t0**
DtTime	**t1**
DtTime	**t2**
DtTime	**lastT**
DtTime	**myLocalTau**
DtTime	**myLocalHtau**
DtTime	**myLocalTauInv**
DtTime*	**myTau**
DtTime*	**myHtau**
DtTime*	**myTauInv**
DtVector	**ps**
DtVector	**vs**
DtVector	**p0**
DtVector	**v0**
DtVector	**dp**
DtVector	**dv**
DtVector	**p1**
DtVector	**v1**
DtVector	**a1**
DtVector	**a2**
DtVector	**ap**
int	**myEnable**
int	**needT1data**
int	**needT2data**
void*	**usr**
static DtTime	**theDfltTau**
static DtTime	**theDfltHtau**
static DtTime	**theDfltTauInv**
static DtBoolean	**theMasterSmoothEnable**

Protected Methods

- **DtSmoothPos** (const DtSmoothPos&)
- virtual void **dfltAction** (DtTime simTime, void *usr, DtVectorRef vs, DtVectorRef ps)
- void **init** (void *u, DtTime t)
- void **setT1data** ()

Documentation

---------- POSITION ----------

- **`DtSmoothPos(void *usr, DtTime smoothT, DtConstVector pos, DtConstVector vel, DtConstVector acc)`**

  Constructors. smoothT is the initial smoothPeriod. Use a value of -1 to indicate use of the default period.

- **`DtSmoothPos(void *usr, DtTime smoothT)`**

  WARNING: not fully constructed until input() & tick() are called

- **`virtual ~DtSmoothPos()`**
- **`void input(DtConstVector pos, DtConstVector vel, DtConstVector acc, DtTime simTime, int freeze=0)`**
- **`void tick(DtTime simTime)`**
- **`DtConstVector position() const`**
- **`DtConstVector velocity() const`**
- **`virtual void setSmoothEnable(int on)`**

  Set/Get the individual smoothing switch. In order for smoothing to be enabled, the "master" smoothing-enabled switch must be on, AND the individual smoothing switch (accessed through these functions), must be on.

- **`virtual int smoothEnable() const`**
- **`virtual void setSmoothPeriod(DtTime t)`**

  Set/Get smoothPeriod. If set to -1, smooth period reverts to the current default smooth period.

- **`virtual DtTime smoothPeriod() const`**
- **`static void setDfltSmoothPeriod(DtTime t)`**

  Set/Get default smooth period

- **`static DtTime dfltSmoothPeriod()`**
- **`static void setMasterSmoothEnable(DtBoolean onOff)`**

  Set/Get the value of the "master" smoothing-enabled switch

- **`static DtBoolean masterSmoothEnable()`**
- **`DtSmoothPos(const DtSmoothPos&)`**
- **`virtual void dfltAction(DtTime simTime, void *usr, DtVectorRef vs, DtVectorRef ps)`**

- void init(void *u, DtTime t)
- void setT1data()
- int frozen
- DtTime t0
- DtTime t1
- DtTime t2
- DtTime lastT
- DtTime myLocalTau
- DtTime myLocalHtau
- DtTime myLocalTauInv
- DtTime* myTau
- DtTime* myHtau
- DtTime* myTauInv
- DtVector ps
- DtVector vs
- DtVector p0
- DtVector v0
- DtVector dp
- DtVector dv
- DtVector p1
- DtVector v1
- DtVector a1
- DtVector a2
- DtVector ap
- int myEnable
- int needT1data
- int needT2data
- void* usr
- static DtTime theDfltTau
- static DtTime theDfltHtau
- static DtTime theDfltTauInv

- **`static DtBoolean theMasterSmoothEnable`**
- **`inline DtSmoothPos(const DtSmoothPos&)`**
- **`inline DtTime smoothPeriod() const`**

Direct child classes:

DtSmoothPosWithDr

Alphabetic index *Hierarchy of classes*

In file smoothutil.h:

class DtSmoothOrient

---------- ORIENTATION ----------

Inheritance:

DtSmoothOrient

Public Methods

	DtSmoothOrient (void *usr, DtTime smoothT, DtConstDcm orient, const DtDirectedRotRate &drr) *Constructors*
	DtSmoothOrient (void *usr, DtTime smoothT) *WARNING: not fully constructed until input() & tick() are called*
void	**input** (DtConstDcm orient, const DtDirectedRotRate &drr, DtTime simTime, int freeze=0)
void	**tick** (DtTime simTime)
DtConstDcm	**orientation** () const
virtual void	**setSmoothEnable** (int on) *Set/Get the individual smoothing switch*
virtual int	**smoothEnable** () const
virtual void	**setSmoothPeriod** (DtTime t) *Set/Get smoothPeriod*
virtual DtTime	**smoothPeriod** () const
static void	**setDfltSmoothPeriod** (DtTime t) *Set/Get default smooth period*
static DtTime	**dfltSmoothPeriod** ()
static void	**setMasterSmoothEnable** (DtBoolean onOff) *Set/Get the value of the "master" smoothing-enabled switch*
static DtBoolean	**masterSmoothEnable** ()
inline DtTime	**smoothPeriod** () const

Protected Fields

int	**frozen**
DtTime	**t0**
DtTime	**t2**

- DtTime **lastT**
- DtTime* **myTau**
- DtTime **myLocalTau**
- DtDcm **os**
- DtDcm **o0**
- DtVector **ws**
- int **myEnable**
- void* **usr**
- static DtTime **theDfltTau**
- static DtBoolean **theMasterSmoothEnable**

Protected Methods

- virtual void **dfltAction** (DtTime simTime, void *usr, DtDcmRef os)
- void **init** (void *u, DtTime t)
- void **setT1data** ()

Documentation

---------- ORIENTATION ----------

- **`DtSmoothOrient(void *usr, DtTime smoothT, DtConstDcm orient, const DtDirectedRotRate &drr)`**

  Constructors. smoothT is the initial smoothPeriod. Use a value of -1 to indicate use of the default period.

- **`DtSmoothOrient(void *usr, DtTime smoothT)`**

  WARNING: not fully constructed until input() & tick() are called

- **`void input(DtConstDcm orient, const DtDirectedRotRate &drr, DtTime simTime, int freeze=0)`**
- **`void tick(DtTime simTime)`**
- **`DtConstDcm orientation() const`**
- **`virtual void setSmoothEnable(int on)`**

  Set/Get the individual smoothing switch. In order for smoothing to be enabled, the "master" smoothing-enabled switch must be on, AND the individual smoothing switch (accessed through these functions), must be on.

- **`virtual int smoothEnable() const`**
- **`virtual void setSmoothPeriod(DtTime t)`**

  Set/Get smoothPeriod. If set to -1, smooth period reverts to the current default smooth period.

- **virtual DtTime smoothPeriod() const**
- **static void setDfltSmoothPeriod(DtTime t)**

  Set/Get default smooth period
- **static DtTime dfltSmoothPeriod()**
- **static void setMasterSmoothEnable(DtBoolean onOff)**

  Set/Get the value of the "master" smoothing-enabled switch
- **static DtBoolean masterSmoothEnable()**
- **virtual void dfltAction(DtTime simTime, void *usr, DtDcmRef os)**
- **void init(void *u, DtTime t)**
- **void setT1data()**
- **int frozen**
- **DtTime t0**
- **DtTime t2**
- **DtTime lastT**
- **DtTime* myTau**
- **DtTime myLocalTau**
- **DtDcm os**
- **DtDcm o0**
- **DtVector ws**
- **int myEnable**
- **void* usr**
- **static DtTime theDfltTau**
- **static DtBoolean theMasterSmoothEnable**
- **inline DtTime smoothPeriod() const**

Direct child classes:

DtSmoothOriWithDr

Alphabetic index *Hierarchy of classes*

In file artpart.h:

class DtArtPart

One Articulated Part

Public Methods

virtual DtBoolean	**setVal** (int paramType, DtFloat32 value) *Set/Get the value for the specified parameter type*
virtual DtBoolean	**getVal** (int paramType, DtFloat32& value) const
virtual DtFloat32	**val** (int paramType) const *Get the value for the specified parameter type*
virtual int	**partType** () const *Get the part type*
virtual int	**partNum** () const *Get the part number*
virtual int	**partAttachedTo** () const *Get the part this part is attached to*
virtual int	**numParams** () const *Get the number of parameters*
virtual int	**firstValidParamType** () const *Get the first valid (added) parameter type*
virtual int	**nextValidParamType** (int paramType) const *Get the next valid parameter type*
virtual int	**paramNum** (int paramType) const *Returns the index of the parameter in a multiple part list as is found in HLA messages and DIS PDUs*
virtual const DtTaitBryan&	**angles** () const *Get the euler angles*
virtual void	**getAngles** (DtTaitBryan& euler) const
virtual DtConstVector	**translations** () const *Get the translations*
virtual void	**getTranslations** (DtVectorRef vec) const
virtual void	**setApproximating** (DtBoolean onOff) *Set/Get whether or not to do approximating*
virtual DtBoolean	**approximating** () const
virtual void	**print** () *Print the part data*
static int	**artParamIsRate** (int paramType) *Get whether the parameter type is actually a rate*
static int	**computePartType** (int partPlusParam) *Get the part type from the part and parameter value*
static int	**computeParameter** (int partPlusParam) *Get the parameter type from the part and parameter value*

	DtArtPart () *default constructor*
inline DtBoolean	**approximating** () const
inline DtFloat32	**lastSetVal** (int paramType) const
inline int	**numParams** () const
inline int	**partAttachedTo** () const
inline int	**partNum** () const
inline int	**partType** () const
inline DtArtPart*	**self** () const
inline void	**setApproximating** (DtBoolean onOff)
inline void	**setPartNum** (int num)
inline DtTime	**timeSet** (int paramType) const

Protected Fields

int	**myPartType**
int	**myPartNum**
int	**myPartAttachedTo**
int	**myNumParams**
DtInternalArtParam**	**myVals** *myVals is an array of DtMAX_PARAMS_PER_PART + 1 items which are themselves pointers*
int	**myFirstParam**
int	**myLastParam**
DtBoolean	**myApproxFlag**
DtTaitBryan	**myAngles**
DtVector	**myVector**

Protected Methods

	DtArtPart (const DtArtPartSpec& spec) *constructor*
virtual	**~DtArtPart** () *destructor*
	DtArtPart (const DtArtPart& orig) *copy constructor*
DtArtPart&	**operator=** (const DtArtPart& orig) *assignment operator*
virtual void	**setParamNum** (int paramType, int num) *Set the parameter number of the specified parameter type*
virtual void	**setPartNum** (int num) *Set the part number*

- virtual void **addParam** (int paramType)
  Add the specified parameter type to the part
- virtual int **findNextValidParamType** (int paramType) const
  Find the next valid parameter type
- virtual DtFloat32 **approxVal** (int paramType) const
  Get the approximated value
- virtual DtFloat32 **lastSetVal** (int paramType) const
  Get the last set (absolute) value
- virtual DtTime **timeSet** (int paramType) const
  Get the time at which the value was last set
- DtArtPart* **self** () const
  Cast off const-ness for trivial internal change

Protected

- *Constructors are protected since this class is managed by DtArtPartList*

Documentation

One Articulated Part

- **`virtual DtBoolean setVal(int paramType, DtFloat32 value)`**

  Set/Get the value for the specified parameter type. Returns DtFalse if the parameter type has not been added, returns DtTrue otherwise.

- **`virtual DtBoolean getVal(int paramType, DtFloat32& value) const`**

- **`virtual DtFloat32 val(int paramType) const`**

  Get the value for the specified parameter type. Note this does not check existence of the parameter type and should only be called when you know that it does exist (as from nextValidParamType()).

- **`virtual int partType() const`**

  Get the part type

- **`virtual int partNum() const`**

  Get the part number

- **`virtual int partAttachedTo() const`**

  Get the part this part is attached to

- **`virtual int numParams() const`**

  Get the number of parameters

- **`virtual int firstValidParamType() const`**

  Get the first valid (added) parameter type

- **`virtual int nextValidParamType(int paramType) const`**

  Get the next valid parameter type

- **`virtual int paramNum(int paramType) const`**

Returns the index of the parameter in a multiple part list as is found in HLA messages and DIS PDUs

- **virtual const DtTaitBryan& angles() const**

  Get the euler angles. Euler is filled with the angles phi, theta and psi of the part with respect to its neutral orientation.

- **virtual void getAngles(DtTaitBryan& euler) const**

- **virtual DtConstVector translations() const**

  Get the translations. Vec is filled in with a vector representing the current translation, in x, y and z of the part from its neutral position.

- **virtual void getTranslations(DtVectorRef vec) const**

- **virtual void setApproximating(DtBoolean onOff)**

  Set/Get whether or not to do approximating

- **virtual DtBoolean approximating() const**

- **virtual void print()**

  Print the part data

- **static int artParamIsRate(int paramType)**

  Get whether the parameter type is actually a rate

- **static int computePartType(int partPlusParam)**

  Get the part type from the part and parameter value

- **static int computeParameter(int partPlusParam)**

  Get the parameter type from the part and parameter value

-

  Constructors are protected since this class is managed by DtArtPartList

- **DtArtPart()**

  default constructor

- **DtArtPart(const DtArtPartSpec& spec)**

  constructor

- **virtual ~DtArtPart()**

  destructor

- **DtArtPart(const DtArtPart& orig)**

  copy constructor

- **DtArtPart& operator=(const DtArtPart& orig)**

  assignment operator

- **virtual void setParamNum(int paramType, int num)**

  Set the parameter number of the specified parameter type

- **virtual void setPartNum(int num)**

  Set the part number

- **virtual void addParam(int paramType)**

Add the specified parameter type to the part. Note that this assumes that parameters are always added in order from lowest to highest.

- **virtual int findNextValidParamType(int paramType) const**

  Find the next valid parameter type

- **virtual DtFloat32 approxVal(int paramType) const**

  Get the approximated value

- **virtual DtFloat32 lastSetVal(int paramType) const**

  Get the last set (absolute) value

- **virtual DtTime timeSet(int paramType) const**

  Get the time at which the value was last set

- **DtArtPart* self() const**

  Cast off const-ness for trivial internal change. Non-virtual since can't override on return type.

- **int myPartType**
- **int myPartNum**
- **int myPartAttachedTo**
- **int myNumParams**
- **DtInternalArtParam** myVals**

  myVals is an array of DtMAX_PARAMS_PER_PART + 1 items which are themselves pointers

- **int myFirstParam**
- **int myLastParam**
- **DtBoolean myApproxFlag**
- **DtTaitBryan myAngles**
- **DtVector myVector**
- **inline DtBoolean approximating() const**
- **inline DtFloat32 lastSetVal(int paramType) const**
- **inline int numParams() const**
- **inline int partAttachedTo() const**
- **inline int partNum() const**
- **inline int partType() const**
- **inline DtArtPart* self() const**
- **inline void setApproximating(DtBoolean onOff)**
- **inline void setPartNum(int num)**
- **inline DtTime timeSet(int paramType) const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file artpartlist.h:

class DtArtPartList

class DtArtPartList: Instances of DtArtPartList are used to manage articulated parts

---

Public Methods

	DtArtPartList () *default constructor*
	DtArtPartList (const DtArtPartSpec* specList) *constructor*
virtual	**~DtArtPartList** () *destructor*
	DtArtPartList (const DtArtPartList& orig) *copy constructor*
DtArtPartList&	**operator=** (const DtArtPartList& orig) *assignment operator*
void	**init** (const DtArtPartSpec* partSpecs) *The init functions don't set values for the parameters, they merely set up the part and parameter types*
virtual void	**init** (int numParams, const DtNetArtParamRecord* netParams) *The init functions don't set values for the parameters, they merely set up the part and parameter types*
virtual void	**setValsFromNet** (int numParams, const DtNetArtParamRecord* netParams) *sets values from net representation, but does not change the set of valid parts and parameters*
virtual void	**setFromNet** (int numParams, const DtNetArtParamRecord* netParams) *initializes the part types and parameters, and sets values as well*
virtual void	**fillNetParams** (DtNetArtParamRecord* netParams) const *fills in netParams based on current values*
void	**clear** () *removes all parts and parameters from list Non-virtual since called from ctor/dtor*
virtual DtBoolean	**setArtParamVal** (int partType, int paramType, DtFloat32 val) *Set/Get the specified part type's specified parameter type value*
virtual DtBoolean	**getArtParamVal** (int partType, int paramType, DtFloat32& val) const
virtual int	**numArtParts** () const *Get the number of articulated parts*
virtual int	**numArtParams** () const *Get the number of articulated parameters*
virtual DtArtPart*	**artPartFromPartNum** (int partNum) const *Get the articulated part from the part number*
virtual DtArtPart*	**artPartFromPartType** (int partType) const *Get the first articulated part with the part type*
virtual int	**partNumFromPartType** (int partType) const *returns 0 if there is no such part, otherwise the part number*

- virtual const DtArtPartSpec* **artPartSpecs** () const
  return the list of DtArtPartSpecs describing the parts in the list
- virtual void **print** () const
  Print the articulated part list
- static void **fillSpecList** (int numParams, const DtNetArtParamRecord* netParams, DtArtPartSpec* specList)
- static DtBoolean **artNotAttached** (const DtNetArtParamRecord* param)
  Returns true if param is an articulated parameter record, false if its an attached part record
- inline int **numArtParams** () const
- inline int **numArtParts** () const
- inline DtArtPartList* **self** () const

Public

- *Fill the articulated part spec list from the net parameter record list*

Protected Fields

- DtArtPart** **myPartList**
- int **myNumArtParts**
- int **myNumArtParams**
- DtArtPartSpec* **mySpecList**

Protected Methods

- virtual DtBoolean **partNumInBounds** (int partNum) const
  Returns whether the part number is within bounds
- DtArtPartList* **self** () const
  Cast off const-ness for trivial internal change

Documentation

class DtArtPartList:

Instances of DtArtPartList are used to manage articulated parts. Articulated part spec lists are mainly used to generate these lists.

- `DtArtPartList()`
  default constructor
- `DtArtPartList(const DtArtPartSpec* specList)`
  constructor
- `virtual ~DtArtPartList()`

destructor

- **DtArtPartList(const DtArtPartList& orig)**

  copy constructor

- **DtArtPartList& operator=(const DtArtPartList& orig)**

  assignment operator

- **void init(const DtArtPartSpec* partSpecs)**

  The init functions don't set values for the parameters, they merely set up the part and parameter types. Non-virtual since called from ctor.

- **virtual void init(int numParams, const DtNetArtParamRecord* netParams)**

  The init functions don't set values for the parameters, they merely set up the part and parameter types. numParams is the total number of records in netParams, which might include attached parameter records that will be ignored.

- **virtual void setValsFromNet(int numParams, const DtNetArtParamRecord* netParams)**

  sets values from net representation, but does not change the set of valid parts and parameters. numParams is the total number of records in netParams, which might include attached parameter records that will be ignored.

- **virtual void setFromNet(int numParams, const DtNetArtParamRecord* netParams)**

  initializes the part types and parameters, and sets values as well. numParams is the total number of records in netParams, which might include attached parameter records that will be ignored.

- **virtual void fillNetParams(DtNetArtParamRecord* netParams) const**

  fills in netParams based on current values

- **void clear()**

  removes all parts and parameters from list Non-virtual since called from ctor/dtor

- **virtual DtBoolean setArtParamVal(int partType, int paramType, DtFloat32 val)**

  Set/Get the specified part type's specified parameter type value. Returns DtFalse if the part or parameter are not found, otherwise returns DtTrue.

- **virtual DtBoolean getArtParamVal(int partType, int paramType, DtFloat32& val) const**

- **virtual int numArtParts() const**

  Get the number of articulated parts

- **virtual int numArtParams() const**

  Get the number of articulated parameters

- **virtual DtArtPart* artPartFromPartNum(int partNum) const**

  Get the articulated part from the part number

- **virtual DtArtPart* artPartFromPartType(int partType) const**

  Get the first articulated part with the part type

- **virtual int partNumFromPartType(int partType) const**

  returns 0 if there is no such part, otherwise the part number

- **virtual const DtArtPartSpec* artPartSpecs() const**

  return the list of DtArtPartSpecs describing the parts in the list

- **`virtual void print() const`**

  Print the articulated part list

-

  Fill the articulated part spec list from the net parameter record list. numParams is the total number of records in netParams, which might include attached parameter records that will be ignored.

- **`static void fillSpecList( int numParams, const DtNetArtParamRecord* netParams, DtArtPartSpec* specList)`**

- **`static DtBoolean artNotAttached(const DtNetArtParamRecord* param)`**

  Returns true if param is an articulated parameter record, false if its an attached part record

- **`virtual DtBoolean partNumInBounds(int partNum) const`**

  Returns whether the part number is within bounds

- **`DtArtPartList* self() const`**

  Cast off const-ness for trivial internal change. Non-virtual since can't override on return type.

- **`DtArtPart** myPartList`**

- **`int myNumArtParts`**

- **`int myNumArtParams`**

- **`DtArtPartSpec* mySpecList`**

- **`inline int numArtParams() const`**

- **`inline int numArtParts() const`**

- **`inline DtArtPartList* self() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file artpartthresh.h:

class DtArtPartThresholder

class DtArtPartThresholder: DtArtPartThresholder is used to store art-part-related thresholds, and to perform some art part thresholding

---

Public Classes

- enum **DtArtPartThresholdType**
  - **DtArtPartDfltThreshMode**
  - **DtArtPartIndividualThresh**
  - **DtArtPartAggregateThresh**

Public Methods

	DtArtPartThresholder () *Constructor*
	~DtArtPartThresholder () *Destructor*
	DtArtPartThresholder (const DtArtPartThresholder& orig) *copy constructor*
DtArtPartThresholder&	**operator=** (const DtArtPartThresholder& orig) *assignment operator*
virtual void	**setRotationThreshold** (DtReal thresh) *Set/Get rotation threshold in radians*
virtual double	**rotationThreshold** () const
virtual double	**rotationTraceThreshold** () const *Returns the "RotationTrace" threshold*
virtual void	**setTranslationThreshold** (DtReal thresh) *Set/Get translation threshold in meters*
virtual double	**translationThreshold** () const
virtual void	**setFractionalThreshold** (DtReal thresh) *Set/Get fractional threshold (fraction between 0 and 1)*
virtual double	**fractionalThreshold** () const
virtual void	**setThreshMode** (DtArtPartThresholdType mode) *Set/Get the theshold mode*
virtual DtArtPartThresholdType	**threshMode** () const
virtual DtBoolean	**artPartsExceedAggThreshold** (const DtArtPartList& localList, const DtArtPartList& remoteList) const *Get whether or not an aggregate threshold (such as position, translation, rotation, etc*

virtual DtBoolean	**artPartsExceedIndivThreshold** (const DtArtPartList& localList, const DtArtPartList& remoteList) const *Get whether or not an individual parameter threshold has been exceeded*
virtual float	**artPartThreshold** (int paramType) const *Get the thresholding value (functional, translational, or rotational) for the parameter type*
static void	**setDfltThreshMode** (DtArtPartThresholdType mode) *Set/Get the theshold mode*
static DtArtPartThresholdType	**dfltThreshMode** ()
static void	**setDfltFractionalThreshold** (DtFloat32 thresh) *Set/Get fractional threshold value*
static DtFloat32	**dfltFractionalThreshold** ()
static void	**setDfltRotationThreshold** (DtReal r) *Set/Get rotational threshold value*
static double	**dfltRotationThreshold** ()
static void	**setDfltTranslationThreshold** (DtReal t) *Set/Get translational threshold value*
static double	**dfltTranslationThreshold** ()
inline DtFloat32	**dfltFractionalThreshold** ()
inline double	**dfltRotationThreshold** ()
inline DtArtPartThresholder::DtArtPartThresholdType	**dfltThreshMode** ()
inline double	**dfltTranslationThreshold** ()
inline void	**setDfltFractionalThreshold** (DtFloat32 thresh)
inline void	**setDfltRotationThreshold** (DtReal thresh)
inline void	**setDfltThreshMode** (DtArtPartThresholdType mode)
inline void	**setDfltTranslationThreshold** (DtReal t)

Protected Fields

DtArtPartThresholdType	**myThreshMode**
float	**myFracThresh**
double	**myTransThresh**
double	**myRotThresh**
double	**myTraceRotThresh**
DtBoolean	**myUsingDfltMode**
DtBoolean	**myUsingDfltTrans**
DtBoolean	**myUsingDfltRot**
DtBoolean	**myUsingDfltFrac**
static DtArtPartThresholdType	**theDfltThreshMode**

- static float **theDfltFracThresh**
- static double **theDfltTransThresh**
- static double **theDfltRotThresh**
- static double **theDfltTraceRotThresh**

Documentation

class DtArtPartThresholder: DtArtPartThresholder is used to store art-part-related thresholds, and to perform some art part thresholding

- **enum DtArtPartThresholdType**
- **DtArtPartDfltThreshMode**
- **DtArtPartIndividualThresh**
- **DtArtPartAggregateThresh**
- **DtArtPartThresholder()**
  Constructor
- **~DtArtPartThresholder()**
  Destructor
- **DtArtPartThresholder(const DtArtPartThresholder& orig)**
  copy constructor
- **DtArtPartThresholder& operator=(const DtArtPartThresholder& orig)**
  assignment operator
- **virtual void setRotationThreshold(DtReal thresh)**
  Set/Get rotation threshold in radians. Negative number means use default.
- **virtual double rotationThreshold() const**
- **virtual double rotationTraceThreshold() const**
  Returns the "RotationTrace" threshold
- **virtual void setTranslationThreshold(DtReal thresh)**
  Set/Get translation threshold in meters. Negative number means use default.
- **virtual double translationThreshold() const**
- **virtual void setFractionalThreshold(DtReal thresh)**
  Set/Get fractional threshold (fraction between 0 and 1). Negative number means use default.
- **virtual double fractionalThreshold() const**
- **virtual void setThreshMode(DtArtPartThresholdType mode)**
  Set/Get the theshold mode. Use DtArtPartDfltThreshMode to indicate use of the default.
- **virtual DtArtPartThresholdType threshMode() const**

- **virtual DtBoolean artPartsExceedAggThreshold(const DtArtPartList& localList, const DtArtPartList& remoteList) const**

  Get whether or not an aggregate threshold (such as position, translation, rotation, etc.) has been exceeded.

- **virtual DtBoolean artPartsExceedIndivThreshold(const DtArtPartList& localList, const DtArtPartList& remoteList) const**

  Get whether or not an individual parameter threshold has been exceeded

- **virtual float artPartThreshold(int paramType) const**

  Get the thresholding value (functional, translational, or rotational) for the parameter type

- **static void setDfltThreshMode(DtArtPartThresholdType mode)**

  Set/Get the theshold mode

- **static DtArtPartThresholdType dfltThreshMode()**

- **static void setDfltFractionalThreshold(DtFloat32 thresh)**

  Set/Get fractional threshold value

- **static DtFloat32 dfltFractionalThreshold()**

- **static void setDfltRotationThreshold(DtReal r)**

  Set/Get rotational threshold value

- **static double dfltRotationThreshold()**

- **static void setDfltTranslationThreshold(DtReal t)**

  Set/Get translational threshold value

- **static double dfltTranslationThreshold()**

- **DtArtPartThresholdType myThreshMode**

- **float myFracThresh**

- **double myTransThresh**

- **double myRotThresh**

- **double myTraceRotThresh**

- **DtBoolean myUsingDfltMode**

- **DtBoolean myUsingDfltTrans**

- **DtBoolean myUsingDfltRot**

- **DtBoolean myUsingDfltFrac**

- **static DtArtPartThresholdType theDfltThreshMode**

- **static float theDfltFracThresh**

- **static double theDfltTransThresh**

- **static double theDfltRotThresh**

- `static double theDfltTraceRotThresh`
- `inline DtFloat32 dfltFractionalThreshold()`
- `inline double dfltRotationThreshold()`
- `inline DtArtPartThresholder::DtArtPartThresholdType dfltThreshMode()`
- `inline double dfltTranslationThreshold()`
- `inline void setDfltFractionalThreshold( DtFloat32 thresh)`
- `inline void setDfltRotationThreshold(DtReal thresh)`
- `inline void setDfltThreshMode( DtArtPartThresholdType mode)`
- `inline void setDfltTranslationThreshold(DtReal t)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file attpartlist.h:

class DtAttachedPartList

class DtAttachedPartList: Instances of DtAttachedPartList are lists of attached parts

---

Public Methods

	DtAttachedPartList () *default constructor*
	DtAttachedPartList (const DtAttachedPartList& orig) *copy constructor*
DtAttachedPartList&	**operator=** (const DtAttachedPartList& orig) *assignment operator*
virtual	**~DtAttachedPartList** () *destructor*
virtual DtBoolean	**setNumAttachedParts** (int num) *Set/Get the number of attached parts*
virtual int	**numAttachedParts** () const
virtual const int*	**stationList** () const *Returns a zero-termination list of station numbers that are valid in this attached part list*
virtual void	**init** (const int* stations) *Stations is a zero-terminated array of stations*
virtual DtBoolean	**setStationNumber** (int partIndex, int station) *Set/Get the station number for a part*
virtual DtBoolean	**getStationNumber** (int partIndex, int& station) const
virtual int	**stationNumber** (int partIndex) const *Returns the station number for a part*
virtual DtBoolean	**setEntityType** (int partIndex, const DtEntityType& type) *Set/Get the entity type for a part*
virtual DtBoolean	**getEntityType** (int partIndex, DtEntityType& type) const
virtual const DtEntityType&	**entityType** (int partIndex) const *Returns the entityType for a part, by index*
virtual DtAttachedPartStatus	**setEntityTypeAtStation** (int station, const DtEntityType& type) *Set/Get the entity type for a part*
virtual DtAttachedPartStatus	**getEntityTypeAtStation** (int station, DtEntityType& type) const
virtual const DtEntityType&	**entityTypeAtStation** (int station) const *Returns the entityType at a particular station*
virtual void	**unloadNetList** (int numParams, const DtNetAttachedPart* partList) *After this call, this attached part list will reflect the information contained in netParams*
virtual void	**loadNetList** (DtNetAttachedPart* partList) const *Load a list from network data representation*
virtual int	**netRecordLength** () const *Get the network data length for this list*

- DtBoolean — **equivalent** (const DtAttachedPartList& other) const
  Returns true if we contain the exact same information as other
- virtual void — **print** () const
  Print the information contained in the record
- inline const int* — **stationList** () const

Protected Fields

- int — **myNumParts**
- int* — **myStationList**
  zero terminated list that should always be of length myNumParts + 1
- DtEntityType* — **myTypeList**
  NULL if myNumArtParts = 0, otherwise has length = myNumParts

---

Documentation

class DtAttachedPartList: Instances of DtAttachedPartList are lists of attached parts

- **`DtAttachedPartList()`**
  default constructor
- **`DtAttachedPartList(const DtAttachedPartList& orig)`**
  copy constructor
- **`DtAttachedPartList& operator=(const DtAttachedPartList& orig)`**
  assignment operator
- **`virtual ~DtAttachedPartList()`**
  destructor
- **`virtual DtBoolean setNumAttachedParts(int num)`**
  Set/Get the number of attached parts. Set may return DtFalse if the number of requested parts is invalid.
- **`virtual int numAttachedParts() const`**
- **`virtual const int* stationList() const`**
  Returns a zero-termination list of station numbers that are valid in this attached part list
- **`virtual void init(const int* stations)`**
  Stations is a zero-terminated array of stations. Init initializes the attPartList with the stations in the array.
- **`virtual DtBoolean setStationNumber(int partIndex, int station)`**
  Set/Get the station number for a part. Returns DtFalse if the index is out of range.
- **`virtual DtBoolean getStationNumber(int partIndex, int& station) const`**
- **`virtual int stationNumber(int partIndex) const`**
  Returns the station number for a part. This version of the function reuires that partIndex be valid (between 0 and numAttachedParts().

- **virtual DtBoolean setEntityType(int partIndex, const DtEntityType& type)**

  Set/Get the entity type for a part. Returns DtFalse if the index is out of range.

- **virtual DtBoolean getEntityType(int partIndex, DtEntityType& type) const**

- **virtual const DtEntityType& entityType(int partIndex) const**

  Returns the entityType for a part, by index. This version of the function reuires that partIndex be valid (between 0 and numAttachedParts().

- **virtual DtAttachedPartStatus setEntityTypeAtStation(int station, const DtEntityType& type)**

  Set/Get the entity type for a part. Returns DtFalse if the index is out of range.

- **virtual DtAttachedPartStatus getEntityTypeAtStation(int station, DtEntityType& type) const**

- **virtual const DtEntityType& entityTypeAtStation(int station) const**

  Returns the entityType at a particular station. If the list does not contain an entry for the station, DtEntityType::nullEntityType is returned;

- **virtual void unloadNetList(int numParams, const DtNetAttachedPart* partList)**

  After this call, this attached part list will reflect the information contained in netParams. Both the set of valid stations, and the entity types attached to those stations might be changed. numParams is the total number of records in netParams, which might include articulated parameter records that will be ignored.

- **virtual void loadNetList(DtNetAttachedPart* partList) const**

  Load a list from network data representation. numAttachedParts() network part records will be written/read.

- **virtual int netRecordLength() const**

  Get the network data length for this list

- **DtBoolean equivalent(const DtAttachedPartList& other) const**

  Returns true if we contain the exact same information as other

- **virtual void print() const**

  Print the information contained in the record

- **int myNumParts**

- **int* myStationList**

  zero terminated list that should always be of length myNumParts + 1

- **DtEntityType* myTypeList**

  NULL if myNumArtParts = 0, otherwise has length = myNumParts

- **inline const int* stationList() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hoststructs.h:

class DtEntityType

Inheritance:

DtEntityType

Public Fields

- int **DtEntityKind**
- int **DtDomain**
  this conflicts with enum of same name, so must be before funcs that use it
- int **DtCountry**
- int **DtCategory**
- int **DtSubCategory**
- int **DtSpecific**
- int **DtExtra**

Public Methods

	DtEntityType (const char* p)
	DtEntityType (int kind, int domain, int country, int category, int subCategory, int specific, int extra)
	DtEntityType ()
	DtEntityType (DtNetEntityType netType)
void	**setFromNet** (DtNetEntityType netType)
	operator DtNetEntityType () const
void	**init** (int kind, int domain, int country, int category, int subCategory, int specific, int extra)
char*	**string** () const
static const DtEntityType&	**nullType** ()
int	**kind** () const *int entityKind() const {return(DtEntityKind);}*
int	**domain** () const
int	**country** () const

int	**category** () const
int	**subCategory** () const
int	**specific** () const
int	**extra** () const
void	**setKind** (int kind) *void setEntityKind(int kind){DtEntityKind = kind;}*
void	**setDomain** (int domain)
void	**setCountry** (int country)
void	**setCategory** (int category)
void	**setSubCategory** (int subCategory)
void	**setSpecific** (int specific)
void	**setExtra** (int extra)
void	**setKind** (long kind) *the funcs used to take longs, so for back-compat:*
void	**setDomain** (long domain)
void	**setCountry** (long country)
void	**setCategory** (long category)
void	**setSubCategory** (long subCategory)
void	**setSpecific** (long specific)
void	**setExtra** (long extra)
DtBoolean	**matchPattern** (const DtEntityType &pat) const
DtBoolean	**DtTypeMatchPattern** (const DtEntityType &pat) const
int	**match** (const DtEntityType &type) const *== returns 1 if they are equal, 0 if they are not equal*
int	**operator==** (const DtEntityType &type) const
int	**operator!=** (const DtEntityType &type) const *!= returns 0 if they are equal, nonzero if they are not equal*

Documentation

- **`DtEntityType(const char* p)`**
- **`DtEntityType(int kind, int domain, int country, int category, int subCategory, int specific, int extra)`**
- **`DtEntityType()`**
- **`DtEntityType(DtNetEntityType netType)`**
- **`void setFromNet(DtNetEntityType netType)`**
- **`operator DtNetEntityType() const`**

- **`void init(int kind, int domain, int country, int category, int subCategory, int specific, int extra)`**
- **`char* string() const`**
- **`static const DtEntityType& nullType()`**
- **`int DtEntityKind`**
- **`int DtDomain`**

  this conflicts with enum of same name, so must be before funcs that use it. Leave for awhile back-compat for awhile.
- **`int DtCountry`**
- **`int DtCategory`**
- **`int DtSubCategory`**
- **`int DtSpecific`**
- **`int DtExtra`**
- **`int kind() const`**

  int entityKind() const {return(DtEntityKind);}
- **`int domain() const`**
- **`int country() const`**
- **`int category() const`**
- **`int subCategory() const`**
- **`int specific() const`**
- **`int extra() const`**
- **`void setKind(int kind)`**

  void setEntityKind(int kind){DtEntityKind = kind;}
- **`void setDomain(int domain)`**
- **`void setCountry(int country)`**
- **`void setCategory(int category)`**
- **`void setSubCategory(int subCategory)`**
- **`void setSpecific(int specific)`**
- **`void setExtra(int extra)`**
- **`void setKind(long kind)`**

  the funcs used to take longs, so for back-compat:
- **`void setDomain(long domain)`**
- **`void setCountry(long country)`**
- **`void setCategory(long category)`**

- **`void setSubCategory(long subCategory)`**
- **`void setSpecific(long specific)`**
- **`void setExtra(long extra)`**
- **`DtBoolean matchPattern(const DtEntityType &pat) const`**
- **`DtBoolean DtTypeMatchPattern(const DtEntityType &pat) const`**
- **`int match(const DtEntityType &type) const`**

  == returns 1 if they are equal, 0 if they are not equal
- **`int operator==(const DtEntityType &type) const`**
- **`int operator!=(const DtEntityType &type) const`**

  != returns 0 if they are equal, nonzero if they are not equal

---

Direct child classes:

DtRadioEntityType

Alphabetic index *Hierarchy of classes*

In file netstructs.h:

class DtNetEntityType

Public Fields

- DtNetU8 **DtEntityKind**
- DtNetU8 **DtDomain**
- DtNetU16 **DtCountry**
- DtNetU8 **DtCategory**
- DtNetU8 **DtSubCategory**
- DtNetU8 **DtSpecific**
- DtNetU8 **DtExtra**

Public Methods

- DtEntityType **operator =** (DtEntityType)
- DtNetEntityType **operator =** (DtNetEntityType)

Documentation

- **`DtNetU8 DtEntityKind`**
- **`DtNetU8 DtDomain`**
- **`DtNetU16 DtCountry`**
- **`DtNetU8 DtCategory`**
- **`DtNetU8 DtSubCategory`**
- **`DtNetU8 DtSpecific`**
- **`DtNetU8 DtExtra`**
- **`DtEntityType operator = (DtEntityType)`**
- **`DtNetEntityType operator = (DtNetEntityType)`**

This class has no child classes.

In file hoststructs.h:

class DtRadioEntityType: public DtEntityType

Inheritance:

DtRadioEntityType - DtEntityType

Public Methods

	DtRadioEntityType ()
	DtRadioEntityType (const char* p)
	DtRadioEntityType (int kind, int domain, int country, int category, int nomenclatureVersion, int nomenclature)
	DtRadioEntityType (DtNetRadioEntityType netType)
void	**setFromNet** (DtNetRadioEntityType netType)
int	**nomenclatureVersion** () const
int	**nomenclature** () const
void	**setNomenclatureVersion** (int version)
void	**setNomenclature** (int nom)
char*	**string** () const
static const DtRadioEntityType&	**nullType** ()
	operator DtNetRadioEntityType () const

Inherited from DtEntityType:

Public Fields

- int **DtEntityKind**
- int **DtDomain**
- int **DtCountry**
- int **DtCategory**
- int **DtSubCategory**

- int **DtSpecific**
- int **DtExtra**

Public Methods

- **operator DtNetEntityType**() const
- void **init**(int kind, int domain, int country, int category, int subCategory, int specific, int extra)
- int **kind**() const
- int **domain**() const
- int **country**() const
- int **category**() const
- int **subCategory**() const
- int **specific**() const
- int **extra**() const
- void **setKind**(int kind)
- void **setDomain**(int domain)
- void **setCountry**(int country)
- void **setCategory**(int category)
- void **setSubCategory**(int subCategory)
- void **setSpecific**(int specific)
- void **setExtra**(int extra)
- void **setKind**(long kind)
- void **setDomain**(long domain)
- void **setCountry**(long country)
- void **setCategory**(long category)
- void **setSubCategory**(long subCategory)
- void **setSpecific**(long specific)
- void **setExtra**(long extra)
- DtBoolean **matchPattern**(const DtEntityType &pat) const
- DtBoolean **DtTypeMatchPattern**(const DtEntityType &pat) const
- int **match**(const DtEntityType &type) const
- int **operator==**(const DtEntityType &type) const
- int **operator!=**(const DtEntityType &type) const

Documentation

- `DtRadioEntityType()`
- `DtRadioEntityType(const char* p)`
- `DtRadioEntityType(int kind, int domain, int country, int category, int nomenclatureVersion, int nomenclature)`
- `DtRadioEntityType(DtNetRadioEntityType netType)`
- `void setFromNet(DtNetRadioEntityType netType)`
- `int nomenclatureVersion() const`
- `int nomenclature() const`
- `void setNomenclatureVersion(int version)`
- `void setNomenclature(int nom)`
- `char* string() const`
- `static const DtRadioEntityType& nullType()`
- `operator DtNetRadioEntityType() const`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file netstructs.h:

class DtNetRadioEntityType

Public Fields

- DtNetU8 **kind**
- DtNetU8 **domain**
- DtNetU16 **country**
- DtNetU8 **category**
- DtNetU8 **nomenclatureVersion**
- DtNetU16 **nomenclature**

Public Methods

- DtRadioEntityType **operator =** (DtRadioEntityType)
- DtNetRadioEntityType **operator =** (DtNetRadioEntityType)

Documentation

- **`DtNetU8 kind`**
- **`DtNetU8 domain`**
- **`DtNetU16 country`**
- **`DtNetU8 category`**
- **`DtNetU8 nomenclatureVersion`**
- **`DtNetU16 nomenclature`**
- **`DtRadioEntityType operator = (DtRadioEntityType)`**
- **`DtNetRadioEntityType operator = (DtNetRadioEntityType)`**

This class has no child classes.

In file attpartlist.h:

enum DtAttachedPartStatus

- **DtAttachedPartSuccess**
- **DtNoSuchAttachStation**
- **DtPartNotAttached**
- **DtPartAttached**

Documentation

- `DtAttachedPartSuccess`
- `DtNoSuchAttachStation`
- `DtPartNotAttached`
- `DtPartAttached`

In file baseexconn.h:

class DtBaseExerciseConn

Inheritance:

DtBaseExerciseConn

Public Methods

virtual	**~DtBaseExerciseConn** () *destructor*
DtBaseExerciseConn&	**operator=** (const DtBaseExerciseConn& orig) *assignment operator*
virtual void	**send** (const DtStateMsg& msg) *Send a state message to the network*
virtual void	**send** (const DtInteraction& msg) *Send an interaction to the network*
virtual void	**sendStamped** (const DtStateMsg& msg) *Send a state message to the network which has been time stamped with the current time*
virtual void	**sendStamped** (const DtInteraction& msg) *Send an interaction to the network which has been time stamped with the current time*
virtual int	**drainInput** (DtTime timeout=-1, int nReadPerTimeCheck=100) *Read and process messages from the network until either no more messages, or for timeout seconds, whichever happpens first*
virtual void	**addPostDrainCallback** (DtPostDrainCb cb, void *usr) *add/remove callbacks to be executed from within drainInput right before it returns*
virtual void	**removePostDrainCallback** (DtPostDrainCb cb, void *usr)
	DtBaseExerciseConn () *default constructor*

Protected Fields

DtList	**myPostDrainCbs**

Protected Methods

	DtBaseExerciseConn (const DtBaseExerciseConn& orig) *copy constructor*

- virtual void **executePostDrainCbs** ()
  should be called by derived class' drainInputs

Protected

- *Constructors are protected since this is an abstract base class*

Documentation

- **`virtual ~DtBaseExerciseConn()`**
  destructor
- **`DtBaseExerciseConn& operator=(const DtBaseExerciseConn& orig)`**
  assignment operator
- **`virtual void send(const DtStateMsg& msg)`**
  Send a state message to the network
- **`virtual void send(const DtInteraction& msg)`**
  Send an interaction to the network
- **`virtual void sendStamped(const DtStateMsg& msg)`**
  Send a state message to the network which has been time stamped with the current time
- **`virtual void sendStamped(const DtInteraction& msg)`**
  Send an interaction to the network which has been time stamped with the current time
- **`virtual int drainInput(DtTime timeout=-1, int nReadPerTimeCheck=100)`**
  Read and process messages from the network until either no more messages, or for timeout seconds, whichever happpens first. Will read nReadPerTimeCheck messages at a time before checking for timeout. Returns number of incoming events which have been processed.
- **`virtual void addPostDrainCallback(DtPostDrainCb cb, void *usr)`**
  add/remove callbacks to be executed from within drainInput right before it returns
- **`virtual void removePostDrainCallback(DtPostDrainCb cb, void *usr)`**
-
  Constructors are protected since this is an abstract base class
- **`DtBaseExerciseConn()`**
  default constructor
- **`DtBaseExerciseConn(const DtBaseExerciseConn& orig)`**
  copy constructor
- **`virtual void executePostDrainCbs()`**
  should be called by derived class' drainInputs
- **`DtList myPostDrainCbs`**

Direct child classes:

DtExerciseConn

Alphabetic index *Hierarchy of classes*

In file hstatemsg.h:

class DtStateMsg

Public Methods

	DtStateMsg (RTI::ObjectHandle id, const RTI::AttributeHandleValuePairSet& attrs) *constructor*
	DtStateMsg (RTI::ObjectHandle id, RTI::AttributeHandleValuePairSet* attrs) *constructor*
virtual	**~DtStateMsg** () *destructor*
	DtStateMsg (const DtStateMsg& orig) *copy constructor*
DtStateMsg&	**operator=** (const DtStateMsg& orig) *assignment operator*
virtual const RTI::AttributeHandleValuePairSet&	**ahvps** () const *Get the attribute handle value pair set*
virtual RTI::AttributeHandleValuePairSet*	**ahvps** () *Get the attribute handle value pair set*
virtual RTI::ObjectHandle	**objectId** () const *Get the object identifier*
virtual void	**print** (DtExerciseConn* exConn, RTI::ObjectClassHandle h) const *Prints prints header and attributes with values*
virtual void	**print** (DtHlaObject *obj) const *Uses the object to print host representation of data*
virtual void	**printHeader** (DtExerciseConn* exConn, RTI::ObjectClassHandle classHandle) const *prints the header information*
virtual void	**printAttrs** (DtExerciseConn* exConn, RTI::ObjectClassHandle classHandle, int withHex) const *Print the attributes (with hex values if withHex is 1)*
virtual void	**setTag** (const DtString& tag) *Set/Get the user-defined tag*
virtual const DtString&	**tag** () const
virtual void	**setTimeStamp** (DtTime time, DtTimeStampType type = DtTimeStampRelative) *Set the time of validity of the message, and whether this represents relative or absolute time*
virtual DtTime	**timeStamp** () const *Return the time of validity of the message (in seconds past an hour) as set by setTimeStamp*
virtual DtTimeStampType	**timeStampType** () const *Return the type of the time stamp*

virtual DtTime	**guessTimeValid** (DtTime referenceTime) const *Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past that hour*
static void	**setTagIsAsciiTime** (DtBoolean val) *Set/Get flag that tells us whether we're using the RPR FOM convention that the UserDefinedTag in updates and interactions represents a DIS-style timestamp*
static DtBoolean	**tagIsAsciiTime** ()
inline const RTI::AttributeHandleValuePairSet&	**ahvps** () const
inline RTI::AttributeHandleValuePairSet*	**ahvps** ()
inline RTI::ObjectHandle	**objectId** () const

Protected Fields

DtString	**myTag**
RTI::AttributeHandleValuePairSet*	**myAvList**
RTI::ObjectHandle	**myId**
DtTime	**myTime**
DtTimeStampType	**myTimeType**
static DtBoolean	**theTagIsAsciiTime**

Documentation

- **`DtStateMsg(RTI::ObjectHandle id, const RTI::AttributeHandleValuePairSet& attrs)`**
  constructor
- **`DtStateMsg(RTI::ObjectHandle id, RTI::AttributeHandleValuePairSet* attrs)`**
  constructor
- **`virtual ~DtStateMsg()`**
  destructor
- **`DtStateMsg(const DtStateMsg& orig)`**
  copy constructor
- **`DtStateMsg& operator=(const DtStateMsg& orig)`**
  assignment operator
- **`virtual const RTI::AttributeHandleValuePairSet& ahvps() const`**
  Get the attribute handle value pair set. (Const version)
- **`virtual RTI::AttributeHandleValuePairSet* ahvps()`**
  Get the attribute handle value pair set. (Non-const version)

- **virtual RTI::ObjectHandle objectId() const**

  Get the object identifier

- **virtual void print(DtExerciseConn* exConn, RTI::ObjectClassHandle h) const**

  Prints prints header and attributes with values

- **virtual void print(DtHlaObject *obj) const**

  Uses the object to print host representation of data

- **virtual void printHeader(DtExerciseConn* exConn, RTI::ObjectClassHandle classHandle) const**

  prints the header information

- **virtual void printAttrs(DtExerciseConn* exConn, RTI::ObjectClassHandle classHandle, int withHex) const**

  Print the attributes (with hex values if withHex is 1)

- **virtual void setTag(const DtString& tag)**

  Set/Get the user-defined tag

- **virtual const DtString& tag() const**

- **virtual void setTimeStamp(DtTime time, DtTimeStampType type = DtTimeStampRelative)**

  Set the time of validity of the message, and whether this represents relative or absolute time. Time is in seconds, but only time modulo an hour is stored.

- **virtual DtTime timeStamp() const**

  Return the time of validity of the message (in seconds past an hour) as set by setTimeStamp

- **virtual DtTimeStampType timeStampType() const**

  Return the type of the time stamp

- **virtual DtTime guessTimeValid(DtTime referenceTime) const**

  Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past that hour

- **static void setTagIsAsciiTime(DtBoolean val)**

  Set/Get flag that tells us whether we're using the RPR FOM convention that the UserDefinedTag in updates and interactions represents a DIS-style timestamp

- **static DtBoolean tagIsAsciiTime()**

- **DtString myTag**

- **RTI::AttributeHandleValuePairSet* myAvList**

- **RTI::ObjectHandle myId**

- **DtTime myTime**

- **DtTimeStampType myTimeType**

- **static DtBoolean theTagIsAsciiTime**

- **inline const RTI::AttributeHandleValuePairSet& ahvps() const**

- `inline RTI::AttributeHandleValuePairSet* ahvps()`
- `inline RTI::ObjectHandle objectId() const`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hexerciseconn.h:

class DtExerciseConn: public DtBaseExerciseConn

class DtExerciseConn: Instances of DtExerciseConn are used to provide an application's interface or connection to the network in an HLA environment (as opposed to DIS)

Inheritance:

DtExerciseConn - DtBaseExerciseConn

Public Methods

	DtExerciseConn (const char* execName, const char* federateName, DtFomMapper* mapper = DtRprFomMapper::create(NULL), const char* fedFileName = NULL) *Constructor*
	DtExerciseConn (const char* execName, const char* federateName, const char* dsoName, const char* fedFileName = NULL, void* fomMapInitData = NULL) *Constructor*
virtual	**~DtExerciseConn** () *destructor*
virtual DtBoolean	**setReflecting** (DtBoolean onOff) *Set/Get a flag that determines whether local objects and interactions will be reflected back to us*
virtual DtBoolean	**reflecting** () const
virtual void	**send** (const DtStateMsg& msg) *Send a state message to the network*
virtual void	**send** (const DtInteraction& msg) *Send an interaction to the network*
virtual void	**sendStamped** (const DtStateMsg& msg) *Send a state message to the network which has been time stamped with the current time*
virtual void	**sendStamped** (const DtInteraction& msg) *Send an interaction to the network which has been time stamped with the current time*
virtual int	**drainInput** (DtTime timeout=-1, int nReadPerTimeCheck=100) *Read and process messages from the network until either no more messages, or for timeout seconds, whichever happpens first*
virtual void	**setObjectIdForEventId** (const DtGlobalObjectDesignator& id) *Set objectID to use in the next event ID*
virtual const DtEventID&	**nextEventId** () *Get a unique event ID*

virtual DtVrlRtiAmbassador*	**rtiAmb** () *Get the RTI ambassador*
virtual DtVrlFederateAmbassador*	**fedAmb** () *Get the federate ambassador*
virtual DtFom*	**fom** () *Get the FOM*
virtual DtInteractionFactory*	**interactionFactory** () *Get the interaction factory*
virtual DtHlaObjectFactory*	**objectFactory** () *Get the object factory*
virtual DtFomMapper*	**fomMapper** () *Get the Fom Mapper*
virtual DtReflectedHlaObjectManager*	**reflectedObjectManager** () *Get the reflectedObjectManager*
virtual DtLocalHlaObjectManager*	**localObjectManager** () *Get the localObjectManager*
virtual void	**setTimeStampType** (DtTimeStampType relOrAbs) *Set/Get time stamp type (DtTimeStampAbsolute or DtTimeStampRelative)*
virtual DtTimeStampType	**timeStampType** () const
virtual void	**addInteractionCallback** (RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr) *Add an interaction callback by class handle, which is called with the user data when the specified interaction is received*
virtual void	**addInteractionCallbackByName** (const char* className, DtReceiveInteractionCb cb, void* usr) *Add an interaction callback by class name, which is called with the user data when the specified interaction is received*
virtual void	**removeInteractionCallback** (RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr) *Remove an interaction callback by class handle*
virtual void	**removeInteractionCallbackByName** (const char* className, DtReceiveInteractionCb cb, void* usr) *Remove an interaction callback by class name Note: The user data is significant for identifying the callback and needs to be the same as when the callback was added*
virtual void	**addDiscoverObjectCallback** (RTI::ObjectClassHandle h, DtDiscoverObjectCb cb, void* usr) *Add a discoverObject callback which is called with the user data when the specified interaction is received*
virtual void	**removeDiscoverObjectCallback** (RTI::ObjectClassHandle h, DtDiscoverObjectCb cb, void* usr) *Remove a discoverObject callback*
virtual void	**addRemoveObjectCallback** (RTI::ObjectHandle id, DtRemoveObjectCb cb, void* usr) *Add a removeObject callback which is called with the user data when the specified interaction is received*
virtual void	**removeRemoveObjectCallback** (RTI::ObjectHandle id, DtRemoveObjectCb cb, void* usr) *Remove a removeObject callback*
virtual void	**setDestroyFedExecFlag** (DtBoolean flag) *Set/Get flag that determines whether or not the DtExerciseConn destructor will attempt to destroy the federation execution*
virtual DtBoolean	**destroyFedExecFlag** ()

- static void — **setFedAmbCreator** (DtFedAmbCreator creator)
  Set/get the function to be used to create a federate ambassador
- static DtFedAmbCreator — **fedAmbCreator** ()
- static void — **setRtiAmbCreator** (DtRtiAmbCreator creator)
  Set/get function to be used to create an RTI ambassador
- static DtRtiAmbCreator — **rtiAmbCreator** ()
- static void — **setFomMapperCreator** (DtFomMapperCreator creator, void* usr = NULL)
  Set/Get the default FomMapper creator - used to create a fomMapper when a NULL FOM Mapper or invalid FOM Mapper DSO name is passed to the constructor
- static DtFomMapperCreator — **fomMapperCreator** ()
- static const char* — **vrlinkVersion** ()
  Returns a string indicating the VR-Link version number
- inline DtVrlFederateAmbassador* — **fedAmb** ()
- inline DtFom* — **fom** ()
- inline DtLocalHlaObjectManager* — **localObjectManager** ()
- inline DtReflectedHlaObjectManager* — **reflectedObjectManager** ()
- inline DtVrlRtiAmbassador* — **rtiAmb** ()

Protected Fields

- *Note: The copy constructor and assignment operator are protected because it is not clear what these would mean*
- char* — **myExecName**
- char* — **myFedFileName**
- char* — **myFedName**
- RTI::FederateHandle — **myFedHandle**
- DtVrlRtiAmbassador* — **myRtiAmb**
- DtVrlFederateAmbassador* — **myFedAmb**
- DtFom* — **myFom**
- DtReflectedHlaObjectManager* — **myReflectedObjectManager**
- DtLocalHlaObjectManager* — **myLocalObjectManager**
- DtInteractionManager* — **myInteractionManager**
- DtFomMapper* — **myFomMapper**
- DtFomMapperCreator — **myDsoCreator**
- DtFomMapperDeletor — **myDsoDeletor**
- DtEventID — **myCurrentEventId**
- DtTimeStampType — **myRelOrAbs**

DtBoolean	**myDestroyFedExFlag**
DtBoolean	**myUseFomMapperDeletor**
static DtFedAmbCreator	**theFedAmbCreator**
static DtRtiAmbCreator	**theRtiAmbCreator**
static DtFomMapperCreator	**theFomMapperCreator**
static void*	**theFomMapInitData**

Protected Methods

	DtExerciseConn (const DtExerciseConn& orig) *copy constructor*
DtExerciseConn&	**operator=** (const DtExerciseConn& orig) *assignment operator*
void	**init** (const char* execName, const char* federateName, const char* fedFileName, DtFomMapper* mapper) *Not virtual since called by ctor*
void	**init** (const char* execName, const char* federateName, const char* fedFileName, const char* dsoName, void* fomMapInitData) *Not virtual since called by ctor*
void	**getFuncsFromSharedLib** (const DtString& libName, DtFomMapperCreator* creator, DtFomMapperDeletor* deletor, DtBoolean complain = DtTrue) *Not virtual since called by ctor*
void	**initFom** (const char* execName, const char* fedFileName) *Not virtual since called by ctor*
virtual DtTime	**currentTimeForStamping** () *Return time to send with outgoing messages (default is real time if absolute timestamping, sim time otherwise*
void	**createRtiAmb** () *Create the RTI ambassador*
void	**createFedEx** () *Create the federation execution*
void	**createFedAmb** () *Create the federate ambassador*
void	**joinFedEx** () *Join the federation execution*

Inherited from DtBaseExerciseConn:

Public Methods

- virtual void **addPostDrainCallback**(DtPostDrainCb cb, void *usr)
- virtual void **removePostDrainCallback**(DtPostDrainCb cb, void *usr)

Protected Fields

- DtList **myPostDrainCbs**

Protected Methods

- virtual void **executePostDrainCbs**()

Documentation

class DtExerciseConn: Instances of DtExerciseConn are used to provide an application's interface or connection to the network in an HLA environment (as opposed to DIS)

- **`DtExerciseConn(const char* execName, const char* federateName, DtFomMapper* mapper = DtRprFomMapper::create(NULL), const char* fedFileName = NULL)`**

  Constructor. Mapper is a pointer to a new'ed FOM Mapper to use. The FOM Mapper's init() function will be called by the DtExerciseConn constructor. The default value for mapper is a new'ed DtRprFomMapper. The mapper is deleted by DtExerciseConn destructor. If mapper is NULL, we create one using the creator function passed to setFomMapperCreator (default is an empty FOM Mapper). The FOM Mapper's init function is called after the RTI is initialized. If fedFileName is NULL, <execName>.fed is used.

- **`DtExerciseConn(const char* execName, const char* federateName, const char* dsoName, const char* fedFileName = NULL, void* fomMapInitData = NULL)`**

  Constructor. Similar to above, but rather than passing a FOM Mapper, pass the name of a shared library (DSO or DLL) that contains a FOM Mapper creation function called DtCreateFomMapper (and a FOM Mapper destruction function called DtDeleteFomMapper. fomMapInitData is a pointer to arbitrary data that you wish to have passed to the FOM Mapper creation function in your FOM Mapper shared library.

- **`virtual ~DtExerciseConn()`**

  destructor

- **`virtual DtBoolean setReflecting(DtBoolean onOff)`**

  Set/Get a flag that determines whether local objects and interactions will be reflected back to us. (Default in HLA is no, since the RTI does not do this. The set call's return value indicates whether or not reflecting could be enabled.

- **`virtual DtBoolean reflecting() const`**

- **`virtual void send(const DtStateMsg& msg)`**

  Send a state message to the network

- **`virtual void send(const DtInteraction& msg)`**

  Send an interaction to the network

- **`virtual void sendStamped(const DtStateMsg& msg)`**

  Send a state message to the network which has been time stamped with the current time

- **`virtual void sendStamped(const DtInteraction& msg)`**

  Send an interaction to the network which has been time stamped with the current time

- **`virtual int drainInput(DtTime timeout=-1, int nReadPerTimeCheck=100)`**

Read and process messages from the network until either no more messages, or for timeout seconds, whichever happpens first. Will read nReadPerTimeCheck messages at a time before checking for timeout. Returns number of incoming events which have been processed.

virtual void setObjectIdForEventId(const DtGlobalObjectDesignator& id)

Set objectID to use in the next event ID

virtual const DtEventID& nextEventId()

Get a unique event ID

virtual DtVrlRtiAmbassador* rtiAmb()

Get the RTI ambassador

virtual DtVrlFederateAmbassador* fedAmb()

Get the federate ambassador

virtual DtFom* fom()

Get the FOM

virtual DtInteractionFactory* interactionFactory()

Get the interaction factory

virtual DtHlaObjectFactory* objectFactory()

Get the object factory

virtual DtFomMapper* fomMapper()

Get the Fom Mapper

virtual DtReflectedHlaObjectManager* reflectedObjectManager()

Get the reflectedObjectManager

virtual DtLocalHlaObjectManager* localObjectManager()

Get the localObjectManager

virtual void setTimeStampType(DtTimeStampType relOrAbs)

Set/Get time stamp type (DtTimeStampAbsolute or DtTimeStampRelative)

virtual DtTimeStampType timeStampType() const

virtual void addInteractionCallback(RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr)

Add an interaction callback by class handle, which is called with the user data when the specified interaction is received. Rather than using this function, user code should typically use each derived DtInteraction's addCallback function.

virtual void addInteractionCallbackByName(const char* className, DtReceiveInteractionCb cb, void* usr)

Add an interaction callback by class name, which is called with the user data when the specified interaction is received. Rather than using this function, user code should typically use each derived DtInteraction's addCallback function.

virtual void removeInteractionCallback(RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr)

Remove an interaction callback by class handle. Note: The user data is significant for identifying the callback and needs to be the same as when the callback was added. Rather than using this function, user code should typically

use each derived DtInteraction's addCallback function.

- **virtual void removeInteractionCallbackByName(const char* className, DtReceiveInteractionCb cb, void* usr)**

  Remove an interaction callback by class name Note: The user data is significant for identifying the callback and needs to be the same as when the callback was added. Rather than using this function, user code should typically use each derived DtInteraction's addCallback function.

- **virtual void addDiscoverObjectCallback(RTI::ObjectClassHandle h, DtDiscoverObjectCb cb, void* usr)**

  Add a discoverObject callback which is called with the user data when the specified interaction is received

- **virtual void removeDiscoverObjectCallback(RTI::ObjectClassHandle h, DtDiscoverObjectCb cb, void* usr)**

  Remove a discoverObject callback. Note: The user data is significant for identifying the callback and needs to be the same as when the callback was added.

- **virtual void addRemoveObjectCallback(RTI::ObjectHandle id, DtRemoveObjectCb cb, void* usr)**

  Add a removeObject callback which is called with the user data when the specified interaction is received

- **virtual void removeRemoveObjectCallback(RTI::ObjectHandle id, DtRemoveObjectCb cb, void* usr)**

  Remove a removeObject callback. Note: The user data is significant for identifying the callback and needs to be the same as when the callback was added.

- **virtual void setDestroyFedExecFlag(DtBoolean flag)**

  Set/Get flag that determines whether or not the DtExerciseConn destructor will attempt to destroy the federation execution. Default is true.

- **virtual DtBoolean destroyFedExecFlag()**

- **static void setFedAmbCreator(DtFedAmbCreator creator)**

  Set/get the function to be used to create a federate ambassador. Call before creating a DtExericseConn if you want to use something other than a DtVrlFedAmbassador.

- **static DtFedAmbCreator fedAmbCreator()**

- **static void setRtiAmbCreator(DtRtiAmbCreator creator)**

  Set/get function to be used to create an RTI ambassador. Call before creating a DtExerciseConn if you want to use something other than a DtDirectRtiAmbassador.

- **static DtRtiAmbCreator rtiAmbCreator()**

- **static void setFomMapperCreator(DtFomMapperCreator creator, void* usr = NULL)**

  Set/Get the default FomMapper creator - used to create a fomMapper when a NULL FOM Mapper or invalid FOM Mapper DSO name is passed to the constructor. Set before creating a DtExConn if you want it to have an effect. The FomMapperCreator defaults to DtFomMapper::create. usr allows you to specify some arbitary initialization data that will be passed to the FOM Mapper creator when it is called. VR-Link maintains a pointer to the object passed as usr as long as the DtExerciseConn and DtFomMapper are around, and does not deleted it.

- **static DtFomMapperCreator fomMapperCreator()**

- **static const char* vrlinkVersion()**

  Returns a string indicating the VR-Link version number

-

Note: The copy constructor and assignment operator are protected because it is not clear what these would mean. An execution name must be unique. These should not be called.

- **DtExerciseConn(const DtExerciseConn& orig)**

  copy constructor

- **DtExerciseConn& operator=(const DtExerciseConn& orig)**

  assignment operator

- **void init(const char* execName, const char* federateName, const char* fedFileName, DtFomMapper* mapper)**

  Not virtual since called by ctor. NULL fedFileName means use <execName>.fed. If FOM Mapper is NULL, use fomMapperCreator().

- **void init(const char* execName, const char* federateName, const char* fedFileName, const char* dsoName, void* fomMapInitData)**

  Not virtual since called by ctor. NULL fedFileName means use <execName>.fed. If can't get functions from dso, use fomMapperCreator().

- **void getFuncsFromSharedLib(const DtString& libName, DtFomMapperCreator* creator, DtFomMapperDeletor* deletor, DtBoolean complain = DtTrue)**

  Not virtual since called by ctor. Get pointers to functions called DtCreateFomMapper and DtDestroyFomMapper from DSO or DLL named libName. If complain is DtFalse, don't print diagnostic if can't find library. Pointers will point to NULL on failure.

- **void initFom(const char* execName, const char* fedFileName)**

  Not virtual since called by ctor. Initializes myFom from fed file.

- **virtual DtTime currentTimeForStamping()**

  Return time to send with outgoing messages (default is real time if absolute timestamping, sim time otherwise

- **void createRtiAmb()**

  Create the RTI ambassador. Not virtual since called by ctor. Generates FatalError if unsuccessful.

- **void createFedEx()**

  Create the federation execution. Not virtual since called by ctor. Generates FatalError if unsuccessful.

- **void createFedAmb()**

  Create the federate ambassador. Not virtual since called by ctor. Generates FatalError if unsuccessful.

- **void joinFedEx()**

  Join the federation execution. Not virtual since called by ctor. Generates FatalError if unsuccessful.

- **char* myExecName**

- **char* myFedFileName**

- **char* myFedName**

- **RTI::FederateHandle myFedHandle**

- **DtVrlRtiAmbassador* myRtiAmb**

- **DtVrlFederateAmbassador* myFedAmb**

- **DtFom* myFom**

- DtReflectedHlaObjectManager* myReflectedObjectManager
- DtLocalHlaObjectManager* myLocalObjectManager
- DtInteractionManager* myInteractionManager
- DtFomMapper* myFomMapper
- DtFomMapperCreator myDsoCreator
- DtFomMapperDeletor myDsoDeletor
- DtEventID myCurrentEventId
- DtTimeStampType myRelOrAbs
- DtBoolean myDestroyFedExFlag
- DtBoolean myUseFomMapperDeletor
- static DtFedAmbCreator theFedAmbCreator
- static DtRtiAmbCreator theRtiAmbCreator
- static DtFomMapperCreator theFomMapperCreator
- static void* theFomMapInitData
- inline DtVrlFederateAmbassador* fedAmb()
- inline DtFom* fom()
- inline DtLocalHlaObjectManager* localObjectManager()
- inline DtReflectedHlaObjectManager* reflectedObjectManager()
- inline DtVrlRtiAmbassador* rtiAmb()

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hlaobj.h:

class DtHlaObject

class DtHlaObject: Instances of DtHlaObject are used to represent HLA objects (as opposed to interactions) which are communicated across the network

---

Inheritance:

DtHlaObject

Public Methods

	DtHlaObject (DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc)
	DtHlaObject (DtExerciseConn* conn, const char* name, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtHlaObject** () *destructor*
virtual const RTI::ObjectHandle&	**objectId** () const *Get the object's indentifier*
virtual const char*	**name** () const *Get the object's indentifier*
virtual RTI::ObjectClassHandle	**objectClassHandle** () const *Get the object's class handle*
virtual int	**numAttributes** () const *Get the number of attributes associated with this object*
virtual const RTI::AttributeHandleSet&	**attrHandleSet** () *Get the attribute handle set*
virtual DtExerciseConn*	**exerciseConn** () const *Get the exercise connection for this object*
virtual DtObjClassDesc*	**classDesc** () *Get the class descriptor for the object's class*
virtual void	**setupUpdating** () *Do the initialization required to begin publishing this object*
virtual void	**update** () *Update (publish) this object on the network*
virtual void	**generateUpdate** (const RTI::AttributeHandleSet& subscribedAttrs, const RTI::AttributeHandleSet& requiredAttrs, DtStateMsg* stateMsg) *Called by update to actually generate an outgoing message*
virtual void	**cleanupFromUpdating** () *Do the deinitialization required to stop publishing this object*

- virtual void — **processProvideUpdate** (const RTI::AttributeHandleSet& theAttributes)
  Set yourself up so that on the next tick, you send an update for the appropriate attributes
- virtual const RTI::AttributeHandleSet& — **requestedAttributes** ()
  returns the union of all sets of attributes requested since the last call to clearRequestedAttributes
- virtual void — **clearRequestedAttributes** ()
  clears the set of requested attributes
- virtual void — **forceUpdate** ()
  Specify a set of attributes that should be included in the ahvps the next time update is called, even if they would not otherwise need to be sent
- virtual void — **forceUpdate** (const RTI::AttributeHandleSet& set)
- virtual void — **clearForcedAttributes** ()
  Clear (empty) the set of forced attributes
- virtual const RTI::AttributeHandleSet& — **attributesNeededByFederation** ()
  returns the set of attributes that are currently needed by the federation, i
- DtBoolean — **classNeededByFederation** ()
  returns DtTrue if the class is needed by the federation, i
- virtual void — **setupClassForUpdating** ()
  If the class has never been published, publish it with all attributes using DtObjClassDesc::publish
- virtual void — **setupReflecting** ()
  Do the initialization required to handle reflection of object
- virtual void — **callReflect** (const DtStateMsg& stateMsg)
  Calls reflect, and any postReflect callbacks Override reflect rather than this function to change the way an HLA object reflects a stateMsg
- virtual void — **cleanupFromReflecting** ()
  Do the deinitialization required to stop handling reflection of object
- virtual void — **requestUpdate** (RTI::AttributeHandleSet* attrs = NULL)
  Request an update for the object for the set of attributes indicated by attrs (or all subscribed attributes if attrs is NULL)
- virtual void — **addRemoveObjectCb** (DtRemoveObjectCb cb, void* usr)
  Add (register) function to be called when object is removed
- virtual void — **removeRemoveObjectCb** (DtRemoveObjectCb cb, void* usr)
  Remove (deregister) function to be called when object is removed
- virtual void — **addPostReflectCb** (DtObjectReflectAttributeValuesCb cb, void* usr)
  Add (register) function to be called after each reflectAttrValues
- virtual void — **removePostReflectCb** (DtObjectReflectAttributeValuesCb cb, void* usr)
  Remove (deregister) function to be called after each reflectAttrValues
- virtual DtTime — **lastSimTimeUpdateReceived** () const
- virtual void — **printMsg** (const DtStateMsg& msg) const
  Prints contents of a state msg
- static int — **maxObjectClasses** ()
  Get maximum number of object class handles supported
- inline const RTI::AttributeHandleSet& — **attrHandleSet** ()
- inline DtExerciseConn* — **exerciseConn** () const
- inline DtTime — **lastSimTimeUpdateReceived** () const
- inline int — **maxObjectClasses** ()
- inline const char* — **name** () const
- inline int — **numAttributes** () const

- inline RTI::ObjectClassHandle **objectClassHandle** () const
- inline const RTI::ObjectHandle& **objectId** () const
- inline const RTI::AttributeHandleSet& **requestedAttributes** ()
- inline void **setLastSimTimeUpdateReceived** (DtTime time)

Public

- *Local Object specific functions*
- *Remote Object specific functions*

Protected Fields

- char* **myName**
- RTI::AttributeHandleSet* **myNeededAttrs**
  Needed attributes is a per-object concept in RTI1
- DtExerciseConn* **myExConn**
- RTI::ObjectHandle **myObjectId**
- DtBoolean **myReflecting**
- DtBoolean **myUpdating**
- DtList **myRemoveCbList**
- DtTime **myLastSimTimeUpdateReceived**
- DtList **myReflectCbList**
- DtObjClassDesc* **myClassDesc**
- RTI::AttributeHandleSet* **myRequestedAttributes**
  Requested by other federates
- RTI::AttributeHandleSet* **myForcedAttributes**
  Requested by a call to setForcedUpdate
- RTI::AttributeHandleSet* **myRequiredAttributes**
  Union of forced and requested - not computed until needed
- static const int **theMaxObjectClasses**

Protected Methods

- **DtHlaObject** (const DtHlaObject& orig)
  copy constructor
- DtHlaObject& **operator=** (const DtHlaObject& orig)
  assignment operator
- virtual void **processRemoveObject** ()
  Process notification of removal of the object from the network
- virtual void **reflect** (const DtStateMsg& stateMsg)
  Handle reflection of object (receipt of state message) from the network

- void **wipeRemoveList** ()
  Remove all callbacks from the remove callback list
- virtual void **clearRemoveList** ()
  Virtual version of above
- void **wipeReflectList** ()
  Remove all callbacks from the reflect callback list
- void **copyRemoveList** (const DtHlaObject& orig)
  Copy another HLA object's remove callback list into this object's remove callback list
- virtual void **setLastSimTimeUpdateReceived** (DtTime time)
  Set the time an update was last received or sent
- virtual void **applyPostReflectCallbacks** (const DtStateMsg& stateMsg)
  apply callbacks after a state msg is reflected
- virtual void **processUpdatesOn** (RTI::ObjectHandle hand, const RTI::AttributeHandleSet& set)
  Callback for handling turnUpdatesOnForObjectInstance RTI call
- virtual void **processUpdatesOff** (RTI::ObjectHandle hand, const RTI::AttributeHandleSet& set)
  Callback for handling turnUpdatesOffForObjectInstance RTI call
- static void **reflectAttributeValuesCb** (const DtStateMsg& stateMsg, void* hlaObj)
  Callback for handling state messages for the HLA object
- static void **updatesOnCb** (RTI::ObjectHandle hand, const RTI::AttributeHandleSet& attributeSet, void* usr)
  Callback for handling turnUpdatesOnForObjectInstance RTI call
- static void **updatesOffCb** (RTI::ObjectHandle hand, const RTI::AttributeHandleSet& attributeSet, void* usr)
  Callback for handling turnUpdatesOffForObjectInstance RTI call

---

Documentation

class DtHlaObject: Instances of DtHlaObject are used to represent HLA objects (as opposed to interactions) which are communicated across the network

- **`DtHlaObject(DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc)`**
- **`DtHlaObject(DtExerciseConn* conn, const char* name, DtObjClassDesc* classDesc)`**

  constructor
- **`virtual ~DtHlaObject()`**

  destructor
- **`virtual const RTI::ObjectHandle& objectId() const`**

  Get the object's indentifier
- **`virtual const char* name() const`**

  Get the object's indentifier
- **`virtual RTI::ObjectClassHandle objectClassHandle() const`**

  Get the object's class handle
- **`virtual int numAttributes() const`**

  Get the number of attributes associated with this object
- **`virtual const RTI::AttributeHandleSet& attrHandleSet()`**

  Get the attribute handle set

- **virtual DtExerciseConn* exerciseConn() const**

  Get the exercise connection for this object

- **virtual DtObjClassDesc* classDesc()**

  Get the class descriptor for the object's class

-

  Local Object specific functions

- **virtual void setupUpdating()**

  Do the initialization required to begin publishing this object

- **virtual void update()**

  Update (publish) this object on the network. Called by publisher's tick functions to generate and send needed data. Default implementation is to call the pure virtual generateUpdate, and then send the generated message.

- **virtual void generateUpdate(const RTI::AttributeHandleSet& subscribedAttrs, const RTI::AttributeHandleSet& requiredAttrs, DtStateMsg* stateMsg)**

  Called by update to actually generate an outgoing message. Derived class implementations should keep track of which attributes are updated when, should check update conditions, and add values for only those attributes that currently need to be sent. subscribedAttrs is the set of attributes to which remote federates have subscribed. requiredAttrs is the set of attributes that have been requested or forced by application code since the attribute was last sent. stateMsg is the message to fill out. Default implementation is NULL, but the function is not pure virtual since some subclasses might want to override update rather than overriding generateUpdate.

- **virtual void cleanupFromUpdating()**

  Do the deinitialization required to stop publishing this object

- **virtual void processProvideUpdate(const RTI::AttributeHandleSet& theAttributes)**

  Set yourself up so that on the next tick, you send an update for the appropriate attributes. (Can't send them within this function since this is called from within an RTI service).

- **virtual const RTI::AttributeHandleSet& requestedAttributes()**

  returns the union of all sets of attributes requested since the last call to clearRequestedAttributes

- **virtual void clearRequestedAttributes()**

  clears the set of requested attributes. Should be called after request has been serviced.

- **virtual void forceUpdate()**

  Specify a set of attributes that should be included in the ahvps the next time update is called, even if they would not otherwise need to be sent. The call with no arguments, indicates the set of all of the object's published attributes. Multiple calls are cumulative, until the set is cleared with clearForcedAttributes, which should happen within update.

- **virtual void forceUpdate(const RTI::AttributeHandleSet& set)**

- **virtual void clearForcedAttributes()**

  Clear (empty) the set of forced attributes

- **virtual const RTI::AttributeHandleSet& attributesNeededByFederation()**

  returns the set of attributes that are currently needed by the federation, i.e. those that have been specified in startUpdates calls, but not those that have been specified in stopUpdates.

- **DtBoolean classNeededByFederation()**

  returns DtTrue if the class is needed by the federation, i.e. we've received a startRegistration or startUpdates

service call from the RTI for the object's class.

- **virtual void setupClassForUpdating()**

  If the class has never been published, publish it with all attributes using DtObjClassDesc::publish

-

  Remote Object specific functions

- **virtual void setupReflecting()**

  Do the initialization required to handle reflection of object

- **virtual void callReflect(const DtStateMsg& stateMsg)**

  Calls reflect, and any postReflect callbacks Override reflect rather than this function to change the way an HLA object reflects a stateMsg

- **virtual void cleanupFromReflecting()**

  Do the deinitialization required to stop handling reflection of object

- **virtual void requestUpdate(RTI::AttributeHandleSet* attrs = NULL)**

  Request an update for the object for the set of attributes indicated by attrs (or all subscribed attributes if attrs is NULL)

- **virtual void addRemoveObjectCb(DtRemoveObjectCb cb, void* usr)**

  Add (register) function to be called when object is removed

- **virtual void removeRemoveObjectCb(DtRemoveObjectCb cb, void* usr)**

  Remove (deregister) function to be called when object is removed

- **virtual void addPostReflectCb(DtObjectReflectAttributeValuesCb cb, void* usr)**

  Add (register) function to be called after each reflectAttrValues

- **virtual void removePostReflectCb(DtObjectReflectAttributeValuesCb cb, void* usr)**

  Remove (deregister) function to be called after each reflectAttrValues

- **virtual DtTime lastSimTimeUpdateReceived() const**

- **virtual void printMsg(const DtStateMsg& msg) const**

  Prints contents of a state msg. Base class prints only header info.

- **static int maxObjectClasses()**

  Get maximum number of object class handles supported

- **DtHlaObject(const DtHlaObject& orig)**

  copy constructor

- **DtHlaObject& operator=(const DtHlaObject& orig)**

  assignment operator

- **virtual void processRemoveObject()**

  Process notification of removal of the object from the network. Call back all the remove object callbacks.

- **virtual void reflect(const DtStateMsg& stateMsg)**

  Handle reflection of object (receipt of state message) from the network. called by callReflect. Override this function rather than callReflect to change the way an HLA object reflects a stateMsg.

- **void wipeRemoveList()**

  Remove all callbacks from the remove callback list. Non-virtual since meant to be called from dtor.

- **virtual void clearRemoveList()**

  Virtual version of above

- **void wipeReflectList()**

  Remove all callbacks from the reflect callback list. Non-virtual since meant to be called from dtor.

- **void copyRemoveList(const DtHlaObject& orig)**

  Copy another HLA object's remove callback list into this object's remove callback list. Non-virtual since meant to be called from ctor.

- **virtual void setLastSimTimeUpdateReceived(DtTime time)**

  Set the time an update was last received or sent

- **virtual void applyPostReflectCallbacks(const DtStateMsg& stateMsg)**

  apply callbacks after a state msg is reflected

- **virtual void processUpdatesOn(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& set)**

  Callback for handling turnUpdatesOnForObjectInstance RTI call

- **virtual void processUpdatesOff(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& set)**

  Callback for handling turnUpdatesOffForObjectInstance RTI call

- **static void reflectAttributeValuesCb(const DtStateMsg& stateMsg, void* hlaObj)**

  Callback for handling state messages for the HLA object. Casts hlaObj to DtHlaObject* and calls reflect().

- **static void updatesOnCb(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& attributeSet, void* usr)**

  Callback for handling turnUpdatesOnForObjectInstance RTI call

- **static void updatesOffCb(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& attributeSet, void* usr)**

  Callback for handling turnUpdatesOffForObjectInstance RTI call

- **char* myName**

- **RTI::AttributeHandleSet* myNeededAttrs**

  Needed attributes is a per-object concept in RTI1

- **DtExerciseConn* myExConn**

- **RTI::ObjectHandle myObjectId**

- **DtBoolean myReflecting**

- **DtBoolean myUpdating**

- **DtList myRemoveCbList**

- **DtTime myLastSimTimeUpdateReceived**

- **DtList myReflectCbList**

- **DtObjClassDesc* myClassDesc**
- **RTI::AttributeHandleSet* myRequestedAttributes**
  Requested by other federates
- **RTI::AttributeHandleSet* myForcedAttributes**
  Requested by a call to setForcedUpdate
- **RTI::AttributeHandleSet* myRequiredAttributes**
  Union of forced and requested - not computed until needed
- **static const int theMaxObjectClasses**
- **inline const RTI::AttributeHandleSet& attrHandleSet()**
- **inline DtExerciseConn* exerciseConn() const**
- **inline DtTime lastSimTimeUpdateReceived() const**
- **inline int maxObjectClasses()**
- **inline const char* name() const**
- **inline int numAttributes() const**
- **inline RTI::ObjectClassHandle objectClassHandle() const**
- **inline const RTI::ObjectHandle& objectId() const**
- **inline const RTI::AttributeHandleSet& requestedAttributes()**
- **inline void setLastSimTimeUpdateReceived(DtTime time)**

Direct child classes:

DtUnknownHlaObject
DtHlaObjectWithStateRep

Alphabetic index *Hierarchy of classes*

In file str.h:

class DtString

class DtString: Instances of DtString are

Inheritance:

DtString

Public Methods

	DtString () *default constructor*
	DtString (const char* str)
virtual	**~DtString** () *destructor*
	DtString (const DtString& orig) *copy constructor*
DtString&	**operator=** (const DtString& orig) *assignment operators*
DtString&	**operator=** (const char* orig)
virtual	**operator const char*** () const *cast operator*
friend DtBoolean	**operator==** (const DtString& str1, const DtString& str2) *comparison operators*
friend DtBoolean	**operator<** (const DtString& str1, const DtString& str2)
friend DtBoolean	**operator<=** (const DtString& str1, const DtString& str2)
friend DtBoolean	**operator>** (const DtString& str1, const DtString& str2)
friend DtBoolean	**operator>=** (const DtString& str1, const DtString& str2)
friend DtBoolean	**operator!=** (const DtString& str1, const DtString& str2)
virtual DtBoolean	**operator==** (const char* str) const
virtual DtBoolean	**operator<** (const char* str) const
virtual DtBoolean	**operator<=** (const char* str) const
virtual DtBoolean	**operator>** (const char* str) const
virtual DtBoolean	**operator>=** (const char* str) const

- virtual DtBoolean **operator!=** (const char* str) const
- virtual DtBoolean **caseCompare** (const DtString& str)
  Case-insensitive string compare
- friend DtString **operator+** (const DtString& str1, const DtString& str2)
  concatenation operators
- virtual DtString **operator+** (const char* str) const
- virtual DtString& **operator+=** (const char* str)
- virtual DtString& **operator+=** (const DtString& str)
- virtual const char& **operator[]** (int index) const
  index operator
- virtual char& **operator[]** (int index)
- virtual DtBoolean **snipFront** (const char ch, DtBoolean toLastCh = DtFalse)
  truncates everything before or after the character, including the character, respectively Normally, snipFront stops at the first occurrence of "ch", and snipBack stops at the last occurrence of "ch"
- virtual DtBoolean **snipBack** (const char ch, DtBoolean fromFirstCh = DtFalse)
- DtBoolean **snipFrontToLast** (const char ch)
- DtBoolean **snipBackToFirst** (const char ch)
- virtual int **count** (const char ch) const
  returns the number of occurrences of the character
- virtual int **length** () const
  return the length of the string
- const char* **string** () const
  returns the char internal representation of the string*
- static const DtString& **nullString** ()

Protected Fields

- char* **myString**
- static const int **theMaxInsertionStringSize**

Protected Methods

- void **clear** ()
  deletes the string, and sets the myString pointer to NULL non-virtual because only called by destructor, and non-virtual members myString is set to NULL, so it should be allocated some memory if operators are to be used on the string
- void **setString** (const char* str)
  Set/Get for myString - char pointer that holds the string non-virtual because only called by constructor

Private Methods

- friend ostream& **operator<<** (ostream& os, const DtString& str)
- friend istream& **operator>>** (istream& is, DtString& str)

Documentation

class DtString: Instances of DtString are

- `friend ostream& operator<<(ostream& os, const DtString& str)`
- `friend istream& operator>>(istream& is, DtString& str)`
- `DtString()`
  default constructor
- `DtString(const char* str)`
- `virtual ~DtString()`
  destructor
- `DtString(const DtString& orig)`
  copy constructor
- `DtString& operator=(const DtString& orig)`
  assignment operators
- `DtString& operator=(const char* orig)`
- `virtual operator const char*() const`
  cast operator
- `friend DtBoolean operator==(const DtString& str1, const DtString& str2)`
  comparison operators
- `friend DtBoolean operator<(const DtString& str1, const DtString& str2)`
- `friend DtBoolean operator<=(const DtString& str1, const DtString& str2)`
- `friend DtBoolean operator>(const DtString& str1, const DtString& str2)`
- `friend DtBoolean operator>=(const DtString& str1, const DtString& str2)`
- `friend DtBoolean operator!=(const DtString& str1, const DtString& str2)`
- `virtual DtBoolean operator==(const char* str) const`
- `virtual DtBoolean operator<(const char* str) const`
- `virtual DtBoolean operator<=(const char* str) const`

- **virtual DtBoolean operator>(const char* str) const**
- **virtual DtBoolean operator>=(const char* str) const**
- **virtual DtBoolean operator!=(const char* str) const**
- **virtual DtBoolean caseCompare(const DtString& str)**

  Case-insensitive string compare. Returns DtTrue if the strings are equal except for differences in case.
- **friend DtString operator+(const DtString& str1, const DtString& str2)**

  concatenation operators
- **virtual DtString operator+(const char* str) const**
- **virtual DtString& operator+=(const char* str)**
- **virtual DtString& operator+=(const DtString& str)**
- **virtual const char& operator[](int index) const**

  index operator
- **virtual char& operator[](int index)**
- **virtual DtBoolean snipFront(const char ch, DtBoolean toLastCh = DtFalse)**

  truncates everything before or after the character, including the character, respectively Normally, snipFront stops at the first occurrence of "ch", and snipBack stops at the last occurrence of "ch". The second argument flags reverse this. Return value is DtTrue iff "ch" was found in the string.
- **virtual DtBoolean snipBack(const char ch, DtBoolean fromFirstCh = DtFalse)**
- **DtBoolean snipFrontToLast(const char ch)**
- **DtBoolean snipBackToFirst(const char ch)**
- **virtual int count(const char ch) const**

  returns the number of occurrences of the character
- **virtual int length() const**

  return the length of the string
- **const char* string() const**

  returns the char* internal representation of the string
- **static const DtString& nullString()**
- **void clear()**

  deletes the string, and sets the myString pointer to NULL non-virtual because only called by destructor, and non-virtual members myString is set to NULL, so it should be allocated some memory if operators are to be used on the string
- **void setString(const char* str)**

  Set/Get for myString - char pointer that holds the string non-virtual because only called by constructor
- **char* myString**
- **static const int theMaxInsertionStringSize**

Direct child classes:

DtFilename

Alphabetic index *Hierarchy of classes*

In file tstamptype.h:

enum DtTimeStampType

- **DtTimeStampRelative**
- **DtTimeStampAbsolute**

Documentation

- **`DtTimeStampRelative`**
- **`DtTimeStampAbsolute`**

In file fommapper.h:

class DtFomMapper

Inheritance:

DtFomMapper

Public Methods

typedef char*	**(*DtInteractionClassChooser)** (DtExerciseConn* exConn, const DtInteraction& inter) *A function that chooses an interaction class from the FOM based on the contents of a derived DtInteraction object*
typedef char*	**(*DtObjectClassChooser)** (const char* vrlinkClassName, DtExerciseConn* conn, void* usr) *A function that chooses an object class from the FOM based based on the name of a C++ class derived from DtObjectPublisher, and arbitary user data (e*
	DtFomMapper () *Default constructor*
virtual	**~DtFomMapper** () *destructor*
virtual DtStateDecoderFactory*	**setStateDecoderFactory** (DtStateDecoderFactory* fact) *Set/Get the current factory that can be used to create DtHlaStateDecoders*
virtual DtStateDecoderFactory*	**stateDecoderFactory** ()
virtual DtStateEncoderFactory*	**setStateEncoderFactory** (DtStateEncoderFactory* fact) *Set/Get the current factory that can be used to create DtHlaStateEncoders*
virtual DtStateEncoderFactory*	**stateEncoderFactory** ()
virtual DtInteractionDecoderFactory*	**setInteractionDecoderFactory** (DtInteractionDecoderFactory* fact) *Set/Get the current factory that can be used to create DtInteractionDecoders*
virtual DtInteractionDecoderFactory*	**interactionDecoderFactory** ()
virtual DtInteractionEncoderFactory*	**setInteractionEncoderFactory** (DtInteractionEncoderFactory* fact) *Set/Get the current factory that can be used to create DtInteractionEncoders*
virtual DtInteractionEncoderFactory*	**interactionEncoderFactory** ()
virtual DtHlaObjectFactory*	**setHlaObjectFactory** (DtHlaObjectFactory* fact) *Set/Get the current factory that can be used to create DtHlaObjects*
virtual DtHlaObjectFactory*	**hlaObjectFactory** ()
virtual DtInteractionFactory*	**setInteractionFactory** (DtInteractionFactory* fact) *Set/Get the current factory that can be used to create DtInteractions*

- virtual DtInteractionFactory* **interactionFactory** ()
- virtual void **setObjectClassToChoose** (const char* vrlinkClassName, const char* fomClassName)
  Tell the FOM Mapper how to choose a FOM object class to use to for objects being published by a particular kind of object publisher
- virtual void **setObjectClassChooser** (const char* vrlinkClassName, DtObjectClassChooser chooser)
- virtual char* **chooseObjectClass** (const char* vrlinkClassName, void* usr = NULL) const
  Chooses an object class to represent an object managed by the C++ class named vrlinkClassName (which should be a class derived from DtObjectPublisher)
- virtual void **setInteractionClassToChoose** (const char* vrlinkClassName, const char* fomClassName)
  Tell the FOM Mapper how to choose a FOM interaction class to use to for a particular kind of derived DtInteraction
- virtual void **setInteractionClassChooser** (const char* vrlinkClassName, DtInteractionClassChooser chooser)
- virtual char* **chooseInteractionClass** (const DtInteraction& inter) const
  Chooses a FOM interaction class to be by a particular derived DtInteraction class named vrlinkClassName
- virtual void **setInteractionClasses** (const char* vrlinkClassName, const DtList& classes)
  Set/Get a list of names (char) of FOM classes that can be decoded into the derived DtInteraction class whose name is name*
- virtual DtList* **interactionClasses** (const char* vrlinkClassName)
- virtual void **setInteractionClass** (const char* vrlinkClassName, const char* fomClassName)
  When there is only a single FOM class to be associated with a VR-Link interaction class, then there is no need to create a DtList
- virtual void **setObjectClassNames** (const char* vrlinkClassName, const DtList& classes)
  Set/Get a list of names (char) of FOM object classes that should be subscribed to by a particular subclass of VR-Link's DtReflectedObjectList*
- virtual DtList* **objectClassNames** (const char* vrlinkClassName)
- virtual DtList* **objectClassHandles** (const char* vrlinkClassName)
  Similar to objectClassNames, but returns a DtList of ObjectClassHandles instead
- virtual void **setObjectClass** (const char* vrlinkClassName, const char* fomClassName)
  When there is only a single FOM class to be associated with a VR-Link DtReflectedObjectList class, then there is no need to create a DtList
- virtual void **init** (DtExerciseConn* conn)
  Initialize the instance, once an exercise connection is available
- virtual void **setVersion** (double vers)
  Returns the version number of the FOM, as passed to setVersion
- virtual double **version** () const

Protected Fields

- DtExerciseConn* **myExConn**
- DtStateDecoderFactory* **myDecFact**
  Current factories and other objects
- DtStateEncoderFactory* **myEncFact**
- DtInteractionDecoderFactory* **myInterDecFact**

DtInteractionEncoderFactory*	**myInterEncFact**
DtHlaObjectFactory*	**myObjFact**
DtInteractionFactory*	**myInterFact**
DtHashlist	**myIntMapList** *Hashlist that maps names of C++ classes derived from DtInteraction to lists of names of FOM classes*
DtHashlist	**myObjMapList** *Hashlist that maps names of C++ classes derived from DtReflectedObjectList to lists of names of FOM classes*
DtHashlist	**myObjChooserList** *Hashlist that maps names of C++ classes derived from DtObjectPublisher to DtObjectClassChoosers - functions that can choose an object class name from the FOM to use for the object*
DtHashlist	**mySimpleObjChooserList** *Hashlist that maps names of C++ classes derived from DtObjectPublisher to single FOM object class names*
DtHashlist	**myInterChooserList** *Hashlist that maps names of C++ classes derived from DtInteraction to DtInteractionClassChoosers - functions that can choose an interaction class name from the FOM to use for the interaction*
DtHashlist	**mySimpleInterChooserList** *Hashlist that maps names of C++ classes derived from DtInteraction to single FOM object class names*
DtList	**myObjHandleList** *The object returned by objectClassHandles*
DtList	**myEmptyList** *Just an empty list to return when no class list has been registered for a particular VR-Link class*
double	**myVersion**

Protected Methods

	DtFomMapper (const DtFomMapper& orig) *copy constructor - not implemented*
DtFomMapper&	**operator=** (const DtFomMapper& orig) *assignment operator - not implemented*
virtual DtObjectClassChooser	**objectClassChooser** (const char* vrlinkClassName) const *Return the FOM-class-chooser function, if any, that has been associated with the name of a C++ class derived from DtObjectPublisher using setObjectClassChooser*
virtual DtInteractionClassChooser	**interactionClassChooser** (const char* vrlinkClassName) const *Return the FOM-class-chooser function, if any, that has been associated with the name of a C++ class derived from DtInteraction using setInteractionClassChooser*

Documentation

`typedef char* (*DtInteractionClassChooser)( DtExerciseConn* exConn, const DtInteraction& inter)`

A function that chooses an interaction class from the FOM based on the contents of a derived DtInteraction object

typedef char* (*DtObjectClassChooser)(const char* vrlinkClassName, DtExerciseConn* conn, void* usr)

A function that chooses an object class from the FOM based based on the name of a C++ class derived from DtObjectPublisher, and arbitary user data (e.g. entity type).

DtFomMapper()

Default constructor. Typically, most initialization takes place within init(), which is called by DtExerciseConn's constructor.

virtual ~DtFomMapper()

destructor

virtual DtStateDecoderFactory* setStateDecoderFactory(DtStateDecoderFactory* fact)

Set/Get the current factory that can be used to create DtHlaStateDecoders

virtual DtStateDecoderFactory* stateDecoderFactory()

virtual DtStateEncoderFactory* setStateEncoderFactory(DtStateEncoderFactory* fact)

Set/Get the current factory that can be used to create DtHlaStateEncoders

virtual DtStateEncoderFactory* stateEncoderFactory()

virtual DtInteractionDecoderFactory* setInteractionDecoderFactory(DtInteractionDecoderFactory* fact)

Set/Get the current factory that can be used to create DtInteractionDecoders

virtual DtInteractionDecoderFactory* interactionDecoderFactory()

virtual DtInteractionEncoderFactory* setInteractionEncoderFactory(DtInteractionEncoderFactory* fact)

Set/Get the current factory that can be used to create DtInteractionEncoders

virtual DtInteractionEncoderFactory* interactionEncoderFactory()

virtual DtHlaObjectFactory* setHlaObjectFactory(DtHlaObjectFactory* fact)

Set/Get the current factory that can be used to create DtHlaObjects

virtual DtHlaObjectFactory* hlaObjectFactory()

virtual DtInteractionFactory* setInteractionFactory(DtInteractionFactory* fact)

Set/Get the current factory that can be used to create DtInteractions

virtual DtInteractionFactory* interactionFactory()

virtual void setObjectClassToChoose(const char* vrlinkClassName, const char* fomClassName)

Tell the FOM Mapper how to choose a FOM object class to use to for objects being published by a particular kind of object publisher. You can either associate a single FOM class name, or a FOM class choosing function with the name of a C++ class derived from DtObjectPublisher.

virtual void setObjectClassChooser(const char* vrlinkClassName, DtObjectClassChooser chooser)

virtual char* chooseObjectClass(const char* vrlinkClassName, void* usr = NULL) const

Chooses an object class to represent an object managed by the C++ class named vrlinkClassName (which should

be a class derived from DtObjectPublisher). usr is additional user data that may be required to make the choice. It is passed as usr to a chooser function that has been registered with the FOM Mapper for this publisher type, if any.

virtual void setInteractionClassToChoose(const char* vrlinkClassName, const char* fomClassName)

Tell the FOM Mapper how to choose a FOM interaction class to use to for a particular kind of derived DtInteraction. You can either associate a single FOM class name, or a FOM class choosing function with the name of a C++ class derived from DtInteraction.

virtual void setInteractionClassChooser(const char* vrlinkClassName, DtInteractionClassChooser chooser)

virtual char* chooseInteractionClass(const DtInteraction& inter) const

Chooses a FOM interaction class to be by a particular derived DtInteraction class named vrlinkClassName

virtual void setInteractionClasses(const char* vrlinkClassName, const DtList& classes)

Set/Get a list of names (char*) of FOM classes that can be decoded into the derived DtInteraction class whose name is name. These classes will be subscribed to, when you use a derived DtInteraction's addCallback function. The DtFomMapper copies the data in the list passed to setInteractionClasses, so the object can be deleted at the caller's leisure. The list returned by interactionClasses is owned by the DtFomMapper and shouldn't be deleted by the caller.

virtual DtList* interactionClasses(const char* vrlinkClassName)

virtual void setInteractionClass(const char* vrlinkClassName, const char* fomClassName)

When there is only a single FOM class to be associated with a VR-Link interaction class, then there is no need to create a DtList. Just use this:

virtual void setObjectClassNames(const char* vrlinkClassName, const DtList& classes)

Set/Get a list of names (char*) of FOM object classes that should be subscribed to by a particular subclass of VR-Link's DtReflectedObjectList. vrlinkClassName is the name of the DtReflectedObjectList subclass for which you are creating a mapping. The DtFomMapper copies the data in the list passed to setObjectClassNames, so the object can be deleted at the caller's leisure. The list returned by objectClassNames is owned by the DtFomMapper and shouldn't be deleted by the caller.

virtual DtList* objectClassNames(const char* vrlinkClassName)

virtual DtList* objectClassHandles(const char* vrlinkClassName)

Similar to objectClassNames, but returns a DtList of ObjectClassHandles instead. The returned list is owned by the DtFomMapper, and should not be deleted. In addition, it is only valid until the next call to this function.

virtual void setObjectClass(const char* vrlinkClassName, const char* fomClassName)

When there is only a single FOM class to be associated with a VR-Link DtReflectedObjectList class, then there is no need to create a DtList. Just use this:

virtual void init(DtExerciseConn* conn)

Initialize the instance, once an exercise connection is available. Must be overridden in subclasses.

virtual void setVersion(double vers)

Returns the version number of the FOM, as passed to setVersion

virtual double version() const

DtFomMapper(const DtFomMapper& orig)

copy constructor - not implemented

DtFomMapper& operator=(const DtFomMapper& orig)

assignment operator - not implemented

virtual DtObjectClassChooser objectClassChooser(const char* vrlinkClassName) const

Return the FOM-class-chooser function, if any, that has been associated with the name of a C++ class derived from DtObjectPublisher using setObjectClassChooser

virtual DtInteractionClassChooser interactionClassChooser(const char* vrlinkClassName) const

Return the FOM-class-chooser function, if any, that has been associated with the name of a C++ class derived from DtInteraction using setInteractionClassChooser

DtExerciseConn* myExConn

DtStateDecoderFactory* myDecFact

Current factories and other objects

DtStateEncoderFactory* myEncFact

DtInteractionDecoderFactory* myInterDecFact

DtInteractionEncoderFactory* myInterEncFact

DtHlaObjectFactory* myObjFact

DtInteractionFactory* myInterFact

DtHashlist myIntMapList

Hashlist that maps names of C++ classes derived from DtInteraction to lists of names of FOM classes. Hashkeys are DtStringHashKeys. Data are DtLists of char*'s that indicate FOM class names.

DtHashlist myObjMapList

Hashlist that maps names of C++ classes derived from DtReflectedObjectList to lists of names of FOM classes. Hashkeys are DtStringHashKeys. Data are DtLists of char*'s that indicate FOM class names.

DtHashlist myObjChooserList

Hashlist that maps names of C++ classes derived from DtObjectPublisher to DtObjectClassChoosers - functions that can choose an object class name from the FOM to use for the object

DtHashlist mySimpleObjChooserList

Hashlist that maps names of C++ classes derived from DtObjectPublisher to single FOM object class names

DtHashlist myInterChooserList

Hashlist that maps names of C++ classes derived from DtInteraction to DtInteractionClassChoosers - functions that can choose an interaction class name from the FOM to use for the interaction

DtHashlist mySimpleInterChooserList

Hashlist that maps names of C++ classes derived from DtInteraction to single FOM object class names

DtList myObjHandleList

The object returned by objectClassHandles. It is reused and overwritten each call.

DtList myEmptyList

Just an empty list to return when no class list has been registered for a particular VR-Link class

double myVersion

Direct child classes:

DtEmptyFomMapper

Alphabetic index *Hierarchy of classes*

In file hinteraction.h:

class DtInteraction

class DtInteraction: DtInteraction is an abstract base class for representing HLA interactions (or events) (as opposed to HLA objects)

Inheritance:

DtInteraction

Public Methods

	DtInteraction () *default constructor*
virtual	**~DtInteraction** () *destructor*
	DtInteraction (const DtInteraction& orig) *copy constructor*
DtInteraction&	**operator=** (const DtInteraction& orig) *assignment operator*
virtual const RTI::ParameterHandleValuePairSet&	**phvps** () const *Get the parameter handle value pair set for the interaction*
virtual void	**setFromPhvps** (const RTI::ParameterHandleValuePairSet& pvList) *Set contents from a parameter handle value pair set*
virtual void	**setExConn** (DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL) *Sets the interaction's ExerciseConn and class descriptor*
virtual RTI::InteractionClassHandle	**interactionClassHandle** () const *Get the interaction class handle*
virtual char*	**interactionClassName** () const *Get the interaction class name*
virtual int	**numParameters** () const *Get the number of parameters*
virtual DtBoolean	**neededByFederation** () const *Returns DtTrue if the interaction class is needed by the federation DtFalse otherwise*
virtual void	**print** () const *Print the interaction's host rep data, including header*
virtual void	**printHeader** () const *Print the header info*
virtual void	**printData** () const *Print the interaction's host rep data*

virtual void	**printParams** (DtBoolean withHex) const *Print the header followed by the interaction's parameters*
virtual const char*	**name** () const *Name of the interaction (Not nec*
virtual DtExerciseConn*	**exerciseConn** () const *Get the current exercise connection*
virtual DtInterClassDesc*	**classDesc** () const *Get the current class descriptor we're using for this interaction*
virtual void	**setTag** (const DtString& tag) *Set/Get the user-defined tag*
virtual const DtString&	**tag** () const
virtual void	**setTimeStamp** (DtTime time, DtTimeStampType type = DtTimeStampRelative) *Set the time of validity of the message, and whether this represents relative or absolute time*
virtual DtTime	**timeStamp** () const *Return the time of validity of the message (in seconds past an hour) as set by setTimeStamp*
virtual DtTimeStampType	**timeStampType** () const *Return the type of the time stamp*
virtual DtTime	**guessTimeValid** (DtTime referenceTime) const *Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past that hour*
static void	**setTagIsAsciiTime** (DtBoolean val) *Set/Get flag that tells us whether we're using the RPR FOM convention that the UserDefinedTag in updates and interactions represents a DIS-style timestamp*
static DtBoolean	**tagIsAsciiTime** ()
static void	**addCallback** (DtExerciseConn* conn, DtList* classes, DtReceiveInteractionCb cb, void* usr) *Register/Deregister callback function cb with conn, on all interaction class name (char*) contained in classes*
static void	**removeCallback** (DtExerciseConn* conn, DtList* classes, DtReceiveInteractionCb cb, void* usr)
static void	**addCallback** (DtExerciseConn* conn, const char* className, DtReceiveInteractionCb cb, void* usr) *Can be used instead of above for a single class*
static void	**removeCallback** (DtExerciseConn* conn, const char* className, DtReceiveInteractionCb cb, void* usr)
static int	**maxInteractionClasses** () *Returns maximum supported number of interaction classes*
inline DtInterClassDesc*	**classDesc** () const
inline DtExerciseConn*	**exerciseConn** () const
inline RTI::InteractionClassHandle	**interactionClassHandle** () const
inline char*	**interactionClassName** () const
inline int	**maxInteractionClasses** ()
inline int	**numParameters** () const

- inline void **printData** () const
- inline DtInteraction* **self** () const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse** (DtExerciseConn* exConn) const
  Returns the FOM interaction class name to use
- virtual RTI::ParameterHandleValuePairSet* **privPvList** () const
  Returns a pointer to a private pvList that can be used as a return value by phvps
- DtInteraction* **self** () const
  Get rid of constness on the "this" pointer in order to access non-const functions from const member functions (such as simple accessors)

Documentation

class DtInteraction:

DtInteraction is an abstract base class for representing HLA interactions (or events) (as opposed to HLA objects).

- **`DtInteraction()`**
  default constructor
- **`virtual ~DtInteraction()`**
  destructor
- **`DtInteraction(const DtInteraction& orig)`**
  copy constructor
- **`DtInteraction& operator=(const DtInteraction& orig)`**
  assignment operator

- **virtual const RTI::ParameterHandleValuePairSet& phvps() const**

  Get the parameter handle value pair set for the interaction

- **virtual void setFromPhvps(const RTI::ParameterHandleValuePairSet& pvList)**

  Set contents from a parameter handle value pair set

- **virtual void setExConn(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)**

  Sets the interaction's ExerciseConn and class descriptor. If classDesc is NULL, derived class' versions choose a reasonable default. Without being told what ExConn or class descriptor to use, many of the HLA-specific functions won't work properly. However, it is usually not necessary for application code to call this function, since it is called by DtExerciseConn::send on the outgoing side, and by the DtInteractionFactory on the incoming side.

- **virtual RTI::InteractionClassHandle interactionClassHandle() const**

  Get the interaction class handle. Returns 0 if no classDesc.

- **virtual char* interactionClassName() const**

  Get the interaction class name. Returns NULL if no classDesc.

- **virtual int numParameters() const**

  Get the number of parameters. Returns 0 if no classDesc.

- **virtual DtBoolean neededByFederation() const**

  Returns DtTrue if the interaction class is needed by the federation DtFalse otherwise. Returns DtFalse if no classDesc.

- **virtual void print() const**

  Print the interaction's host rep data, including header

- **virtual void printHeader() const**

  Print the header info

- **virtual void printData() const**

  Print the interaction's host rep data

- **virtual void printParams(DtBoolean withHex) const**

  Print the header followed by the interaction's parameters

- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).

- **virtual DtExerciseConn* exerciseConn() const**

  Get the current exercise connection

- **virtual DtInterClassDesc* classDesc() const**

  Get the current class descriptor we're using for this interaction

- **virtual void setTag(const DtString& tag)**

  Set/Get the user-defined tag

- **virtual const DtString& tag() const**

- **virtual void setTimeStamp(DtTime time, DtTimeStampType type = DtTimeStampRelative)**

  Set the time of validity of the message, and whether this represents relative or absolute time. Time is in seconds,

but only time modulo an hour is stored.

- **virtual DtTime timeStamp() const**

  Return the time of validity of the message (in seconds past an hour) as set by setTimeStamp

- **virtual DtTimeStampType timeStampType() const**

  Return the type of the time stamp

- **virtual DtTime guessTimeValid(DtTime referenceTime) const**

  Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past that hour

- **static void setTagIsAsciiTime(DtBoolean val)**

  Set/Get flag that tells us whether we're using the RPR FOM convention that the UserDefinedTag in updates and interactions represents a DIS-style timestamp

- **static DtBoolean tagIsAsciiTime()**

- **static void addCallback(DtExerciseConn* conn, DtList* classes, DtReceiveInteractionCb cb, void* usr)**

  Register/Deregister callback function cb with conn, on all interaction class name (char*) contained in classes

- **static void removeCallback(DtExerciseConn* conn, DtList* classes, DtReceiveInteractionCb cb, void* usr)**

- **static void addCallback(DtExerciseConn* conn, const char* className, DtReceiveInteractionCb cb, void* usr)**

  Can be used instead of above for a single class

- **static void removeCallback(DtExerciseConn* conn, const char* className, DtReceiveInteractionCb cb, void* usr)**

- **static int maxInteractionClasses()**

  Returns maximum supported number of interaction classes

- **virtual char* interactionClassToUse(DtExerciseConn* exConn) const**

  Returns the FOM interaction class name to use. Default implementation is to ask the fomMapper.

- **virtual RTI::ParameterHandleValuePairSet* privPvList() const**

  Returns a pointer to a private pvList that can be used as a return value by phvps. It's guaranteed to be able to fit at least numParameters() parameters.

- **DtInteraction* self() const**

  Get rid of constness on the "this" pointer in order to access non-const functions from const member functions (such as simple accessors). This should be used sparingly since it bypasses the const protection. Note: This is essentially like an operator and is non-virtual since can't override based on return type. Allows derived classes to also define this member.

- **DtString myTag**

- **DtInterClassDesc* myClassDesc**

- **DtExerciseConn* myExConn**

- **DtTime myTime**

- **DtTimeStampType myTimeType**
- **static DtBoolean theTagIsAsciiTime**
- **static RTI::ParameterHandleValuePairSet* thePvList**
- **static int thePvListMaxSize**
- **static const int theMaxInteractionClasses**
- **inline DtInterClassDesc* classDesc() const**
- **inline DtExerciseConn* exerciseConn() const**
- **inline RTI::InteractionClassHandle interactionClassHandle() const**
- **inline char* interactionClassName() const**
- **inline int maxInteractionClasses()**
- **inline int numParameters() const**
- **inline void printData() const**
- **inline DtInteraction* self() const**

Direct child classes:

DtUnknownInteraction
DtInteractionWithEncDec

Alphabetic index *Hierarchy of classes*

In file vrlrtiamb.h:

class DtVrlRtiAmbassador

class DtVrlRtiAmbassador: DtVrlRtiAmbassador is a purely virtual class, designed to provide a foundation for different ways to send RTI messages

---

Inheritance:

DtVrlRtiAmbassador

Public Methods

	DtVrlRtiAmbassador () *default constructor*
virtual	**~DtVrlRtiAmbassador** () *destructor*
	DtVrlRtiAmbassador (const DtVrlRtiAmbassador& orig) *copy constructor*
DtVrlRtiAmbassador&	**operator=** (const DtVrlRtiAmbassador& orig) *assignment operator*
virtual RTI::RTIambassador*	**rtiAmb** () *Get the real RTI ambassador, if the implementation of the derived object permits*
virtual DtBoolean	**setReflecting** (DtBoolean onOff) *Set/Get whether to reflect back local objects and interactions, if supported by the derived class*
virtual DtBoolean	**reflecting** () const
virtual void	**createFederationExecution** (const char *executionName, const char *FED)
virtual void	**destroyFederationExecution** (const char *executionName)
virtual RTI::FederateHandle	**joinFederationExecution** (const char* yourName, const char* executionName, RTI::FederateAmbassadorPtr federateAmbassadorReference)
virtual void	**resignFederationExecution** (RTI::ResignAction theAction)
virtual void	**registerFederationSynchronizationPoint** (const char* label, const char* theTag)
virtual void	**registerFederationSynchronizationPoint** (const char* label, const char* theTag, const RTI::FederateHandleSet& syncSet)
virtual void	**synchronizationPointAchieved** (const char* label)
virtual void	**requestFederationSave** (const char* label, const RTI::FedTime& theTime)
virtual void	**requestFederationSave** (const char *label)
virtual void	**federateSaveBegun** ()

virtual void	**federateSaveComplete** ()
virtual void	**federateSaveNotComplete** ()
virtual void	**requestFederationRestore** (const char* label)
virtual void	**federateRestoreComplete** ()
virtual void	**federateRestoreNotComplete** ()
virtual void	**publishObjectClass** (RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& attributeList)
virtual void	**unpublishObjectClass** (RTI::ObjectClassHandle theClass)
virtual void	**publishInteractionClass** (RTI::InteractionClassHandle theInteraction)
virtual void	**unpublishInteractionClass** (RTI::InteractionClassHandle theInteraction)
virtual void	**subscribeObjectClassAttributes** (RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& attributeList, RTI::Boolean active = RTI::RTI_TRUE)
virtual void	**unsubscribeObjectClass** (RTI::ObjectClassHandle theClass)
virtual void	**subscribeInteractionClass** (RTI::InteractionClassHandle theClass, RTI::Boolean active = RTI::RTI_TRUE)
virtual void	**unsubscribeInteractionClass** (RTI::InteractionClassHandle theClass)
virtual RTI::ObjectHandle	**registerObjectInstance** (RTI::ObjectClassHandle theClass, const char* theObject)
virtual RTI::ObjectHandle	**registerObjectInstance** (RTI::ObjectClassHandle theClass)
virtual RTI::EventRetractionHandle	**updateAttributeValues** (RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const RTI::FedTime& theTime, const char* theTag)
virtual void	**updateAttributeValues** (RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const char* theTag)
virtual RTI::EventRetractionHandle	**sendInteraction** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const RTI::FedTime& theTime, const char* theTag)
virtual void	**sendInteraction** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const char* theTag)
virtual RTI::EventRetractionHandle	**deleteObjectInstance** (RTI::ObjectHandle theObject, const RTI::FedTime& theTime, const char* theTag)
virtual void	**deleteObjectInstance** (RTI::ObjectHandle theObject, const char* theTag)
virtual void	**localDeleteObjectInstance** (RTI::ObjectHandle theObject)
virtual void	**changeAttributeTransportationType** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, RTI::TransportationHandle theType)
virtual void	**changeInteractionTransportationType** (RTI::InteractionClassHandle theClass, RTI::TransportationHandle theType)
virtual void	**requestObjectAttributeValueUpdate** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)
virtual void	**requestClassAttributeValueUpdate** (RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& theAttributes)
virtual void	**unconditionalAttributeOwnershipDivestiture** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)
virtual void	**negotiatedAttributeOwnershipDivestiture** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, const char* theTag)
virtual void	**attributeOwnershipAcquisition** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& desiredAttributes, const char* theTag)

virtual void	**attributeOwnershipAcquisitionIfAvailable** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& desiredAttributes)
virtual RTI::AttributeHandleSet*	**attributeOwnershipReleaseResponse** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)
virtual void	**cancelNegotiatedAttributeOwnershipDivestiture** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)
virtual void	**cancelAttributeOwnershipAcquisition** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)
virtual void	**queryAttributeOwnership** (RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute)
virtual RTI::Boolean	**isAttributeOwnedByFederate** (RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute)
virtual void	**enableTimeRegulation** (const RTI::FedTime& theFederateTime, const RTI::FedTime& theLookahead)
virtual void	**disableTimeRegulation** ()
virtual void	**enableTimeConstrained** ()
virtual void	**disableTimeConstrained** ()
virtual void	**timeAdvanceRequest** (const RTI::FedTime& theTime)
virtual void	**timeAdvanceRequestAvailable** (const RTI::FedTime& theTime)
virtual void	**nextEventRequest** (const RTI::FedTime& theTime)
virtual void	**nextEventRequestAvailable** (const RTI::FedTime& theTime)
virtual void	**flushQueueRequest** (const RTI::FedTime& theTime)
virtual void	**enableAsynchronousDelivery** ()
virtual void	**disableAsynchronousDelivery** ()
virtual void	**queryLBTS** (RTI::FedTime& theTime)
virtual void	**queryFederateTime** (RTI::FedTime& theTime)
virtual void	**queryMinNextEventTime** (RTI::FedTime& theTime)
virtual void	**modifyLookahead** (const RTI::FedTime& theLookahead)
virtual void	**queryLookahead** (RTI::FedTime& theTime)
virtual void	**retract** (RTI::EventRetractionHandle theHandle)
virtual void	**changeAttributeOrderType** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, RTI::OrderingHandle theType)
virtual void	**changeInteractionOrderType** (RTI::InteractionClassHandle theClass, RTI::OrderingHandle theType)
virtual void	**createRegion** (RTI::SpaceHandle theSpace, RTI::ULong numberOfExtents)
virtual void	**notifyAboutRegionModification** (RTI::Region &theRegion)
virtual void	**deleteRegion** (RTI::Region *theRegion)
virtual RTI::ObjectHandle	**registerObjectInstanceWithRegion** (RTI::ObjectClassHandle theClass, const char* theObject, RTI::AttributeHandle theAttributes[], RTI::Region* theRegions[], RTI::ULong theNumberOfHandles)
virtual RTI::ObjectHandle	**registerObjectInstanceWithRegion** (RTI::ObjectClassHandle theClass, RTI::AttributeHandle theAttributes[], RTI::Region* theRegions[], RTI::ULong theNumberOfHandles)

- virtual void **associateRegionForUpdates** (RTI::Region& theRegion, RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)
- virtual void **unassociateRegionForUpdates** (RTI::Region& theRegion, RTI::ObjectHandle theObject)
- virtual void **subscribeObjectClassAttributesWithRegion** (RTI::ObjectClassHandle theClass, RTI::Region& theRegion, const RTI::AttributeHandleSet &attributeList, RTI::Boolean active = RTI::RTI_TRUE)
- virtual void **unsubscribeObjectClassWithRegion** (RTI::ObjectClassHandle theClass, RTI::Region& theRegion)
- virtual void **subscribeInteractionClassWithRegion** (RTI::InteractionClassHandle theClass, RTI::Region& theRegion, RTI::Boolean active = RTI::RTI_TRUE)
- virtual void **unsubscribeInteractionClassWithRegion** (RTI::InteractionClassHandle theClass, RTI::Region& theRegion)
- virtual RTI::EventRetractionHandle **sendInteractionWithRegion** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet &theParameters, const RTI::FedTime& theTime, const char* theTag, const RTI::Region& theRegion)
- virtual void **sendInteractionWithRegion** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet &theParameters, const char* theTag, const RTI::Region& theRegion)
- virtual void **requestClassAttributeValueUpdateWithRegion** (RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& theAttributes, const RTI::Region& theRegion)
- virtual RTI::ObjectClassHandle **getObjectClassHandle** (const char* theName)
- virtual char* **getObjectClassName** (RTI::ObjectClassHandle theHandle)
- virtual RTI::AttributeHandle **getAttributeHandle** (const char* theName, RTI::ObjectClassHandle whichClass)
- virtual char* **getAttributeName** (RTI::AttributeHandle theHandle, RTI::ObjectClassHandle whichClass)
- virtual RTI::InteractionClassHandle **getInteractionClassHandle** (const char* theName)
- virtual char* **getInteractionClassName** (RTI::InteractionClassHandle theHandle)
- virtual RTI::ParameterHandle **getParameterHandle** (const char* theName, RTI::InteractionClassHandle whichClass)
- virtual char* **getParameterName** (RTI::ParameterHandle theHandle, RTI::InteractionClassHandle whichClass)
- virtual RTI::ObjectHandle **getObjectInstanceHandle** (const char* theName)
- virtual char* **getObjectInstanceName** (RTI::ObjectHandle theHandle)
- virtual RTI::SpaceHandle **getRoutingSpaceHandle** (const char* theName)
- virtual const char* **getRoutingSpaceName** (const RTI::SpaceHandle theHandle)
- virtual RTI::DimensionHandle **getDimensionHandle** (const char* theName, RTI::SpaceHandle whichSpace)
- virtual char* **getDimensionName** (RTI::DimensionHandle theHandle, RTI::SpaceHandle whichSpace)
- virtual RTI::SpaceHandle **getAttributeRoutingSpaceHandle** (RTI::AttributeHandle theHandle, RTI::ObjectClassHandle whichClass)
- virtual RTI::ObjectClassHandle **getObjectClass** (RTI::ObjectHandle theObject)
- virtual RTI::SpaceHandle **getInteractionRoutingSpaceHandle** (RTI::InteractionClassHandle theHandle)
- virtual RTI::TransportationHandle **getTransportationHandle** (const char *theName)
- virtual char* **getTransportationName** (RTI::TransportationHandle theHandle)

- virtual RTI::OrderingHandle **getOrderingHandle** (const char *theName)
- virtual char* **getOrderingName** (RTI::OrderingHandle theHandle)
- virtual void **enableClassRelevanceAdvisorySwitch** ()
- virtual void **disableClassRelevanceAdvisorySwitch** ()
- virtual void **enableAttributeRelevanceAdvisorySwitch** ()
- virtual void **disableAttributeRelevanceAdvisorySwitch** ()
- virtual void **enableAttributeScopeAdvisorySwitch** ()
- virtual void **disableAttributeScopeAdvisorySwitch** ()
- virtual void **enableInteractionRelevanceAdvisorySwitch** ()
- virtual void **disableInteractionRelevanceAdvisorySwitch** ()
- virtual RTI::Boolean **tick** ()
- virtual RTI::Boolean **tick** (RTI::TickTime minimum, RTI::TickTime maximum)
- virtual RTI::RegionToken **getRegionToken** (RTI::Region*)
- virtual RTI::Region* **getRegion** (RTI::RegionToken)

Documentation

class DtVrlRtiAmbassador: DtVrlRtiAmbassador is a purely virtual class, designed to provide a foundation for different ways to send RTI messages

- **`DtVrlRtiAmbassador()`**

  default constructor

- **`virtual ~DtVrlRtiAmbassador()`**

  destructor

- **`DtVrlRtiAmbassador(const DtVrlRtiAmbassador& orig)`**

  copy constructor

- **`DtVrlRtiAmbassador& operator=(const DtVrlRtiAmbassador& orig)`**

  assignment operator

- **`virtual RTI::RTIambassador* rtiAmb()`**

  Get the real RTI ambassador, if the implementation of the derived object permits. Otherwise returns NULL.

- **`virtual DtBoolean setReflecting(DtBoolean onOff)`**

  Set/Get whether to reflect back local objects and interactions, if supported by the derived class. The set call's return value indicates the value of the flag as a result of the call.

- **`virtual DtBoolean reflecting() const`**

- **`virtual void createFederationExecution( const char *executionName, const char *FED)`**

- **`virtual void destroyFederationExecution( const char *executionName)`**

- virtual RTI::FederateHandle joinFederationExecution(const char* yourName, const char* executionName, RTI::FederateAmbassadorPtr federateAmbassadorReference)
- virtual void resignFederationExecution(RTI::ResignAction theAction)
- virtual void registerFederationSynchronizationPoint(const char* label, const char* theTag)
- virtual void registerFederationSynchronizationPoint(const char* label, const char* theTag, const RTI::FederateHandleSet& syncSet)
- virtual void synchronizationPointAchieved(const char* label)
- virtual void requestFederationSave(const char* label, const RTI::FedTime& theTime)
- virtual void requestFederationSave(const char *label)
- virtual void federateSaveBegun()
- virtual void federateSaveComplete()
- virtual void federateSaveNotComplete()
- virtual void requestFederationRestore(const char* label)
- virtual void federateRestoreComplete()
- virtual void federateRestoreNotComplete()
- virtual void publishObjectClass(RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& attributeList)
- virtual void unpublishObjectClass(RTI::ObjectClassHandle theClass)
- virtual void publishInteractionClass(RTI::InteractionClassHandle theInteraction)
- virtual void unpublishInteractionClass(RTI::InteractionClassHandle theInteraction)
- virtual void subscribeObjectClassAttributes(RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& attributeList, RTI::Boolean active = RTI::RTI_TRUE)
- virtual void unsubscribeObjectClass(RTI::ObjectClassHandle theClass)
- virtual void subscribeInteractionClass(RTI::InteractionClassHandle theClass, RTI::Boolean active = RTI::RTI_TRUE)
- virtual void unsubscribeInteractionClass(RTI::InteractionClassHandle theClass)
- virtual RTI::ObjectHandle registerObjectInstance(RTI::ObjectClassHandle theClass, const char* theObject)
- virtual RTI::ObjectHandle registerObjectInstance(RTI::ObjectClassHandle theClass)
- virtual RTI::EventRetractionHandle updateAttributeValues(RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const RTI::FedTime& theTime, const char* theTag)
- virtual void updateAttributeValues(RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const char* theTag)
- virtual RTI::EventRetractionHandle sendInteraction(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const RTI::FedTime& theTime, const char* theTag)

- **virtual void sendInteraction(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const char* theTag)**
- **virtual RTI::EventRetractionHandle deleteObjectInstance(RTI::ObjectHandle theObject, const RTI::FedTime& theTime, const char* theTag)**
- **virtual void deleteObjectInstance(RTI::ObjectHandle theObject, const char* theTag)**
- **virtual void localDeleteObjectInstance(RTI::ObjectHandle theObject)**
- **virtual void changeAttributeTransportationType(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, RTI::TransportationHandle theType)**
- **virtual void changeInteractionTransportationType(RTI::InteractionClassHandle theClass, RTI::TransportationHandle theType)**
- **virtual void requestObjectAttributeValueUpdate(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)**
- **virtual void requestClassAttributeValueUpdate(RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& theAttributes)**
- **virtual void unconditionalAttributeOwnershipDivestiture(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)**
- **virtual void negotiatedAttributeOwnershipDivestiture(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, const char* theTag)**
- **virtual void attributeOwnershipAcquisition(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& desiredAttributes, const char* theTag)**
- **virtual void attributeOwnershipAcquisitionIfAvailable(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& desiredAttributes)**
- **virtual RTI::AttributeHandleSet* attributeOwnershipReleaseResponse(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)**
- **virtual void cancelNegotiatedAttributeOwnershipDivestiture(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)**
- **virtual void cancelAttributeOwnershipAcquisition(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)**
- **virtual void queryAttributeOwnership(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute)**
- **virtual RTI::Boolean isAttributeOwnedByFederate(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute)**
- **virtual void enableTimeRegulation(const RTI::FedTime& theFederateTime, const RTI::FedTime& theLookahead)**
- **virtual void disableTimeRegulation()**
- **virtual void enableTimeConstrained()**
- **virtual void disableTimeConstrained()**
- **virtual void timeAdvanceRequest(const RTI::FedTime& theTime)**
- **virtual void timeAdvanceRequestAvailable(const RTI::FedTime& theTime)**

- virtual void nextEventRequest(const RTI::FedTime& theTime)
- virtual void nextEventRequestAvailable(const RTI::FedTime& theTime)
- virtual void flushQueueRequest(const RTI::FedTime& theTime)
- virtual void enableAsynchronousDelivery()
- virtual void disableAsynchronousDelivery()
- virtual void queryLBTS(RTI::FedTime& theTime)
- virtual void queryFederateTime(RTI::FedTime& theTime)
- virtual void queryMinNextEventTime(RTI::FedTime& theTime)
- virtual void modifyLookahead(const RTI::FedTime& theLookahead)
- virtual void queryLookahead(RTI::FedTime& theTime)
- virtual void retract(RTI::EventRetractionHandle theHandle)
- virtual void changeAttributeOrderType(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, RTI::OrderingHandle theType)
- virtual void changeInteractionOrderType(RTI::InteractionClassHandle theClass, RTI::OrderingHandle theType)
- virtual void createRegion(RTI::SpaceHandle theSpace, RTI::ULong numberOfExtents)
- virtual void notifyAboutRegionModification(RTI::Region &theRegion)
- virtual void deleteRegion(RTI::Region *theRegion)
- virtual RTI::ObjectHandle registerObjectInstanceWithRegion(RTI::ObjectClassHandle theClass, const char* theObject, RTI::AttributeHandle theAttributes[], RTI::Region* theRegions[], RTI::ULong theNumberOfHandles)
- virtual RTI::ObjectHandle registerObjectInstanceWithRegion(RTI::ObjectClassHandle theClass, RTI::AttributeHandle theAttributes[], RTI::Region* theRegions[], RTI::ULong theNumberOfHandles)
- virtual void associateRegionForUpdates(RTI::Region& theRegion, RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)
- virtual void unassociateRegionForUpdates(RTI::Region& theRegion, RTI::ObjectHandle theObject)
- virtual void subscribeObjectClassAttributesWithRegion(RTI::ObjectClassHandle theClass, RTI::Region& theRegion, const RTI::AttributeHandleSet &attributeList, RTI::Boolean active = RTI::RTI_TRUE)
- virtual void unsubscribeObjectClassWithRegion(RTI::ObjectClassHandle theClass, RTI::Region& theRegion)
- virtual void subscribeInteractionClassWithRegion(RTI::InteractionClassHandle theClass, RTI::Region& theRegion, RTI::Boolean active = RTI::RTI_TRUE)
- virtual void unsubscribeInteractionClassWithRegion(RTI::InteractionClassHandle theClass, RTI::Region& theRegion)
- virtual RTI::EventRetractionHandle sendInteractionWithRegion(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet

&theParameters, const RTI::FedTime& theTime, const char* theTag, const RTI::Region& theRegion)

- virtual void sendInteractionWithRegion(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet &theParameters, const char* theTag, const RTI::Region& theRegion)
- virtual void requestClassAttributeValueUpdateWithRegion(RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& theAttributes, const RTI::Region& theRegion)
- virtual RTI::ObjectClassHandle getObjectClassHandle(const char* theName)
- virtual char* getObjectClassName(RTI::ObjectClassHandle theHandle)
- virtual RTI::AttributeHandle getAttributeHandle(const char* theName, RTI::ObjectClassHandle whichClass)
- virtual char* getAttributeName(RTI::AttributeHandle theHandle, RTI::ObjectClassHandle whichClass)
- virtual RTI::InteractionClassHandle getInteractionClassHandle(const char* theName)
- virtual char* getInteractionClassName(RTI::InteractionClassHandle theHandle)
- virtual RTI::ParameterHandle getParameterHandle(const char* theName, RTI::InteractionClassHandle whichClass)
- virtual char* getParameterName(RTI::ParameterHandle theHandle, RTI::InteractionClassHandle whichClass)
- virtual RTI::ObjectHandle getObjectInstanceHandle(const char* theName)
- virtual char* getObjectInstanceName(RTI::ObjectHandle theHandle)
- virtual RTI::SpaceHandle getRoutingSpaceHandle(const char* theName)
- virtual const char* getRoutingSpaceName(const RTI::SpaceHandle theHandle)
- virtual RTI::DimensionHandle getDimensionHandle(const char* theName, RTI::SpaceHandle whichSpace)
- virtual char* getDimensionName(RTI::DimensionHandle theHandle, RTI::SpaceHandle whichSpace)
- virtual RTI::SpaceHandle getAttributeRoutingSpaceHandle(RTI::AttributeHandle theHandle, RTI::ObjectClassHandle whichClass)
- virtual RTI::ObjectClassHandle getObjectClass(RTI::ObjectHandle theObject)
- virtual RTI::SpaceHandle getInteractionRoutingSpaceHandle(RTI::InteractionClassHandle theHandle)
- virtual RTI::TransportationHandle getTransportationHandle(const char *theName)
- virtual char* getTransportationName(RTI::TransportationHandle theHandle)
- virtual RTI::OrderingHandle getOrderingHandle(const char *theName)
- virtual char* getOrderingName(RTI::OrderingHandle theHandle)
- virtual void enableClassRelevanceAdvisorySwitch()
- virtual void disableClassRelevanceAdvisorySwitch()

- `virtual void enableAttributeRelevanceAdvisorySwitch()`
- `virtual void disableAttributeRelevanceAdvisorySwitch()`
- `virtual void enableAttributeScopeAdvisorySwitch()`
- `virtual void disableAttributeScopeAdvisorySwitch()`
- `virtual void enableInteractionRelevanceAdvisorySwitch()`
- `virtual void disableInteractionRelevanceAdvisorySwitch()`
- `virtual RTI::Boolean tick()`
- `virtual RTI::Boolean tick( RTI::TickTime minimum, RTI::TickTime maximum)`
- `virtual RTI::RegionToken getRegionToken(RTI::Region*)`
- `virtual RTI::Region* getRegion(RTI::RegionToken)`

Direct child classes:

DtDirectRtiAmbassador

Alphabetic index *Hierarchy of classes*

In file vrlfedamb.h:

class DtVrlFederateAmbassador: public DtNullFederateAmbassador

class DtVrlFederateAmbassador: Instances of DtVrlFederateAmbassador are used to manage the interface between the RTI and the VR-Link Federate

---

Inheritance:

DtVrlFederateAmbassador - DtNullFederateAmbassador

Public Methods

	DtVrlFederateAmbassador () *default constructor*
virtual	**~DtVrlFederateAmbassador** () *destructor*
virtual void	**init** (DtReflectedHlaObjectManager* reflectedObjMgr, DtLocalHlaObjectManager* localObjMgr, DtInteractionManager* interMgr) *Initialize the ambassador*
virtual void	**addReflectAttributeValuesCb** (RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr) *Add/Remove (register/deregister) function to be called when the RTI's reflectAttributeValues service is invoked*
virtual void	**removeReflectAttributeValuesCb** (RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr)
virtual void	**addDiscoverObjectCb** (RTI::ObjectClassHandle hand, DtDiscoverObjectCb cb, void* usr) *Add/Remove (register/deregister) function to be called when the RTI's discoverObject service is invoked*
virtual void	**removeDiscoverObjectCb** (RTI::ObjectClassHandle hand, DtDiscoverObjectCb cb, void* usr)
virtual void	**addRemoveObjectCb** (RTI::ObjectHandle id, DtRemoveObjectCb, void* usr) *Add/Remove (register/deregister) function to be called when the RTI's removeObject service is invoked*
virtual void	**removeRemoveObjectCb** (RTI::ObjectHandle id, DtRemoveObjectCb, void* usr)
virtual void	**addStartRegCb** (RTI::ObjectClassHandle hand, DtStartRegCb, void* usr) *Add/Remove (register/deregister) function to be called when the RTI's startRegistration service is invoked*
virtual void	**removeStartRegCb** (RTI::ObjectClassHandle hand, DtStartRegCb, void* usr)

- virtual void — **addStopRegCb** (RTI::ObjectClassHandle hand, DtStopRegCb, void* usr)
  Add/Remove (register/deregister) function to be called when the RTI's stopReg service is invoked
- virtual void — **removeStopRegCb** (RTI::ObjectClassHandle hand, DtStopRegCb, void* usr)
- virtual void — **addUpdatesOnCb** (RTI::ObjectHandle hand, DtUpdatesOnCb, void* usr)
  Add/Remove (register/deregister) function to be called when the RTI's turnUpdatesOn service is invoked
- virtual void — **removeUpdatesOnCb** (RTI::ObjectHandle hand, DtUpdatesOnCb, void* usr)
- virtual void — **addUpdatesOffCb** (RTI::ObjectHandle hand, DtUpdatesOffCb, void* usr)
  Add/Remove (register/deregister) function to be called when the RTI's turnUpdatesOff service is invoked
- virtual void — **removeUpdatesOffCb** (RTI::ObjectHandle hand, DtUpdatesOffCb, void* usr)
- virtual void — **addStartInteractionsCb** (RTI::InteractionClassHandle hand, DtStartStopInterCb, void* usr)
  Add/Remove (register/deregister) function to be called when the RTI's startInteractionGeneration service is invoked
- virtual void — **removeStartInteractionsCb** (RTI::InteractionClassHandle hand, DtStartStopInterCb, void* usr)
- virtual void — **addStopInteractionsCb** (RTI::InteractionClassHandle hand, DtStartStopInterCb, void* usr)
  Add/Remove (register/deregister) function to be called when the RTI's stopInteractionGeneration service is invoked
- virtual void — **removeStopInteractionsCb** (RTI::InteractionClassHandle hand, DtStartStopInterCb, void* usr)
- virtual void — **addReceiveInteractionCb** (RTI::InteractionClassHandle hand, DtReceiveInteractionCb cb, void* usr)
  Add/Remove (register/deregister) function to be called when the RTI's receiveInteraction service is invoked
- virtual void — **removeReceiveInteractionCb** (RTI::InteractionClassHandle hand, DtReceiveInteractionCb cb, void* usr)
- static DtVrlFederateAmbassador* — **create** ()
  returns a new DtVrlFedAmbassador

Protected Fields

- DtInteractionManager* — **myInteractionManager**
- DtReflectedHlaObjectManager* — **myReflectedObjectManager**
- DtLocalHlaObjectManager* — **myLocalObjectManager**
- DtReflectAttributeValuesCbList — **myRavCbList**
- DtUpdatesOnCbList — **myUpdatesOnCbList**
- DtUpdatesOffCbList — **myUpdatesOffCbList**
- DtStartRegCbList — **myStartRegCbList**
- DtStopRegCbList — **myStopRegCbList**
- DtStartStopInterCbList — **myStartIntCbList**
- DtStartStopInterCbList — **myStopIntCbList**

Protected Methods

	DtVrlFederateAmbassador (const DtVrlFederateAmbassador& orig) *copy constructor - not implemented*
DtVrlFederateAmbassador&	**operator=** (const DtVrlFederateAmbassador& orig) *assignment operator - not implemented*
virtual void	**startRegistrationForObjectClass** (RTI::ObjectClassHandle theClass) throw(RTI::ObjectClassNotPublished, RTI::FederateInternalError)
virtual void	**stopRegistrationForObjectClass** (RTI::ObjectClassHandle theClass) throw(RTI::ObjectClassNotPublished, RTI::FederateInternalError)
virtual void	**turnInteractionsOn** (RTI::InteractionClassHandle theHandle) throw(RTI::InteractionClassNotPublished, RTI::FederateInternalError)
virtual void	**turnInteractionsOff** (RTI::InteractionClassHandle theHandle) throw(RTI::InteractionClassNotPublished, RTI::FederateInternalError)
virtual void	**discoverObjectInstance** (RTI::ObjectHandle theObject, RTI::ObjectClassHandle theObjectClass, const char* theObjectName) throw(RTI::CouldNotDiscover, RTI::ObjectClassNotKnown, RTI::FederateInternalError)
virtual void	**reflectAttributeValues** (RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const RTI::FedTime& theTime, const char* theTag, RTI::EventRetractionHandle theHandle) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateOwnsAttributes, RTI::InvalidFederationTime, RTI::FederateInternalError)
virtual void	**reflectAttributeValues** (RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const char* theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateOwnsAttributes, RTI::FederateInternalError)
virtual void	**receiveInteraction** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const RTI::FedTime& theTime, const char* theTag, RTI::EventRetractionHandle theHandle) throw(RTI::InteractionClassNotKnown, RTI::InteractionParameterNotKnown, RTI::InvalidFederationTime, RTI::FederateInternalError)
virtual void	**receiveInteraction** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const char* theTag) throw(RTI::InteractionClassNotKnown, RTI::InteractionParameterNotKnown, RTI::FederateInternalError)
virtual void	**removeObjectInstance** (RTI::ObjectHandle theObject, const RTI::FedTime& theTime, const char* theTag, RTI::EventRetractionHandle theHandle) throw(RTI::ObjectNotKnown, RTI::InvalidFederationTime, RTI::FederateInternalError)
virtual void	**removeObjectInstance** (RTI::ObjectHandle theObject, const char* theTag) throw(RTI::ObjectNotKnown, RTI::FederateInternalError)
virtual void	**provideAttributeValueUpdate** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)
virtual void	**turnUpdatesOnForObjectInstance** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)
virtual void	**turnUpdatesOffForObjectInstance** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)

Inherited from DtNullFederateAmbassador:

Public Methods

- * virtual void **synchronizationPointRegistrationSucceeded**(const char *label) throw(RTI::FederateInternalError)
- virtual void **synchronizationPointRegistrationFailed**(const char *label) throw(RTI::FederateInternalError)
- virtual void **announceSynchronizationPoint**(const char *label, const char *tag) throw(RTI::FederateInternalError)
- virtual void **federationSynchronized**(const char *label) throw(RTI::FederateInternalError)
- virtual void **initiateFederateSave**(const char *label) throw(RTI::UnableToPerformSave, RTI::FederateInternalError)
- virtual void **federationSaved**() throw(RTI::FederateInternalError)
- virtual void **federationNotSaved**() throw(RTI::FederateInternalError)
- virtual void **requestFederationRestoreSucceeded**(const char *label) throw(RTI::FederateInternalError)
- virtual void **requestFederationRestoreFailed**(const char *label, const char *reason) throw(RTI::FederateInternalError)
- virtual void **federationRestoreBegun**() throw(RTI::FederateInternalError)
- virtual void **initiateFederateRestore**(const char *label, RTI::FederateHandle handle) throw(RTI::SpecifiedSaveLabelDoesNotExist, RTI::CouldNotRestore, RTI::FederateInternalError)
- virtual void **federationRestored**() throw(RTI::FederateInternalError)
- virtual void **federationNotRestored**() throw(RTI::FederateInternalError)
- virtual void **attributesInScope**(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)
- virtual void **attributesOutOfScope**(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)
- * virtual void **requestAttributeOwnershipAssumption**(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& offeredAttributes, const char *theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned, RTI::AttributeNotPublished, RTI::FederateInternalError)
- virtual void **attributeOwnershipDivestitureNotification**(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& releasedAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::AttributeDivestitureWasNotRequested, RTI::FederateInternalError)
- virtual void **attributeOwnershipAcquisitionNotification**(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& securedAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAcquisitionWasNotRequested, RTI::AttributeAlreadyOwned, RTI::AttributeNotPublished, RTI::FederateInternalError)
- virtual void **attributeOwnershipUnavailable**(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned, RTI::AttributeAcquisitionWasNotRequested, RTI::FederateInternalError)
- virtual void **requestAttributeOwnershipRelease**(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& candidateAttributes, const char *theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)
- virtual void **confirmAttributeOwnershipAcquisitionCancellation**(RTI::ObjectHandle theObject, const

RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned, RTI::AttributeAcquisitionWasNotCanceled, RTI::FederateInternalError)

- virtual void **informAttributeOwnership**(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute, RTI::FederateHandle theOwner) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)

- virtual void **attributeIsNotOwned**(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)

- virtual void **attributeOwnedByRTI**(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)

- * virtual void **timeRegulationEnabled**(const RTI::FedTime& theFederateTime) throw(RTI::InvalidFederationTime, RTI::EnableTimeRegulationWasNotPending, RTI::FederateInternalError)

- virtual void **timeConstrainedEnabled**(const RTI::FedTime& theFederateTime) throw(RTI::InvalidFederationTime, RTI::EnableTimeConstrainedWasNotPending, RTI::FederateInternalError)

- virtual void **timeAdvanceGrant**(const RTI::FedTime& theTime) throw(RTI::InvalidFederationTime, RTI::TimeAdvanceWasNotInProgress, RTI::FederateInternalError)

- virtual void **requestRetraction**(RTI::EventRetractionHandle theHandle) throw(RTI::EventNotKnown, RTI::FederateInternalError)

Public

- * Federation Management **Services***
- * Declaration Management **Services***
- * Object Management **Services***
- * Ownership Management **Services***
- * Time Management **Services***

Documentation

class DtVrlFederateAmbassador: Instances of DtVrlFederateAmbassador are used to manage the interface between the RTI and the VR-Link Federate

- **`DtVrlFederateAmbassador()`**

  default constructor

- **`virtual ~DtVrlFederateAmbassador()`**

  destructor

- **`virtual void init(DtReflectedHlaObjectManager* reflectedObjMgr, DtLocalHlaObjectManager* localObjMgr, DtInteractionManager* interMgr)`**

  Initialize the ambassador. Needs to be called after construction.

- **`virtual void addReflectAttributeValuesCb(RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr)`**

  Add/Remove (register/deregister) function to be called when the RTI's reflectAttributeValues service is invoked

- **`virtual void removeReflectAttributeValuesCb(RTI::ObjectHandle id,`**

DtReflectAttributeValuesCb cb, void* usr)

- **virtual void addDiscoverObjectCb(RTI::ObjectClassHandle hand, DtDiscoverObjectCb cb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's discoverObject service is invoked

- **virtual void removeDiscoverObjectCb(RTI::ObjectClassHandle hand, DtDiscoverObjectCb cb, void* usr)**

- **virtual void addRemoveObjectCb(RTI::ObjectHandle id, DtRemoveObjectCb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's removeObject service is invoked

- **virtual void removeRemoveObjectCb(RTI::ObjectHandle id, DtRemoveObjectCb, void* usr)**

- **virtual void addStartRegCb(RTI::ObjectClassHandle hand, DtStartRegCb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's startRegistration service is invoked

- **virtual void removeStartRegCb(RTI::ObjectClassHandle hand, DtStartRegCb, void* usr)**

- **virtual void addStopRegCb(RTI::ObjectClassHandle hand, DtStopRegCb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's stopReg service is invoked

- **virtual void removeStopRegCb(RTI::ObjectClassHandle hand, DtStopRegCb, void* usr)**

- **virtual void addUpdatesOnCb(RTI::ObjectHandle hand, DtUpdatesOnCb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's turnUpdatesOn service is invoked

- **virtual void removeUpdatesOnCb(RTI::ObjectHandle hand, DtUpdatesOnCb, void* usr)**

- **virtual void addUpdatesOffCb(RTI::ObjectHandle hand, DtUpdatesOffCb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's turnUpdatesOff service is invoked

- **virtual void removeUpdatesOffCb(RTI::ObjectHandle hand, DtUpdatesOffCb, void* usr)**

- **virtual void addStartInteractionsCb(RTI::InteractionClassHandle hand, DtStartStopInterCb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's startInteractionGeneration service is invoked

- **virtual void removeStartInteractionsCb(RTI::InteractionClassHandle hand, DtStartStopInterCb, void* usr)**

- **virtual void addStopInteractionsCb(RTI::InteractionClassHandle hand, DtStartStopInterCb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's stopInteractionGeneration service is invoked

- **virtual void removeStopInteractionsCb(RTI::InteractionClassHandle hand, DtStartStopInterCb, void* usr)**

- **virtual void addReceiveInteractionCb(RTI::InteractionClassHandle hand, DtReceiveInteractionCb cb, void* usr)**

  Add/Remove (register/deregister) function to be called when the RTI's receiveInteraction service is invoked

- **virtual void removeReceiveInteractionCb(RTI::InteractionClassHandle hand, DtReceiveInteractionCb cb, void* usr)**

- **DtVrlFederateAmbassador(const DtVrlFederateAmbassador& orig)**

copy constructor - not implemented

- **DtVrlFederateAmbassador& operator=(const DtVrlFederateAmbassador& orig)**

  assignment operator - not implemented

- **virtual void startRegistrationForObjectClass(RTI::ObjectClassHandle theClass) throw(RTI::ObjectClassNotPublished, RTI::FederateInternalError)**

- **virtual void stopRegistrationForObjectClass(RTI::ObjectClassHandle theClass) throw(RTI::ObjectClassNotPublished, RTI::FederateInternalError)**

- **virtual void turnInteractionsOn(RTI::InteractionClassHandle theHandle) throw(RTI::InteractionClassNotPublished, RTI::FederateInternalError)**

- **virtual void turnInteractionsOff(RTI::InteractionClassHandle theHandle) throw(RTI::InteractionClassNotPublished, RTI::FederateInternalError)**

- **virtual void discoverObjectInstance(RTI::ObjectHandle theObject, RTI::ObjectClassHandle theObjectClass, const char* theObjectName) throw(RTI::CouldNotDiscover, RTI::ObjectClassNotKnown, RTI::FederateInternalError)**

- **virtual void reflectAttributeValues(RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const RTI::FedTime& theTime, const char* theTag, RTI::EventRetractionHandle theHandle) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateOwnsAttributes, RTI::InvalidFederationTime, RTI::FederateInternalError)**

- **virtual void reflectAttributeValues(RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const char* theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateOwnsAttributes, RTI::FederateInternalError)**

- **virtual void receiveInteraction(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const RTI::FedTime& theTime, const char* theTag, RTI::EventRetractionHandle theHandle) throw(RTI::InteractionClassNotKnown, RTI::InteractionParameterNotKnown, RTI::InvalidFederationTime, RTI::FederateInternalError)**

- **virtual void receiveInteraction(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const char* theTag) throw(RTI::InteractionClassNotKnown, RTI::InteractionParameterNotKnown, RTI::FederateInternalError)**

- **virtual void removeObjectInstance(RTI::ObjectHandle theObject, const RTI::FedTime& theTime, const char* theTag, RTI::EventRetractionHandle theHandle) throw(RTI::ObjectNotKnown, RTI::InvalidFederationTime, RTI::FederateInternalError)**

- **virtual void removeObjectInstance(RTI::ObjectHandle theObject, const char* theTag) throw(RTI::ObjectNotKnown, RTI::FederateInternalError)**

- **virtual void provideAttributeValueUpdate(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)**

- **virtual void turnUpdatesOnForObjectInstance(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)**

- **virtual void turnUpdatesOffForObjectInstance(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)**

- **static DtVrlFederateAmbassador* create()**

  returns a new DtVrlFedAmbassador

- **DtInteractionManager* myInteractionManager**
- **DtReflectedHlaObjectManager* myReflectedObjectManager**
- **DtLocalHlaObjectManager* myLocalObjectManager**
- **DtReflectAttributeValuesCbList myRavCbList**
- **DtUpdatesOnCbList myUpdatesOnCbList**
- **DtUpdatesOffCbList myUpdatesOffCbList**
- **DtStartRegCbList myStartRegCbList**
- **DtStopRegCbList myStopRegCbList**
- **DtStartStopInterCbList myStartIntCbList**
- **DtStartStopInterCbList myStopIntCbList**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file fom.h:

class DtFom: public DtFedTarget

class DtFom: An instances of DtFom is used to represent a particular FOM, as described by a FED file

---

Inheritance:

DtFom - DtFedTarget

Public Methods

	DtFom (DtExerciseConn* exConn) *Constructor*
virtual	**~DtFom** () *Destructor*
	DtFom (const DtFom& orig) *copy constructor*
DtFom&	**operator=** (const DtFom& orig) *assignment operator*
virtual void	**addInterClass** (const char* name, const char* baseName, DtTransportType transport, DtOrderType order) *Define a class*
virtual void	**addObjClass** (const char* name, const char* baseName) *Define a class*
virtual void	**addParameter** (const char* className, const char* paramName) *Define an parameter of a class*
virtual void	**addAttribute** (const char* className, const char* attrName, DtTransportType transport, DtOrderType order) *Define an attribute of a class*
virtual DtInterClassDesc*	**interClassByName** (const char *name) const *Returns the class descriptor for the name*
virtual DtObjClassDesc*	**objClassByName** (const char *name) const *Returns the class descriptor for the name*
virtual DtInterClassDesc*	**interClassByHandle** (RTI::InteractionClassHandle handle) const *Returns the class descriptor for the handle*
virtual DtObjClassDesc*	**objClassByHandle** (RTI::ObjectClassHandle handle) const *Returns the class descriptor for the handle*
virtual DtObjClassDesc*	**firstObjClass** () const *Returns the first, last and next object classes in the fom*
virtual DtObjClassDesc*	**lastObjClass** () const
virtual DtObjClassDesc*	**nextObjClass** (DtObjClassDesc* current) const

- virtual DtInterClassDesc* **firstInterClass** () const
  Returns the first, last and next interaction class in the fom
- virtual DtInterClassDesc* **lastInterClass** () const
- virtual DtInterClassDesc* **nextInterClass** (DtInterClassDesc* current) const
- virtual DtVrlRtiAmbassador* **rtiAmb** ()
  Returns our RTI ambassador
- virtual void **print** () const
  prints the contents of the FOM
- static void **setMaxClasses** (int num)
  Set/Get max number of object or interaction classes
- static int **maxClasses** ()
- inline DtInterClassDesc* **interClassByHandle** (RTI::InteractionClassHandle handle) const
- inline int **maxClasses** ()
- inline DtObjClassDesc* **objClassByHandle** (RTI::ObjectClassHandle handle) const
- inline DtVrlRtiAmbassador* **rtiAmb** ()
- inline void **setMaxClasses** (int num)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc** **myInterClasses**
- DtObjClassDesc** **myObjClasses**
- int **myLowestObj**
- int **myHighestObj**
- int **myLowestInter**
- int **myHighestInter**
- char* **myObjRootName**
- char* **myInterRootName**
- static int **theMaxClasses**

Inherited from DtFedTarget:

Documentation

class DtFom:

An instances of DtFom is used to represent a particular FOM, as described by a FED file. Object classes are represented by DtObjClassDescs; Interaction classes by DtInterClassDescs. Being a DtFedTarget, a

DtFom can be filled in by a DtFedReader. Alternatively, you can fill it in programatically, with addObjClass, addInterClass, addParameter and addAttribute. Once so initialized, you can query for a class descriptor associated with a class.

- **`DtFom(DtExerciseConn* exConn)`**

  Constructor

- **`virtual ~DtFom()`**

  Destructor

- **`DtFom(const DtFom& orig)`**

  copy constructor

- **`DtFom& operator=(const DtFom& orig)`**

  assignment operator

- **`virtual void addInterClass(const char* name, const char* baseName, DtTransportType transport, DtOrderType order)`**

  Define a class. Used when initializing from FED file.

- **`virtual void addObjClass(const char* name, const char* baseName)`**

  Define a class. Used when initializing from FED file.

- **`virtual void addParameter(const char* className, const char* paramName)`**

  Define an parameter of a class. Used when initializing from FED file.

- **`virtual void addAttribute(const char* className, const char* attrName, DtTransportType transport, DtOrderType order)`**

  Define an attribute of a class. Used when initializing from FED file.

- **`virtual DtInterClassDesc* interClassByName(const char *name) const`**

  Returns the class descriptor for the name

- **`virtual DtObjClassDesc* objClassByName(const char *name) const`**

  Returns the class descriptor for the name

- **`virtual DtInterClassDesc* interClassByHandle( RTI::InteractionClassHandle handle) const`**

  Returns the class descriptor for the handle

- **`virtual DtObjClassDesc* objClassByHandle( RTI::ObjectClassHandle handle) const`**

  Returns the class descriptor for the handle

- **`virtual DtObjClassDesc* firstObjClass() const`**

  Returns the first, last and next object classes in the fom. Next returns NULL if current is the last class, or if current is NULL. These functions facilitate looping through classes like this: for (desc = firstObjClass; desc; desc = nextObjClass(desc))

- **`virtual DtObjClassDesc* lastObjClass() const`**

- **`virtual DtObjClassDesc* nextObjClass(DtObjClassDesc* current) const`**

- **`virtual DtInterClassDesc* firstInterClass() const`**

  Returns the first, last and next interaction class in the fom. Next returns NULL if current is the last class, or if current is NULL. These functions facilitate looping through classes like this: for (desc = firstInterClass; desc; desc

= nextInterClass(desc))

- **virtual DtInterClassDesc* lastInterClass() const**
- **virtual DtInterClassDesc* nextInterClass(DtInterClassDesc* current) const**
- **virtual DtVrlRtiAmbassador* rtiAmb()**
  Returns our RTI ambassador
- **virtual void print() const**
  prints the contents of the FOM
- **static void setMaxClasses(int num)**
  Set/Get max number of object or interaction classes
- **static int maxClasses()**
- **DtExerciseConn* myExConn**
- **DtInterClassDesc** myInterClasses**
- **DtObjClassDesc** myObjClasses**
- **int myLowestObj**
- **int myHighestObj**
- **int myLowestInter**
- **int myHighestInter**
- **char* myObjRootName**
- **char* myInterRootName**
- **static int theMaxClasses**
- **inline DtInterClassDesc* interClassByHandle(RTI::InteractionClassHandle handle) const**
- **inline int maxClasses()**
- **inline DtObjClassDesc* objClassByHandle(RTI::ObjectClassHandle handle) const**
- **inline DtVrlRtiAmbassador* rtiAmb()**
- **inline void setMaxClasses(int num)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hinterfactory.h:

class DtInteractionFactory

class DtInteractionFactory: Instances of DtInteractionFactory are used to create derived DtInteractions based on Interaction Class

---

Inheritance:

DtInteractionFactory

Public Methods

typedef DtInteraction*	**(*DtInteractionCreator)** () *Interaction creation function signature*
	DtInteractionFactory (DtExerciseConn* exConn) *constructor*
virtual	**~DtInteractionFactory** () *destructor*
	DtInteractionFactory (const DtInteractionFactory& orig) *copy constructor*
DtInteractionFactory&	**operator=** (const DtInteractionFactory& orig) *assignment operator*
virtual DtInteraction*	**createInteraction** (RTI::InteractionClassHandle handle, const RTI::ParameterHandleValuePairSet& pairSet) *Create the specified interaction*
virtual DtInteractionCreator	**addCreator** (RTI::InteractionClassHandle handle, DtInteractionCreator creator) *Add an interaction creation function to the factory*
virtual DtInteractionCreator	**addCreator** (const char* className, DtInteractionCreator creator)

Protected Fields

DtInteractionCreator*	**myCreators** *myCreators is an array of DtInteraction::theMaxInteractionClasses items*
DtExerciseConn*	**myExConn**

Documentation

class DtInteractionFactory: Instances of DtInteractionFactory are used to create derived DtInteractions based on Interaction Class

typedef DtInteraction* (*DtInteractionCreator)()

Interaction creation function signature

DtInteractionFactory(DtExerciseConn* exConn)

constructor

virtual ~DtInteractionFactory()

destructor

DtInteractionFactory(const DtInteractionFactory& orig)

copy constructor

DtInteractionFactory& operator=(const DtInteractionFactory& orig)

assignment operator

virtual DtInteraction* createInteraction(RTI::InteractionClassHandle handle, const RTI::ParameterHandleValuePairSet& pairSet)

Create the specified interaction. Note: Caller is responsible for deleting. Returns NULL if interaction creation function is not in the factory.

virtual DtInteractionCreator addCreator(RTI::InteractionClassHandle handle, DtInteractionCreator creator)

Add an interaction creation function to the factory. The function must return a new'ed interaction object which will be deleted by the user. Returns any previous interaction creation function which was replaced by the add. Second version registers the creator by name, rather than by handle.

virtual DtInteractionCreator addCreator(const char* className, DtInteractionCreator creator)

DtInteractionCreator* myCreators

myCreators is an array of DtInteraction::theMaxInteractionClasses items

DtExerciseConn* myExConn

Direct child classes:

DtRprInteractionFactory

Alphabetic index *Hierarchy of classes*

In file hlaobjfact.h:

class DtHlaObjectFactory

class DtHlaObjectFactory: Instances of DtHlaObjectFactory are used to create HLA objects

---

Inheritance:

DtHlaObjectFactory

Public Methods

typedef DtHlaObject*	**(*DtHlaObjectCreator)** (DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* cDesc) *HLA object creation function signature*
typedef DtHlaObject*	**(*DtHlaObjectCreator)** (DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* cDesc) *HLA object creation function signature*
	DtHlaObjectFactory (DtExerciseConn* conn) *constructor*
virtual	**~DtHlaObjectFactory** () *destructor*
	DtHlaObjectFactory (const DtHlaObjectFactory& orig) *copy constructor*
DtHlaObjectFactory&	**operator=** (const DtHlaObjectFactory& orig) *assignment operator*
virtual DtHlaObject*	**createHlaObject** (RTI::ObjectClassHandle handle, RTI::ObjectHandle id) *Create the specified object*
virtual DtHlaObject*	**createHlaObject** (RTI::ObjectClassHandle handle, RTI::ObjectHandle id, const char* name) *Create the specified object*
virtual DtHlaObjectCreator	**addCreator** (RTI::ObjectClassHandle handle, DtHlaObjectCreator creator) *Add an object creation function to the factory*

Protected Fields

DtHlaObjectCreator*	**myCreators** *myCreators is an array of DtHlaObject::theMaxObjectClasses items*
DtExerciseConn*	**myConn**

Documentation

class DtHlaObjectFactory: Instances of DtHlaObjectFactory are used to create HLA objects

typedef DtHlaObject* (*DtHlaObjectCreator)(DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* cDesc)

HLA object creation function signature

typedef DtHlaObject* (*DtHlaObjectCreator)(DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* cDesc)

HLA object creation function signature

DtHlaObjectFactory(DtExerciseConn* conn)

constructor

virtual ~DtHlaObjectFactory()

destructor

DtHlaObjectFactory(const DtHlaObjectFactory& orig)

copy constructor

DtHlaObjectFactory& operator=(const DtHlaObjectFactory& orig)

assignment operator

virtual DtHlaObject* createHlaObject(RTI::ObjectClassHandle handle, RTI::ObjectHandle id)

Create the specified object. Note: Caller is responsible for deleting. Returns NULL if object creation function is not in the factory.

virtual DtHlaObject* createHlaObject(RTI::ObjectClassHandle handle, RTI::ObjectHandle id, const char* name)

Create the specified object. Note: Caller is responsible for deleting. Returns NULL if object creation function is not in the factory.

virtual DtHlaObjectCreator addCreator(RTI::ObjectClassHandle handle, DtHlaObjectCreator creator)

Add an object creation function to the factory. The function must return a new'ed object which will be deleted by the user. Returns any previous object creation function which was replaced by the add.

DtHlaObjectCreator* myCreators

myCreators is an array of DtHlaObject::theMaxObjectClasses items

DtExerciseConn* myConn

Direct child classes:

DtRprHlaObjectFactory

Alphabetic index *Hierarchy of classes*

In file refobjmgr.h:

class DtReflectedHlaObjectManager

class DtReflectedHlaObjectManager: Instances of DtReflectedHlaObjectManager are used to manage the creation and destruction of HLA objects from the network

---

Public Methods

	DtReflectedHlaObjectManager (DtHlaObjectFactory* factory) *constructor*
virtual	**~DtReflectedHlaObjectManager** () *destructor*
virtual void	**addDiscoverObjectCallback** (RTI::ObjectClassHandle handle, DtDiscoverObjectCb cb, void* usr) *Add/remove a callback to be called when an object which meets the federate's subscription criteria is discovered*
virtual void	**removeDiscoverObjectCallback** (RTI::ObjectClassHandle handle, DtDiscoverObjectCb cb, void* usr)
virtual void	**addRemoveObjectCallback** (RTI::ObjectHandle id, DtRemoveObjectCb cb, void* usr) *Add/remove a callback to be called when an object has been removed from the federation or no longer meets the interest criteria of the federate*
virtual void	**removeRemoveObjectCallback** (RTI::ObjectHandle id, DtRemoveObjectCb cb, void* usr)
virtual void	**processDiscoverObject** (RTI::ObjectHandle id, RTI::ObjectClassHandle handle) *Process the discovery of an object*
virtual void	**processDiscoverObject** (RTI::ObjectHandle id, RTI::ObjectClassHandle handle, const char* name) *Process the discovery of an object*
virtual void	**processRemoveObject** (RTI::ObjectHandle id) *Process the removal of an object*
virtual void	**processReflectAttributeValues** (const DtStateMsg &msg) *Process an update message*
virtual const DtList&	**allHlaObjects** () const *Get the list of all HLA objects*
virtual DtHlaObject*	**hlaObject** (RTI::ObjectHandle id) const *Get the HLA object based on its identifier*
	DtReflectedHlaObjectManager (const DtReflectedHlaObjectManager& orig) *copy constructor*
inline const DtList&	**allHlaObjects** () const

Protected Fields

DtList*	**myDiscoverCbList** *myDiscoverCbList is an array of DtHlaObject::theMaxObjectClasses items*
DtHlaObjectFactory*	**myFactory**

- DtObjectIdHashlist **myObjectList**

Protected Methods

virtual void	**objectProcessRemoveObject** (DtHlaObject* hlaObj) *Call DtHlaObject::processRemoveObject()*
virtual void	**objectProcessReflect** (DtHlaObject* hlaObj, const DtStateMsg &msg) *Call DtHlaObject::callReflect()*
DtReflectedHlaObjectManager&	**operator=** (const DtReflectedHlaObjectManager& orig) *assignment operator*

Protected

- *Note: The copy constructor and assignment operator are protected because there should only be one object manager per exercise connection*

Documentation

class DtReflectedHlaObjectManager: Instances of DtReflectedHlaObjectManager are used to manage the creation and destruction of HLA objects from the network

- **`DtReflectedHlaObjectManager(DtHlaObjectFactory* factory)`**

  constructor

- **`virtual ~DtReflectedHlaObjectManager()`**

  destructor

- **`virtual void addDiscoverObjectCallback(RTI::ObjectClassHandle handle, DtDiscoverObjectCb cb, void* usr)`**

  Add/remove a callback to be called when an object which meets the federate's subscription criteria is discovered

- **`virtual void removeDiscoverObjectCallback(RTI::ObjectClassHandle handle, DtDiscoverObjectCb cb, void* usr)`**

- **`virtual void addRemoveObjectCallback(RTI::ObjectHandle id, DtRemoveObjectCb cb, void* usr)`**

  Add/remove a callback to be called when an object has been removed from the federation or no longer meets the interest criteria of the federate

- **`virtual void removeRemoveObjectCallback(RTI::ObjectHandle id, DtRemoveObjectCb cb, void* usr)`**

- **`virtual void processDiscoverObject(RTI::ObjectHandle id, RTI::ObjectClassHandle handle)`**

  Process the discovery of an object. Creates the object, sets it up for reflecting, and calls discover callbacks.

- **`virtual void processDiscoverObject(RTI::ObjectHandle id, RTI::ObjectClassHandle handle, const char* name)`**

  Process the discovery of an object. Creates the object, sets it up for reflecting, and calls discover callbacks.

- **`virtual void processRemoveObject(RTI::ObjectHandle id)`**

Process the removal of an object. Cleans up the object from reflecting.

- **virtual void processReflectAttributeValues(const DtStateMsg &msg)**

  Process an update message

- **virtual const DtList& allHlaObjects() const**

  Get the list of all HLA objects

- **virtual DtHlaObject* hlaObject(RTI::ObjectHandle id) const**

  Get the HLA object based on its identifier

- **virtual void objectProcessRemoveObject(DtHlaObject* hlaObj)**

  Call DtHlaObject::processRemoveObject(). Provides friendship towards DtHlaObject to derived classes.

- **virtual void objectProcessReflect(DtHlaObject* hlaObj, const DtStateMsg &msg)**

  Call DtHlaObject::callReflect(). Provides friendship towards DtHlaObject to derived classes.

-

  Note: The copy constructor and assignment operator are protected because there should only be one object manager per exercise connection. These should not be called.

- **DtReflectedHlaObjectManager(const DtReflectedHlaObjectManager& orig)**

  copy constructor

- **DtReflectedHlaObjectManager& operator=(const DtReflectedHlaObjectManager& orig)**

  assignment operator

- **DtList* myDiscoverCbList**

  myDiscoverCbList is an array of DtHlaObject::theMaxObjectClasses items

- **DtHlaObjectFactory* myFactory**

- **DtObjectIdHashlist myObjectList**

- **inline const DtList& allHlaObjects() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file locobjmgr.h:

class DtLocalHlaObjectManager

class DtLocalHlaObjectManager: Instances of DtLocalHlaObjectManager are used to manage local HLA objects (objects which are being published by this application)

Public Methods

	DtLocalHlaObjectManager () *default constructor*
virtual	**~DtLocalHlaObjectManager** () *destructor*
virtual const DtList&	**allHlaObjects** () const *Get the list of all HLA objects*
virtual DtHlaObject*	**hlaObject** (RTI::ObjectHandle id) const *Get the HLA object based on its identifier*
virtual void	**processProvideUpdate** (RTI::ObjectHandle id, const RTI::AttributeHandleSet& theAttributes) *process RTI's provideUpdate service*
virtual void	**manage** (DtHlaObject* obj) *add the object to the list of local objects*
virtual void	**unManage** (DtHlaObject* obj) *remove the object from the list of local objects*
	DtLocalHlaObjectManager (const DtLocalHlaObjectManager& orig) *copy constructor*
inline const DtList&	**allHlaObjects** () const

Protected Fields

DtObjectIdHashlist	**myObjectList**

Protected Methods

DtLocalHlaObjectManager&	**operator=** (const DtLocalHlaObjectManager& orig) *assignment operator*

Protected

Note: The copy constructor and assignment operator are protected because there should only be one object manager per exercise connection

---

Documentation

class DtLocalHlaObjectManager:

Instances of DtLocalHlaObjectManager are used to manage local HLA objects (objects which are being published by this application). This manager handles providing updates to the network.

- **`DtLocalHlaObjectManager()`**

  default constructor

- **`virtual ~DtLocalHlaObjectManager()`**

  destructor

- **`virtual const DtList& allHlaObjects() const`**

  Get the list of all HLA objects

- **`virtual DtHlaObject* hlaObject(RTI::ObjectHandle id) const`**

  Get the HLA object based on its identifier

- **`virtual void processProvideUpdate(RTI::ObjectHandle id, const RTI::AttributeHandleSet& theAttributes)`**

  process RTI's provideUpdate service

- **`virtual void manage(DtHlaObject* obj)`**

  add the object to the list of local objects. This enables it to get RTI callbacks destined for it.

- **`virtual void unManage(DtHlaObject* obj)`**

  remove the object from the list of local objects. Indicates that the object is no longer being published.

-

  Note: The copy constructor and assignment operator are protected because there should only be one object manager per exercise connection. These should not be called.

- **`DtLocalHlaObjectManager(const DtLocalHlaObjectManager& orig)`**

  copy constructor

- **`DtLocalHlaObjectManager& operator=(const DtLocalHlaObjectManager& orig)`**

  assignment operator

- **`DtObjectIdHashlist myObjectList`**

- **`inline const DtList& allHlaObjects() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file intermgr.h:

class DtInteractionManager

class DtInteractionManager: Instances of DtInteractionManager are used to manage the creation and processing of HLA interactions

---

Public Methods

	DtInteractionManager (DtInteractionFactory* factory) *constructor*
virtual	**~DtInteractionManager** () *destructor*
	DtInteractionManager (const DtInteractionManager& orig) *copy constructor Care needs to be taken since this copies the callbacks*
DtInteractionManager&	**operator=** (const DtInteractionManager& orig) *assignment operator Care needs to be taken since this copies the callbacks*
virtual void	**addInteractionCallback** (RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr) *Add/remove a callback to be called when an interaction is received from another federate*
virtual void	**removeInteractionCallback** (RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr)
virtual void	**processInteraction** (RTI::InteractionClassHandle h, const RTI::ParameterHandleValuePairSet& p, const char* tag = NULL) *Create an interaction object, call the registered callbacks, then delete the interaction object*

Protected Fields

DtInteractionFactory*	**myFactory**
DtList*	**myCbList** *myCbList is an array of DtInteraction::maxInteractionClasses() items*

Protected Methods

void	**copyCbList** (const DtInteractionManager& orig) *Copy callback list*
void	**emptyCbList** () *Empty callback list*

Documentation

class DtInteractionManager: Instances of DtInteractionManager are used to manage the creation and processing of HLA interactions

- **`DtInteractionManager(DtInteractionFactory* factory)`**

  constructor

- **`virtual ~DtInteractionManager()`**

  destructor

- **`DtInteractionManager(const DtInteractionManager& orig)`**

  copy constructor Care needs to be taken since this copies the callbacks. This means the manager will have a pointer to usr data supplied to the other manager which the original owner may not be aware of or anticipate.

- **`DtInteractionManager& operator=(const DtInteractionManager& orig)`**

  assignment operator Care needs to be taken since this copies the callbacks. This means the manager will have a pointer to usr data supplied to the other manager which the original owner may not be aware of or anticipate.

- **`virtual void addInteractionCallback(RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr)`**

  Add/remove a callback to be called when an interaction is received from another federate

- **`virtual void removeInteractionCallback(RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr)`**

- **`virtual void processInteraction(RTI::InteractionClassHandle h, const RTI::ParameterHandleValuePairSet& p, const char* tag = NULL)`**

  Create an interaction object, call the registered callbacks, then delete the interaction object

- **`void copyCbList(const DtInteractionManager& orig)`**

  Copy callback list. Non-virtual since meant to be called from constructor.

- **`void emptyCbList()`**

  Empty callback list. Non-virtual since meant to be called from destructor.

- **`DtInteractionFactory* myFactory`**

- **`DtList* myCbList`**

  myCbList is an array of DtInteraction::maxInteractionClasses() items

---

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vllist.h:

class DtList

A DtList is a list of elements of arbitrary type

Inheritance:

DtList

Public Methods

	DtList () *Constructor*
virtual	**~DtList** () *Destructor - does NOT free memory of data associated with the items*
	DtList (const DtList& orig) *Copy constructor*
DtList&	**operator=** (const DtList& orig) *Assignment operator*
DtListItem*	**first** () const *Return first and last item on the list*
DtListItem*	**last** () const
unsigned	**count** () const *Returns the number of elements in the list*
DtListItem*	**add** (void* data) *Add element to the end of the list*
DtListItem*	**addToStart** (void* data) *Add element to the beginning of the list*
DtListItem*	**addToEnd** (void* data) *Add element to the end of the list*
DtListItem*	**addBefore** (void* data, DtListItem* before) *Add element to the list just before the indicated item*
DtListItem*	**addAfter** (void* data, DtListItem* after) *Add element to the list just after the indicated item*
void*	**remove** (DtListItem* item) *Remove an element from the list*
void	**removeAll** () *Calls remove for all elements in the list*
DtListItem*	**itemLookup** (void* data) const *Returns the DtListItem envelope object containing the indicated data element*
inline DtListItem*	**add** (void* data)

- inline DtListItem* **addBefore** (void* data, DtListItem* before)
- inline DtListItem* **addToEnd** (void* data)
- inline DtListItem* **addToStart** (void* data)
- inline unsigned **count** () const
- inline DtListItem* **first** () const
- inline DtListItem* **last** () const

Protected Fields

- DtListItem* **myFirst**
- DtListItem* **myLast**
- unsigned **myCount**

Protected Methods

- virtual void **addToList** (DtListItem* item, DtListItem* after)
  Add item to the list after after
- virtual void **removeFromList** (DtListItem* item)
  Remove item from the list

Documentation

A DtList is a list of elements of arbitrary type. Data elements are added to, and extracted from a DtList as void*. DtList encloses each element in a DtListItem object. You can iterate through the elements in a DtList like this: for (DtListItem* item = list.first(); item; item = item->next()) { void* dataElement = item->data(); }

- **`DtList()`**

  Constructor

- **`virtual ~DtList()`**

  Destructor - does NOT free memory of data associated with the items. It only frees the items.

- **`DtList(const DtList& orig)`**

  Copy constructor. Shallow copy. Copies the pointers you've added to the list, but not the items that are pointed to.

- **`DtList& operator=(const DtList& orig)`**

  Assignment operator. Shallow copy. Copies the pointers you've added to the list, but not the items that are pointed to. Old list items are removed from the list, and the elements they contain are not deleted.

- **`DtListItem* first() const`**

  Return first and last item on the list. DtListItem is an envelope for your data. The data can be obtained using DtListItem::data().

- **`DtListItem* last() const`**

- **unsigned count() const**

  Returns the number of elements in the list

- **DtListItem* add(void* data)**

  Add element to the end of the list. Returns the DtListItem envelope object that contains your new element.

- **DtListItem* addToStart(void* data)**

  Add element to the beginning of the list. Returns the DtListItem envelope object that contains your new element.

- **DtListItem* addToEnd(void* data)**

  Add element to the end of the list. Returns the DtListItem envelope object that contains your new element.

- **DtListItem* addBefore(void* data, DtListItem* before)**

  Add element to the list just before the indicated item. Returns the DtListItem envelope object that contains your new element.

- **DtListItem* addAfter(void* data, DtListItem* after)**

  Add element to the list just after the indicated item. Returns the DtListItem envelope object that contains your new element.

- **void* remove(DtListItem* item)**

  Remove an element from the list. Item is the envelope containing the data element you would like to remove. Note: remove does not free memory of data associated with the item. It only frees the item. The data is returned to the caller for deletion if required.

- **void removeAll()**

  Calls remove for all elements in the list. It does NOT delete the data in the list. Merely removes the data from the list.

- **DtListItem* itemLookup(void* data) const**

  Returns the DtListItem envelope object containing the indicated data element

- **virtual void addToList(DtListItem* item, DtListItem* after)**

  Add item to the list after after

- **virtual void removeFromList(DtListItem* item)**

  Remove item from the list. Does NOT delete the data contained in item.

- **DtListItem* myFirst**

- **DtListItem* myLast**

- **unsigned myCount**

- **inline DtListItem* add(void* data)**

- **inline DtListItem* addBefore(void* data, DtListItem* before)**

- **inline DtListItem* addToEnd(void* data)**

- **inline DtListItem* addToStart(void* data)**

- **inline unsigned count() const**

- **inline DtListItem* first() const**

inline DtListItem* last() const

Direct child classes:

DtScript
DtIntrusiveList

Alphabetic index *Hierarchy of classes*

In file objclassdesc.h:

class DtObjClassDesc

class DtObjClassDesc: An instance of DtObjClassDesc is a "class descriptor" for an HLA Object class

---

Public Methods

	DtObjClassDesc (const char* className, DtObjClassDesc* base, RTI::ObjectClassHandle handle, DtExerciseConn* exConn) *Constructor*
virtual	**~DtObjClassDesc** () *destructor*
virtual void	**addAttr** (const char* attrName) *Add an attribute (Used when reading FED file)*
virtual RTI::ObjectClassHandle	**handle** () const *Get the object's class handle*
virtual char*	**name** () const *Get the object's class handle*
virtual const DtObjClassDesc*	**baseClass** () const *Return a pointer to the base class's descriptor, NULL if none*
virtual int	**isOfClass** (RTI::ObjectClassHandle hand) const *Returns 1 if the class is of class hand or a subclass of hand, 0 otherwise*
virtual int	**numAttributes** () const *Get the number of attributes associated with this object, including priviledge to delete (usually handle 0)*
virtual const RTI::AttributeHandleSet&	**attrHandleSet** () const *Get the set of valid attribute handles for this class*
virtual const RTI::AttributeHandleSet&	**ownAttrHandles** () const *Get the set of non-inherited attribute handles for this class*
virtual DtBoolean	**classNeededByFederation** () *Returns DtTrue if the class is currently needed by the federation, i*
virtual void	**publish** () *Publishes the class, along with the attributes specified by hSet (or all attributes if no hSet is given)*
virtual void	**publish** (const RTI::AttributeHandleSet& hSet)
virtual void	**subscribe** (const RTI::AttributeHandleSet& hSet) *Subscribes to the class, along with the attributes specified by hSet (or all attributes is no hSet is given)*
virtual void	**subscribe** ()
virtual void	**unsubscribe** () *Unsubscribe to the class*
virtual void	**requestUpdates** () *Request updates of all objects of this class, include the attributes specified by hSet (or all subscribed attributes if no hSet is given)*
virtual void	**requestUpdates** (const RTI::AttributeHandleSet& hSet)

- virtual const RTI::AttributeHandleSet& **publishedAttrs** () const
  Returns the set of published attributes
- virtual DtBoolean **everPublished** () const
  Return true if we have ever published the class using DtObjClassDesc::publish, false otherwise
- virtual const RTI::AttributeHandleSet& **subscribedAttrs** () const
  Returns the set of subscribed attributes
- virtual DtBoolean **everSubscribed** () const
  Return true if we have ever subscribed to the class using DtObjClassDesc::subscribe, false otherwise
- virtual DtVrlRtiAmbassador* **rtiAmb** ()
  Get my RTI ambassador
- virtual DtVrlFederateAmbassador* **fedAmb** ()
  Get my federate ambassador
- static int **maxAttributes** ()
- static void **setMaxAttributes** (int num)
- inline const RTI::AttributeHandleSet& **attrHandleSet** () const
- if DtRTI103 inline const RTI::AttributeHandleSet& **attributesNeededByFederation** ()
- inline const DtObjClassDesc* **baseClass** () const
- else inline DtBoolean **classNeededByFederation** ()
- inline RTI::ObjectClassHandle **handle** () const
- endif inline int **maxAttributes** ()
- inline char* **name** () const
- inline int **numAttributes** () const
- inline const RTI::AttributeHandleSet& **ownAttrHandles** () const
- inline DtObjClassDesc* **self** () const
- inline void **setMaxAttributes** (int num)

Protected Fields

- char* **myName**
- RTI::ObjectClassHandle **myHandle**
- RTI::AttributeHandleSet* **myAttrs**
- RTI::AttributeHandleSet* **myOwnAttrs**
- RTI::AttributeHandleSet* **mySubAttrs**
- RTI::AttributeHandleSet* **myPubAttrs**
- DtBoolean **myClassNeeded**
- DtObjClassDesc* **myBase**
- DtExerciseConn* **myExConn**
- DtBoolean **myEverPublished**

DtBoolean	**myEverSubscribed**
static int	**theMaxAttrs**

Protected Methods

static void	**startRegCb** (RTI::ObjectClassHandle h, void* usr) *Callback for handling startRegistration RTI callback*
static void	**stopRegCb** (RTI::ObjectClassHandle h, void* usr) *Callback for handling stopRegistration RTI callback*
virtual void	**processStartReg** (RTI::ObjectClassHandle h) *Callback for handling startRegistration RTI callback*
virtual void	**processStopReg** (RTI::ObjectClassHandle h) *Callback for handling stopRegistration RTI callback*
virtual DtObjClassDesc*	**self** () const *Casts away const-ness*

Documentation

class DtObjClassDesc:

An instance of DtObjClassDesc is a "class descriptor" for an HLA Object class. A list of all DtObjClassDescs for the various classes in a FOM can be obtained from a DtFom. A DtObjClassDesc can tell you the set of attributes associated with the class, the class's handle, name, and base class. Publication, subscription and update requests are done through the DtObjClassDesc as well, and the DtObjClassDesc maintains information on which attributes of the class are subscribed to by remote federates. By registering callbacks on start/stopUpdates, a DtObjClassDesc can tell you which attributes of the class have been subscribed to by other federates.

`DtObjClassDesc(const char* className, DtObjClassDesc* base, RTI::ObjectClassHandle handle, DtExerciseConn* exConn)`

Constructor. Adds the base class' attributes to our attribute set. Used when reading FED file.

`virtual ~DtObjClassDesc()`

destructor

`virtual void addAttr(const char* attrName)`

Add an attribute (Used when reading FED file)

`virtual RTI::ObjectClassHandle handle() const`

Get the object's class handle

`virtual char* name() const`

Get the object's class handle

`virtual const DtObjClassDesc* baseClass() const`

Return a pointer to the base class's descriptor, NULL if none

`virtual int isOfClass(RTI::ObjectClassHandle hand) const`

Returns 1 if the class is of class hand or a subclass of hand, 0 otherwise

`virtual int numAttributes() const`

Get the number of attributes associated with this object, including priviledge to delete (usually handle 0)

- **`virtual const RTI::AttributeHandleSet& attrHandleSet() const`**

  Get the set of valid attribute handles for this class. Includes attributes inherited from base classes.

- **`virtual const RTI::AttributeHandleSet& ownAttrHandles() const`**

  Get the set of non-inherited attribute handles for this class

- **`virtual DtBoolean classNeededByFederation()`**

  Returns DtTrue if the class is currently needed by the federation, i.e. we've received a startRegistrationForObjectClass call for the class. We will only receive those calls if the class has been published using DtObjClassDesc::publish.

- **`virtual void publish()`**

  Publishes the class, along with the attributes specified by hSet (or all attributes if no hSet is given). Also, registers start and stopUpdates callbacks on the published classes. Can be called multiple times to change the set of published classes.

- **`virtual void publish(const RTI::AttributeHandleSet& hSet)`**

- **`virtual void subscribe(const RTI::AttributeHandleSet& hSet)`**

  Subscribes to the class, along with the attributes specified by hSet (or all attributes is no hSet is given)

- **`virtual void subscribe()`**

- **`virtual void unsubscribe()`**

  Unsubscribe to the class

- **`virtual void requestUpdates()`**

  Request updates of all objects of this class, include the attributes specified by hSet (or all subscribed attributes if no hSet is given)

- **`virtual void requestUpdates(const RTI::AttributeHandleSet& hSet)`**

- **`virtual const RTI::AttributeHandleSet& publishedAttrs() const`**

  Returns the set of published attributes. For this to work correctly, do not publish directly through the RTI. Instead, use DtObjClassDesc::publish.

- **`virtual DtBoolean everPublished() const`**

  Return true if we have ever published the class using DtObjClassDesc::publish, false otherwise

- **`virtual const RTI::AttributeHandleSet& subscribedAttrs() const`**

  Returns the set of subscribed attributes. For this to work correctly, do not publish directly through the RTI. Instead, use DtObjClassDesc::subscribe.

- **`virtual DtBoolean everSubscribed() const`**

  Return true if we have ever subscribed to the class using DtObjClassDesc::subscribe, false otherwise

- **`virtual DtVrlRtiAmbassador* rtiAmb()`**

  Get my RTI ambassador

- **`virtual DtVrlFederateAmbassador* fedAmb()`**

  Get my federate ambassador

- **`static int maxAttributes()`**

- **static void setMaxAttributes(int num)**
- **static void startRegCb(RTI::ObjectClassHandle h, void* usr)**
  Callback for handling startRegistration RTI callback
- **static void stopRegCb(RTI::ObjectClassHandle h, void* usr)**
  Callback for handling stopRegistration RTI callback
- **virtual void processStartReg(RTI::ObjectClassHandle h)**
  Callback for handling startRegistration RTI callback
- **virtual void processStopReg(RTI::ObjectClassHandle h)**
  Callback for handling stopRegistration RTI callback
- **virtual DtObjClassDesc* self() const**
  Casts away const-ness. Use with caution.
- **char* myName**
- **RTI::ObjectClassHandle myHandle**
- **RTI::AttributeHandleSet* myAttrs**
- **RTI::AttributeHandleSet* myOwnAttrs**
- **RTI::AttributeHandleSet* mySubAttrs**
- **RTI::AttributeHandleSet* myPubAttrs**
- **DtBoolean myClassNeeded**
- **DtObjClassDesc* myBase**
- **DtExerciseConn* myExConn**
- **DtBoolean myEverPublished**
- **DtBoolean myEverSubscribed**
- **static int theMaxAttrs**
- **inline const RTI::AttributeHandleSet& attrHandleSet() const**
- **if DtRTI103 inline const RTI::AttributeHandleSet& attributesNeededByFederation()**
- **inline const DtObjClassDesc* baseClass() const**
- **else inline DtBoolean classNeededByFederation()**
- **inline RTI::ObjectClassHandle handle() const**
- **endif inline int maxAttributes()**
- **inline char* name() const**
- **inline int numAttributes() const**
- **inline const RTI::AttributeHandleSet& ownAttrHandles() const**

- `inline DtObjClassDesc* self() const`
- `inline void setMaxAttributes(int num)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file unknownhobj.h:

class DtUnknownHlaObject: public DtHlaObject

class DtUnknownHlaObject: Instances of DtUnknownHlaObject

Inheritance:

DtUnknownHlaObject - DtHlaObject

Public Methods

	DtUnknownHlaObject (DtExerciseConn* exConn, const char* name, DtObjClassDesc* classDesc) *Constructor for local objects*
	DtUnknownHlaObject (DtExerciseConn* exConn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc) *Constructor for reflected objects*
virtual	**~DtUnknownHlaObject** () *destructor*
virtual void	**update** () *Does nothing*
virtual void	**update** (const RTI::AttributeHandleValuePairSet& ahvps) *Sends the ahpvs through our DtExerciseConn*

Protected Methods

	DtUnknownHlaObject (const DtUnknownHlaObject& orig) *copy constructor*
DtUnknownHlaObject&	**operator=** (const DtUnknownHlaObject& orig) *assignment operator*
virtual void	**reflect** (const DtStateMsg& stateMsg) *Does nothing*

Inherited from DtHlaObject:

Public Methods

- virtual const RTI::ObjectHandle& **objectId**() const
- virtual const char* **name**() const

- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual int **numAttributes**() const
- virtual const RTI::AttributeHandleSet& **attrHandleSet**()
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtObjClassDesc* **classDesc**()
- virtual void **setupUpdating**()
- virtual void **generateUpdate**(const RTI::AttributeHandleSet& subscribedAttrs, const RTI::AttributeHandleSet& requiredAttrs, DtStateMsg* stateMsg)
- virtual void **cleanupFromUpdating**()
- virtual void **processProvideUpdate**(const RTI::AttributeHandleSet& theAttributes)
- virtual const RTI::AttributeHandleSet& **requestedAttributes**()
- virtual void **clearRequestedAttributes**()
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **clearForcedAttributes**()
- virtual const RTI::AttributeHandleSet& **attributesNeededByFederation**()
- DtBoolean **classNeededByFederation**()
- virtual void **setupClassForUpdating**()
- virtual void **setupReflecting**()
- virtual void **callReflect**(const DtStateMsg& stateMsg)
- virtual void **cleanupFromReflecting**()
- virtual void **requestUpdate**(RTI::AttributeHandleSet* attrs = NULL)
- virtual void **addRemoveObjectCb**(DtRemoveObjectCb cb, void* usr)
- virtual void **removeRemoveObjectCb**(DtRemoveObjectCb cb, void* usr)
- virtual void **addPostReflectCb**(DtObjectReflectAttributeValuesCb cb, void* usr)
- virtual void **removePostReflectCb**(DtObjectReflectAttributeValuesCb cb, void* usr)
- virtual DtTime **lastSimTimeUpdateReceived**() const
- virtual void **printMsg**(const DtStateMsg& msg) const
- static int **maxObjectClasses**()
- inline const RTI::AttributeHandleSet& **attrHandleSet**()
- inline DtExerciseConn* **exerciseConn**() const
- inline DtTime **lastSimTimeUpdateReceived**() const

- inline int **maxObjectClasses**()
- inline const char* **name**() const
- inline int **numAttributes**() const
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline const RTI::ObjectHandle& **objectId**() const
- inline const RTI::AttributeHandleSet& **requestedAttributes**()
- inline void **setLastSimTimeUpdateReceived**(DtTime time)

Public

-
-

Protected Fields

- char* **myName**
- RTI::AttributeHandleSet* **myNeededAttrs**
- DtExerciseConn* **myExConn**
- RTI::ObjectHandle **myObjectId**
- DtBoolean **myReflecting**
- DtBoolean **myUpdating**
- DtList **myRemoveCbList**
- DtTime **myLastSimTimeUpdateReceived**
- DtList **myReflectCbList**
- DtObjClassDesc* **myClassDesc**
- RTI::AttributeHandleSet* **myRequestedAttributes**
- RTI::AttributeHandleSet* **myForcedAttributes**
- RTI::AttributeHandleSet* **myRequiredAttributes**
- static const int **theMaxObjectClasses**

Protected Methods

- virtual void **processRemoveObject**()
- void **wipeRemoveList**()
- virtual void **clearRemoveList**()

- void **wipeReflectList**()
- void **copyRemoveList**(const DtHlaObject& orig)
- virtual void **setLastSimTimeUpdateReceived**(DtTime time)
- virtual void **applyPostReflectCallbacks**(const DtStateMsg& stateMsg)
- virtual void **processUpdatesOn**(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& set)
- virtual void **processUpdatesOff**(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& set)
- static void **reflectAttributeValuesCb**(const DtStateMsg& stateMsg, void* hlaObj)
- static void **updatesOnCb**(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& attributeSet, void* usr)
- static void **updatesOffCb**(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& attributeSet, void* usr)

Documentation

class DtUnknownHlaObject: Instances of DtUnknownHlaObject

- **`DtUnknownHlaObject( DtExerciseConn* exConn, const char* name, DtObjClassDesc* classDesc)`**

  Constructor for local objects

- **`DtUnknownHlaObject( DtExerciseConn* exConn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc)`**

  Constructor for reflected objects

- **`virtual ~DtUnknownHlaObject()`**

  destructor

- **`virtual void update()`**

  Does nothing. Use the version of update that takes an ahpvs to send data for a DtUnknownHlaObject.

- **`virtual void update(const RTI::AttributeHandleValuePairSet& ahvps)`**

  Sends the ahpvs through our DtExerciseConn. Does not look at the contents of ahvps. It is the caller's responsibility to insure that all attributes in the message are in the set of attributes needed by the federation (obtained using attributesNeededByFederation). It is also the responsibility of the user of DtUnknownHlaObject to insure that requested attributes (obtained via requestedAttributes) get updated in a timely manner. clearRequestedAttributes should be called after each time requested attributes are updated.

- **`DtUnknownHlaObject(const DtUnknownHlaObject& orig)`**

  copy constructor

- **`DtUnknownHlaObject& operator=(const DtUnknownHlaObject& orig)`**

  assignment operator

- **`virtual void reflect(const DtStateMsg& stateMsg)`**

  Does nothing. You can access information from incoming update messages by using DtHlaObject::addPostReflectCb to register a function that will be called whenever an update msg is received.

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hlaobjwithsr.h:

class DtHlaObjectWithStateRep: public DtHlaObject

class DtHlaObjectWithStateRep: Instances of DtHlaObjectWithStateRep are used to represent an HLA object via a state representation

---

Inheritance:

DtHlaObjectWithStateRep - DtHlaObject

Public Methods

	DtHlaObjectWithStateRep (DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* classDesc) *constructor*
	DtHlaObjectWithStateRep (DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc) *constructor*
	DtHlaObjectWithStateRep (DtExerciseConn* conn, const char* name, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtHlaObjectWithStateRep** () *destructor*
	DtHlaObjectWithStateRep (const DtHlaObjectWithStateRep& orig) *copy constructor*
DtHlaObjectWithStateRep&	**operator=** (const DtHlaObjectWithStateRep& orig) *assignment operator*
virtual void	**useTheseForUpdating** (DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder, DtHlaStateDecoder* decoder) *Set the state repositories (local representation and remote reflection), and the encoder and decoder which are used to send and receive HLA state messages from/into the repository*
virtual void	**generateUpdate** (const RTI::AttributeHandleSet& subscribedAttrs, const RTI::AttributeHandleSet& requiredAttrs, DtStateMsg* stateMsg) *Called by update() to actually generate an outgoing message*
virtual void	**useTheseForReflecting** (DtStateRepository* stateRep, DtHlaStateDecoder* decoder) *Set the state repository for holding a remote reflection and the decoder which is used to receive HLA state messages into the repository*
virtual void	**reflect** (const DtStateMsg& stateMsg) *Receive representation (remote reflection) from the network*
virtual void	**printMsg** (const DtStateMsg& msg) const *Prints contents of a state msg*

- static DtHlaObject* **create** (DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* classDesc)
  Creator function for this class
- static DtHlaObject* **create** (DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc)
  Creator function for this class
- inline DtHlaObject* **create** (DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* classDesc)
- else inline DtHlaObject* **create** (DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc)

Public

- *Local Entity specific functions*
- *Remote Entity specific functions*

Protected Fields

- DtHlaStateDecoder* **myDecoder**
- DtHlaStateEncoder* **myEncoder**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**

Inherited from DtHlaObject:

Public Methods

- virtual const RTI::ObjectHandle& **objectId**() const
- virtual const char* **name**() const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual int **numAttributes**() const
- virtual const RTI::AttributeHandleSet& **attrHandleSet**()
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtObjClassDesc* **classDesc**()
- virtual void **setupUpdating**()
- virtual void **update**()
- virtual void **cleanupFromUpdating**()
- virtual void **processProvideUpdate**(const RTI::AttributeHandleSet& theAttributes)

- virtual const RTI::AttributeHandleSet& **requestedAttributes**()
- virtual void **clearRequestedAttributes**()
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **clearForcedAttributes**()
- virtual const RTI::AttributeHandleSet& **attributesNeededByFederation**()
- DtBoolean **classNeededByFederation**()
- virtual void **setupClassForUpdating**()
- virtual void **setupReflecting**()
- virtual void **callReflect**(const DtStateMsg& stateMsg)
- virtual void **cleanupFromReflecting**()
- virtual void **requestUpdate**(RTI::AttributeHandleSet* attrs = NULL)
- virtual void **addRemoveObjectCb**(DtRemoveObjectCb cb, void* usr)
- virtual void **removeRemoveObjectCb**(DtRemoveObjectCb cb, void* usr)
- virtual void **addPostReflectCb**(DtObjectReflectAttributeValuesCb cb, void* usr)
- virtual void **removePostReflectCb**(DtObjectReflectAttributeValuesCb cb, void* usr)
- virtual DtTime **lastSimTimeUpdateReceived**() const
- static int **maxObjectClasses**()
- inline const RTI::AttributeHandleSet& **attrHandleSet**()
- inline DtExerciseConn* **exerciseConn**() const
- inline DtTime **lastSimTimeUpdateReceived**() const
- inline int **maxObjectClasses**()
- inline const char* **name**() const
- inline int **numAttributes**() const
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline const RTI::ObjectHandle& **objectId**() const
- inline const RTI::AttributeHandleSet& **requestedAttributes**()
- inline void **setLastSimTimeUpdateReceived**(DtTime time)

Protected Fields

- char* **myName**
- RTI::AttributeHandleSet* **myNeededAttrs**
- DtExerciseConn* **myExConn**
- RTI::ObjectHandle **myObjectId**
- DtBoolean **myReflecting**
- DtBoolean **myUpdating**
- DtList **myRemoveCbList**
- DtTime **myLastSimTimeUpdateReceived**
- DtList **myReflectCbList**
- DtObjClassDesc* **myClassDesc**
- RTI::AttributeHandleSet* **myRequestedAttributes**
- RTI::AttributeHandleSet* **myForcedAttributes**
- RTI::AttributeHandleSet* **myRequiredAttributes**
- static const int **theMaxObjectClasses**

Protected Methods

- virtual void **processRemoveObject**()
- void **wipeRemoveList**()
- virtual void **clearRemoveList**()
- void **wipeReflectList**()
- void **copyRemoveList**(const DtHlaObject& orig)
- virtual void **setLastSimTimeUpdateReceived**(DtTime time)
- virtual void **applyPostReflectCallbacks**(const DtStateMsg& stateMsg)
- virtual void **processUpdatesOn**(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& set)
- virtual void **processUpdatesOff**(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& set)
- static void **reflectAttributeValuesCb**(const DtStateMsg& stateMsg, void* hlaObj)
- static void **updatesOnCb**(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& attributeSet, void* usr)
- static void **updatesOffCb**(RTI::ObjectHandle hand, const RTI::AttributeHandleSet& attributeSet, void* usr)

Documentation

class DtHlaObjectWithStateRep: Instances of DtHlaObjectWithStateRep are used to represent an HLA object via a state representation

- **DtHlaObjectWithStateRep(DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* classDesc)**

  constructor

- **DtHlaObjectWithStateRep(DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc)**

  constructor

- **DtHlaObjectWithStateRep(DtExerciseConn* conn, const char* name, DtObjClassDesc* classDesc)**

  constructor

- **virtual ~DtHlaObjectWithStateRep()**

  destructor

- **DtHlaObjectWithStateRep(const DtHlaObjectWithStateRep& orig)**

  copy constructor

- **DtHlaObjectWithStateRep& operator=(const DtHlaObjectWithStateRep& orig)**

  assignment operator

- Local Entity specific functions

- **virtual void useTheseForUpdating(DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder, DtHlaStateDecoder* decoder)**

  Set the state repositories (local representation and remote reflection), and the encoder and decoder which are used to send and receive HLA state messages from/into the repository

- **virtual void generateUpdate(const RTI::AttributeHandleSet& subscribedAttrs, const RTI::AttributeHandleSet& requiredAttrs, DtStateMsg* stateMsg)**

  Called by update() to actually generate an outgoing message. This class's implementation uses an encoder, and stateRepositories to perform this function. subscribedAttrs is the set of attributes to which remote federates have subscribed. requiredAttrs is the set of attributes that have been requested or forced by application code since the attribute was last sent. stateMsg is the message to fill out.

- Remote Entity specific functions

- **virtual void useTheseForReflecting(DtStateRepository* stateRep, DtHlaStateDecoder* decoder)**

  Set the state repository for holding a remote reflection and the decoder which is used to receive HLA state messages into the repository

- **virtual void reflect(const DtStateMsg& stateMsg)**

  Receive representation (remote reflection) from the network

- **virtual void printMsg(const DtStateMsg& msg) const**

  Prints contents of a state msg. Base class prints only header info.

static DtHlaObject* create(DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* classDesc)

Creator function for this class

static DtHlaObject* create(DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc)

Creator function for this class

DtHlaStateDecoder* myDecoder

DtHlaStateEncoder* myEncoder

DtStateRepository* myStateRep

DtStateRepository* myAsSeenByRemote

inline DtHlaObject* create(DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* classDesc)

else inline DtHlaObject* create(DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* classDesc)

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file filename.h:

class DtFilename: public DtString

class DtFilename: Instances of DtFilename are DtStrings which convert filenames to the native format

---

Inheritance:

DtFilename - DtString

Public Methods

	DtFilename () *default constructor*
	DtFilename (const char* str) *constructor*
virtual	**~DtFilename** () *destructor*
	DtFilename (const DtFilename& orig) *copy constructor*
DtFilename&	**operator=** (const DtFilename& orig) *assignment operator*
virtual DtBoolean	**absolute** () const *Returns DtTrue if the filename is an absolute filename, or a relative filename, respectively*
virtual DtBoolean	**relative** () const
static DtFilename	**combine** (const DtFilename& path, const DtFilename& filename) *Constructs a new filename by combining the path with filename, which is an absolute or relative path*
static char	**separator** () *returns the separator used in representing directories*

Protected Fields

static const char	**theSeparator**

Protected Methods

void	**replaceSlashes** () *Replaces slashes*

Inherited from DtString:

Public Methods

- virtual **operator const char***() const
- friend DtBoolean **operator==**(const DtString& str1, const DtString& str2)
- friend DtBoolean **operator<**(const DtString& str1, const DtString& str2)
- friend DtBoolean **operator<=**(const DtString& str1, const DtString& str2)
- friend DtBoolean **operator>**(const DtString& str1, const DtString& str2)
- friend DtBoolean **operator>=**(const DtString& str1, const DtString& str2)
- friend DtBoolean **operator!=**(const DtString& str1, const DtString& str2)
- virtual DtBoolean **operator==**(const char* str) const
- virtual DtBoolean **operator<**(const char* str) const
- virtual DtBoolean **operator<=**(const char* str) const
- virtual DtBoolean **operator>**(const char* str) const
- virtual DtBoolean **operator>=**(const char* str) const
- virtual DtBoolean **operator!=**(const char* str) const
- virtual DtBoolean **caseCompare**(const DtString& str)
- friend DtString **operator+**(const DtString& str1, const DtString& str2)
- virtual DtString **operator+**(const char* str) const
- virtual DtString& **operator+=**(const char* str)
- virtual DtString& **operator+=**(const DtString& str)
- virtual const char& **operator[]**(int index) const
- virtual char& **operator[]**(int index)
- virtual DtBoolean **snipFront**(const char ch, DtBoolean toLastCh = DtFalse)
- virtual DtBoolean **snipBack**(const char ch, DtBoolean fromFirstCh = DtFalse)
- DtBoolean **snipFrontToLast**(const char ch)
- DtBoolean **snipBackToFirst**(const char ch)
- virtual int **count**(const char ch) const
- virtual int **length**() const
- const char* **string**() const

- static const DtString& **nullString**()

Protected Fields

- char* **myString**
- static const int **theMaxInsertionStringSize**

Protected Methods

- void **clear**()
- void **setString**(const char* str)

Private Methods

- friend ostream& **operator<<**(ostream& os, const DtString& str)
- friend istream& **operator>>**(istream& is, DtString& str)

Documentation

class DtFilename: Instances of DtFilename are DtStrings which convert filenames to the native format

- **`DtFilename()`**
  default constructor
- **`DtFilename(const char* str)`**
  constructor
- **`virtual ~DtFilename()`**
  destructor
- **`DtFilename(const DtFilename& orig)`**
  copy constructor
- **`DtFilename& operator=(const DtFilename& orig)`**
  assignment operator
- **`virtual DtBoolean absolute() const`**
  Returns DtTrue if the filename is an absolute filename, or a relative filename, respectively
- **`virtual DtBoolean relative() const`**
- **`static DtFilename combine(const DtFilename& path, const DtFilename& filename)`**
  Constructs a new filename by combining the path with filename, which is an absolute or relative path. If the filename is an absolute filename, then filename is returned. If the filename is a relative filename, then path is prepended to it.

- **`static char separator()`**

  returns the separator used in representing directories

- **`void replaceSlashes()`**

  Replaces slashes. On unix, this replaces backslashes with forward slashes. Vice versa on win32. Hail Mr Gates. This is not virtual because we call this from the constructors and destructor.

- **`static const char theSeparator`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file intclassdesc.h:

class DtInterClassDesc

class DtInterClassDesc: Instances of DtInterClassDesc are used to represent HLA Object and Interaction classes

---

Public Methods

	DtInterClassDesc (const char* className, DtInterClassDesc* base, RTI::InteractionClassHandle handle, DtExerciseConn* exConn) *constructor*
virtual	**~DtInterClassDesc** () *destructor*
	DtInterClassDesc (const DtInterClassDesc& orig) *copy constructor*
DtInterClassDesc&	**operator=** (const DtInterClassDesc& orig) *assignment operator*
virtual void	**addParam** (const char* paramName) *Add a parameter (Used when reading FED file)*
virtual RTI::InteractionClassHandle	**handle** () const *Get the object's class handle*
virtual char*	**name** () const *Get the object's class name*
virtual const DtInterClassDesc*	**baseClass** () const *Return a pointer to the base class's descriptor, NULL if none*
virtual int	**isOfClass** (RTI::InteractionClassHandle hand) const *Returns 1 if the class is of class hand or a subclass of hand, 0 otherwise*
virtual int	**numParameters** () const *Get the number of parameters associated with this object*
virtual const RTI::AttributeHandleSet&	**paramHandleSet** () const *Get the set of valid parameter handles for this class*
virtual const RTI::AttributeHandleSet&	**ownParamHandles** () const *Get the set of non-inherited parameter handles for this class*
virtual void	**publish** () *Publish the class, and register start and stopUpdates callbacks*
virtual void	**subscribe** () *Subscribe to the class*
virtual void	**unsubscribe** () *Unsubscribe to the class*
virtual DtBoolean	**published** () const *Returns DtTrue if the class is published, DtFalse otherwise*
virtual DtBoolean	**subscribed** () const *Returns DtTrue if the class is subscribed, DtFalse otherwise*
virtual DtVrlRtiAmbassador*	**rtiAmb** () *Get my RTI ambassador*

virtual DtBoolean	**neededByFederation** () const *Returns DtTrue if the interaction class is needed by the federation false otherwise*
virtual DtBoolean	**everPublished** () const *Return true if we have ever published the class using DtInterClassDesc::publish, false otherwise*
virtual DtBoolean	**everSubscribed** () const *Return true if we have ever subscribed to the class using DtInterClassDesc::subscribe, false otherwise*
static void	**setMaxParameters** (int num) *Set/Get maximum number of allowable parameters for an interaction*
static int	**maxParameters** ()
inline const DtInterClassDesc*	**baseClass** () const
inline RTI::InteractionClassHandle	**handle** () const
inline int	**maxParameters** ()
inline char*	**name** () const
inline int	**numParameters** () const
inline const RTI::AttributeHandleSet&	**ownParamHandles** () const
inline const RTI::AttributeHandleSet&	**paramHandleSet** () const
inline void	**setMaxParameters** (int num)

Protected Fields

char*	**myName**
RTI::InteractionClassHandle	**myHandle**
RTI::AttributeHandleSet*	**myParams**
RTI::AttributeHandleSet*	**myOwnParams**
DtInterClassDesc*	**myBase**
DtExerciseConn*	**myExConn**
DtBoolean	**myNeededFlag**
DtBoolean	**myPublished**
DtBoolean	**mySubscribed**
DtBoolean	**myEverPublished**
DtBoolean	**myEverSubscribed**
static int	**theMaxParams**

Protected Methods

- static void **startIntersCb** (RTI::InteractionClassHandle hand, void* usr)
  Callback for handling startInters RTI callback
- static void **stopIntersCb** (RTI::InteractionClassHandle hand, void* usr)
  Callback for handling stopInters RTI callback
- virtual void **processStartInters** (RTI::InteractionClassHandle hand)
  Callback for handling startInters RTI callback
- virtual void **processStopInters** (RTI::InteractionClassHandle hand)
  Callback for handling stopInters RTI callback

Documentation

class DtInterClassDesc: Instances of DtInterClassDesc are used to represent HLA Object and Interaction classes

- **`DtInterClassDesc(const char* className, DtInterClassDesc* base, RTI::InteractionClassHandle handle, DtExerciseConn* exConn)`**

  constructor

- **`virtual ~DtInterClassDesc()`**

  destructor

- **`DtInterClassDesc(const DtInterClassDesc& orig)`**

  copy constructor

- **`DtInterClassDesc& operator=(const DtInterClassDesc& orig)`**

  assignment operator

- **`virtual void addParam(const char* paramName)`**

  Add a parameter (Used when reading FED file)

- **`virtual RTI::InteractionClassHandle handle() const`**

  Get the object's class handle

- **`virtual char* name() const`**

  Get the object's class name

- **`virtual const DtInterClassDesc* baseClass() const`**

  Return a pointer to the base class's descriptor, NULL if none

- **`virtual int isOfClass(RTI::InteractionClassHandle hand) const`**

  Returns 1 if the class is of class hand or a subclass of hand, 0 otherwise

- **`virtual int numParameters() const`**

  Get the number of parameters associated with this object

- **`virtual const RTI::AttributeHandleSet& paramHandleSet() const`**

  Get the set of valid parameter handles for this class. Includes parameters inherited from base classes.

- **`virtual const RTI::AttributeHandleSet& ownParamHandles() const`**

Get the set of non-inherited parameter handles for this class

- **`virtual void publish()`**
  Publish the class, and register start and stopUpdates callbacks
- **`virtual void subscribe()`**
  Subscribe to the class
- **`virtual void unsubscribe()`**
  Unsubscribe to the class
- **`virtual DtBoolean published() const`**
  Returns DtTrue if the class is published, DtFalse otherwise
- **`virtual DtBoolean subscribed() const`**
  Returns DtTrue if the class is subscribed, DtFalse otherwise
- **`virtual DtVrlRtiAmbassador* rtiAmb()`**
  Get my RTI ambassador
- **`virtual DtBoolean neededByFederation() const`**
  Returns DtTrue if the interaction class is needed by the federation false otherwise
- **`virtual DtBoolean everPublished() const`**
  Return true if we have ever published the class using DtInterClassDesc::publish, false otherwise
- **`virtual DtBoolean everSubscribed() const`**
  Return true if we have ever subscribed to the class using DtInterClassDesc::subscribe, false otherwise
- **`static void setMaxParameters(int num)`**
  Set/Get maximum number of allowable parameters for an interaction
- **`static int maxParameters()`**
- **`static void startIntersCb(RTI::InteractionClassHandle hand, void* usr)`**
  Callback for handling startInters RTI callback
- **`static void stopIntersCb(RTI::InteractionClassHandle hand, void* usr)`**
  Callback for handling stopInters RTI callback
- **`virtual void processStartInters(RTI::InteractionClassHandle hand)`**
  Callback for handling startInters RTI callback
- **`virtual void processStopInters(RTI::InteractionClassHandle hand)`**
  Callback for handling stopInters RTI callback
- **`char* myName`**
- **`RTI::InteractionClassHandle myHandle`**
- **`RTI::AttributeHandleSet* myParams`**
- **`RTI::AttributeHandleSet* myOwnParams`**
- **`DtInterClassDesc* myBase`**

- DtExerciseConn* myExConn
- DtBoolean myNeededFlag
- DtBoolean myPublished
- DtBoolean mySubscribed
- DtBoolean myEverPublished
- DtBoolean myEverSubscribed
- static int theMaxParams
- inline const DtInterClassDesc* baseClass() const
- inline RTI::InteractionClassHandle handle() const
- inline int maxParameters()
- inline char* name() const
- inline int numParameters() const
- inline const RTI::AttributeHandleSet& ownParamHandles() const
- inline const RTI::AttributeHandleSet& paramHandleSet() const
- inline void setMaxParameters(int num)

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vllist.h:

class DtListItem

DtListItem is an envelope class used by DtList

Public Methods

- virtual **~DtListItem** ()
  Destructor
- DtListItem* **prev** () const
  Return previous (next) item
- DtListItem* **next** () const
- void* **data** () const
  Returns the data element contained in the DtListItem envelope
- inline void* **data** () const
- inline DtListItem* **next** () const
- inline DtListItem* **prev** () const
  Inline Functions

Protected Fields

- DtListItem* **myPrev**
- DtListItem* **myNext**
- void* **myData**

Protected Methods

- **DtListItem** (void* d)
  Constructor - called only by DtList

Documentation

DtListItem is an envelope class used by DtList. When you add an element to a DtList, it creates a DtListItem to contain the element. The DtListItem stores the next and previous pointer, as well as the data element itself.

- `virtual ~DtListItem()`

  Destructor

- `DtListItem* prev() const`

Return previous (next) item. If you hit the beginning (end) of the list, returns NULL.

- **DtListItem* next() const**
- **void* data() const**

  Returns the data element contained in the DtListItem envelope

- **DtListItem(void* d)**

  Constructor - called only by DtList

- **DtListItem* myPrev**
- **DtListItem* myNext**
- **void* myData**
- **inline void* data() const**
- **inline DtListItem* next() const**
- **inline DtListItem* prev() const**

  Inline Functions

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file script.h:

class DtScript: public DtList

A DtScript is a DtList of StealthInstructions

Inheritance:

DtScript - DtList

Public Methods

- **DtScript** (char *fname)
- **DtScript** ()
- **~DtScript** ()
- void **addInstruction** (DtTime time, DtViewerCommand command, char *args)
- void **addInstruction** (DtTime time, DtViewControlPdu *pdu)
- void **clear** ()

Inherited from DtList:

Public Methods

- DtList& **operator=**(const DtList& orig)
- DtListItem* **first**() const
- DtListItem* **last**() const
- unsigned **count**() const
- DtListItem* **add**(void* data)
- DtListItem* **addToStart**(void* data)
- DtListItem* **addToEnd**(void* data)
- DtListItem* **addBefore**(void* data, DtListItem* before)
- DtListItem* **addAfter**(void* data, DtListItem* after)

- void* **remove**(DtListItem* item)
- void **removeAll**()
- DtListItem* **itemLookup**(void* data) const
- inline DtListItem* **add**(void* data)
- inline DtListItem* **addBefore**(void* data, DtListItem* before)
- inline DtListItem* **addToEnd**(void* data)
- inline DtListItem* **addToStart**(void* data)
- inline unsigned **count**() const
- inline DtListItem* **first**() const
- inline DtListItem* **last**() const

Protected Fields

- DtListItem* **myFirst**
- DtListItem* **myLast**
- unsigned **myCount**

Protected Methods

- virtual void **addToList**(DtListItem* item, DtListItem* after)
- virtual void **removeFromList**(DtListItem* item)

Documentation

A DtScript is a DtList of StealthInstructions

- **`DtScript(char *fname)`**
- **`DtScript()`**
- **`~DtScript()`**
- **`void addInstruction(DtTime time, DtViewerCommand command, char *args)`**
- **`void addInstruction(DtTime time, DtViewControlPdu *pdu)`**
- **`void clear()`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vllist.h:

class DtIntrusiveList: public DtList

An intrusive list is a list that assumes that the elements that you add are pointers to instances of classes derived from DtListedObject

---

Inheritance:

DtIntrusiveList - DtList

Public Methods

- virtual void **addToList** (DtListItem* item, DtListItem* after)
  Override of virtual function
- virtual void **removeFromList** (DtListItem* item)
  Override of virtual function

Inherited from DtList:

Public Methods

- DtList& **operator=**(const DtList& orig)
- DtListItem* **first**() const
- DtListItem* **last**() const
- unsigned **count**() const
- DtListItem* **add**(void* data)
- DtListItem* **addToStart**(void* data)
- DtListItem* **addToEnd**(void* data)
- DtListItem* **addBefore**(void* data, DtListItem* before)
- DtListItem* **addAfter**(void* data, DtListItem* after)
- void* **remove**(DtListItem* item)
- void **removeAll**()

- DtListItem* **itemLookup**(void* data) const
- inline DtListItem* **add**(void* data)
- inline DtListItem* **addBefore**(void* data, DtListItem* before)
- inline DtListItem* **addToEnd**(void* data)
- inline DtListItem* **addToStart**(void* data)
- inline unsigned **count**() const
- inline DtListItem* **first**() const
- inline DtListItem* **last**() const

Protected Fields

- DtListItem* **myFirst**
- DtListItem* **myLast**
- unsigned **myCount**

Documentation

An intrusive list is a list that assumes that the elements that you add are pointers to instances of classes derived from DtListedObject. When an intrusive list is use, you can get previous and next pointers directly from the listed elements, (through the base DtListedObject class), without having to go through the DtList itself.

- **`virtual void addToList(DtListItem* item, DtListItem* after)`**

  Override of virtual function. Saves pointer to the list and to the item in the DtListedObj.

- **`virtual void removeFromList(DtListItem* item)`**

  Override of virtual function. Removes the item from this list, and cleans up the DtListedObj's item and list pointers.

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hunknowninter.h:

class DtUnknownInteraction: public DtInteraction

Inheritance:

DtUnknownInteraction - DtInteraction

Public Methods

	DtUnknownInteraction () *default constructor*
	DtUnknownInteraction (RTI::InteractionClassHandle handle, const RTI::ParameterHandleValuePairSet& pvList) *constructor*
virtual	**~DtUnknownInteraction** () *destructor*
	DtUnknownInteraction (const DtUnknownInteraction& orig) *copy constructor*
DtUnknownInteraction&	**operator=** (const DtUnknownInteraction& orig) *assignment operator*
virtual const RTI::ParameterHandleValuePairSet&	**phvps** () const *Get the parameter handle value pair set for the interaction*
virtual void	**setFromPhvps** (const RTI::ParameterHandleValuePairSet& pvList) *Decode into this object from parameter value list*
virtual void	**setExConn** (DtExerciseConn* exConn, DtInterClassDesc* classDesc) *Sets the interaction's ExerciseConn and class descriptor*
virtual void	**printData** () const *Just prints the parameters*
virtual const char*	**name** () const *Name of the interaction*
static DtInteraction*	**create** () *Create a DtUnknownInteraction*
static void	**addCallback** (RTI::InteractionClassHandle handle, DtExerciseConn* conn, DtUnknownInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction with specified class handle occurs*
static void	**removeCallback** (RTI::InteractionClassHandle handle, DtExerciseConn* conn, DtUnknownInteractionCb cb, void* usr)

Protected Fields

- RTI::ParameterHandleValuePairSet* **myParams**
- RTI::InteractionClassHandle **myHandle**

Protected Methods

- void **internalSetFromPhvps** (const RTI::ParameterHandleValuePairSet& pvList)
  Decode into this object from parameter value list

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const

- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtUnknownInteraction()`**
  default constructor
- **`DtUnknownInteraction(RTI::InteractionClassHandle handle, const RTI::ParameterHandleValuePairSet& pvList)`**
  constructor
- **`virtual ~DtUnknownInteraction()`**
  destructor
- **`DtUnknownInteraction(const DtUnknownInteraction& orig)`**
  copy constructor

- **DtUnknownInteraction& operator=(const DtUnknownInteraction& orig)**

  assignment operator

- **virtual const RTI::ParameterHandleValuePairSet& phvps() const**

  Get the parameter handle value pair set for the interaction

- **virtual void setFromPhvps(const RTI::ParameterHandleValuePairSet& pvList)**

  Decode into this object from parameter value list

- **virtual void setExConn(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**

  Sets the interaction's ExerciseConn and class descriptor. If classDesc is NULL, we use the classDesc associated with the handle on which the interaction was created. Without being told what ExConn or class descriptor to use, many of the HLA-specific functions won't work properly.

- **virtual void printData() const**

  Just prints the parameters

- **virtual const char* name() const**

  Name of the interaction

- **static DtInteraction* create()**

  Create a DtUnknownInteraction. Caller is reponsible for deletion.

- **static void addCallback(RTI::InteractionClassHandle handle, DtExerciseConn* conn, DtUnknownInteractionCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction with specified class handle occurs

- **static void removeCallback(RTI::InteractionClassHandle handle, DtExerciseConn* conn, DtUnknownInteractionCb cb, void* usr)**

- **void internalSetFromPhvps(const RTI::ParameterHandleValuePairSet& pvList)**

  Decode into this object from parameter value list. Non-virtual since meant to be called from ctor.

- **RTI::ParameterHandleValuePairSet* myParams**

- **RTI::InteractionClassHandle myHandle**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hintwencdec.h:

class DtInteractionWithEncDec: public DtInteraction

class DtInteractionWithEncDec: DtInteractionWithEncDec is a DtInteraction that uses an encoder and decoder to implement the virtual functions phvps and setFromPhvps

---

Inheritance:

DtInteractionWithEncDec - DtInteraction

Public Methods

	DtInteractionWithEncDec () *default constructor*
virtual	**~DtInteractionWithEncDec** () *destructor*
	DtInteractionWithEncDec (const DtInteractionWithEncDec& orig) *copy constructor*
DtInteractionWithEncDec&	**operator=** (const DtInteractionWithEncDec& orig) *assignment operator*
virtual void	**setExConn** (DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL) *Sets the interaction's ExerciseConn and class descriptor*
virtual const RTI::ParameterHandleValuePairSet&	**phvps** () const *Get the parameter handle value pair set for the interaction*
virtual void	**setFromPhvps** (const RTI::ParameterHandleValuePairSet& pvList) *Set contents from a parameter handle value pair set*
virtual void	**useThese** (DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data) *Specify an encoder, decoder, and interaction data repository to use in encoding and decoding*
virtual DtInteractionEncoder*	**encoder** () const *Return the encoder we are using*
virtual DtInteractionDecoder*	**decoder** () const *Return the decoder we are using*
inline DtInteractionDecoder*	**decoder** () const
inline DtInteractionEncoder*	**encoder** () const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder** (DtExerciseConn* exConn) const
  Called by decoderToUse to obtain a new'ed decoder
- virtual DtInteractionEncoder* **createEncoder** (DtExerciseConn* exConn) const
  Called by encoderToUse to obtain a new'ed encoder
- virtual DtInteractionDecoder* **decoderToUse** (DtExerciseConn* exConn) const
  Returns a decoder to use for decoding the interaction
- virtual DtInteractionEncoder* **encoderToUse** (DtExerciseConn* exConn) const
  Returns an encoder to use for encoding the interaction

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printData**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual const char* **name**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const

- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static void **addCallback**(DtExerciseConn* conn, DtList* classes, DtReceiveInteractionCb cb, void* usr)
- static void **removeCallback**(DtExerciseConn* conn, DtList* classes, DtReceiveInteractionCb cb, void* usr)
- static void **addCallback**(DtExerciseConn* conn, const char* className, DtReceiveInteractionCb cb, void* usr)
- static void **removeCallback**(DtExerciseConn* conn, const char* className, DtReceiveInteractionCb cb, void* usr)
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline void **printData**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

class DtInteractionWithEncDec: DtInteractionWithEncDec is a DtInteraction that uses an encoder and decoder to implement the virtual functions phvps and setFromPhvps

- **`DtInteractionWithEncDec()`**

  default constructor

- **`virtual ~DtInteractionWithEncDec()`**

  destructor

- **`DtInteractionWithEncDec(const DtInteractionWithEncDec& orig)`**

  copy constructor

- **`DtInteractionWithEncDec& operator=(const DtInteractionWithEncDec& orig)`**

  assignment operator

- **`virtual void setExConn(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)`**

  Sets the interaction's ExerciseConn and class descriptor. If classDesc is NULL, a default is chosen by calling interactionClassToUse(). Without being told what ExConn or class descriptor to use, many of the HLA-specific functions won't work properly. However, it is usually not necessary for application code to call this function, since it is called by DtExerciseConn::send on the outgoing side, and by the DtInteractionFactory on the incoming side.

- **`virtual const RTI::ParameterHandleValuePairSet& phvps() const`**

  Get the parameter handle value pair set for the interaction

- **`virtual void setFromPhvps( const RTI::ParameterHandleValuePairSet& pvList)`**

  Set contents from a parameter handle value pair set

- **`virtual void useThese(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)`**

  Specify an encoder, decoder, and interaction data repository to use in encoding and decoding. Called by setExConn (using encoder and decoder obtained through encoderToUse() and decoderToUse()) so call AFTER setExConn if you want to override defaults. Caller is responsible for deleting enc, dec and data, but should do so only after they are no longer used by the interaction.

- **`virtual DtInteractionEncoder* encoder() const`**

  Return the encoder we are using

- **`virtual DtInteractionDecoder* decoder() const`**

  Return the decoder we are using

- **`virtual DtInteractionDecoder* createDecoder( DtExerciseConn* exConn) const`**

  Called by decoderToUse to obtain a new'ed decoder. Default implmentation is to ask the FOM Mapper. The object

is automatically deleted in the interaction's destructor.

- **virtual DtInteractionEncoder* createEncoder(DtExerciseConn* exConn) const**

  Called by encoderToUse to obtain a new'ed encoder. Default implmentation is to ask the FOM Mapper. The object is automatically deleted in the interaction's destructor.

- **virtual DtInteractionDecoder* decoderToUse(DtExerciseConn* exConn) const**

  Returns a decoder to use for decoding the interaction

- **virtual DtInteractionEncoder* encoderToUse(DtExerciseConn* exConn) const**

  Returns an encoder to use for encoding the interaction

- **DtInteractionDecoder* myDecoder**
- **DtInteractionEncoder* myEncoder**
- **DtInteractionDecoder* myCreatedDecoder**
- **DtInteractionEncoder* myCreatedEncoder**
- **DtInteractionRep* myData**
- **inline DtInteractionDecoder* decoder() const**
- **inline DtInteractionEncoder* encoder() const**

Direct child classes:

DtViewControlInteraction
DtStopFreezeInteraction
DtStartResumeInteraction
DtSignalInteraction
DtServiceRequestInteraction
DtResupplyRecInteraction
DtResupplyOfferInteraction
DtResupplyCancelInteraction
DtRepairResponseInteraction
DtRepairCompleteInteraction
DtRemoveObjectResultInter
DtRemoveObjectRequestInter
DtRemoveEntityInteraction
DtLgrControlInteraction
DtInteractionWithData
DtFireInteraction
DtDetonationInteraction
DtDataQueryInteraction
DtCreateObjectResultInter
DtCreateObjReqInter
DtCreateEntityInteraction
DtCollisionInteraction
DtAttributeChangeResInter
DtAttributeChangeReqInter
DtActionResponseFromObjInter
DtActionRequestToObjInter

DtAcknowledgeInteraction

Alphabetic index *Hierarchy of classes*

In file interencoder.h:

class DtInteractionEncoder

class DtInteractionEncoder: DtInteractionEncoder is used to provide encoding of an interaction into an RTI::ParameterHandleValuePairSet in an HLA (as opposed to DIS) environment

---

Inheritance:

DtInteractionEncoder

Public Methods

	DtInteractionEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc) *constructor*
virtual	**~DtInteractionEncoder** () *destructor*
	DtInteractionEncoder (const DtInteractionEncoder& orig) *copy constructor*
DtInteractionEncoder&	**operator=** (const DtInteractionEncoder& orig) *assignment operator*
typedef void	**(*DtParameterEncoder)** (const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle) *Attribute encoding function declaration*
virtual DtParameterEncoder	**addEncoder** (RTI::ParameterHandle h, DtParameterEncoder func) *Add a parameter encoder function by parameter handle*
virtual DtParameterEncoder	**addEncoder** (char* name, DtParameterEncoder proc) *Add a parameter encoder function by parameter name*
virtual DtParameterEncoder	**encoder** (int handle) const *Get the parameter encoder for the specified handle*
virtual void	**encode** (const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList) *Encode necessary values from the protocol-independent interaction*
virtual RTI::InteractionClassHandle	**interClassHandle** () *Get the object's class handle*
virtual DtInteractionEncoder*	**clone** () const *Returns a new-ed clone of this object*
virtual void	**setClassDesc** (DtInterClassDesc* desc) *Set the class descriptor of the encoder*
inline DtInteractionEncoder::DtParameterEncoder	**encoder** (int h) const
inline RTI::InteractionClassHandle	**interClassHandle** ()

- inline void **setClassDesc** (DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
  myList is an array of theMaxParametersPerInteraction items
- static const int **theMaxParametersPerInteraction**

Documentation

class DtInteractionEncoder: DtInteractionEncoder is used to provide encoding of an interaction into an RTI::ParameterHandleValuePairSet in an HLA (as opposed to DIS) environment

- `DtInteractionEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`

  constructor

- `virtual ~DtInteractionEncoder()`

  destructor

- `DtInteractionEncoder(const DtInteractionEncoder& orig)`

  copy constructor

- `DtInteractionEncoder& operator=(const DtInteractionEncoder& orig)`

  assignment operator

- `typedef void (*DtParameterEncoder)(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)`

  Attribute encoding function declaration

- `virtual DtParameterEncoder addEncoder(RTI::ParameterHandle h, DtParameterEncoder func)`

  Add a parameter encoder function by parameter handle. Returns any previous parameter encoder function which was replaced by the add.

- `virtual DtParameterEncoder addEncoder(char* name, DtParameterEncoder proc)`

  Add a parameter encoder function by parameter name. Returns any previous parameter encoder function which was replaced by the add.

- `virtual DtParameterEncoder encoder(int handle) const`

  Get the parameter encoder for the specified handle

- `virtual void encode(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)`

  Encode necessary values from the protocol-independent interaction

- **`virtual RTI::InteractionClassHandle interClassHandle()`**
  Get the object's class handle
- **`virtual DtInteractionEncoder* clone() const`**
  Returns a new-ed clone of this object
- **`virtual void setClassDesc(DtInterClassDesc* desc)`**
  Set the class descriptor of the encoder
- **`DtExerciseConn* myExConn`**
- **`DtInterClassDesc* myClassDesc`**
- **`DtVrlRtiAmbassador* myRtiAmb`**
- **`RTI::InteractionClassHandle myHandle`**
- **`DtParameterEncoder* myList`**
  myList is an array of theMaxParametersPerInteraction items
- **`static const int theMaxParametersPerInteraction`**
- **`inline DtInteractionEncoder::DtParameterEncoder encoder(int h) const`**
- **`inline RTI::InteractionClassHandle interClassHandle()`**
- **`inline void setClassDesc(DtInterClassDesc* desc)`**

Direct child classes:

DtViewControlEncoder
DtStopFreezeEncoder
DtStartResumeEncoder
DtSetDataEncoder
DtServiceRequestEncoder
DtResupplyReceivedEncoder
DtResupplyOfferEncoder
DtResupplyCancelEncoder
DtRepairResponseEncoder
DtRepairCompleteEncoder
DtRemoveObjectResultEncoder
DtRemoveObjectRequestEncoder
DtRemoveEntityEncoder
DtRawBinaryRadioSignalEncoder
DtLgrControlEncoder
DtFireEncoder
DtEventReportEncoder
DtEncodedAudioRadioSignalEncoder
DtDetonationEncoder
DtDatabaseIndexRadioSignalEncoder
DtDataQueryEncoder
DtDataEncoder
DtCreateObjectResultEncoder
DtCreateObjReqEncoder

DtCreateEntityEncoder
DtCommentEncoder
DtCollisionEncoder
DtAttributeChangeResEnc
DtAttributeChangeReqEnc
DtAppSpecificRadioSignalEncoder
DtActionResponseFromObjEncoder
DtActionResponseEncoder
DtActionRequestToObjEncoder
DtActionRequestEncoder
DtAcknowledgeEncoder

Alphabetic index *Hierarchy of classes*

In file interdecoder.h:

class DtInteractionDecoder

class DtInteractionDecoder: Instances of DtInteractionDecoder are used to decode interactions from the network

---

Inheritance:

DtInteractionDecoder

Public Methods

typedef void	**(*DtParameterDecoder)** (DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i) *Parameter decoder function signature*
	DtInteractionDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc) *constructor*
virtual	**~DtInteractionDecoder** () *destructor*
	DtInteractionDecoder (const DtInteractionDecoder& orig) *copy constructor*
DtInteractionDecoder&	**operator=** (const DtInteractionDecoder& orig) *assignment operator*
DtParameterDecoder	**addDecoder** (RTI::ParameterHandle handle, DtParameterDecoder proc) *Add/Remove a decoding function for the parameter specified by handle*
DtParameterDecoder	**addDecoder** (char* name, DtParameterDecoder proc) *Add/Remove a decoding function for the parameter specified by name*
virtual DtParameterDecoder	**decoder** (int handle) const *Get the parameter decoder for the specified handle*
virtual void	**decode** (DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList) *Decode the interaction*
virtual RTI::InteractionClassHandle	**interactionClassHandle** () *Get the interaction's class handle*
virtual DtInteractionDecoder*	**clone** () const *Returns a new-ed clone of this object*
virtual void	**setClassDesc** (DtInterClassDesc* desc) *Set the class descriptor of the decoder*
static char*	**decodingBuffer** () *Returns a pointer to a large static buffer that can be used to copy data obtained from a phvps*
inline char*	**decodingBuffer** ()

- inline RTI::InteractionClassHandle **interactionClassHandle** ()
- inline void **setClassDesc** (DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
  myList is an array of theMaxParametersPerInteraction items
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer** [2000]

Protected Methods

- virtual void **decodeParameter** (DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)
  Process (decode) the specified parameter

Documentation

class DtInteractionDecoder: Instances of DtInteractionDecoder are used to decode interactions from the network

- `typedef void (*DtParameterDecoder)(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)`

  Parameter decoder function signature

- `DtInteractionDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`

  constructor

- `virtual ~DtInteractionDecoder()`

  destructor

- `DtInteractionDecoder(const DtInteractionDecoder& orig)`

  copy constructor

- `DtInteractionDecoder& operator=(const DtInteractionDecoder& orig)`

  assignment operator

- `DtParameterDecoder addDecoder(RTI::ParameterHandle handle, DtParameterDecoder proc)`

  Add/Remove a decoding function for the parameter specified by handle. Returns the decoding function which is replaced by the add/remove. Note: This is non-virtual since it is typically called by constructors.

- `DtParameterDecoder addDecoder(char* name, DtParameterDecoder proc)`

Add/Remove a decoding function for the parameter specified by name. Returns the decoding function which is replaced by the add/remove. Note: This is non-virtual since it is typically called by constructors.

- **`virtual DtParameterDecoder decoder(int handle) const`**

  Get the parameter decoder for the specified handle

- **`virtual void decode(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)`**

  Decode the interaction

- **`virtual RTI::InteractionClassHandle interactionClassHandle()`**

  Get the interaction's class handle

- **`virtual DtInteractionDecoder* clone() const`**

  Returns a new-ed clone of this object

- **`virtual void setClassDesc(DtInterClassDesc* desc)`**

  Set the class descriptor of the decoder

- **`static char* decodingBuffer()`**

  Returns a pointer to a large static buffer that can be used to copy data obtained from a phvps.

- **`virtual void decodeParameter(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)`**

  Process (decode) the specified parameter

- **`DtExerciseConn* myExConn`**
- **`DtInterClassDesc* myClassDesc`**
- **`DtVrlRtiAmbassador* myRtiAmb`**
- **`RTI::InteractionClassHandle myClassHandle`**
- **`DtParameterDecoder* myList`**

  myList is an array of theMaxParametersPerInteraction items

- **`static const int theMaxParametersPerInteraction`**
- **`static char theDecBuffer[2000]`**
- **`inline char* decodingBuffer()`**
- **`inline RTI::InteractionClassHandle interactionClassHandle()`**
- **`inline void setClassDesc(DtInterClassDesc* desc)`**

Direct child classes:

DtViewControlDecoder
DtStopFreezeDecoder
DtStartResumeDecoder
DtSetDataDecoder
DtServiceRequestDecoder
DtResupplyReceivedDecoder
DtResupplyOfferDecoder
DtResupplyCancelDecoder

DtRepairResponseDecoder
DtRepairCompleteDecoder
DtRemoveObjectResultDecoder
DtRemoveObjectRequestDecoder
DtRemoveEntityDecoder
DtRawBinaryRadioSignalDecoder
DtLgrControlDecoder
DtFireDecoder
DtEventReportDecoder
DtEncodedAudioRadioSignalDecoder
DtDetonationDecoder
DtDatabaseIndexRadioSignalDecoder
DtDataQueryDecoder
DtDataDecoder
DtCreateObjectResultDecoder
DtCreateObjReqDecoder
DtCreateEntityDecoder
DtCommentDecoder
DtCollisionDecoder
DtAttributeChangeResDec
DtAttributeChangeReqDec
DtAppSpecificRadioSignalDecoder
DtActionResponseFromObjDecoder
DtActionResponseDecoder
DtActionRequestToObjDecoder
DtActionRequestDecoder
DtAcknowledgeDecoder

Alphabetic index *Hierarchy of classes*

In file hvcinter.h:

class DtViewControlInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtViewControlInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtViewControlInteraction () *default constructor*
virtual	**~DtViewControlInteraction** () *destructor*
	DtViewControlInteraction (const DtViewControlInteraction& orig) *copy constructor*
DtViewControlInteraction&	**operator=** (const DtViewControlInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual const DtViewControlPdu*	**vcPdu** () const *Get/Set a view control PDU*
virtual void	**setVcPdu** (const DtViewControlPdu& pdu)
virtual void	**setFromNetPacket** (const DtNetPacket* pack)
virtual const char*	**name** () const *Name of the interaction*
static DtInteraction*	**create** () *Create a DtViewControlInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtViewControlInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtViewControlInteractionCb cb, void* usr)
inline const char*	**name** () const
inline const DtViewControlPdu*	**vcPdu** () const

Protected Fields

- DtViewControlPdu* **myVcPdu**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtViewControlInteraction()`**
  default constructor
- **`virtual ~DtViewControlInteraction()`**
  destructor
- **`DtViewControlInteraction(const DtViewControlInteraction& orig)`**
  copy constructor
- **`DtViewControlInteraction& operator=(const DtViewControlInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual const DtViewControlPdu* vcPdu() const`**
  Get/Set a view control PDU
- **`virtual void setVcPdu(const DtViewControlPdu& pdu)`**
- **`virtual void setFromNetPacket(const DtNetPacket* pack)`**

- **`virtual const char* name() const`**

  Name of the interaction

- **`static DtInteraction* create()`**

  Create a DtViewControlInteraction. Caller is reponsible for deletion.

- **`static void addCallback(DtExerciseConn* conn, DtViewControlInteractionCb cb, void* usr)`**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **`static void removeCallback(DtExerciseConn* conn, DtViewControlInteractionCb cb, void* usr)`**

- **`DtViewControlPdu* myVcPdu`**

- **`inline const char* name() const`**

- **`inline const DtViewControlPdu* vcPdu() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hstopinter.h:

class DtStopFreezeInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtStopFreezeInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtStopFreezeInteraction () *default constructor*
virtual	**~DtStopFreezeInteraction** () *destructor*
	DtStopFreezeInteraction (const DtStopFreezeInteraction& orig) *copy constructor*
DtStopFreezeInteraction&	**operator=** (const DtStopFreezeInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRealWorldTime** (const DtTime time) *Set/Get real world time*
virtual DtTime	**realWorldTime** () const
virtual void	**setRequestId** (int id) *Set/Get request ID*
virtual int	**requestId** () const
virtual DtStopReason	**reason** () const *Set/Get stopReason*
virtual void	**setReason** (DtStopReason reason)
virtual DtBoolean	**reflectWhenFrozen** () const *SetGet reflectWhenFrozen flag*
virtual void	**setReflectWhenFrozen** (DtBoolean flag)

virtual DtBoolean	**runClockWhenFrozen** () const *SetGet runClockWhenFrozen flag*
virtual void	**setRunClockWhenFrozen** (DtBoolean flag)
virtual DtBoolean	**updateWhenFrozen** () const *SetGet updateWhenFrozen flag*
virtual void	**setUpdateWhenFrozen** (DtBoolean flag)
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtStopFreezeInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtStopFreezeInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtStopFreezeInteractionCb cb, void* usr)

Protected Fields

DtEntityIdentifier	**myOrigEnt**
DtEntityIdentifier	**myReceiveEnt**
int	**myRequestId**
DtTime	**myRealTime**
DtStopReason	**myReason**
DtBoolean	**myReflect**
DtBoolean	**myUpdate**
DtBoolean	**myRunClock**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const

- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtStopFreezeInteraction()`**
  default constructor
- **`virtual ~DtStopFreezeInteraction()`**
  destructor
- **`DtStopFreezeInteraction(const DtStopFreezeInteraction& orig)`**
  copy constructor
- **`DtStopFreezeInteraction& operator=(const DtStopFreezeInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setSenderId(const DtEntityIdentifier& id)`**
  Set/Get sender ID
- **`virtual const DtEntityIdentifier& senderId() const`**
- **`virtual void setReceiverId(const DtEntityIdentifier& id)`**
  Set/Get receiver ID
- **`virtual const DtEntityIdentifier& receiverId() const`**
- **`virtual void setRealWorldTime(const DtTime time)`**
  Set/Get real world time
- **`virtual DtTime realWorldTime() const`**
- **`virtual void setRequestId(int id)`**
  Set/Get request ID
- **`virtual int requestId() const`**
- **`virtual DtStopReason reason() const`**
  Set/Get stopReason
- **`virtual void setReason(DtStopReason reason)`**
- **`virtual DtBoolean reflectWhenFrozen() const`**
  SetGet reflectWhenFrozen flag
- **`virtual void setReflectWhenFrozen(DtBoolean flag)`**
- **`virtual DtBoolean runClockWhenFrozen() const`**
  SetGet runClockWhenFrozen flag
- **`virtual void setRunClockWhenFrozen(DtBoolean flag)`**
- **`virtual DtBoolean updateWhenFrozen() const`**

SetGet updateWhenFrozen flag

- **virtual void setUpdateWhenFrozen(DtBoolean flag)**
- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**

  Create a DtStopFreezeInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtStopFreezeInteractionCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtStopFreezeInteractionCb cb, void* usr)**
- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **int myRequestId**
- **DtTime myRealTime**
- **DtStopReason myReason**
- **DtBoolean myReflect**
- **DtBoolean myUpdate**
- **DtBoolean myRunClock**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hstartinter.h:

class DtStartResumeInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtStartResumeInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtStartResumeInteraction () *default constructor*
virtual	**~DtStartResumeInteraction** () *destructor*
	DtStartResumeInteraction (const DtStartResumeInteraction& orig) *copy constructor*
DtStartResumeInteraction&	**operator=** (const DtStartResumeInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRealWorldTime** (const DtTime time) *Set/Get real world time*
virtual DtTime	**realWorldTime** () const
virtual void	**setSimulationTime** (const DtTime time) *Set/Get simulation time*
virtual DtTime	**simulationTime** () const
virtual void	**setRequestId** (int id) *Set/Get request ID*
virtual int	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*

- static DtInteraction* **create** ()
  Create a DtStartResumeInteraction
- static void **addCallback** (DtExerciseConn* conn, DtStartResumeInteractionCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtStartResumeInteractionCb cb, void* usr)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- int **myRequestId**
- DtTime **myRealTime**
- DtTime **mySimTime**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const

- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtStartResumeInteraction()`**
  default constructor
- **`virtual ~DtStartResumeInteraction()`**
  destructor
- **`DtStartResumeInteraction(const DtStartResumeInteraction& orig)`**
  copy constructor
- **`DtStartResumeInteraction& operator=(const DtStartResumeInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data

- **virtual void setSenderId(const DtEntityIdentifier& id)**
  Set/Get sender ID
- **virtual const DtEntityIdentifier& senderId() const**
- **virtual void setReceiverId(const DtEntityIdentifier& id)**
  Set/Get receiver ID
- **virtual const DtEntityIdentifier& receiverId() const**
- **virtual void setRealWorldTime(const DtTime time)**
  Set/Get real world time
- **virtual DtTime realWorldTime() const**
- **virtual void setSimulationTime(const DtTime time)**
  Set/Get simulation time
- **virtual DtTime simulationTime() const**
- **virtual void setRequestId(int id)**
  Set/Get request ID
- **virtual int requestId() const**
- **virtual const char* name() const**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**
  Create a DtStartResumeInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtStartResumeInteractionCb cb, void* usr)**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtStartResumeInteractionCb cb, void* usr)**
- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **int myRequestId**
- **DtTime myRealTime**
- **DtTime mySimTime**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hsignalinter.h:

class DtSignalInteraction: public DtInteractionWithEncDec

Inheritance:

DtSignalInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtSignalInteraction () *default constructor*
virtual	**~DtSignalInteraction** () *destructor*
	DtSignalInteraction (const DtSignalInteraction& orig) *copy constructor*
DtSignalInteraction&	**operator=** (const DtSignalInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setRadioId** (const DtGlobalObjectDesignator& id) *Set/Get Host radio id*
virtual const DtGlobalObjectDesignator&	**radioId** () const
virtual void	**setSampleRate** (DtU32 rate) *Set/Get data rate*
virtual DtU32	**sampleRate** () const
virtual void	**setNumSamples** (DtU16 count) *Set/Get sample count*
virtual DtU16	**numSamples** () const
virtual void	**setNumDataBytes** (DtU16 len) *Set/Get signal data length in bytes or bits*
virtual DtU16	**numDataBytes** () const
virtual void	**setNumDataBits** (DtU16 len)
virtual DtU16	**numDataBits** () const
virtual void	**setData** (const void* data, int numBytes) *Set/Get signal data*
virtual const void*	**data** () const

virtual DtU8	**dataByte** (int byteNum) const *Set/Get the byteNum'th data byte*
virtual void	**setDataByte** (int byteNum, DtU8 data)
virtual void	**setTdlType** (DtTacticalDataLinkType link) *Set/Get tactical data link*
virtual DtTacticalDataLinkType	**tdlType** () const
virtual void	**setEncodingType** (DtEncodingType encoding) *Set/Get encoding type*
virtual DtEncodingType	**encodingType** () const
virtual DtEncodingClass	**encodingClass** () const *Set/Get encoding class*
virtual void	**setEncodingClass** (DtEncodingClass encodingClass)
virtual void	**setTdlMessageCount** (DtU16 count) *Set/Get TDL message count*
virtual DtU16	**tdlMessageCount** () const
virtual void	**setProtocolId** (DtUserProtocol id) *Set/Get user protocol id*
virtual DtUserProtocol	**protocolId** () const
virtual void	**setDatabaseIndex** (DtU32 index) *Set/Get database index*
virtual DtU32	**databaseIndex** () const
virtual void	**setDuration** (DtU32 duration) *Set/Get duration*
virtual DtU32	**duration** () const
virtual void	**setStartOffset** (DtU32 offset) *Set/Get start offset*
virtual DtU32	**startOffset** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtSignalInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtSignalInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtSignalInteractionCb cb, void* usr)

Protected Fields

DtGlobalObjectDesignator	**myHostRadioId**
DtU16	**mySignalDataLength**
DtU8*	**mySignalData** *bits*
DtTacticalDataLinkType	**myTacticalDataLink**
DtU16	**myTDLMessageCount**

- DtUserProtocol **myUserProtocolId**
- DtU32 **myDatabaseIndex**
- DtU32 **myDuration**
- DtU32 **myStartOffset**
- DtEncodingType **myEncodingType**
- DtEncodingClass **myEncodingClass**
- DtU32 **mySampleRate**
- DtU16 **mySampleCount**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const

- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtSignalInteraction()`**
  default constructor
- **`virtual ~DtSignalInteraction()`**
  destructor
- **`DtSignalInteraction(const DtSignalInteraction& orig)`**
  copy constructor
- **`DtSignalInteraction& operator=(const DtSignalInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setRadioId(const DtGlobalObjectDesignator& id)`**
  Set/Get Host radio id
- **`virtual const DtGlobalObjectDesignator& radioId() const`**
- **`virtual void setSampleRate(DtU32 rate)`**

Set/Get data rate

- **virtual DtU32 sampleRate() const**
- **virtual void setNumSamples(DtU16 count)**

  Set/Get sample count
- **virtual DtU16 numSamples() const**
- **virtual void setNumDataBytes(DtU16 len)**

  Set/Get signal data length in bytes or bits
- **virtual DtU16 numDataBytes() const**
- **virtual void setNumDataBits(DtU16 len)**
- **virtual DtU16 numDataBits() const**
- **virtual void setData(const void* data, int numBytes)**

  Set/Get signal data
- **virtual const void* data() const**
- **virtual DtU8 dataByte(int byteNum) const**

  Set/Get the byteNum'th data byte
- **virtual void setDataByte(int byteNum, DtU8 data)**
- **virtual void setTdlType(DtTacticalDataLinkType link)**

  Set/Get tactical data link
- **virtual DtTacticalDataLinkType tdlType() const**
- **virtual void setEncodingType(DtEncodingType encoding)**

  Set/Get encoding type
- **virtual DtEncodingType encodingType() const**
- **virtual DtEncodingClass encodingClass() const**

  Set/Get encoding class
- **virtual void setEncodingClass(DtEncodingClass encodingClass)**
- **virtual void setTdlMessageCount(DtU16 count)**

  Set/Get TDL message count
- **virtual DtU16 tdlMessageCount() const**
- **virtual void setProtocolId(DtUserProtocol id)**

  Set/Get user protocol id
- **virtual DtUserProtocol protocolId() const**
- **virtual void setDatabaseIndex(DtU32 index)**

  Set/Get database index
- **virtual DtU32 databaseIndex() const**

- **virtual void setDuration(DtU32 duration)**
  Set/Get duration
- **virtual DtU32 duration() const**
- **virtual void setStartOffset(DtU32 offset)**
  Set/Get start offset
- **virtual DtU32 startOffset() const**
- **virtual const char* name() const**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**
  Create a DtSignalInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtSignalInteractionCb cb, void* usr)**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtSignalInteractionCb cb, void* usr)**
- **DtGlobalObjectDesignator myHostRadioId**
- **DtU16 mySignalDataLength**
- **DtU8* mySignalData**
  bits
- **DtTacticalDataLinkType myTacticalDataLink**
- **DtU16 myTDLMessageCount**
- **DtUserProtocol myUserProtocolId**
- **DtU32 myDatabaseIndex**
- **DtU32 myDuration**
- **DtU32 myStartOffset**
- **DtEncodingType myEncodingType**
- **DtEncodingClass myEncodingClass**
- **DtU32 mySampleRate**
- **DtU16 mySampleCount**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hservreqinter.h:

class DtServiceRequestInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtServiceRequestInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtServiceRequestInteraction () *default constructor*
virtual	**~DtServiceRequestInteraction** () *destructor*
	DtServiceRequestInteraction (const DtServiceRequestInteraction& orig) *copy constructor*
DtServiceRequestInteraction&	**operator=** (const DtServiceRequestInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setRequestingId** (const DtGlobalObjectDesignator &requester) *Set/Get the receiving object*
virtual const DtGlobalObjectDesignator&	**requestingId** () const
virtual void	**setServicingId** (const DtGlobalObjectDesignator &servicer) *Set/Get the repairing object*
virtual const DtGlobalObjectDesignator&	**servicingId** () const
virtual void	**setServiceType** (DtServiceType type) *Set/Get the service type*
virtual DtServiceType	**serviceType** () const
virtual void	**setSupplies** (const void* data, int numBytes) *Set/Get the supplies memory block all at once*
virtual const void*	**supplies** () const
virtual void	**setNumSupplyTypes** (int numTypes) *Supporting functions to get/set the Supply Array*
virtual int	**numSupplyTypes** () const
virtual void	**setNthSupply** (int supplyIndex, const DtEntityType &supply, float quantity, int *retCode = NULL)

virtual const DtEntityType&	**nthSupply** (int supplyIndex, float *quantity = NULL, int *retCode = NULL) const
virtual float	**supplyQuantity** (const DtEntityType &supply) const
virtual void	**changeSuppliesSize** (int newSize)
virtual int	**indexInBounds** (int ind) const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtServiceRequestInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtServiceRequestInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtServiceRequestInteractionCb cb, void* usr)
inline const char*	**name** () const
inline DtServiceType	**serviceType** () const
inline void	**setRequestingId** (const DtGlobalObjectDesignator &id)
inline void	**setServiceType** (DtServiceType type)
inline void	**setServicingId** (const DtGlobalObjectDesignator &id)

Protected Fields

DtGlobalObjectDesignator	**myRequestingObject**
DtGlobalObjectDesignator	**myServicingObject**
DtServiceType	**myServiceType**
int	**myNumSupplyTypes**
DtNetSupplyQuantity*	**mySupplies**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const

- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)

- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtServiceRequestInteraction()`**
  default constructor
- **`virtual ~DtServiceRequestInteraction()`**
  destructor
- **`DtServiceRequestInteraction(const DtServiceRequestInteraction& orig)`**
  copy constructor
- **`DtServiceRequestInteraction& operator=( const DtServiceRequestInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setRequestingId(const DtGlobalObjectDesignator &requester)`**
  Set/Get the receiving object
- **`virtual const DtGlobalObjectDesignator& requestingId() const`**
- **`virtual void setServicingId(const DtGlobalObjectDesignator &servicer)`**
  Set/Get the repairing object
- **`virtual const DtGlobalObjectDesignator& servicingId() const`**
- **`virtual void setServiceType(DtServiceType type)`**
  Set/Get the service type
- **`virtual DtServiceType serviceType() const`**
- **`virtual void setSupplies(const void* data, int numBytes)`**
  Set/Get the supplies memory block all at once
- **`virtual const void* supplies() const`**
- **`virtual void setNumSupplyTypes(int numTypes)`**
  Supporting functions to get/set the Supply Array
- **`virtual int numSupplyTypes() const`**
- **`virtual void setNthSupply(int supplyIndex, const DtEntityType &supply, float quantity, int *retCode = NULL)`**
- **`virtual const DtEntityType& nthSupply(int supplyIndex, float *quantity = NULL, int *retCode = NULL) const`**
- **`virtual float supplyQuantity(const DtEntityType &supply) const`**
- **`virtual void changeSuppliesSize(int newSize)`**
- **`virtual int indexInBounds(int ind) const`**
- **`virtual const char* name() const`**

Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtServiceRequestInteraction. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtServiceRequestInteractionCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtServiceRequestInteractionCb cb, void* usr)**
- **DtGlobalObjectDesignator myRequestingObject**
- **DtGlobalObjectDesignator myServicingObject**
- **DtServiceType myServiceType**
- **int myNumSupplyTypes**
- **DtNetSupplyQuantity* mySupplies**
- **inline const char* name() const**
- **inline DtServiceType serviceType() const**
- **inline void setRequestingId(const DtGlobalObjectDesignator &id)**
- **inline void setServiceType(DtServiceType type)**
- **inline void setServicingId(const DtGlobalObjectDesignator &id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hresrecinter.h:

class DtResupplyRecInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtResupplyRecInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtResupplyRecInteraction () *default constructor*
virtual	**~DtResupplyRecInteraction** () *destructor*
	DtResupplyRecInteraction (const DtResupplyRecInteraction& orig) *copy constructor*
DtResupplyRecInteraction&	**operator=** (const DtResupplyRecInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setReceivingId** (const DtGlobalObjectDesignator &requester) *Set/Get the receiving object*
virtual const DtGlobalObjectDesignator&	**receivingId** () const
virtual void	**setSupplyingId** (const DtGlobalObjectDesignator &servicer) *Set/Get the repairing object*
virtual const DtGlobalObjectDesignator&	**supplyingId** () const
virtual void	**setSupplies** (const void* data, int numBytes) *Set/Get the supplies memory block all at once*
virtual const void*	**supplies** () const
virtual void	**setNumSupplyTypes** (int numTypes) *Supporting functions to get/set the Supply Array*
virtual int	**numSupplyTypes** () const
virtual void	**setNthSupply** (int supplyIndex, const DtEntityType &supply, float quantity, int *retCode = NULL)
virtual const DtEntityType&	**nthSupply** (int supplyIndex, float *quantity = NULL, int *retCode = NULL) const
virtual float	**supplyQuantity** (const DtEntityType &supply) const
virtual void	**changeSuppliesSize** (int newSize)

- virtual int **indexInBounds** (int ind) const
- virtual const char* **name** () const
  Name of the interaction (Not nec
- static DtInteraction* **create** ()
  Create a DtResupplyRecInteraction
- static void **addCallback** (DtExerciseConn* conn, DtResupplyRecInteractionCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtResupplyRecInteractionCb cb, void* usr)
- inline const char* **name** () const
- inline void **setReceivingId** (const DtGlobalObjectDesignator &id)
- inline void **setSupplyingId** (const DtGlobalObjectDesignator &id)

Protected Fields

- DtGlobalObjectDesignator **myReceivingObject**
- DtGlobalObjectDesignator **mySupplyingObject**
- int **myNumSupplyTypes**
- DtNetSupplyQuantity* **mySupplies**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const

- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtResupplyRecInteraction()`**
  default constructor
- **`virtual ~DtResupplyRecInteraction()`**
  destructor
- **`DtResupplyRecInteraction(const DtResupplyRecInteraction& orig)`**
  copy constructor
- **`DtResupplyRecInteraction& operator=(const DtResupplyRecInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setReceivingId(const DtGlobalObjectDesignator &requester)`**
  Set/Get the receiving object
- **`virtual const DtGlobalObjectDesignator& receivingId() const`**
- **`virtual void setSupplyingId(const DtGlobalObjectDesignator &servicer)`**
  Set/Get the repairing object
- **`virtual const DtGlobalObjectDesignator& supplyingId() const`**
- **`virtual void setSupplies(const void* data, int numBytes)`**
  Set/Get the supplies memory block all at once
- **`virtual const void* supplies() const`**
- **`virtual void setNumSupplyTypes(int numTypes)`**
  Supporting functions to get/set the Supply Array
- **`virtual int numSupplyTypes() const`**
- **`virtual void setNthSupply(int supplyIndex, const DtEntityType &supply, float quantity, int *retCode = NULL)`**
- **`virtual const DtEntityType& nthSupply(int supplyIndex, float *quantity = NULL, int *retCode = NULL) const`**
- **`virtual float supplyQuantity(const DtEntityType &supply) const`**
- **`virtual void changeSuppliesSize(int newSize)`**
- **`virtual int indexInBounds(int ind) const`**
- **`virtual const char* name() const`**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **`static DtInteraction* create()`**
  Create a DtResupplyRecInteraction. Caller is reponsible for deletion.

- **`static void addCallback(DtExerciseConn* conn, DtResupplyRecInteractionCb cb, void* usr)`**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **`static void removeCallback(DtExerciseConn* conn, DtResupplyRecInteractionCb cb, void* usr)`**
- **`DtGlobalObjectDesignator myReceivingObject`**
- **`DtGlobalObjectDesignator mySupplyingObject`**
- **`int myNumSupplyTypes`**
- **`DtNetSupplyQuantity* mySupplies`**
- **`inline const char* name() const`**
- **`inline void setReceivingId( const DtGlobalObjectDesignator &id)`**
- **`inline void setSupplyingId( const DtGlobalObjectDesignator &id)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hresoffinter.h:

class DtResupplyOfferInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtResupplyOfferInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtResupplyOfferInteraction () *default constructor*
virtual	**~DtResupplyOfferInteraction** () *destructor*
	DtResupplyOfferInteraction (const DtResupplyOfferInteraction& orig) *copy constructor*
DtResupplyOfferInteraction&	**operator=** (const DtResupplyOfferInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setReceivingId** (const DtGlobalObjectDesignator &receiver) *Set/Get the receiving object*
virtual const DtGlobalObjectDesignator&	**receivingId** () const
virtual void	**setSupplyingId** (const DtGlobalObjectDesignator &supplier) *Set/Get the repairing object*
virtual const DtGlobalObjectDesignator&	**supplyingId** () const
virtual void	**setSupplies** (const void* data, int numBytes) *Set/Get the supplies memory block all at once*
virtual const void*	**supplies** () const
virtual void	**setNumSupplyTypes** (int numTypes) *Supporting functions to get/set the Supply Array*
virtual int	**numSupplyTypes** () const
virtual void	**setNthSupply** (int supplyIndex, const DtEntityType &supply, float quantity, int *retCode = NULL)
virtual const DtEntityType&	**nthSupply** (int supplyIndex, float *quantity = NULL, int *retCode = NULL) const
virtual float	**supplyQuantity** (const DtEntityType &supply) const

- virtual void **changeSuppliesSize** (int newSize)
- virtual int **indexInBounds** (int ind) const
- virtual const char* **name** () const
  Name of the interaction (Not nec
- static DtInteraction* **create** ()
  Create a DtResupplyOfferInteraction
- static void **addCallback** (DtExerciseConn* conn, DtResupplyOfferInteractionCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtResupplyOfferInteractionCb cb, void* usr)
- inline const char* **name** () const
- inline void **setReceivingId** (const DtGlobalObjectDesignator &receiver)
- inline void **setSupplyingId** (const DtGlobalObjectDesignator &supplier)

Protected Fields

- DtGlobalObjectDesignator **myReceivingObject**
- DtGlobalObjectDesignator **mySupplyingObject**
- int **myNumSupplyTypes**
- DtNetSupplyQuantity* **mySupplies**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const

- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtResupplyOfferInteraction()`**
  default constructor
- **`virtual ~DtResupplyOfferInteraction()`**
  destructor
- **`DtResupplyOfferInteraction(const DtResupplyOfferInteraction& orig)`**
  copy constructor
- **`DtResupplyOfferInteraction& operator=( const DtResupplyOfferInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setReceivingId(const DtGlobalObjectDesignator &receiver)`**
  Set/Get the receiving object
- **`virtual const DtGlobalObjectDesignator& receivingId() const`**
- **`virtual void setSupplyingId(const DtGlobalObjectDesignator &supplier)`**
  Set/Get the repairing object
- **`virtual const DtGlobalObjectDesignator& supplyingId() const`**
- **`virtual void setSupplies(const void* data, int numBytes)`**
  Set/Get the supplies memory block all at once
- **`virtual const void* supplies() const`**
- **`virtual void setNumSupplyTypes(int numTypes)`**
  Supporting functions to get/set the Supply Array
- **`virtual int numSupplyTypes() const`**
- **`virtual void setNthSupply(int supplyIndex, const DtEntityType &supply, float quantity, int *retCode = NULL)`**
- **`virtual const DtEntityType& nthSupply(int supplyIndex, float *quantity = NULL, int *retCode = NULL) const`**
- **`virtual float supplyQuantity(const DtEntityType &supply) const`**
- **`virtual void changeSuppliesSize(int newSize)`**
- **`virtual int indexInBounds(int ind) const`**
- **`virtual const char* name() const`**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **`static DtInteraction* create()`**
  Create a DtResupplyOfferInteraction. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtResupplyOfferInteractionCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtResupplyOfferInteractionCb cb, void* usr)**
- **DtGlobalObjectDesignator myReceivingObject**
- **DtGlobalObjectDesignator mySupplyingObject**
- **int myNumSupplyTypes**
- **DtNetSupplyQuantity* mySupplies**
- **inline const char* name() const**
- **inline void setReceivingId(const DtGlobalObjectDesignator &receiver)**
- **inline void setSupplyingId(const DtGlobalObjectDesignator &supplier)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrescancinter.h:

class DtResupplyCancelInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtResupplyCancelInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtResupplyCancelInteraction () *default constructor*
virtual	**~DtResupplyCancelInteraction** () *destructor*
	DtResupplyCancelInteraction (const DtResupplyCancelInteraction& orig) *copy constructor*
DtResupplyCancelInteraction&	**operator=** (const DtResupplyCancelInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setReceivingId** (const DtGlobalObjectDesignator &requester) *Set/Get the receiving object*
virtual const DtGlobalObjectDesignator&	**receivingId** () const
virtual void	**setSupplyingId** (const DtGlobalObjectDesignator &servicer) *Set/Get the supplying object*
virtual const DtGlobalObjectDesignator&	**supplyingId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtResupplyCancelInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtResupplyCancelInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtResupplyCancelInteractionCb cb, void* usr)
inline const char*	**name** () const

- inline void **setReceivingId** (const DtGlobalObjectDesignator &id)
- inline void **setSupplyingId** (const DtGlobalObjectDesignator &id)

Protected Fields

- DtGlobalObjectDesignator **myReceivingObject**
- DtGlobalObjectDesignator **mySupplyingObject**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtResupplyCancelInteraction()`**
  default constructor
- **`virtual ~DtResupplyCancelInteraction()`**
  destructor
- **`DtResupplyCancelInteraction(const DtResupplyCancelInteraction& orig)`**
  copy constructor
- **`DtResupplyCancelInteraction& operator=( const DtResupplyCancelInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setReceivingId(const DtGlobalObjectDesignator &requester)`**
  Set/Get the receiving object
- **`virtual const DtGlobalObjectDesignator& receivingId() const`**
- **`virtual void setSupplyingId(const DtGlobalObjectDesignator &servicer)`**
  Set/Get the supplying object

- **virtual const DtGlobalObjectDesignator& supplyingId() const**
- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**

  Create a DtResupplyCancelInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtResupplyCancelInteractionCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtResupplyCancelInteractionCb cb, void* usr)**
- **DtGlobalObjectDesignator myReceivingObject**
- **DtGlobalObjectDesignator mySupplyingObject**
- **inline const char* name() const**
- **inline void setReceivingId(const DtGlobalObjectDesignator &id)**
- **inline void setSupplyingId(const DtGlobalObjectDesignator &id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrepinter.h:

class DtRepairResponseInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtRepairResponseInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtRepairResponseInteraction () *default constructor*
virtual	**~DtRepairResponseInteraction** () *destructor*
	DtRepairResponseInteraction (const DtRepairResponseInteraction& orig) *copy constructor*
DtRepairResponseInteraction&	**operator=** (const DtRepairResponseInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setReceivingId** (const DtGlobalObjectDesignator &id) *Set/Get the receiving object*
virtual const DtGlobalObjectDesignator&	**receivingId** () const
virtual void	**setRepairingId** (const DtGlobalObjectDesignator &id) *Set/Get the repairing object*
virtual const DtGlobalObjectDesignator&	**repairingId** () const
virtual void	**setRepairResult** (DtRepairResult result) *Set/Get the repair type*
virtual DtRepairResult	**repairResult** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtRepairResponseInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtRepairResponseInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*

- static void **removeCallback** (DtExerciseConn* conn, DtRepairResponseInteractionCb cb, void* usr)
- inline const char* **name** () const
- inline DtRepairResult **repairResult** () const
- inline void **setReceivingId** (const DtGlobalObjectDesignator &id)
- inline void **setRepairResult** (DtRepairResult r)
- inline void **setRepairingId** (const DtGlobalObjectDesignator &id)

Protected Fields

- DtGlobalObjectDesignator **myReceivingObject**
- DtGlobalObjectDesignator **myRepairingObject**
- DtRepairResult **myRepairResult**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const

- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtRepairResponseInteraction()`**
  default constructor
- **`virtual ~DtRepairResponseInteraction()`**
  destructor
- **`DtRepairResponseInteraction(const DtRepairResponseInteraction& orig)`**
  copy constructor
- **`DtRepairResponseInteraction& operator=( const DtRepairResponseInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data

- **virtual void setReceivingId(const DtGlobalObjectDesignator &id)**
  Set/Get the receiving object
- **virtual const DtGlobalObjectDesignator& receivingId() const**
- **virtual void setRepairingId(const DtGlobalObjectDesignator &id)**
  Set/Get the repairing object
- **virtual const DtGlobalObjectDesignator& repairingId() const**
- **virtual void setRepairResult(DtRepairResult result)**
  Set/Get the repair type
- **virtual DtRepairResult repairResult() const**
- **virtual const char* name() const**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**
  Create a DtRepairResponseInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtRepairResponseInteractionCb cb, void* usr)**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtRepairResponseInteractionCb cb, void* usr)**
- **DtGlobalObjectDesignator myReceivingObject**
- **DtGlobalObjectDesignator myRepairingObject**
- **DtRepairResult myRepairResult**
- **inline const char* name() const**
- **inline DtRepairResult repairResult() const**
- **inline void setReceivingId(const DtGlobalObjectDesignator &id)**
- **inline void setRepairResult(DtRepairResult r)**
- **inline void setRepairingId(const DtGlobalObjectDesignator &id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrepcompinter.h:

class DtRepairCompleteInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtRepairCompleteInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtRepairCompleteInteraction () *default constructor*
virtual	**~DtRepairCompleteInteraction** () *destructor*
	DtRepairCompleteInteraction (const DtRepairCompleteInteraction& orig) *copy constructor*
DtRepairCompleteInteraction&	**operator=** (const DtRepairCompleteInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setReceivingId** (const DtGlobalObjectDesignator &id) *Set/Get the receiving object*
virtual const DtGlobalObjectDesignator&	**receivingId** () const
virtual void	**setRepairingId** (const DtGlobalObjectDesignator &id) *Set/Get the repairing object*
virtual const DtGlobalObjectDesignator&	**repairingId** () const
virtual void	**setRepairType** (DtRepairType type) *Set/Get the repair type*
virtual DtRepairType	**repairType** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtRepairCompleteInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtRepairCompleteInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*

- static void **removeCallback** (DtExerciseConn* conn, DtRepairCompleteInteractionCb cb, void* usr)
- inline const char* **name** () const
- inline DtRepairType **repairType** () const
- inline void **setReceivingId** (const DtGlobalObjectDesignator &id)
- inline void **setRepairType** (DtRepairType t)
- inline void **setRepairingId** (const DtGlobalObjectDesignator &id)

Protected Fields

- DtGlobalObjectDesignator **myReceivingObject**
- DtGlobalObjectDesignator **myRepairingObject**
- DtRepairType **myRepairType**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const

- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtRepairCompleteInteraction()`**
  default constructor
- **`virtual ~DtRepairCompleteInteraction()`**
  destructor
- **`DtRepairCompleteInteraction(const DtRepairCompleteInteraction& orig)`**
  copy constructor
- **`DtRepairCompleteInteraction& operator=( const DtRepairCompleteInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data

- **virtual void setReceivingId(const DtGlobalObjectDesignator &id)**
  Set/Get the receiving object
- **virtual const DtGlobalObjectDesignator& receivingId() const**
- **virtual void setRepairingId(const DtGlobalObjectDesignator &id)**
  Set/Get the repairing object
- **virtual const DtGlobalObjectDesignator& repairingId() const**
- **virtual void setRepairType(DtRepairType type)**
  Set/Get the repair type
- **virtual DtRepairType repairType() const**
- **virtual const char* name() const**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**
  Create a DtRepairCompleteInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtRepairCompleteInteractionCb cb, void* usr)**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtRepairCompleteInteractionCb cb, void* usr)**
- **DtGlobalObjectDesignator myReceivingObject**
- **DtGlobalObjectDesignator myRepairingObject**
- **DtRepairType myRepairType**
- **inline const char* name() const**
- **inline DtRepairType repairType() const**
- **inline void setReceivingId(const DtGlobalObjectDesignator &id)**
- **inline void setRepairType(DtRepairType t)**
- **inline void setRepairingId(const DtGlobalObjectDesignator &id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremobjresint.h:

class DtRemoveObjectResultInter: public DtInteractionWithEncDec

---

Inheritance:

DtRemoveObjectResultInter - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtRemoveObjectResultInter () *default constructor*
virtual	**~DtRemoveObjectResultInter** () *destructor*
	DtRemoveObjectResultInter (const DtRemoveObjectResultInter& orig) *copy constructor*
DtRemoveObjectResultInter&	**operator=** (const DtRemoveObjectResultInter& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setRemoveObjectResult** (DtRemoveObjectResult result) *Set/Get create object result*
virtual DtRemoveObjectResult	**removeObjectResult** () const
virtual void	**setRequestId** (DtU32 id) *Set/Get request ID*
virtual DtU32	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtRemoveObjectResultInter*
static void	**addCallback** (DtExerciseConn* conn, DtRemoveObjectResultInterCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtRemoveObjectResultInterCb cb, void* usr)
inline const char*	**name** () const
inline DtU32	**requestId** () const

- inline void **setRemoveObjectResult** (DtRemoveObjectResult result)
- inline void **setRequestId** (DtU32 id)

Protected Fields

- DtRemoveObjectResult **myRemoveObjectResult**
- DtU32 **myRequestId**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const

- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtRemoveObjectResultInter()`**
  default constructor
- **`virtual ~DtRemoveObjectResultInter()`**
  destructor
- **`DtRemoveObjectResultInter(const DtRemoveObjectResultInter& orig)`**
  copy constructor
- **`DtRemoveObjectResultInter& operator=( const DtRemoveObjectResultInter& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setRemoveObjectResult(DtRemoveObjectResult result)`**
  Set/Get create object result
- **`virtual DtRemoveObjectResult removeObjectResult() const`**
- **`virtual void setRequestId(DtU32 id)`**

Set/Get request ID

- **virtual DtU32 requestId() const**
- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtRemoveObjectResultInter. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtRemoveObjectResultInterCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtRemoveObjectResultInterCb cb, void* usr)**
- **DtRemoveObjectResult myRemoveObjectResult**
- **DtU32 myRequestId**
- **static char* theClassName**
- **inline const char* name() const**
- **inline DtU32 requestId() const**
- **inline void setRemoveObjectResult(DtRemoveObjectResult result)**
- **inline void setRequestId(DtU32 id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremobjreqint.h:

class DtRemoveObjectRequestInter: public DtInteractionWithEncDec

---

Inheritance:

DtRemoveObjectRequestInter - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtRemoveObjectRequestInter () *default constructor*
virtual	**~DtRemoveObjectRequestInter** () *destructor*
	DtRemoveObjectRequestInter (const DtRemoveObjectRequestInter& orig) *copy constructor*
DtRemoveObjectRequestInter&	**operator=** (const DtRemoveObjectRequestInter& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual const DtGlobalObjectDesignatorList&	**objectIDs** () const *Return the Object ID list*
virtual DtGlobalObjectDesignatorList*	**objectIDs** ()
virtual void	**setNumIds** (DtU8 numIds) *Set/Get number of IDs*
virtual DtU8	**numIds** () const
virtual void	**setNthObjectId** (DtInt32 idIndex, const DtGlobalObjectDesignator& id) *Set/Get the nth ObjectId*
virtual const DtGlobalObjectDesignator&	**nthObjectId** (DtInt32 idIndex) const
virtual void	**setRequestId** (DtU32 id) *Set/Get request ID*
virtual DtU32	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtRemoveObjectRequestInter*

- static void — **addCallback** (DtExerciseConn* conn, DtRemoveObjectRequestInterCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void — **removeCallback** (DtExerciseConn* conn, DtRemoveObjectRequestInterCb cb, void* usr)
- inline const char* — **name** () const
- inline DtU32 — **requestId** () const
- inline void — **setRequestId** (DtU32 id)

Protected Fields

- DtGlobalObjectDesignatorList — **myObjectList**
- DtU32 — **myRequestId**
- static char* — **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const

- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtRemoveObjectRequestInter()`**
  default constructor
- **`virtual ~DtRemoveObjectRequestInter()`**
  destructor
- **`DtRemoveObjectRequestInter(const DtRemoveObjectRequestInter& orig)`**
  copy constructor
- **`DtRemoveObjectRequestInter& operator=( const DtRemoveObjectRequestInter& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data

- **virtual const DtGlobalObjectDesignatorList& objectIDs() const**

  Return the Object ID list

- **virtual DtGlobalObjectDesignatorList* objectIDs()**
- **virtual void setNumIds(DtU8 numIds)**

  Set/Get number of IDs

- **virtual DtU8 numIds() const**
- **virtual void setNthObjectId(DtInt32 idIndex, const DtGlobalObjectDesignator& id)**

  Set/Get the nth ObjectId

- **virtual const DtGlobalObjectDesignator& nthObjectId(DtInt32 idIndex) const**
- **virtual void setRequestId(DtU32 id)**

  Set/Get request ID

- **virtual DtU32 requestId() const**
- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtRemoveObjectRequestInter. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtRemoveObjectRequestInterCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtRemoveObjectRequestInterCb cb, void* usr)**
- **DtGlobalObjectDesignatorList myObjectList**
- **DtU32 myRequestId**
- **static char* theClassName**
- **inline const char* name() const**
- **inline DtU32 requestId() const**
- **inline void setRequestId(DtU32 id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremoventint.h:

class DtRemoveEntityInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtRemoveEntityInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtRemoveEntityInteraction () *default constructor*
virtual	**~DtRemoveEntityInteraction** () *destructor*
	DtRemoveEntityInteraction (const DtRemoveEntityInteraction& orig) *copy constructor*
DtRemoveEntityInteraction&	**operator=** (const DtRemoveEntityInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRequestId** (int id) *Set/Get request ID*
virtual int	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtRemoveEntityInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtRemoveEntityInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtRemoveEntityInteractionCb cb, void* usr)
inline const char*	**name** () const

- inline const DtEntityIdentifier& **receiverId** () const
- inline int **requestId** () const
- inline const DtEntityIdentifier& **senderId** () const
- inline void **setReceiverId** (const DtEntityIdentifier& id)
- inline void **setRequestId** (int id)
- inline void **setSenderId** (const DtEntityIdentifier& id)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- int **myRequestId**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const

- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtRemoveEntityInteraction()`**
  default constructor
- **`virtual ~DtRemoveEntityInteraction()`**
  destructor
- **`DtRemoveEntityInteraction(const DtRemoveEntityInteraction& orig)`**
  copy constructor
- **`DtRemoveEntityInteraction& operator=(const DtRemoveEntityInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data

- **virtual void setSenderId(const DtEntityIdentifier& id)**

  Set/Get sender ID

- **virtual const DtEntityIdentifier& senderId() const**
- **virtual void setReceiverId(const DtEntityIdentifier& id)**

  Set/Get receiver ID

- **virtual const DtEntityIdentifier& receiverId() const**
- **virtual void setRequestId(int id)**

  Set/Get request ID

- **virtual int requestId() const**
- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtRemoveEntityInteraction. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtRemoveEntityInteractionCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtRemoveEntityInteractionCb cb, void* usr)**
- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **int myRequestId**
- **static char* theClassName**
- **inline const char* name() const**
- **inline const DtEntityIdentifier& receiverId() const**
- **inline int requestId() const**
- **inline const DtEntityIdentifier& senderId() const**
- **inline void setReceiverId(const DtEntityIdentifier& id)**
- **inline void setRequestId(int id)**
- **inline void setSenderId(const DtEntityIdentifier& id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hlcinter.h:

class DtLgrControlInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtLgrControlInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtLgrControlInteraction () *default constructor*
virtual	**~DtLgrControlInteraction** () *destructor*
	DtLgrControlInteraction (const DtLgrControlInteraction& orig) *copy constructor*
DtLgrControlInteraction&	**operator=** (const DtLgrControlInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual const DtLgrControlPdu*	**lcPdu** () const *Get/Set a logger control PDU*
virtual void	**setLcPdu** (const DtLgrControlPdu& pdu)
virtual void	**setFromNetPacket** (const DtNetPacket* pack)
virtual const char*	**name** () const *Name of the interaction*
static DtInteraction*	**create** () *Create a DtLgrControlInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtLgrControlInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtLgrControlInteractionCb cb, void* usr)
inline const DtLgrControlPdu*	**lcPdu** () const
inline const char*	**name** () const

Protected Fields

- DtLgrControlPdu* **myLcPdu**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtLgrControlInteraction()`**
  default constructor
- **`virtual ~DtLgrControlInteraction()`**
  destructor
- **`DtLgrControlInteraction(const DtLgrControlInteraction& orig)`**
  copy constructor
- **`DtLgrControlInteraction& operator=(const DtLgrControlInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual const DtLgrControlPdu* lcPdu() const`**
  Get/Set a logger control PDU
- **`virtual void setLcPdu(const DtLgrControlPdu& pdu)`**
- **`virtual void setFromNetPacket(const DtNetPacket* pack)`**

- **`virtual const char* name() const`**

  Name of the interaction

- **`static DtInteraction* create()`**

  Create a DtLgrControlInteraction. Caller is reponsible for deletion.

- **`static void addCallback(DtExerciseConn* conn, DtLgrControlInteractionCb cb, void* usr)`**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **`static void removeCallback(DtExerciseConn* conn, DtLgrControlInteractionCb cb, void* usr)`**

- **`DtLgrControlPdu* myLcPdu`**

- **`inline const DtLgrControlPdu* lcPdu() const`**

- **`inline const char* name() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hinterwdata.h:

class DtInteractionWithData: public DtInteractionWithEncDec

class DtInteractionWithData: Instances of DtInteractionWithData are used to represent RPR FOM Interactions that use fixed and var datum records

---

Inheritance:

DtInteractionWithData - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtInteractionWithData () *default constructor*
virtual	**~DtInteractionWithData** () *destructor*
	DtInteractionWithData (const DtInteractionWithData& orig) *copy constructor*
DtInteractionWithData&	**operator=** (const DtInteractionWithData& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual int	**numFixedFields** () const
virtual int	**numVarFields** () const *returns number of variable length fields*
virtual DtDatumParam	**varDatumId** (int index) const *Returns the datum param found in the index'th variable datum record*
virtual DtDatumParam	**fixedDatumId** (int index) const *Returns the datum param found in the index'th fixed datum record*
virtual DtDatumParam	**datumId** (DtDatumRecordType fixedOrVar, int index) const *Returns the datum param found in the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
virtual int	**datumSizeBits** (DtDatumRecordType fixedOrVar, int index) const *Returns the number of bits of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
virtual int	**datumSizeBytes** (DtDatumRecordType fixedOrVar, int index) const *Returns the number of bytes of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*

- virtual void — **getIndex** (DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const
  Looks through all fixed and variable records for one containing param as the parameter ID
- virtual DtFloat32 — **datumValFloat32** (DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
  ******* INSPECTING DATUM VALUES BY INDEX ********* If you know the data type of the value contained in a particular record, use one of the following to return it*
- virtual DtFloat64 — **datumValFloat64** (DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long — **datumValUnsigned32** (DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual long — **datumValInt32** (DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual const char* — **datumValByteArray** (DtDatumRecordType fixedOrVar, int index) const
- virtual DtFloat32 — **datumValFloat32** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
  ******* INSPECTING DATUM VALUES BY PARAM ID ********* If you know the data type of the value contained in a particular record, use one of the following to return it*
- virtual DtFloat64 — **datumValFloat64** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long — **datumValUnsigned32** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual long — **datumValInt32** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual const char* — **datumValByteArray** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual void — **initDataSection** (const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)
  Initialize to reflect the fixed parameters and variable parameters and sizes indicated
- virtual void — **setNumFixedFields** (int newNum)
  Sets the number of fixed or variable fields to newNum
- virtual void — **setNumVarFields** (int newNum)
- virtual void — **setDatumParam** (DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)
  Set the datum ID in the index'th fixed or variable datum record to param, depending on whether fixedOrVar is DtFixed or DtVar
- virtual void — **setVarDataBytes** (int index, int length, DtDatumRetCode* retcode = NULL)
  Allocates length bytes for the index'th variable datum record
- virtual void — **setDatumValFloat64** (DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)
  ******* SETTING DATA VALUES BY INDEX ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data*
- virtual void — **setDatumValFloat32** (DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void — **setDatumValInt32** (DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)
- virtual void — **setDatumValUnsigned32** (DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)

- virtual void **setDatumValByteArray** (DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64** (DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)
  ****** SETTING DATA VALUES BY PARAM ID ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data*
- virtual void **setDatumValFloat32** (DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32** (DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32** (DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray** (DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)
- virtual DtDatumRecordSet* **datumRecSet** ()
  Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records
- virtual const DtDatumRecordSet* **datumRecSet** () const

Public

- ****************** *MUTATOR FUNCTIONS* ******************

Protected Fields

- DtDatumRecordSet **myRecSet**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual const char* **name**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const

- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static void **addCallback**(DtExerciseConn* conn, DtList* classes, DtReceiveInteractionCb cb, void* usr)
- static void **removeCallback**(DtExerciseConn* conn, DtList* classes, DtReceiveInteractionCb cb, void* usr)
- static void **addCallback**(DtExerciseConn* conn, const char* className, DtReceiveInteractionCb cb, void* usr)
- static void **removeCallback**(DtExerciseConn* conn, const char* className, DtReceiveInteractionCb cb, void* usr)
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const

DtInteraction* **self**() const

Documentation

class DtInteractionWithData: Instances of DtInteractionWithData are used to represent RPR FOM Interactions that use fixed and var datum records

`DtInteractionWithData()`

default constructor

`virtual ~DtInteractionWithData()`

destructor

`DtInteractionWithData(const DtInteractionWithData& orig)`

copy constructor

`DtInteractionWithData& operator=( const DtInteractionWithData& orig)`

assignment operator

`virtual void printData() const`

Print the interaction's data

`virtual int numFixedFields() const`

`virtual int numVarFields() const`

returns number of variable length fields

`virtual DtDatumParam varDatumId(int index) const`

Returns the datum param found in the index'th variable datum record. Index is between 1 and numRecords. Returns DtDatumParam(0) if index is out of bounds.

`virtual DtDatumParam fixedDatumId(int index) const`

Returns the datum param found in the index'th fixed datum record. Index is between 1 and numRecords. Returns DtDatumParam(0) if index is out of bounds.

`virtual DtDatumParam datumId(DtDatumRecordType fixedOrVar, int index) const`

Returns the datum param found in the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar. Index is between 1 and numRecords. Returns DtDatumParam(0) if index is out of bounds.

`virtual int datumSizeBits(DtDatumRecordType fixedOrVar, int index) const`

Returns the number of bits of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar. If DtFixed, the answer is always 32.

`virtual int datumSizeBytes(DtDatumRecordType fixedOrVar, int index) const`

Returns the number of bytes of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar. If DtFixed, the answer is always 4.

`virtual void getIndex(DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const`

Looks through all fixed and variable records for one containing param as the parameter ID. If it finds one, fills supplied fixedOrVar pointer with DtFixed or DtVar, and index with the index to indicate which record was found, and fills retcode with DtDatumValid. If param was not found, retcode is filled with DtDatumBadId, and index and

fixedOrVar are invalid.

virtual DtFloat32 datumValFloat32(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const

******* INSPECTING DATUM VALUES BY INDEX ********* If you know the data type of the value contained in a particular record, use one of the following to return it. fixedOrVar (either DtFixed or DtVar) and index (between 1 and numRecords) indicate which record you are inspecting. datumValBytesArray is used when the data type is a character string, or something other than a long, unsigned long, float or double. It returns a pointer to a character string containing the exact bytes in the network representation that represents the value being inspected. The supplied retcode pointer will be filled with DtDatumValid if the value was successfully inspected, and DtDatumBadLength if either the index was invalid, or the size of the data indicated that it could not be interpreted as the chosen data type.

virtual DtFloat64 datumValFloat64(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const

virtual unsigned long datumValUnsigned32(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const

virtual long datumValInt32(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const

virtual const char* datumValByteArray(DtDatumRecordType fixedOrVar, int index) const

virtual DtFloat32 datumValFloat32(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

******* INSPECTING DATUM VALUES BY PARAM ID ********* If you know the data type of the value contained in a particular record, use one of the following to return it. The particular record inspected will be the first one containing param as the datum ID. datumValByteArray is used when the data type is a character string, or something other than a long, unsigned long, float or double. It returns a pointer to a character string containing the exact bytes in the network representation that represents the value being inspected. The supplied retcode pointer will be filled with DtDatumValid if the value was successfully inspected, and DtDatumBadLength if either param was not found, or the size of the data indicated that it could not be interpreted as the chosen data type.

virtual DtFloat64 datumValFloat64(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

virtual unsigned long datumValUnsigned32(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

virtual long datumValInt32(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

virtual const char* datumValByteArray(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

****************** MUTATOR FUNCTIONS ******************

virtual void initDataSection(const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)

Initialize to reflect the fixed parameters and variable parameters and sizes indicated. fixedParams is a NULL-terminated array of DtDatumParams. specList is an array of DtVarDatumSpecs terminated with the DtNullVarDatumSpec constant. This call can be made instead of calls to setNumVarDatumRecords, setNumFixedDatumRecord, and repeated calls to setDatumParam and setVarDataBytes.

virtual void setNumFixedFields(int newNum)

Sets the number of fixed or variable fields to newNum. Previously existing fixed fields with indices greater than newNum are deleted. If newNum exceeds the number of existing fields, trailing new fields are created with zeros

as datum params, sizes (for var only) and values.

virtual void setNumVarFields(int newNum)

virtual void setDatumParam(DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)

Set the datum ID in the index'th fixed or variable datum record to param, depending on whether fixedOrVar is DtFixed or DtVar. Supplied retcode pointer is filled with DtDatumValid if successful, or DtDatumBadIndex if the index is not valid.

virtual void setVarDataBytes(int index, int length, DtDatumRetCode* retcode = NULL)

Allocates length bytes for the index'th variable datum record. Should be called before using one of the setDatumVal functions to actually set the data. Supplied retcode pointer is set to DtDatumValid if successful, or DtDatumBadIndex if the index is not valid.

virtual void setDatumValFloat64(DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)

******* SETTING DATA VALUES BY INDEX ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data. fixedOrVar (either DtFixed or DtVar) and index (between 1 and numRecords) indicate which record you are modifying. setDatumValBytesArray is used when the data type is a character string, or something other than a long, unsigned long, float or double. It takes a pointer to a character string containing the exact bytes that should be used in the network representation. The supplied retcode pointer will be filled with DtDatumValid if the value was successfully set, DtDatumBadIndex if the index is invalid, and DtDatumBadLength if the size of the data indicated that it could not be set from as the chosen data type.

virtual void setDatumValFloat32(DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValInt32(DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValUnsigned32(DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValByteArray(DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValFloat64(DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)

****** SETTING DATA VALUES BY PARAM ID ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data. The particular record modified will be the first one containing param as its parameter ID. setDatumValBytesArray is used when the data type is a character string, or something other than a long, unsigned long, float or double. It takes a pointer to a character string containing the exact bytes that should be used in the network representation. The supplied retcode pointer will be filled with DtDatumValid if the value was successfully set, DtDatumBadId if param is invalid, and DtDatumBadLength if the size of the data indicated that it could not be set from the chosen data type.

virtual void setDatumValFloat32(DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValInt32(DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValUnsigned32(DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValByteArray(DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)

virtual DtDatumRecordSet* datumRecSet()

Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records. In general, the wrapper functions above should be used rather than going directly through the DtDatumRecordSet, which is necessary only when doing encoding and decoding of network representations.

virtual const DtDatumRecordSet* datumRecSet() const

DtDatumRecordSet myRecSet

Direct child classes:

DtSetDataInteraction
DtEventReportInteraction
DtDataInteraction
DtCommentInteraction
DtActionResponseInteraction
DtActionRequestInteraction

Alphabetic index *Hierarchy of classes*

In file hfireinter.h:

class DtFireInteraction: public DtInteractionWithEncDec

Inheritance:

DtFireInteraction - DtInteractionWithEncDec - DtInteraction

Public Methods

	DtFireInteraction () *default constructor*
virtual	**~DtFireInteraction** () *destructor*
	DtFireInteraction (const DtFireInteraction& orig) *copy constructor*
DtFireInteraction&	**operator=** (const DtFireInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setAttackerId** (const DtGlobalObjectDesignator& id) *Set/Get the attacker identifier*
virtual const DtGlobalObjectDesignator&	**attackerId** () const
virtual void	**setTargetId** (const DtGlobalObjectDesignator& id) *Set/Get the target identifier*
virtual const DtGlobalObjectDesignator&	**targetId** () const
virtual void	**setMunitionId** (const DtGlobalObjectDesignator& id) *Set/Get the munition identifier*
virtual const DtGlobalObjectDesignator&	**munitionId** () const
virtual void	**setEventId** (const DtEventID& id) *Set/Get the event id*
virtual const DtEventID&	**eventId** () const
virtual void	**setFireMissionIndex** (unsigned long i) *Set/Get the fire mission index*
virtual unsigned long	**fireMissionIndex** () const
virtual void	**setLocation** (DtConstVector location) *Set/Get the location in world coordinates*
virtual DtConstVector	**location** () const

virtual void	**setVelocity** (DtConstVector velocity) *Set/Get the velocity*
virtual DtConstVector	**velocity** () const
virtual void	**setRange** (double range) *Set/Get the range*
virtual double	**range** () const
virtual void	**setFuseType** (DtDetonatorFuze f) *Set/Get the detonator fuze type*
virtual DtDetonatorFuze	**fuseType** () const
virtual void	**setMunitionType** (const DtEntityType& t) *Set/Get the munition type*
virtual const DtEntityType&	**munitionType** () const
virtual void	**setQuantity** (int q) *Set/Get the quantity*
virtual int	**quantity** () const
virtual void	**setRate** (int r) *Set/Get the rate*
virtual int	**rate** () const
virtual void	**setWarheadType** (DtWarheadType t) *Set/Get the warhead type*
virtual DtWarheadType	**warheadType** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtFireInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtFireInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtFireInteractionCb cb, void* usr)

Protected Fields

DtGlobalObjectDesignator	**myAttackerId**
DtGlobalObjectDesignator	**myTargetId**
DtGlobalObjectDesignator	**myMunitionId**
DtEventID	**myEventId**
unsigned long	**myFireMissionIndex**
DtVector	**myLocation**
DtVector	**myVelocity**
double	**myRange**
DtDetonatorFuze	**myFuseType**
DtEntityType	**myMunitionType**

- int **myQuantity**
- int **myRate**
- DtWarheadType **myWarheadType**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtFireInteraction()`**
  default constructor
- **`virtual ~DtFireInteraction()`**
  destructor
- **`DtFireInteraction(const DtFireInteraction& orig)`**
  copy constructor
- **`DtFireInteraction& operator=(const DtFireInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setAttackerId(const DtGlobalObjectDesignator& id)`**
  Set/Get the attacker identifier
- **`virtual const DtGlobalObjectDesignator& attackerId() const`**
- **`virtual void setTargetId(const DtGlobalObjectDesignator& id)`**
  Set/Get the target identifier

- **virtual const DtGlobalObjectDesignator& targetId() const**
- **virtual void setMunitionId(const DtGlobalObjectDesignator& id)**
  Set/Get the munition identifier
- **virtual const DtGlobalObjectDesignator& munitionId() const**
- **virtual void setEventId(const DtEventID& id)**
  Set/Get the event id
- **virtual const DtEventID& eventId() const**
- **virtual void setFireMissionIndex(unsigned long i)**
  Set/Get the fire mission index
- **virtual unsigned long fireMissionIndex() const**
- **virtual void setLocation(DtConstVector location)**
  Set/Get the location in world coordinates
- **virtual DtConstVector location() const**
- **virtual void setVelocity(DtConstVector velocity)**
  Set/Get the velocity
- **virtual DtConstVector velocity() const**
- **virtual void setRange(double range)**
  Set/Get the range
- **virtual double range() const**
- **virtual void setFuseType(DtDetonatorFuze f)**
  Set/Get the detonator fuze type
- **virtual DtDetonatorFuze fuseType() const**
- **virtual void setMunitionType(const DtEntityType& t)**
  Set/Get the munition type
- **virtual const DtEntityType& munitionType() const**
- **virtual void setQuantity(int q)**
  Set/Get the quantity
- **virtual int quantity() const**
- **virtual void setRate(int r)**
  Set/Get the rate
- **virtual int rate() const**
- **virtual void setWarheadType(DtWarheadType t)**
  Set/Get the warhead type
- **virtual DtWarheadType warheadType() const**

- **`virtual const char* name() const`**

  Name of the interaction (Not nec. the ClassName from the FOM).

- **`static DtInteraction* create()`**

  Create a DtFireInteraction. Caller is reponsible for deletion.

- **`static void addCallback(DtExerciseConn* conn, DtFireInteractionCb cb, void* usr)`**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **`static void removeCallback(DtExerciseConn* conn, DtFireInteractionCb cb, void* usr)`**
- **`DtGlobalObjectDesignator myAttackerId`**
- **`DtGlobalObjectDesignator myTargetId`**
- **`DtGlobalObjectDesignator myMunitionId`**
- **`DtEventID myEventId`**
- **`unsigned long myFireMissionIndex`**
- **`DtVector myLocation`**
- **`DtVector myVelocity`**
- **`double myRange`**
- **`DtDetonatorFuze myFuseType`**
- **`DtEntityType myMunitionType`**
- **`int myQuantity`**
- **`int myRate`**
- **`DtWarheadType myWarheadType`**
- **`static char* theClassName`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdetinter.h:

class DtDetonationInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtDetonationInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtDetonationInteraction () *default constructor*
	DtDetonationInteraction (const DtFireInteraction& fire) *Constructor Fills in common fields with data from Fire Interaction*
virtual	**~DtDetonationInteraction** () *destructor*
	DtDetonationInteraction (const DtDetonationInteraction& orig) *copy constructor*
DtDetonationInteraction&	**operator=** (const DtDetonationInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setAttackerId** (const DtGlobalObjectDesignator& id) *Set/Get the attacker identifier*
virtual const DtGlobalObjectDesignator&	**attackerId** () const
virtual void	**setTargetId** (const DtGlobalObjectDesignator& id) *Set/Get the target identifier*
virtual const DtGlobalObjectDesignator&	**targetId** () const
virtual void	**setMunitionId** (const DtGlobalObjectDesignator& id) *Set/Get the munition identifier*
virtual const DtGlobalObjectDesignator&	**munitionId** () const
virtual void	**setEventId** (const DtEventID& id) *Set/Get the detonation event id*
virtual const DtEventID&	**eventId** () const
virtual void	**setVelocity** (DtConstVector velocity) *Set/Get the velocity*
virtual DtConstVector	**velocity** () const

virtual void	**setWorldLocation** (DtConstVector location) *Set/Get the location of the detonation in world coordinates*
virtual DtConstVector	**worldLocation** () const
virtual void	**setEntityLocation** (DtConstVector location) *Set/Get the location of the detonation in entity coordinates*
virtual DtConstVector	**entityLocation** () const
virtual void	**setResult** (DtDetonationResult result) *Set/Get the detonation result*
virtual DtDetonationResult	**result** () const
virtual void	**setFuseType** (DtDetonatorFuze fuse) *Set/Get the detonator fuze type*
virtual DtDetonatorFuze	**fuseType** () const
virtual void	**setMunitionType** (const DtEntityType& munType) *Set/Get the munition type*
virtual const DtEntityType&	**munitionType** () const
virtual void	**setQuantity** (int quant) *Set/Get the quantity*
virtual int	**quantity** () const
virtual void	**setRate** (int rate) *Set/Get the rate*
virtual int	**rate** () const
virtual void	**setWarheadType** (DtWarheadType type) *Set/Get the warhead type*
virtual DtWarheadType	**warheadType** () const
virtual void	**initArtParts** (const DtArtPartSpec* specList) *Set/Get ArticulatedParts*
virtual const DtArtPartSpec*	**artPartSpecs** () const
virtual int	**numArtParams** () const *Get number of articulation parameters*
virtual int	**setArtParamVal** (int partType, int param, DtFloat32 value) *Set/Get articulation parameter values by part type and parameter type*
virtual int	**getArtParamVal** (int partType, int param, DtFloat32* value) const
virtual DtArtPartList*	**artPartList** () *Get a pointer to an DtArtPartList containing the interaction's articulated part information*
virtual const DtArtPartList*	**artPartList** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtDetonationInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtDetonationInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtDetonationInteractionCb cb, void* usr)

Protected Fields

- DtVector **myWorldLocation**
- DtDetonationResult **myResult**
- DtEventID **myEventId**
- DtGlobalObjectDesignator **myAttackerId**
- DtVector **myVelocity**
- DtDetonatorFuze **myFuseType**
- DtGlobalObjectDesignator **myMunitionId**
- DtEntityType **myMunitionType**
- int **myQuantity**
- int **myRate**
- DtVector **myEntityLocation**
- DtGlobalObjectDesignator **myTargetId**
- DtWarheadType **myWarheadType**
- DtArtPartList **myPartList**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**

- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()

- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtDetonationInteraction()`**
  default constructor
- **`DtDetonationInteraction(const DtFireInteraction& fire)`**
  Constructor Fills in common fields with data from Fire Interaction
- **`virtual ~DtDetonationInteraction()`**
  destructor

- **DtDetonationInteraction(const DtDetonationInteraction& orig)**
  copy constructor
- **DtDetonationInteraction& operator=(const DtDetonationInteraction& orig)**
  assignment operator
- **virtual void printData() const**
  Print the interaction's data
- **virtual void setAttackerId(const DtGlobalObjectDesignator& id)**
  Set/Get the attacker identifier
- **virtual const DtGlobalObjectDesignator& attackerId() const**
- **virtual void setTargetId(const DtGlobalObjectDesignator& id)**
  Set/Get the target identifier
- **virtual const DtGlobalObjectDesignator& targetId() const**
- **virtual void setMunitionId(const DtGlobalObjectDesignator& id)**
  Set/Get the munition identifier
- **virtual const DtGlobalObjectDesignator& munitionId() const**
- **virtual void setEventId(const DtEventID& id)**
  Set/Get the detonation event id
- **virtual const DtEventID& eventId() const**
- **virtual void setVelocity(DtConstVector velocity)**
  Set/Get the velocity
- **virtual DtConstVector velocity() const**
- **virtual void setWorldLocation(DtConstVector location)**
  Set/Get the location of the detonation in world coordinates
- **virtual DtConstVector worldLocation() const**
- **virtual void setEntityLocation(DtConstVector location)**
  Set/Get the location of the detonation in entity coordinates
- **virtual DtConstVector entityLocation() const**
- **virtual void setResult(DtDetonationResult result)**
  Set/Get the detonation result
- **virtual DtDetonationResult result() const**
- **virtual void setFuseType(DtDetonatorFuze fuse)**
  Set/Get the detonator fuze type
- **virtual DtDetonatorFuze fuseType() const**
- **virtual void setMunitionType(const DtEntityType& munType)**
  Set/Get the munition type

- **virtual const DtEntityType& munitionType() const**
- **virtual void setQuantity(int quant)**
  Set/Get the quantity
- **virtual int quantity() const**
- **virtual void setRate(int rate)**
  Set/Get the rate
- **virtual int rate() const**
- **virtual void setWarheadType(DtWarheadType type)**
  Set/Get the warhead type
- **virtual DtWarheadType warheadType() const**
- **virtual void initArtParts(const DtArtPartSpec* specList)**
  Set/Get ArticulatedParts
- **virtual const DtArtPartSpec* artPartSpecs() const**
- **virtual int numArtParams() const**
  Get number of articulation parameters
- **virtual int setArtParamVal(int partType, int param, DtFloat32 value)**
  Set/Get articulation parameter values by part type and parameter type
- **virtual int getArtParamVal(int partType, int param, DtFloat32* value) const**
- **virtual DtArtPartList* artPartList()**
  Get a pointer to an DtArtPartList containing the interaction's articulated part information
- **virtual const DtArtPartList* artPartList() const**
- **virtual const char* name() const**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**
  Create a DtDetonationInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtDetonationInteractionCb cb, void* usr)**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtDetonationInteractionCb cb, void* usr)**
- **DtVector myWorldLocation**
- **DtDetonationResult myResult**
- **DtEventID myEventId**
- **DtGlobalObjectDesignator myAttackerId**
- **DtVector myVelocity**

- DtDetonatorFuze myFuseType
- DtGlobalObjectDesignator myMunitionId
- DtEntityType myMunitionType
- int myQuantity
- int myRate
- DtVector myEntityLocation
- DtGlobalObjectDesignator myTargetId
- DtWarheadType myWarheadType
- DtArtPartList myPartList

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdataqinter.h:

class DtDataQueryInteraction: public DtInteractionWithEncDec

Inheritance:

DtDataQueryInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtDataQueryInteraction () *default constructor*
virtual	**~DtDataQueryInteraction** () *destructor*
	DtDataQueryInteraction (const DtDataQueryInteraction& orig) *copy constructor*
DtDataQueryInteraction&	**operator=** (const DtDataQueryInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *`Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRequestId** (unsigned long id) *Set/Get request ID*
virtual unsigned long	**requestId** () const
virtual void	**setTimeInterval** (DtTime seconds) *Set/Get the time interval*
virtual DtTime	**timeInterval** () const
virtual void	**initDatumIds** (unsigned long numFixedFields, unsigned long numVarFields) *Initialize fixed and variable datum ids*
virtual void	**setNumFixedFields** (unsigned long numFixedFields) *Set/Get fixed fields*
virtual unsigned long	**numFixedFields** () const
virtual int	**setFixedDatumId** (DtU32 index, DtDatumParam id)

- virtual DtDatumParam **fixedDatumId** (DtU32 index) const
- virtual void **setNumVarFields** (unsigned long numVarFields)
  Set/Get variable fields
- virtual unsigned long **numVarFields** () const
- virtual int **setVarDatumId** (DtU32 index, DtDatumParam id)
- virtual DtDatumParam **varDatumId** (DtU32 index) const
- virtual void **setFixedDatumIDs** (const void* data, DtInt32 numBytes)
  Set/Get the fixed fields memory block all at once
- virtual const void* **fixedDatumIDs** () const
- virtual void **setVariableDatumIDs** (const void* data, DtInt32 numBytes)
  Set/Get the variable fields memory block all at once
- virtual const void* **variableDatumIDs** () const
- virtual void **changeFixedSize** (DtU32 newSize)
  Change size of fixed or variable fields
- virtual void **changeVariableSize** (DtU32 newSize)
- virtual const char* **name** () const
  Name of the interaction (Not nec
- static DtInteraction* **create** ()
  Create a DtDataQueryInteraction
- static void **addCallback** (DtExerciseConn* conn, DtDataQueryInteractionCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtDataQueryInteractionCb cb, void* usr)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- DtU32 **myRequestId**
- DtNetTimeStamp **myTimeInterval**
- DtNetU32* **myFixedDatumIDs**
- DtNetU32* **myVariableDatumIDs**
- DtU32 **myNumFixed**
- DtU32 **myNumVar**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const

- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const

- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtDataQueryInteraction()`**
  default constructor
- **`virtual ~DtDataQueryInteraction()`**
  destructor
- **`DtDataQueryInteraction(const DtDataQueryInteraction& orig)`**
  copy constructor
- **`DtDataQueryInteraction& operator=( const DtDataQueryInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setSenderId(const DtEntityIdentifier& id)`**
  `Set/Get sender ID
- **`virtual const DtEntityIdentifier& senderId() const`**
- **`virtual void setReceiverId(const DtEntityIdentifier& id)`**
  Set/Get receiver ID
- **`virtual const DtEntityIdentifier& receiverId() const`**
- **`virtual void setRequestId(unsigned long id)`**
  Set/Get request ID
- **`virtual unsigned long requestId() const`**
- **`virtual void setTimeInterval(DtTime seconds)`**
  Set/Get the time interval
- **`virtual DtTime timeInterval() const`**
- **`virtual void initDatumIds(unsigned long numFixedFields, unsigned long numVarFields)`**
  Initialize fixed and variable datum ids
- **`virtual void setNumFixedFields(unsigned long numFixedFields)`**
  Set/Get fixed fields
- **`virtual unsigned long numFixedFields() const`**

- **virtual int setFixedDatumId(DtU32 index, DtDatumParam id)**
- **virtual DtDatumParam fixedDatumId(DtU32 index) const**
- **virtual void setNumVarFields(unsigned long numVarFields)**
  Set/Get variable fields
- **virtual unsigned long numVarFields() const**
- **virtual int setVarDatumId(DtU32 index, DtDatumParam id)**
- **virtual DtDatumParam varDatumId(DtU32 index) const**
- **virtual void setFixedDatumIDs(const void* data, DtInt32 numBytes)**
  Set/Get the fixed fields memory block all at once
- **virtual const void* fixedDatumIDs() const**
- **virtual void setVariableDatumIDs(const void* data, DtInt32 numBytes)**
  Set/Get the variable fields memory block all at once
- **virtual const void* variableDatumIDs() const**
- **virtual void changeFixedSize(DtU32 newSize)**
  Change size of fixed or variable fields
- **virtual void changeVariableSize(DtU32 newSize)**
- **virtual const char* name() const**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**
  Create a DtDataQueryInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtDataQueryInteractionCb cb, void* usr)**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtDataQueryInteractionCb cb, void* usr)**
- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **DtU32 myRequestId**
- **DtNetTimeStamp myTimeInterval**
- **DtNetU32* myFixedDatumIDs**
- **DtNetU32* myVariableDatumIDs**
- **DtU32 myNumFixed**
- **DtU32 myNumVar**

This class has no child classes.

In file hcrtobjresint.h:

class DtCreateObjectResultInter: public DtInteractionWithEncDec

---

Inheritance:

DtCreateObjectResultInter - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtCreateObjectResultInter () *default constructor*
virtual	**~DtCreateObjectResultInter** () *destructor*
	DtCreateObjectResultInter (const DtCreateObjectResultInter& orig) *copy constructor*
DtCreateObjectResultInter&	**operator=** (const DtCreateObjectResultInter& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setCreateObjectResult** (DtCreateObjectResult result) *Set/Get create object result*
virtual DtCreateObjectResult	**createObjectResult** () const
virtual void	**setRequestId** (DtU32 id) *Set/Get request ID*
virtual DtU32	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtCreateObjectResultInter*
static void	**addCallback** (DtExerciseConn* conn, DtCreateObjectResultInterCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtCreateObjectResultInterCb cb, void* usr)
inline const char*	**name** () const
inline DtU32	**requestId** () const
inline void	**setCreateObjectResult** (DtCreateObjectResult result)
inline void	**setRequestId** (DtU32 id)

Protected Fields

- DtCreateObjectResult **myCreateObjectResult**
- DtU32 **myRequestId**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtCreateObjectResultInter()`**
  default constructor
- **`virtual ~DtCreateObjectResultInter()`**
  destructor
- **`DtCreateObjectResultInter(const DtCreateObjectResultInter& orig)`**
  copy constructor
- **`DtCreateObjectResultInter& operator=( const DtCreateObjectResultInter& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setCreateObjectResult(DtCreateObjectResult result)`**
  Set/Get create object result
- **`virtual DtCreateObjectResult createObjectResult() const`**
- **`virtual void setRequestId(DtU32 id)`**

Set/Get request ID

- **virtual DtU32 requestId() const**
- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtCreateObjectResultInter. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtCreateObjectResultInterCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtCreateObjectResultInterCb cb, void* usr)**
- **DtCreateObjectResult myCreateObjectResult**
- **DtU32 myRequestId**
- **static char* theClassName**
- **inline const char* name() const**
- **inline DtU32 requestId() const**
- **inline void setCreateObjectResult(DtCreateObjectResult result)**
- **inline void setRequestId(DtU32 id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcrtobjreqint.h:

class DtCreateObjReqInter: public DtInteractionWithEncDec

Inheritance:

DtCreateObjReqInter - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtCreateObjReqInter () *default constructor*
virtual	**~DtCreateObjReqInter** () *destructor*
	DtCreateObjReqInter (const DtCreateObjReqInter& orig) *copy constructor*
DtCreateObjReqInter&	**operator=** (const DtCreateObjReqInter& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setObjectClass** (DtU32 id) *Set/Get object class*
virtual DtU32	**objectClass** () const
virtual void	**setNumPairs** (DtU32 pairs) *Set/Get the number of attribute/value pairs*
virtual DtU32	**numPairs** () const
virtual void	**setAvPairSet** (const RTI::AttributeHandleValuePairSet& atts)
virtual const RTI::AttributeHandleValuePairSet&	**avPairSet** () const
virtual void	**setAttributeHandle** (DtU32 index, DtU32 handle) *Set/Get the attribute handle given an index*
virtual DtU32	**attributeHandle** (DtU32 index) const
virtual void	**setAttributeValueLength** (DtU32 index, DtU32 length) *Set/Get the attribute value length given and index*
virtual DtU32	**attributeValueLength** (DtU32 index) const
virtual void	**setAttributeValue** (DtU32 index, const char* usr)
virtual const char*	**attributeValue** (DtU32 index) const

- virtual DtDatumRecordSet* **datumRecSet** ()
  Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records
- virtual const DtDatumRecordSet* **datumRecSet** () const
- virtual void **setRequestId** (DtU32 id)
  Set/Get request ID
- virtual DtU32 **requestId** () const
- virtual const char* **name** () const
  Name of the interaction (Not nec
- static DtInteraction* **create** ()
  Create a DtCreateObjReqInter
- static void **addCallback** (DtExerciseConn* conn, DtCreateObjReqInterCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtCreateObjReqInterCb cb, void* usr)
- inline const char* **name** () const
- inline DtU32 **objectClass** () const
- inline DtU32 **requestId** () const
- inline void **setObjectClass** (DtU32 id)
- inline void **setRequestId** (DtU32 id)

Protected Fields

- DtU32 **myObjectClass**
- DtDatumRecordSet **myRecordSet**
- DtU32 **myRequestId**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const

- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)

- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtCreateObjReqInter()`**
  default constructor
- **`virtual ~DtCreateObjReqInter()`**
  destructor
- **`DtCreateObjReqInter(const DtCreateObjReqInter& orig)`**
  copy constructor
- **`DtCreateObjReqInter& operator=(const DtCreateObjReqInter& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setObjectClass(DtU32 id)`**
  Set/Get object class
- **`virtual DtU32 objectClass() const`**
- **`virtual void setNumPairs(DtU32 pairs)`**
  Set/Get the number of attribute/value pairs
- **`virtual DtU32 numPairs() const`**
- **`virtual void setAvPairSet( const RTI::AttributeHandleValuePairSet& atts)`**
- **`virtual const RTI::AttributeHandleValuePairSet& avPairSet() const`**
- **`virtual void setAttributeHandle(DtU32 index, DtU32 handle)`**
  Set/Get the attribute handle given an index
- **`virtual DtU32 attributeHandle(DtU32 index) const`**
- **`virtual void setAttributeValueLength(DtU32 index, DtU32 length)`**
  Set/Get the attribute value length given and index
- **`virtual DtU32 attributeValueLength(DtU32 index) const`**
- **`virtual void setAttributeValue(DtU32 index, const char* usr)`**
- **`virtual const char* attributeValue(DtU32 index) const`**
- **`virtual DtDatumRecordSet* datumRecSet()`**
  Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records. In general, the wrapper functions above should be used rather than going directly through the DtDatumRecordSet, which is necessary only when doing encoding and decoding of network representations.
- **`virtual const DtDatumRecordSet* datumRecSet() const`**
- **`virtual void setRequestId(DtU32 id)`**
  Set/Get request ID

- **virtual DtU32 requestId() const**
- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**

  Create a DtCreateObjReqInter. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtCreateObjReqInterCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtCreateObjReqInterCb cb, void* usr)**
- **DtU32 myObjectClass**
- **DtDatumRecordSet myRecordSet**
- **DtU32 myRequestId**
- **static char* theClassName**
- **inline const char* name() const**
- **inline DtU32 objectClass() const**
- **inline DtU32 requestId() const**
- **inline void setObjectClass(DtU32 id)**
- **inline void setRequestId(DtU32 id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcreatentint.h:

class DtCreateEntityInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtCreateEntityInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtCreateEntityInteraction () *default constructor*
virtual	**~DtCreateEntityInteraction** () *destructor*
	DtCreateEntityInteraction (const DtCreateEntityInteraction& orig) *copy constructor*
DtCreateEntityInteraction&	**operator=** (const DtCreateEntityInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRequestId** (int id) *Set/Get request ID*
virtual int	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtCreateEntityInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtCreateEntityInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtCreateEntityInteractionCb cb, void* usr)
inline const char*	**name** () const

- inline const DtEntityIdentifier& **receiverId** () const
- inline int **requestId** () const
- inline const DtEntityIdentifier& **senderId** () const
- inline void **setReceiverId** (const DtEntityIdentifier& id)
- inline void **setRequestId** (int id)
- inline void **setSenderId** (const DtEntityIdentifier& id)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- int **myRequestId**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const

- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtCreateEntityInteraction()`**
  default constructor
- **`virtual ~DtCreateEntityInteraction()`**
  destructor
- **`DtCreateEntityInteraction(const DtCreateEntityInteraction& orig)`**
  copy constructor
- **`DtCreateEntityInteraction& operator=(const DtCreateEntityInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data

- **virtual void setSenderId(const DtEntityIdentifier& id)**
  Set/Get sender ID
- **virtual const DtEntityIdentifier& senderId() const**
- **virtual void setReceiverId(const DtEntityIdentifier& id)**
  Set/Get receiver ID
- **virtual const DtEntityIdentifier& receiverId() const**
- **virtual void setRequestId(int id)**
  Set/Get request ID
- **virtual int requestId() const**
- **virtual const char* name() const**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**
  Create a DtCreateEntityInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtCreateEntityInteractionCb cb, void* usr)**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtCreateEntityInteractionCb cb, void* usr)**
- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **int myRequestId**
- **static char* theClassName**
- **inline const char* name() const**
- **inline const DtEntityIdentifier& receiverId() const**
- **inline int requestId() const**
- **inline const DtEntityIdentifier& senderId() const**
- **inline void setReceiverId(const DtEntityIdentifier& id)**
- **inline void setRequestId(int id)**
- **inline void setSenderId(const DtEntityIdentifier& id)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcollideinter.h:

class DtCollisionInteraction: public DtInteractionWithEncDec

Inheritance:

DtCollisionInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtCollisionInteraction () *default constructor*
virtual	**~DtCollisionInteraction** () *destructor*
	DtCollisionInteraction (const DtCollisionInteraction& orig) *copy constructor*
DtCollisionInteraction&	**operator=** (const DtCollisionInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setIssueId** (const DtGlobalObjectDesignator& id) *Set/Get the issuer identifier*
virtual const DtGlobalObjectDesignator&	**issueId** () const
virtual void	**setCollideId** (const DtGlobalObjectDesignator& id) *Set/Get the collide identifier*
virtual const DtGlobalObjectDesignator&	**collideId** () const
virtual void	**setEventId** (const DtEventID& id) *Set/Get the collision event id*
virtual const DtEventID&	**eventId** () const
virtual void	**setCollisionType** (DtCollisionType t) *Set/Get the collision type*
virtual DtCollisionType	**collisionType** () const
virtual void	**setVelocity** (DtConstVector veloc) *Set/Get the velocity*
virtual DtConstVector	**velocity** () const
virtual void	**setMass** (DtFloat32 mass) *Set/Get the mass involved*
virtual DtFloat32	**mass** () const

virtual void	**setEntityCoords** (DtConstVector coords) *Set/Get the entity coordinates*
virtual DtConstVector	**entityCoords** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtCollisionInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtCollisionInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtCollisionInteractionCb cb, void* usr)

Protected Fields

DtGlobalObjectDesignator	**myIssueId**
DtGlobalObjectDesignator	**myCollideId**
DtEventID	**myEventId**
DtCollisionType	**myCollisionType**
DtVector	**myVelocity**
DtFloat32	**myMass**
DtVector	**myEntityCoords**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const

- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtCollisionInteraction()`**
  default constructor
- **`virtual ~DtCollisionInteraction()`**
  destructor
- **`DtCollisionInteraction(const DtCollisionInteraction& orig)`**
  copy constructor
- **`DtCollisionInteraction& operator=(const DtCollisionInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setIssueId(const DtGlobalObjectDesignator& id)`**
  Set/Get the issuer identifier
- **`virtual const DtGlobalObjectDesignator& issueId() const`**
- **`virtual void setCollideId(const DtGlobalObjectDesignator& id)`**
  Set/Get the collide identifier
- **`virtual const DtGlobalObjectDesignator& collideId() const`**
- **`virtual void setEventId(const DtEventID& id)`**
  Set/Get the collision event id
- **`virtual const DtEventID& eventId() const`**
- **`virtual void setCollisionType(DtCollisionType t)`**
  Set/Get the collision type
- **`virtual DtCollisionType collisionType() const`**
- **`virtual void setVelocity(DtConstVector veloc)`**
  Set/Get the velocity
- **`virtual DtConstVector velocity() const`**
- **`virtual void setMass(DtFloat32 mass)`**
  Set/Get the mass involved
- **`virtual DtFloat32 mass() const`**
- **`virtual void setEntityCoords(DtConstVector coords)`**
  Set/Get the entity coordinates
- **`virtual DtConstVector entityCoords() const`**
- **`virtual const char* name() const`**
  Name of the interaction (Not nec. the ClassName from the FOM).

- **`static DtInteraction* create()`**

  Create a DtCollisionInteraction. Caller is reponsible for deletion.

- **`static void addCallback(DtExerciseConn* conn, DtCollisionInteractionCb cb, void* usr)`**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **`static void removeCallback(DtExerciseConn* conn, DtCollisionInteractionCb cb, void* usr)`**
- **`DtGlobalObjectDesignator myIssueId`**
- **`DtGlobalObjectDesignator myCollideId`**
- **`DtEventID myEventId`**
- **`DtCollisionType myCollisionType`**
- **`DtVector myVelocity`**
- **`DtFloat32 myMass`**
- **`DtVector myEntityCoords`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hattchgresint.h:

class DtAttributeChangeResInter: public DtInteractionWithEncDec

---

Inheritance:

DtAttributeChangeResInter - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtAttributeChangeResInter () *default constructor*
virtual	**~DtAttributeChangeResInter** () *destructor*
	DtAttributeChangeResInter (const DtAttributeChangeResInter& orig) *copy constructor*
DtAttributeChangeResInter&	**operator=** (const DtAttributeChangeResInter& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setObjectId** (const DtGlobalObjectDesignator &id) *Set/Get the objectID*
virtual const DtGlobalObjectDesignator&	**objectId** () const
virtual void	**setNumPairs** (DtU32 pairs) *Set/Get the number of attribute/value pairs*
virtual DtU32	**numPairs** () const
virtual void	**setAvPairSet** (const RTI::AttributeHandleValuePairSet& atts)
virtual const RTI::AttributeHandleValuePairSet&	**avPairSet** () const
virtual void	**setAttributeHandle** (DtU32 index, DtU32 handle) *Set/Get the attribute handle given an index*
virtual DtU32	**attributeHandle** (DtU32 index) const
virtual void	**setAttributeValueLength** (DtU32 index, DtU32 length) *Set/Get the attribute value length given and index*
virtual DtU32	**attributeValueLength** (DtU32 index) const
virtual void	**setAttributeValue** (DtU32 index, const char* usr)
virtual const char*	**attributeValue** (DtU32 index) const

virtual DtDatumRecordSet*	**datumRecSet** () *Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records*
virtual const DtDatumRecordSet*	**datumRecSet** () const
virtual void	**setAttributeChangeResult** (DtAttributeChangeResult result) *Set/Get the AttributeChangeResult type*
virtual DtAttributeChangeResult	**attributeChangeResult** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtAttributeChangeResInter*
static void	**addCallback** (DtExerciseConn* conn, DtAttributeChangeResInterCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtAttributeChangeResInterCb cb, void* usr)
inline const char*	**name** () const

Protected Fields

DtGlobalObjectDesignator	**myObjectId**
DtDatumRecordSet	**myRecordSet**
DtAttributeChangeResult	**myAttributeChangeResult**
static char*	**theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const

- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtAttributeChangeResInter()`**
  default constructor
- **`virtual ~DtAttributeChangeResInter()`**
  destructor
- **`DtAttributeChangeResInter(const DtAttributeChangeResInter& orig)`**
  copy constructor
- **`DtAttributeChangeResInter& operator=(const DtAttributeChangeResInter& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setObjectId(const DtGlobalObjectDesignator &id)`**
  Set/Get the objectID
- **`virtual const DtGlobalObjectDesignator& objectId() const`**
- **`virtual void setNumPairs(DtU32 pairs)`**
  Set/Get the number of attribute/value pairs
- **`virtual DtU32 numPairs() const`**
- **`virtual void setAvPairSet( const RTI::AttributeHandleValuePairSet& atts)`**
- **`virtual const RTI::AttributeHandleValuePairSet& avPairSet() const`**
- **`virtual void setAttributeHandle(DtU32 index, DtU32 handle)`**
  Set/Get the attribute handle given an index
- **`virtual DtU32 attributeHandle(DtU32 index) const`**
- **`virtual void setAttributeValueLength(DtU32 index, DtU32 length)`**
  Set/Get the attribute value length given and index
- **`virtual DtU32 attributeValueLength(DtU32 index) const`**
- **`virtual void setAttributeValue(DtU32 index, const char* usr)`**
- **`virtual const char* attributeValue(DtU32 index) const`**
- **`virtual DtDatumRecordSet* datumRecSet()`**
  Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records. In general, the wrapper functions above should be used rather than going directly through the DtDatumRecordSet, which is necessary only when doing encoding and decoding of network representations.
- **`virtual const DtDatumRecordSet* datumRecSet() const`**
- **`virtual void setAttributeChangeResult(DtAttributeChangeResult result)`**
  Set/Get the AttributeChangeResult type

- **virtual DtAttributeChangeResult attributeChangeResult() const**
- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**

  Create a DtAttributeChangeResInter. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtAttributeChangeResInterCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtAttributeChangeResInterCb cb, void* usr)**
- **DtGlobalObjectDesignator myObjectId**
- **DtDatumRecordSet myRecordSet**
- **DtAttributeChangeResult myAttributeChangeResult**
- **static char* theClassName**
- **inline const char* name() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hattchgreqint.h:

class DtAttributeChangeReqInter: public DtInteractionWithEncDec

---

Inheritance:

DtAttributeChangeReqInter - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtAttributeChangeReqInter () *default constructor*
virtual	**~DtAttributeChangeReqInter** () *destructor*
	DtAttributeChangeReqInter (const DtAttributeChangeReqInter& orig) *copy constructor*
DtAttributeChangeReqInter&	**operator=** (const DtAttributeChangeReqInter& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual const DtGlobalObjectDesignatorList&	**objectIDs** () const *Return the Object ID list*
virtual DtGlobalObjectDesignatorList*	**objectIDs** ()
virtual void	**setNumIds** (DtU8 numIds) *Set/Get number of IDs*
virtual DtU8	**numIds** () const
virtual void	**setNthObjectId** (DtInt32 idIndex, const DtGlobalObjectDesignator& id) *Set/Get the nth ObjectId*
virtual const DtGlobalObjectDesignator&	**nthObjectId** (DtInt32 idIndex) const
virtual void	**setNumPairs** (DtU32 pairs) *Set/Get the number of attribute/value pairs*
virtual DtU32	**numPairs** () const
virtual void	**setAvPairSet** (const RTI::AttributeHandleValuePairSet& atts)
virtual const RTI::AttributeHandleValuePairSet&	**avPairSet** () const

- virtual void **setAttributeHandle** (DtU32 index, DtU32 handle)
  Set/Get the attribute handle given an index
- virtual DtU32 **attributeHandle** (DtU32 index) const
- virtual void **setAttributeValueLength** (DtU32 index, DtU32 length)
  Set/Get the attribute value length given and index
- virtual DtU32 **attributeValueLength** (DtU32 index) const
- virtual void **setAttributeValue** (DtU32 index, const char* usr)
- virtual const char* **attributeValue** (DtU32 index) const
- virtual DtDatumRecordSet* **datumRecSet** ()
  Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records
- virtual const DtDatumRecordSet* **datumRecSet** () const
- virtual const char* **name** () const
  Name of the interaction (Not nec
- static DtInteraction* **create** ()
  Create a DtAttributeChangeReqInter
- static void **addCallback** (DtExerciseConn* conn, DtAttributeChangeReqInterCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtAttributeChangeReqInterCb cb, void* usr)
- inline const char* **name** () const

Protected Fields

- DtGlobalObjectDesignatorList **myObjectList**
- DtDatumRecordSet **myRecordSet**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const

- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)

- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtAttributeChangeReqInter()`**
  default constructor
- **`virtual ~DtAttributeChangeReqInter()`**
  destructor
- **`DtAttributeChangeReqInter(const DtAttributeChangeReqInter& orig)`**
  copy constructor
- **`DtAttributeChangeReqInter& operator=(const DtAttributeChangeReqInter& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual const DtGlobalObjectDesignatorList& objectIDs() const`**
  Return the Object ID list
- **`virtual DtGlobalObjectDesignatorList* objectIDs()`**
- **`virtual void setNumIds(DtU8 numIds)`**
  Set/Get number of IDs
- **`virtual DtU8 numIds() const`**
- **`virtual void setNthObjectId(DtInt32 idIndex, const DtGlobalObjectDesignator& id)`**
  Set/Get the nth ObjectId
- **`virtual const DtGlobalObjectDesignator& nthObjectId(DtInt32 idIndex) const`**
- **`virtual void setNumPairs(DtU32 pairs)`**
  Set/Get the number of attribute/value pairs
- **`virtual DtU32 numPairs() const`**
- **`virtual void setAvPairSet( const RTI::AttributeHandleValuePairSet& atts)`**
- **`virtual const RTI::AttributeHandleValuePairSet& avPairSet() const`**
- **`virtual void setAttributeHandle(DtU32 index, DtU32 handle)`**
  Set/Get the attribute handle given an index
- **`virtual DtU32 attributeHandle(DtU32 index) const`**
- **`virtual void setAttributeValueLength(DtU32 index, DtU32 length)`**
  Set/Get the attribute value length given and index
- **`virtual DtU32 attributeValueLength(DtU32 index) const`**
- **`virtual void setAttributeValue(DtU32 index, const char* usr)`**
- **`virtual const char* attributeValue(DtU32 index) const`**

- **virtual DtDatumRecordSet* datumRecSet()**

  Return the DtDatumRecordSet which an InteractionWithData uses to store its fixed and var records. In general, the wrapper functions above should be used rather than going directly through the DtDatumRecordSet, which is necessary only when doing encoding and decoding of network representations.

- **virtual const DtDatumRecordSet* datumRecSet() const**

- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtAttributeChangeReqInter. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtAttributeChangeReqInterCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtAttributeChangeReqInterCb cb, void* usr)**

- **DtGlobalObjectDesignatorList myObjectList**

- **DtDatumRecordSet myRecordSet**

- **static char* theClassName**

- **inline const char* name() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactrspobjint.h:

class DtActionResponseFromObjInter: public DtInteractionWithEncDec

---

Inheritance:

DtActionResponseFromObjInter - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtActionResponseFromObjInter () *default constructor*
virtual	**~DtActionResponseFromObjInter** () *destructor*
	DtActionResponseFromObjInter (const DtActionResponseFromObjInter& orig) *copy constructor*
DtActionResponseFromObjInter&	**operator=** (const DtActionResponseFromObjInter& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setActionResult** (DtResponseFlag response) *Set/Get action enum*
virtual DtResponseFlag	**actionResult** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtActionResponseFromObjInter*
static void	**addCallback** (DtExerciseConn* conn, DtActionResponseFromObjInterCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtActionResponseFromObjInterCb cb, void* usr)
inline DtResponseFlag	**actionResult** () const
inline const char*	**name** () const

Protected Fields

- DtResponseFlag **myActionResult**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtActionResponseFromObjInter()`**
  default constructor
- **`virtual ~DtActionResponseFromObjInter()`**
  destructor
- **`DtActionResponseFromObjInter(const DtActionResponseFromObjInter& orig)`**
  copy constructor
- **`DtActionResponseFromObjInter& operator=( const DtActionResponseFromObjInter& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setActionResult(DtResponseFlag response)`**
  Set/Get action enum
- **`virtual DtResponseFlag actionResult() const`**
- **`virtual const char* name() const`**

Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtActionResponseFromObjInter. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtActionResponseFromObjInterCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtActionResponseFromObjInterCb cb, void* usr)**

- **DtResponseFlag myActionResult**

- **static char* theClassName**

- **inline DtResponseFlag actionResult() const**

- **inline const char* name() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactreqobjint.h:

class DtActionRequestToObjInter: public DtInteractionWithEncDec

---

Inheritance:

DtActionRequestToObjInter - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtActionRequestToObjInter () *default constructor*
virtual	**~DtActionRequestToObjInter** () *destructor*
	DtActionRequestToObjInter (const DtActionRequestToObjInter& orig) *copy constructor*
DtActionRequestToObjInter&	**operator=** (const DtActionRequestToObjInter& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual const DtGlobalObjectDesignatorList&	**objectIDs** () const *Return the Object ID list*
virtual DtGlobalObjectDesignatorList*	**objectIDs** ()
virtual void	**setNumIds** (DtU8 numIds) *Set/Get number of IDs*
virtual DtU8	**numIds** () const
virtual void	**setNthObjectId** (DtInt32 idIndex, const DtGlobalObjectDesignator& id) *Set/Get the nth ObjectId*
virtual const DtGlobalObjectDesignator&	**nthObjectId** (DtInt32 idIndex) const
virtual void	**setAction** (DtActionId action) *Set/Get action enum*
virtual DtActionId	**action** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtActionRequestToObjInter*

- static void — **addCallback** (DtExerciseConn* conn, DtActionRequestToObjInterCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void — **removeCallback** (DtExerciseConn* conn, DtActionRequestToObjInterCb cb, void* usr)

Protected Fields

- DtGlobalObjectDesignatorList **myObjectList**
- DtActionId **myAction**
- static char* **theClassName**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const

- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtActionRequestToObjInter()`**
  default constructor
- **`virtual ~DtActionRequestToObjInter()`**
  destructor
- **`DtActionRequestToObjInter( const DtActionRequestToObjInter& orig)`**
  copy constructor
- **`DtActionRequestToObjInter& operator=( const DtActionRequestToObjInter& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data

- **virtual const DtGlobalObjectDesignatorList& objectIDs() const**

  Return the Object ID list

- **virtual DtGlobalObjectDesignatorList* objectIDs()**

- **virtual void setNumIds(DtU8 numIds)**

  Set/Get number of IDs

- **virtual DtU8 numIds() const**

- **virtual void setNthObjectId(DtInt32 idIndex, const DtGlobalObjectDesignator& id)**

  Set/Get the nth ObjectId

- **virtual const DtGlobalObjectDesignator& nthObjectId(DtInt32 idIndex) const**

- **virtual void setAction(DtActionId action)**

  Set/Get action enum

- **virtual DtActionId action() const**

- **virtual const char* name() const**

  Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtActionRequestToObjInter. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtActionRequestToObjInterCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtActionRequestToObjInterCb cb, void* usr)**

- **DtGlobalObjectDesignatorList myObjectList**

- **DtActionId myAction**

- **static char* theClassName**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hackinter.h:

class DtAcknowledgeInteraction: public DtInteractionWithEncDec

---

Inheritance:

DtAcknowledgeInteraction - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtAcknowledgeInteraction () *default constructor*
virtual	**~DtAcknowledgeInteraction** () *destructor*
	DtAcknowledgeInteraction (const DtAcknowledgeInteraction& orig) *copy constructor*
DtAcknowledgeInteraction&	**operator=** (const DtAcknowledgeInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setAcknowledgeFlag** (DtAcknowledgeFlag flag) *Set/Get acknowledge flag*
virtual DtAcknowledgeFlag	**acknowledgeFlag** () const
virtual void	**setResponseFlag** (DtResponseFlag flag) *Set/Get response flag*
virtual DtResponseFlag	**responseFlag** () const
virtual void	**setRequestId** (int id) *Set/Get request ID*
virtual int	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*

- static DtInteraction* **create** ()
  Create a DtAcknowledgeInteraction
- static void **addCallback** (DtExerciseConn* conn, DtAcknowledgeInteractionCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtAcknowledgeInteractionCb cb, void* usr)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- int **myRequestId**
- DtAcknowledgeFlag **myAckFlag**
- DtResponseFlag **myRespFlag**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const

- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtAcknowledgeInteraction()`**
  default constructor
- **`virtual ~DtAcknowledgeInteraction()`**
  destructor
- **`DtAcknowledgeInteraction(const DtAcknowledgeInteraction& orig)`**
  copy constructor
- **`DtAcknowledgeInteraction& operator=(const DtAcknowledgeInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data

- **virtual void setSenderId(const DtEntityIdentifier& id)**
  Set/Get sender ID
- **virtual const DtEntityIdentifier& senderId() const**
- **virtual void setReceiverId(const DtEntityIdentifier& id)**
  Set/Get receiver ID
- **virtual const DtEntityIdentifier& receiverId() const**
- **virtual void setAcknowledgeFlag(DtAcknowledgeFlag flag)**
  Set/Get acknowledge flag
- **virtual DtAcknowledgeFlag acknowledgeFlag() const**
- **virtual void setResponseFlag(DtResponseFlag flag)**
  Set/Get response flag
- **virtual DtResponseFlag responseFlag() const**
- **virtual void setRequestId(int id)**
  Set/Get request ID
- **virtual int requestId() const**
- **virtual const char* name() const**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **static DtInteraction* create()**
  Create a DtAcknowledgeInteraction. Caller is reponsible for deletion.
- **static void addCallback(DtExerciseConn* conn, DtAcknowledgeInteractionCb cb, void* usr)**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **static void removeCallback(DtExerciseConn* conn, DtAcknowledgeInteractionCb cb, void* usr)**
- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **int myRequestId**
- **DtAcknowledgeFlag myAckFlag**
- **DtResponseFlag myRespFlag**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtViewerCommand

- **DtViewerAbsolute**
- **DtViewerTether**
- **DtViewerCompass**
- **DtViewerMimic**
- **DtViewerTurret**
- **DtViewerOrbit**
- **DtViewerTrack**
- **DtViewerTetherTrack**
- **DtViewerMimicTrack**
- **DtViewerGroup**
- **DtViewerModeless**

Documentation

- `DtViewerAbsolute`
- `DtViewerTether`
- `DtViewerCompass`
- `DtViewerMimic`
- `DtViewerTurret`
- `DtViewerOrbit`
- `DtViewerTrack`
- `DtViewerTetherTrack`
- `DtViewerMimicTrack`
- `DtViewerGroup`
- `DtViewerModeless`

In file vcpdu.h:

class DtViewControlPdu: public DtPdu

Inheritance:

DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

static DtPdu*	**create** (const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
const DtGlobalObjectDesignator&	**viewerId** () const
const DtGlobalObjectDesignator&	**senderId** () const
DtViewerCommand	**viewerCommand** () const
unsigned long	**ignoreFlags** () const
void	**setViewerId** (const DtGlobalObjectDesignator& id)
void	**setSenderId** (const DtGlobalObjectDesignator& id)
void	**setIgnoreFlags** (unsigned long flags)
virtual void	**setFromString** (const char *)
virtual char*	**string** () const
void	**printData** () const

Protected Fields

DtGlobalObjectDesignator	**retId**

Protected Methods

DtPduKind	**internalGetPduKind** () const
void	**setViewerCommand** (DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)

- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- **`static DtPdu* create(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**
- **`const DtGlobalObjectDesignator& viewerId() const`**
- **`const DtGlobalObjectDesignator& senderId() const`**
- **`DtViewerCommand viewerCommand() const`**
- **`unsigned long ignoreFlags() const`**
- **`void setViewerId(const DtGlobalObjectDesignator& id)`**
- **`void setSenderId(const DtGlobalObjectDesignator& id)`**
- **`void setIgnoreFlags(unsigned long flags)`**
- **`virtual void setFromString(const char *)`**
- **`virtual char* string() const`**
- **`void printData() const`**
- **`DtPduKind internalGetPduKind() const`**
- **`DtGlobalObjectDesignator retId`**
- **`void setViewerCommand(DtViewerCommand com)`**

Direct child classes:

DtUnknownVCPdu
DtTrackVCPdu
DtTetherVCPdu
DtTetherTrackVCPdu
DtOrbitVCPdu
DtModelessVCPdu
DtMimicVCPdu
DtMimicTrackVCPdu
DtGroupVCPdu
DtCompassVCPdu
DtAbsoluteVCPdu

Alphabetic index *Hierarchy of classes*

In file decfactory.h:

class DtStateDecoderFactory

class DtStateDecoderFactory: Instances of DtStateDecoderFactory are used to create decoders which know how to decode an HLA state message (or network representation) into a state repository (or host representation)

Inheritance:

DtStateDecoderFactory

Public Methods

	DtStateDecoderFactory (DtExerciseConn* exConn) *constructor*
virtual	**~DtStateDecoderFactory** () *destructor*
	DtStateDecoderFactory (const DtStateDecoderFactory& orig) *copy constructor Copies across added decoder creation functions*
DtStateDecoderFactory&	**operator=** (const DtStateDecoderFactory& orig) *assignment operator Copies across added decoder creation functions*
virtual DtHlaStateDecoder*	**createDecoder** (RTI::ObjectClassHandle hand) *Create a decoder for the specified object class*
virtual DtHlaStateDecoder*	**addDecoder** (RTI::ObjectClassHandle hand, DtHlaStateDecoder* decoder) *Add a decoder prototype to the factory*
virtual void	**addAttributeDecoder** (RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand, DtHlaStateDecoder::DtAttributeDecoder decoder, DtBoolean applyToSubclasses = DtTrue) *This function is used to add an attribute decoder function to several prototype decoders all at once - namely, those associated with the object class denoted by hand, as well as all of its subclasses*
void	**addAttributeDecoder** (char* className, RTI::AttributeHandle attrHand, DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify class by name*
void	**addAttributeDecoder** (RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify attr by name*
void	**addAttributeDecoder** (char* className, char* attrName, DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify class and attr by name*
virtual DtHlaStateDecoder*	**decoder** (RTI::ObjectClassHandle hand, DtBoolean defaultToBase = DtFalse) *Returns the decoder prototype associated with handle by a previous call to addDecoder*

Protected Fields

- DtHlaStateDecoder** **myProtos**
  myProtos is an array of DtHlaObject::maxObjectClasses() items
- DtExerciseConn* **myExConn**

Documentation

class DtStateDecoderFactory: Instances of DtStateDecoderFactory are used to create decoders which know how to decode an HLA state message (or network representation) into a state repository (or host representation)

- **`DtStateDecoderFactory(DtExerciseConn* exConn)`**

  constructor

- **`virtual ~DtStateDecoderFactory()`**

  destructor

- **`DtStateDecoderFactory(const DtStateDecoderFactory& orig)`**

  copy constructor Copies across added decoder creation functions

- **`DtStateDecoderFactory& operator=(const DtStateDecoderFactory& orig)`**

  assignment operator Copies across added decoder creation functions

- **`virtual DtHlaStateDecoder* createDecoder(RTI::ObjectClassHandle hand)`**

  Create a decoder for the specified object class. Note: Caller is responsible for deleting. Returns NULL if decoder creation function is not in the factory.

- **`virtual DtHlaStateDecoder* addDecoder( RTI::ObjectClassHandle hand, DtHlaStateDecoder* decoder)`**

  Add a decoder prototype to the factory. The decoder prototype will be cloned by createDecoder when called for the appropriate handle. Caller deletes the decoder, but AFTER the decoder factory no longer needs it. Returns any previous decoder which was replaced by the add.

- **`virtual void addAttributeDecoder( RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand, DtHlaStateDecoder::DtAttributeDecoder decoder, DtBoolean applyToSubclasses = DtTrue)`**

  This function is used to add an attribute decoder function to several prototype decoders all at once - namely, those associated with the object class denoted by hand, as well as all of its subclasses. If applyToSubclasses is DtFalse, then the function is only registered with the one decoder prototype, making it equivalent to decoder(hand)->addDecoder(hand, decoder));

- **`void addAttributeDecoder( char* className, RTI::AttributeHandle attrHand, DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue)`**

  Same as above, but identify class by name

- **`void addAttributeDecoder( RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue)`**

  Same as above, but identify attr by name

- **`void addAttributeDecoder( char* className, char* attrName,`**

DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify class and attr by name

virtual DtHlaStateDecoder* decoder(RTI::ObjectClassHandle hand, DtBoolean defaultToBase = DtFalse)

Returns the decoder prototype associated with handle by a previous call to addDecoder. If defaultToBase is DtTrue, then if there is no decoder registered with hand, rather than returning NULL, we search up the class hierarchy, and return the decoder registered with the most specific superclass for which a decoder has been registered.

DtHlaStateDecoder myProtos**

myProtos is an array of DtHlaObject::maxObjectClasses() items

DtExerciseConn* myExConn

Direct child classes:

DtRprStateDecoderFactory

Alphabetic index *Hierarchy of classes*

In file encfactory.h:

class DtStateEncoderFactory

class DtStateEncoderFactory: Instances of DtStateEncoderFactory are used to create encoders which know how to encode a state repository (or host representation) into an HLA state message (or network representation)

Inheritance:

DtStateEncoderFactory

Public Methods

	DtStateEncoderFactory (DtExerciseConn* exConn) *constructor*
virtual	**~DtStateEncoderFactory** () *destructor*
	DtStateEncoderFactory (const DtStateEncoderFactory& orig) *copy constructor*
DtStateEncoderFactory&	**operator=** (const DtStateEncoderFactory& orig) *assignment operator*
virtual DtHlaStateEncoder*	**createEncoder** (RTI::ObjectClassHandle hand) *Create an encoder for the specified object class*
virtual DtHlaStateEncoder*	**addEncoder** (RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder) *Add a encoder prototype to the factory*
virtual void	**addAttributeEncoder** (RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeEncoder encoder, DtBoolean applyToSubclasses = DtTrue) *This function is used to add an attribute encoder function to several prototype encoders all at once - namely, those associated with the object class denoted by hand, as well as all of its subclasses*
virtual void	**addAttributeEncoder** (char* className, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify class by name*
virtual void	**addAttributeEncoder** (RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify attr by name*
virtual void	**addAttributeEncoder** (char* className, char* attrName, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify class and attr by name*

- virtual void **addAttributeChecker** (RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeChecker encoder, DtBoolean applyToSubclasses = DtTrue)
  This function is used to add an attribute checker function to several prototype encoders all at once - namely, those associated with the object class denoted by hand, as well as all of its subclasses
- virtual void **addAttributeChecker** (char* className, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)
  Same as above, but identify class by name
- virtual void **addAttributeChecker** (RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)
  Same as above, but identify attr by name
- virtual void **addAttributeChecker** (char* className, char* attrName, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)
  Same as above, but identify class and attr by name
- virtual DtHlaStateEncoder* **encoder** (RTI::ObjectClassHandle hand, DtBoolean defaultToBase = DtFalse)
  Returns the encoder prototype associated with handle by a previous call to addEncoder

Protected Fields

- DtHlaStateEncoder** **myProtos**
  myProtos is an array of DtHlaObject::maxObjectClasses() items
- DtExerciseConn* **myExConn**

Documentation

class DtStateEncoderFactory: Instances of DtStateEncoderFactory are used to create encoders which know how to encode a state repository (or host representation) into an HLA state message (or network representation)

- **`DtStateEncoderFactory(DtExerciseConn* exConn)`**

  constructor

- **`virtual ~DtStateEncoderFactory()`**

  destructor

- **`DtStateEncoderFactory(const DtStateEncoderFactory& orig)`**

  copy constructor

- **`DtStateEncoderFactory& operator=(const DtStateEncoderFactory& orig)`**

  assignment operator

- **`virtual DtHlaStateEncoder* createEncoder(RTI::ObjectClassHandle hand)`**

  Create an encoder for the specified object class. Note: Caller is responsible for deleting. Returns NULL if encoder creation function is not in the factory.

- **`virtual DtHlaStateEncoder* addEncoder( RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder)`**

  Add a encoder prototype to the factory. The encoder prototype will be cloned by createEncoder when called for the appropriate handle. Caller deletes the encoder, but AFTER the encoder factory no longer needs it. Returns

any previous encoder which was replaced by the add.

virtual void addAttributeEncoder(RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeEncoder encoder, DtBoolean applyToSubclasses = DtTrue)

This function is used to add an attribute encoder function to several prototype encoders all at once - namely, those associated with the object class denoted by hand, as well as all of its subclasses. If applyToSubclasses is DtFalse, then the function is only registered with the one encoder prototype, making it equivalent to encoder(hand)->addEncoder(hand, encoder));

virtual void addAttributeEncoder(char* className, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify class by name

virtual void addAttributeEncoder(RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify attr by name

virtual void addAttributeEncoder(char* className, char* attrName, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify class and attr by name

virtual void addAttributeChecker(RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeChecker encoder, DtBoolean applyToSubclasses = DtTrue)

This function is used to add an attribute checker function to several prototype encoders all at once - namely, those associated with the object class denoted by hand, as well as all of its subclasses. If applyToSubclasses is DtFalse, then the function is only registered with the one encoder prototype, making it equivalent to encoder(hand)->addChecker(hand, encoder));

virtual void addAttributeChecker(char* className, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify class by name

virtual void addAttributeChecker(RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify attr by name

virtual void addAttributeChecker(char* className, char* attrName, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify class and attr by name

virtual DtHlaStateEncoder* encoder(RTI::ObjectClassHandle hand, DtBoolean defaultToBase = DtFalse)

Returns the encoder prototype associated with handle by a previous call to addEncoder. If defaultToBase is DtTrue, then if there is no encoder registered with hand, rather than returning NULL, we search up the class hierarchy, and return the encoder registered with the most specific superclass for which a encoder has been registered.

DtHlaStateEncoder myProtos**

myProtos is an array of DtHlaObject::maxObjectClasses() items

DtExerciseConn* myExConn

Direct child classes:

DtRprStateEncoderFactory

Alphabetic index *Hierarchy of classes*

In file intdecfact.h:

class DtInteractionDecoderFactory

class DtInteractionDecoderFactory: Instances of DtInteractionDecoderFactory are used to create decoders which know how to decode an HLA interaction message (or network representation) into an interaction repository (or host representation)

Inheritance:

DtInteractionDecoderFactory

Public Methods

	DtInteractionDecoderFactory (DtExerciseConn* exConn) *constructor*
virtual	**~DtInteractionDecoderFactory** () *destructor*
	DtInteractionDecoderFactory (const DtInteractionDecoderFactory& orig) *copy constructor Copies across added decoder creation functions*
DtInteractionDecoderFactory&	**operator=** (const DtInteractionDecoderFactory& orig) *assignment operator Copies across added decoder creation functions*
virtual DtInteractionDecoder*	**createDecoder** (RTI::InteractionClassHandle hand) *Create a decoder for the specified interaction class*
virtual DtInteractionDecoder*	**addDecoder** (RTI::InteractionClassHandle hand, DtInteractionDecoder* decoder) *Add a decoder prototype to the factory*
virtual void	**addParameterDecoder** (RTI::InteractionClassHandle classHand, RTI::ParameterHandle paramHand, DtInteractionDecoder::DtParameterDecoder decoder, DtBoolean applyToSubclasses = DtTrue) *This function is used to add an parameter decoder function to several prototype decoders all at once - namely, those associated with the interaction class denoted by hand, as well as all of its subclasses*
void	**addParameterDecoder** (char* className, RTI::ParameterHandle paramHand, DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify class by name*
void	**addParameterDecoder** (RTI::InteractionClassHandle classHand, char* paramName, DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify param by name*
void	**addParameterDecoder** (char* className, char* paramName, DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify class and param by name*

- virtual DtInteractionDecoder* **decoder** (RTI::InteractionClassHandle hand)
  Returns the decoder prototype associated with handle by a previous call to addDecoder

Protected Fields

- DtInteractionDecoder** **myProtos**
  myProtos is an array of DtInteraction::maxInteractionClasses() items
- DtExerciseConn* **myExConn**

Documentation

class DtInteractionDecoderFactory: Instances of DtInteractionDecoderFactory are used to create decoders which know how to decode an HLA interaction message (or network representation) into an interaction repository (or host representation)

- **`DtInteractionDecoderFactory(DtExerciseConn* exConn)`**

  constructor

- **`virtual ~DtInteractionDecoderFactory()`**

  destructor

- **`DtInteractionDecoderFactory(const DtInteractionDecoderFactory& orig)`**

  copy constructor Copies across added decoder creation functions

- **`DtInteractionDecoderFactory& operator=( const DtInteractionDecoderFactory& orig)`**

  assignment operator Copies across added decoder creation functions

- **`virtual DtInteractionDecoder* createDecoder( RTI::InteractionClassHandle hand)`**

  Create a decoder for the specified interaction class. Note: Caller is responsible for deleting. Returns NULL if decoder creation function is not in the factory.

- **`virtual DtInteractionDecoder* addDecoder( RTI::InteractionClassHandle hand, DtInteractionDecoder* decoder)`**

  Add a decoder prototype to the factory. The decoder prototype will be cloned by createDecoder when called for the appropriate handle. Caller deletes the decoder, but AFTER the decoder factory no longer needs it. Returns any previous decoder which was replaced by the add.

- **`virtual void addParameterDecoder( RTI::InteractionClassHandle classHand, RTI::ParameterHandle paramHand, DtInteractionDecoder::DtParameterDecoder decoder, DtBoolean applyToSubclasses = DtTrue)`**

  This function is used to add an parameter decoder function to several prototype decoders all at once - namely, those associated with the interaction class denoted by hand, as well as all of its subclasses. If applyToSubclasses is DtFalse, then the function is only registered with the one decoder prototype, making it equivalent to decoder(hand)->addDecoder(hand, decoder));

- **`void addParameterDecoder( char* className, RTI::ParameterHandle paramHand, DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue)`**

  Same as above, but identify class by name

- **`void addParameterDecoder( RTI::InteractionClassHandle classHand, char* paramName,`**

DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify param by name

- **void addParameterDecoder(char* className, char* paramName, DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue)**

Same as above, but identify class and param by name

- **virtual DtInteractionDecoder* decoder(RTI::InteractionClassHandle hand)**

Returns the decoder prototype associated with handle by a previous call to addDecoder

- **DtInteractionDecoder** myProtos**

myProtos is an array of DtInteraction::maxInteractionClasses() items

- **DtExerciseConn* myExConn**

Direct child classes:

DtRprInteractionDecoderFactory

Alphabetic index *Hierarchy of classes*

In file intencfact.h:

class DtInteractionEncoderFactory

class DtInteractionEncoderFactory: Instances of DtInteractionEncoderFactory are used to create encoders which know how to encode an interaction repository (or host representation) into an HLA interaction message (or network representation)

Inheritance:

DtInteractionEncoderFactory

Public Methods

	DtInteractionEncoderFactory (DtExerciseConn* exConn) *constructor*
virtual	**~DtInteractionEncoderFactory** () *destructor*
	DtInteractionEncoderFactory (const DtInteractionEncoderFactory& orig) *copy constructor*
DtInteractionEncoderFactory&	**operator=** (const DtInteractionEncoderFactory& orig) *assignment operator*
virtual DtInteractionEncoder*	**createEncoder** (RTI::InteractionClassHandle hand) *Create an encoder for the specified interaction class*
virtual DtInteractionEncoder*	**addEncoder** (RTI::InteractionClassHandle hand, DtInteractionEncoder* encoder) *Add a encoder prototype to the factory*
virtual void	**addParameterEncoder** (RTI::InteractionClassHandle classHand, RTI::ParameterHandle paramHand, DtInteractionEncoder::DtParameterEncoder encoder, DtBoolean applyToSubclasses = DtTrue) *This function is used to add an parameter encoder function to several prototype encoders all at once - namely, those associated with the interaction class denoted by hand, as well as all of its subclasses*
void	**addParameterEncoder** (char* className, RTI::ParameterHandle paramHand, DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify class by name*
void	**addParameterEncoder** (RTI::InteractionClassHandle classHand, char* paramName, DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify param by name*
void	**addParameterEncoder** (char* className, char* paramName, DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue) *Same as above, but identify class and param by name*

- virtual DtInteractionEncoder* **encoder** (RTI::InteractionClassHandle hand)
  Returns the encoder prototype associated with handle by a previous call to addEncoder

Protected Fields

- DtInteractionEncoder** **myProtos**
  myProtos is an array of DtInteraction::maxInteractionClasses() items
- DtExerciseConn* **myExConn**

Documentation

class DtInteractionEncoderFactory: Instances of DtInteractionEncoderFactory are used to create encoders which know how to encode an interaction repository (or host representation) into an HLA interaction message (or network representation)

- **`DtInteractionEncoderFactory(DtExerciseConn* exConn)`**

  constructor

- **`virtual ~DtInteractionEncoderFactory()`**

  destructor

- **`DtInteractionEncoderFactory(const DtInteractionEncoderFactory& orig)`**

  copy constructor

- **`DtInteractionEncoderFactory& operator=( const DtInteractionEncoderFactory& orig)`**

  assignment operator

- **`virtual DtInteractionEncoder* createEncoder( RTI::InteractionClassHandle hand)`**

  Create an encoder for the specified interaction class. Note: Caller is responsible for deleting. Returns NULL if encoder creation function is not in the factory.

- **`virtual DtInteractionEncoder* addEncoder( RTI::InteractionClassHandle hand, DtInteractionEncoder* encoder)`**

  Add a encoder prototype to the factory. The encoder prototype will be cloned by createEncoder when called for the appropriate handle. Caller deletes the encoder, but AFTER the encoder factory no longer needs it. Returns any previous encoder which was replaced by the add.

- **`virtual void addParameterEncoder( RTI::InteractionClassHandle classHand, RTI::ParameterHandle paramHand, DtInteractionEncoder::DtParameterEncoder encoder, DtBoolean applyToSubclasses = DtTrue)`**

  This function is used to add an parameter encoder function to several prototype encoders all at once - namely, those associated with the interaction class denoted by hand, as well as all of its subclasses. If applyToSubclasses is DtFalse, then the function is only registered with the one encoder prototype, making it equivalent to encoder(hand)->addEncoder(hand, encoder));

- **`void addParameterEncoder( char* className, RTI::ParameterHandle paramHand, DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue)`**

  Same as above, but identify class by name

- **`void addParameterEncoder( RTI::InteractionClassHandle classHand, char* paramName,`**

DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue)

Same as above, but identify param by name

- **void addParameterEncoder(char* className, char* paramName, DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue)**

Same as above, but identify class and param by name

- **virtual DtInteractionEncoder* encoder(RTI::InteractionClassHandle hand)**

Returns the encoder prototype associated with handle by a previous call to addEncoder

- **DtInteractionEncoder** myProtos**

myProtos is an array of DtInteraction::maxInteractionClasses() items

- **DtExerciseConn* myExConn**

Direct child classes:

DtRprInteractionEncoderFactory

Alphabetic index *Hierarchy of classes*

In file hashlist.h:

class DtHashlist

Public Methods

	DtHashlist (int numBuckets = dfltNumBuckets) *Default constructor*
	DtHashlist (DtList* list, int numBuckets = dfltNumBuckets) *Constructor for a DtHashlist that uses an external DtList constructed by the user*
virtual	**~DtHashlist** () *Destructor*
	DtHashlist (const DtHashlist& orig) *Copy constructor - always uses an internal list, so if you want your copy to use an external list, do not use the copy contructor*
DtHashlist&	**operator=** (const DtHashlist& orig) *Assignment operator - Does not replace the DtHashlist's list*
virtual DtListItem*	**add** (const DtBaseHashKey& key, void* data, void** oldData = NULL) *Add something to the list*
virtual void*	**remove** (const DtBaseHashKey& key) *Remove something from the list*
virtual void*	**lookup** (const DtBaseHashKey& key) const *Lookup something in the list, return data pointer*
virtual void*	**operator[]** (const DtBaseHashKey& key) const
virtual const DtList*	**list** () const *Returns a DtList of void*'s which are the data that you have added to the hashlist (independent of hash keys)*
virtual void	**print** () const *Prints the data in the hashlist*
virtual const DtList*	**keys** () const *Returns a DtList of DtHashKey*'s representing the set of keys with which data is currently associated*
virtual void	**empty** () *Remove all keys and data from the list*
virtual void	**setFromOther** (const DtHashlist& orig) *Sets the contents of this hashlist to match those of other*
inline void*	**operator[]** (const DtBaseHashKey& key) const

Protected Fields

int	**myNumBuckets**
const DtBoolean	**myListIsMine**
DtList*	**myList** *Regular DtList containing all of the items added to the hashlist, without regard to key information*

- DtList* **myBuckets**
  Array of buckets (DtLists of DtHashItems)
- DtList **myKeys**
  DtList of keys (DtBaseHashKeys)
- static int **theNumBucketsDefault**

Protected Methods

- virtual DtListItem* **itemLookup** (const DtBaseHashKey& key) const
  Lookup something in the list, return DtListItem
- virtual DtList& **bucketFromHashKey** (const DtBaseHashKey& key) const
  Determine the bucket from the hashkey
- virtual DtListItem* **bucketItemLookup** (const DtBaseHashKey& key) const
  Lookup an item in a bucket

Documentation

- **`DtHashlist(int numBuckets = dfltNumBuckets)`**

  Default constructor. Creates and uses a regular DtList to store the list of data items that you add.

- **`DtHashlist(DtList* list, int numBuckets = dfltNumBuckets)`**

  Constructor for a DtHashlist that uses an external DtList constructed by the user

- **`virtual ~DtHashlist()`**

  Destructor

- **`DtHashlist(const DtHashlist& orig)`**

  Copy constructor - always uses an internal list, so if you want your copy to use an external list, do not use the copy contructor. Instead, use the standard with-external-list constructor to create an empty hash list, then use setFromOther or the assignment operator to copy the contents.

- **`DtHashlist& operator=(const DtHashlist& orig)`**

  Assignment operator - Does not replace the DtHashlist's list. Just empties the DtHashlist, and then sets its contents to those of the original using setFromOther.

- **`virtual DtListItem* add(const DtBaseHashKey& key, void* data, void** oldData = NULL)`**

  Add something to the list. The actual pointer is stored, so you should make sure the pointer remains valid until after you remove the data from the hash list. At any given time, only one data item may be associated with a particular key. If add is called with a key with which an item is already associated, the old item is replaced with the new, and the old item is returned in the optional oldData argument for deletion by the user. Otherwise, oldData will be set to NULL.

- **`virtual void* remove(const DtBaseHashKey& key)`**

  Remove something from the list. Returns the data that was associated with the key, so that you can cast it to the right type and delete it.

- **`virtual void* lookup(const DtBaseHashKey& key) const`**

  Lookup something in the list, return data pointer

- **virtual void* operator[] (const DtBaseHashKey& key) const**
- **virtual const DtList* list() const**

  Returns a DtList of void*'s which are the data that you have added to the hashlist (independent of hash keys)
- **virtual void print() const**

  Prints the data in the hashlist.
- **virtual const DtList* keys() const**

  Returns a DtList of DtHashKey*'s representing the set of keys with which data is currently associated
- **virtual void empty()**

  Remove all keys and data from the list
- **virtual void setFromOther(const DtHashlist& orig)**

  Sets the contents of this hashlist to match those of other. Data pointers that have been added to orig are copied, but the items that they point to are not.
- **virtual DtListItem* itemLookup(const DtBaseHashKey& key) const**

  Lookup something in the list, return DtListItem
- **virtual DtList& bucketFromHashKey(const DtBaseHashKey& key) const**

  Determine the bucket from the hashkey
- **virtual DtListItem* bucketItemLookup(const DtBaseHashKey& key) const**

  Lookup an item in a bucket
- **int myNumBuckets**
- **const DtBoolean myListIsMine**
- **DtList* myList**

  Regular DtList containing all of the items added to the hashlist, without regard to key information
- **DtList* myBuckets**

  Array of buckets (DtLists of DtHashItems)
- **DtList myKeys**

  DtList of keys (DtBaseHashKeys)
- **static int theNumBucketsDefault**
- **inline void* operator[] (const DtBaseHashKey& key) const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file emptyfommap.h:

class DtEmptyFomMapper: public DtFomMapper

class DtEmptyFomMapper: An instances of DtEmptyFomMapper is a DtFomMapper that has no mappings registered between FOM classes and VR-Link classes

---

Inheritance:

DtEmptyFomMapper - DtFomMapper

Public Methods

	DtEmptyFomMapper () *Default constructor*
virtual	**~DtEmptyFomMapper** () *destructor*
virtual void	**init** (DtExerciseConn* conn) *Initialize the instance, once an exercise connection is available*
DtStateDecoderFactory*	**dfltStateDecoderFactory** () *Returns the default state decoder factory*
DtStateEncoderFactory*	**dfltStateEncoderFactory** () *Returns the default state encoder factory*
DtInteractionDecoderFactory*	**dfltInteractionDecoderFactory** () *Returns the default interaction decoder factory*
DtInteractionEncoderFactory*	**dfltInteractionEncoderFactory** () *Returns the default state interaction factory*
DtHlaObjectFactory*	**dfltHlaObjectFactory** () *Returns the default HlaObject factory*
DtInteractionFactory*	**dfltInteractionFactory** () *Returns the default Interaction factory*
static DtFomMapper*	**create** (void* usr = NULL) *A Creator function that returns a DtEmptyFomMapper configured with no mappings at all*

Protected Fields

DtStateDecoderFactory*	**myDfltDecFact** *Default factories and other objects*
DtStateEncoderFactory*	**myDfltEncFact**
DtInteractionDecoderFactory*	**myDfltInterDecFact**
DtInteractionEncoderFactory*	**myDfltInterEncFact**

- DtHlaObjectFactory* **myDfltObjFact**
- DtInteractionFactory* **myDfltInterFact**

Protected Methods

- **DtEmptyFomMapper** (const DtEmptyFomMapper& orig)
  copy constructor - not implemented
- DtEmptyFomMapper& **operator=** (const DtEmptyFomMapper& orig)
  assignment operator - not implemented

Inherited from DtFomMapper:

Public Methods

- typedef char***(*DtInteractionClassChooser)**(DtExerciseConn* exConn, const DtInteraction& inter)
- typedef char***(*DtObjectClassChooser)**(const char* vrlinkClassName, DtExerciseConn* conn, void* usr)
- virtual DtStateDecoderFactory* **setStateDecoderFactory**(DtStateDecoderFactory* fact)
- virtual DtStateDecoderFactory* **stateDecoderFactory**()
- virtual DtStateEncoderFactory* **setStateEncoderFactory**(DtStateEncoderFactory* fact)
- virtual DtStateEncoderFactory* **stateEncoderFactory**()
- virtual DtInteractionDecoderFactory* **setInteractionDecoderFactory**(DtInteractionDecoderFactory* fact)
- virtual DtInteractionDecoderFactory* **interactionDecoderFactory**()
- virtual DtInteractionEncoderFactory* **setInteractionEncoderFactory**(DtInteractionEncoderFactory* fact)
- virtual DtInteractionEncoderFactory* **interactionEncoderFactory**()
- virtual DtHlaObjectFactory* **setHlaObjectFactory**(DtHlaObjectFactory* fact)
- virtual DtHlaObjectFactory* **hlaObjectFactory**()
- virtual DtInteractionFactory* **setInteractionFactory**(DtInteractionFactory* fact)
- virtual DtInteractionFactory* **interactionFactory**()
- virtual void **setObjectClassToChoose**(const char* vrlinkClassName, const char* fomClassName)
- virtual void **setObjectClassChooser**(const char* vrlinkClassName, DtObjectClassChooser chooser)
- virtual char* **chooseObjectClass**(const char* vrlinkClassName, void* usr = NULL) const
- virtual void **setInteractionClassToChoose**(const char* vrlinkClassName, const char* fomClassName)
- virtual void **setInteractionClassChooser**(const char* vrlinkClassName, DtInteractionClassChooser chooser)
- virtual char* **chooseInteractionClass**(const DtInteraction& inter) const

- virtual void **setInteractionClasses**(const char* vrlinkClassName, const DtList& classes)
- virtual DtList* **interactionClasses**(const char* vrlinkClassName)
- virtual void **setInteractionClass**(const char* vrlinkClassName, const char* fomClassName)
- virtual void **setObjectClassNames**(const char* vrlinkClassName, const DtList& classes)
- virtual DtList* **objectClassNames**(const char* vrlinkClassName)
- virtual DtList* **objectClassHandles**(const char* vrlinkClassName)
- virtual void **setObjectClass**(const char* vrlinkClassName, const char* fomClassName)
- virtual void **setVersion**(double vers)
- virtual double **version**() const

Protected Fields

- DtExerciseConn* **myExConn**
- DtStateDecoderFactory* **myDecFact**
- DtStateEncoderFactory* **myEncFact**
- DtInteractionDecoderFactory* **myInterDecFact**
- DtInteractionEncoderFactory* **myInterEncFact**
- DtHlaObjectFactory* **myObjFact**
- DtInteractionFactory* **myInterFact**
- DtHashlist **myIntMapList**
- DtHashlist **myObjMapList**
- DtHashlist **myObjChooserList**
- DtHashlist **mySimpleObjChooserList**
- DtHashlist **myInterChooserList**
- DtHashlist **mySimpleInterChooserList**
- DtList **myObjHandleList**
- DtList **myEmptyList**
- double **myVersion**

Protected Methods

- virtual DtObjectClassChooser **objectClassChooser**(const char* vrlinkClassName) const
- virtual DtInteractionClassChooser **interactionClassChooser**(const char* vrlinkClassName) const

---

Documentation

class DtEmptyFomMapper:

An instances of DtEmptyFomMapper is a DtFomMapper that has no mappings registered between FOM classes and VR-Link classes. Mappings can be added later.

- **`DtEmptyFomMapper()`**

  Default constructor. Typically, most initialization takes place within init(), which is called by DtExerciseConn's constructor.

- **`virtual ~DtEmptyFomMapper()`**

  destructor

- **`virtual void init(DtExerciseConn* conn)`**

  Initialize the instance, once an exercise connection is available. Called by DtExerciseConn constructor.

- **`DtStateDecoderFactory* dfltStateDecoderFactory()`**

  Returns the default state decoder factory

- **`DtStateEncoderFactory* dfltStateEncoderFactory()`**

  Returns the default state encoder factory

- **`DtInteractionDecoderFactory* dfltInteractionDecoderFactory()`**

  Returns the default interaction decoder factory

- **`DtInteractionEncoderFactory* dfltInteractionEncoderFactory()`**

  Returns the default state interaction factory

- **`DtHlaObjectFactory* dfltHlaObjectFactory()`**

  Returns the default HlaObject factory

- **`DtInteractionFactory* dfltInteractionFactory()`**

  Returns the default Interaction factory

- **`static DtFomMapper* create(void* usr = NULL)`**

  A Creator function that returns a DtEmptyFomMapper configured with no mappings at all

- **`DtEmptyFomMapper(const DtEmptyFomMapper& orig)`**

  copy constructor - not implemented

- **`DtEmptyFomMapper& operator=(const DtEmptyFomMapper& orig)`**

  assignment operator - not implemented

- **`DtStateDecoderFactory* myDfltDecFact`**

Default factories and other objects

- **DtStateEncoderFactory* myDfltEncFact**
- **DtInteractionDecoderFactory* myDfltInterDecFact**
- **DtInteractionEncoderFactory* myDfltInterEncFact**
- **DtHlaObjectFactory* myDfltObjFact**
- **DtInteractionFactory* myDfltInterFact**

Direct child classes:

DtRprFomMapper

Alphabetic index *Hierarchy of classes*

In file dirrtiamb.h:

class DtDirectRtiAmbassador: public DtVrlRtiAmbassador

class DtDirectRtiAmbassador: Instances of DtDirectRtiAmbassador are for wrapping the calls to The RTI ambassador so that we may override them

---

Inheritance:

DtDirectRtiAmbassador - DtVrlRtiAmbassador

Public Methods

	DtDirectRtiAmbassador () *default constructor*
virtual	**~DtDirectRtiAmbassador** () *destructor*
virtual RTI::RTIambassador*	**rtiAmb** () *Get the real RTI ambassador*
virtual DtBoolean	**setReflecting** (DtBoolean onOff) *Set/Get whether to reflect back local objects and interactions, if supported by the derived class*
virtual DtBoolean	**reflecting** () const
virtual void	**createFederationExecution** (const char *executionName, const char *FED) *Calls down to the RTI function of the same name*
virtual void	**destroyFederationExecution** (const char *executionName) *Calls down to the RTI function of the same name*
virtual RTI::FederateHandle	**joinFederationExecution** (const char* yourName, const char* executionName, RTI::FederateAmbassadorPtr federateAmbassadorReference) *Calls down to the RTI function of the same name*
virtual void	**resignFederationExecution** (RTI::ResignAction theAction) *Calls down to the RTI function of the same name*
virtual void	**registerFederationSynchronizationPoint** (const char* label, const char* theTag) *Calls down to the RTI function of the same name*
virtual void	**registerFederationSynchronizationPoint** (const char* label, const char* theTag, const RTI::FederateHandleSet& syncSet) *Calls down to the RTI function of the same name*
virtual void	**synchronizationPointAchieved** (const char* label) *Calls down to the RTI function of the same name*
virtual void	**requestFederationSave** (const char* label, const RTI::FedTime& theTime) *Calls down to the RTI function of the same name*

virtual void	**requestFederationSave** (const char *label) *Calls down to the RTI function of the same name*
virtual void	**federateSaveBegun** () *Calls down to the RTI function of the same name*
virtual void	**federateSaveComplete** () *Calls down to the RTI function of the same name*
virtual void	**federateSaveNotComplete** () *Calls down to the RTI function of the same name*
virtual void	**requestFederationRestore** (const char* label) *Calls down to the RTI function of the same name*
virtual void	**federateRestoreComplete** () *Calls down to the RTI function of the same name*
virtual void	**federateRestoreNotComplete** () *Calls down to the RTI function of the same name*
virtual void	**publishObjectClass** (RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& attributeList) *Calls down to the RTI function of the same name*
virtual void	**unpublishObjectClass** (RTI::ObjectClassHandle theClass) *Calls down to the RTI function of the same name*
virtual void	**publishInteractionClass** (RTI::InteractionClassHandle theInteraction) *Calls down to the RTI function of the same name*
virtual void	**unpublishInteractionClass** (RTI::InteractionClassHandle theInteraction) *Calls down to the RTI function of the same name*
virtual void	**subscribeObjectClassAttributes** (RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& attributeList, RTI::Boolean active = RTI::RTI_TRUE) *Calls down to the RTI function of the same name*
virtual void	**unsubscribeObjectClass** (RTI::ObjectClassHandle theClass) *Calls down to the RTI function of the same name*
virtual void	**subscribeInteractionClass** (RTI::InteractionClassHandle theClass, RTI::Boolean active = RTI::RTI_TRUE) *Calls down to the RTI function of the same name*
virtual void	**unsubscribeInteractionClass** (RTI::InteractionClassHandle theClass) *Calls down to the RTI function of the same name*
virtual RTI::ObjectHandle	**registerObjectInstance** (RTI::ObjectClassHandle theClass, const char* theObject) *Calls down to the RTI function of the same name*
virtual RTI::ObjectHandle	**registerObjectInstance** (RTI::ObjectClassHandle theClass) *Calls down to the RTI function of the same name*
virtual RTI::EventRetractionHandle	**updateAttributeValues** (RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const RTI::FedTime& theTime, const char* theTag) *Calls down to the RTI function of the same name*
virtual void	**updateAttributeValues** (RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const char* theTag) *Calls down to the RTI function of the same name*
virtual RTI::EventRetractionHandle	**sendInteraction** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const RTI::FedTime& theTime, const char* theTag) *Calls down to the RTI function of the same name*
virtual void	**sendInteraction** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const char* theTag) *Calls down to the RTI function of the same name*
virtual RTI::EventRetractionHandle	**deleteObjectInstance** (RTI::ObjectHandle theObject, const RTI::FedTime& theTime, const char* theTag) *Calls down to the RTI function of the same name*

virtual void	**deleteObjectInstance** (RTI::ObjectHandle theObject, const char* theTag) *Calls down to the RTI function of the same name*
virtual void	**localDeleteObjectInstance** (RTI::ObjectHandle theObject) *Calls down to the RTI function of the same name*
virtual void	**changeAttributeTransportationType** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, RTI::TransportationHandle theType) *Calls down to the RTI function of the same name*
virtual void	**changeInteractionTransportationType** (RTI::InteractionClassHandle theClass, RTI::TransportationHandle theType) *Calls down to the RTI function of the same name*
virtual void	**requestObjectAttributeValueUpdate** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) *Calls down to the RTI function of the same name*
virtual void	**requestClassAttributeValueUpdate** (RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& theAttributes) *Calls down to the RTI function of the same name*
virtual void	**unconditionalAttributeOwnershipDivestiture** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) *Calls down to the RTI function of the same name*
virtual void	**negotiatedAttributeOwnershipDivestiture** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, const char* theTag) *Calls down to the RTI function of the same name*
virtual void	**attributeOwnershipAcquisition** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& desiredAttributes, const char* theTag) *Calls down to the RTI function of the same name*
virtual void	**attributeOwnershipAcquisitionIfAvailable** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& desiredAttributes) *Calls down to the RTI function of the same name*
virtual RTI::AttributeHandleSet*	**attributeOwnershipReleaseResponse** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) *Calls down to the RTI function of the same name*
virtual void	**cancelNegotiatedAttributeOwnershipDivestiture** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) *Calls down to the RTI function of the same name*
virtual void	**cancelAttributeOwnershipAcquisition** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) *Calls down to the RTI function of the same name*
virtual void	**queryAttributeOwnership** (RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute) *Calls down to the RTI function of the same name*
virtual RTI::Boolean	**isAttributeOwnedByFederate** (RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute) *Calls down to the RTI function of the same name*
virtual void	**enableTimeRegulation** (const RTI::FedTime& theFederateTime, const RTI::FedTime& theLookahead) *Calls down to the RTI function of the same name*
virtual void	**disableTimeRegulation** () *Calls down to the RTI function of the same name*
virtual void	**enableTimeConstrained** () *Calls down to the RTI function of the same name*
virtual void	**disableTimeConstrained** () *Calls down to the RTI function of the same name*
virtual void	**timeAdvanceRequest** (const RTI::FedTime& theTime) *Calls down to the RTI function of the same name*

virtual void **timeAdvanceRequestAvailable** (const RTI::FedTime& theTime)
Calls down to the RTI function of the same name

virtual void **nextEventRequest** (const RTI::FedTime& theTime)
Calls down to the RTI function of the same name

virtual void **nextEventRequestAvailable** (const RTI::FedTime& theTime)
Calls down to the RTI function of the same name

virtual void **flushQueueRequest** (const RTI::FedTime& theTime)
Calls down to the RTI function of the same name

virtual void **enableAsynchronousDelivery** ()
Calls down to the RTI function of the same name

virtual void **disableAsynchronousDelivery** ()
Calls down to the RTI function of the same name

virtual void **queryLBTS** (RTI::FedTime& theTime)
Calls down to the RTI function of the same name

virtual void **queryFederateTime** (RTI::FedTime& theTime)
Calls down to the RTI function of the same name

virtual void **queryMinNextEventTime** (RTI::FedTime& theTime)
Calls down to the RTI function of the same name

virtual void **modifyLookahead** (const RTI::FedTime& theLookahead)
Calls down to the RTI function of the same name

virtual void **queryLookahead** (RTI::FedTime& theTime)
Calls down to the RTI function of the same name

virtual void **retract** (RTI::EventRetractionHandle theHandle)
Calls down to the RTI function of the same name

virtual void **changeAttributeOrderType** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, RTI::OrderingHandle theType)
Calls down to the RTI function of the same name

virtual void **changeInteractionOrderType** (RTI::InteractionClassHandle theClass, RTI::OrderingHandle theType)
Calls down to the RTI function of the same name

virtual void **createRegion** (RTI::SpaceHandle theSpace, RTI::ULong numberOfExtents)
Calls down to the RTI function of the same name

virtual void **notifyAboutRegionModification** (RTI::Region &theRegion)
Calls down to the RTI function of the same name

virtual void **deleteRegion** (RTI::Region *theRegion)
Calls down to the RTI function of the same name

virtual RTI::ObjectHandle **registerObjectInstanceWithRegion** (RTI::ObjectClassHandle theClass, const char* theObject, RTI::AttributeHandle theAttributes[], RTI::Region* theRegions[], RTI::ULong theNumberOfHandles)
Calls down to the RTI function of the same name

virtual RTI::ObjectHandle **registerObjectInstanceWithRegion** (RTI::ObjectClassHandle theClass, RTI::AttributeHandle theAttributes[], RTI::Region* theRegions[], RTI::ULong theNumberOfHandles)
Calls down to the RTI function of the same name

virtual void **associateRegionForUpdates** (RTI::Region& theRegion, RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)
Calls down to the RTI function of the same name

virtual void **unassociateRegionForUpdates** (RTI::Region& theRegion, RTI::ObjectHandle theObject)
Calls down to the RTI function of the same name

virtual void **subscribeObjectClassAttributesWithRegion** (RTI::ObjectClassHandle theClass, RTI::Region& theRegion, const RTI::AttributeHandleSet &attributeList, RTI::Boolean active = RTI::RTI_TRUE)
Calls down to the RTI function of the same name

- virtual void — **unsubscribeObjectClassWithRegion** (RTI::ObjectClassHandle theClass, RTI::Region& theRegion)
  Calls down to the RTI function of the same name
- virtual void — **subscribeInteractionClassWithRegion** (RTI::InteractionClassHandle theClass, RTI::Region& theRegion, RTI::Boolean active = RTI::RTI_TRUE)
  Calls down to the RTI function of the same name
- virtual void — **unsubscribeInteractionClassWithRegion** (RTI::InteractionClassHandle theClass, RTI::Region& theRegion)
  Calls down to the RTI function of the same name
- virtual RTI::EventRetractionHandle — **sendInteractionWithRegion** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet &theParameters, const RTI::FedTime& theTime, const char* theTag, const RTI::Region& theRegion)
  Calls down to the RTI function of the same name
- virtual void — **sendInteractionWithRegion** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet &theParameters, const char* theTag, const RTI::Region& theRegion)
  Calls down to the RTI function of the same name
- virtual void — **requestClassAttributeValueUpdateWithRegion** (RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& theAttributes, const RTI::Region& theRegion)
  Calls down to the RTI function of the same name
- virtual RTI::ObjectClassHandle — **getObjectClassHandle** (const char* theName)
  Calls down to the RTI function of the same name
- virtual char* — **getObjectClassName** (RTI::ObjectClassHandle theHandle)
  Calls down to the RTI function of the same name
- virtual RTI::AttributeHandle — **getAttributeHandle** (const char* theName, RTI::ObjectClassHandle whichClass)
  Calls down to the RTI function of the same name
- virtual char* — **getAttributeName** (RTI::AttributeHandle theHandle, RTI::ObjectClassHandle whichClass)
  Calls down to the RTI function of the same name
- virtual RTI::InteractionClassHandle — **getInteractionClassHandle** (const char* theName)
  Calls down to the RTI function of the same name
- virtual char* — **getInteractionClassName** (RTI::InteractionClassHandle theHandle)
  Calls down to the RTI function of the same name
- virtual RTI::ParameterHandle — **getParameterHandle** (const char* theName, RTI::InteractionClassHandle whichClass)
  Calls down to the RTI function of the same name
- virtual char* — **getParameterName** (RTI::ParameterHandle theHandle, RTI::InteractionClassHandle whichClass)
  Calls down to the RTI function of the same name
- virtual RTI::ObjectHandle — **getObjectInstanceHandle** (const char* theName)
  Calls down to the RTI function of the same name
- virtual char* — **getObjectInstanceName** (RTI::ObjectHandle theHandle)
  Calls down to the RTI function of the same name
- virtual RTI::SpaceHandle — **getRoutingSpaceHandle** (const char* theName)
  Calls down to the RTI function of the same name
- virtual const char* — **getRoutingSpaceName** (const RTI::SpaceHandle theHandle)
  Calls down to the RTI function of the same name
- virtual RTI::DimensionHandle — **getDimensionHandle** (const char* theName, RTI::SpaceHandle whichSpace)
  Calls down to the RTI function of the same name
- virtual char* — **getDimensionName** (RTI::DimensionHandle theHandle, RTI::SpaceHandle whichSpace)
  Calls down to the RTI function of the same name

- virtual RTI::SpaceHandle — **getAttributeRoutingSpaceHandle** (RTI::AttributeHandle theHandle, RTI::ObjectClassHandle whichClass)
  Calls down to the RTI function of the same name
- virtual RTI::ObjectClassHandle — **getObjectClass** (RTI::ObjectHandle theObject)
  Calls down to the RTI function of the same name
- virtual RTI::SpaceHandle — **getInteractionRoutingSpaceHandle** (RTI::InteractionClassHandle theHandle)
  Calls down to the RTI function of the same name
- virtual RTI::TransportationHandle — **getTransportationHandle** (const char *theName)
  Calls down to the RTI function of the same name
- virtual char* — **getTransportationName** (RTI::TransportationHandle theHandle)
  Calls down to the RTI function of the same name
- virtual RTI::OrderingHandle — **getOrderingHandle** (const char *theName)
  Calls down to the RTI function of the same name
- virtual char* — **getOrderingName** (RTI::OrderingHandle theHandle)
  Calls down to the RTI function of the same name
- virtual void — **enableClassRelevanceAdvisorySwitch** ()
  Calls down to the RTI function of the same name
- virtual void — **disableClassRelevanceAdvisorySwitch** ()
  Calls down to the RTI function of the same name
- virtual void — **enableAttributeRelevanceAdvisorySwitch** ()
  Calls down to the RTI function of the same name
- virtual void — **disableAttributeRelevanceAdvisorySwitch** ()
  Calls down to the RTI function of the same name
- virtual void — **enableAttributeScopeAdvisorySwitch** ()
  Calls down to the RTI function of the same name
- virtual void — **disableAttributeScopeAdvisorySwitch** ()
  Calls down to the RTI function of the same name
- virtual void — **enableInteractionRelevanceAdvisorySwitch** ()
  Calls down to the RTI function of the same name
- virtual void — **disableInteractionRelevanceAdvisorySwitch** ()
  Calls down to the RTI function of the same name
- virtual RTI::Boolean — **tick** ()
  Calls down to the RTI function of the same name
- virtual RTI::Boolean — **tick** (RTI::TickTime minimum, RTI::TickTime maximum)
  Calls down to the RTI function of the same name
- virtual RTI::RegionToken — **getRegionToken** (RTI::Region*)
  Calls down to the RTI function of the same name
- virtual RTI::Region* — **getRegion** (RTI::RegionToken)
  Calls down to the RTI function of the same name
- static DtVrlRtiAmbassador* — **create** ()
  returns a new DtDirectRtiAmbassador

Protected Fields

- RTI::RTIambassador* — **myRtiAmb**
- RTI::FederateAmbassadorPtr — **myFedAmb**
- DtBoolean — **myReflecting**

Protected Methods

	DtDirectRtiAmbassador (const DtDirectRtiAmbassador& orig) *copy constructor - not implemented*
DtDirectRtiAmbassador&	**operator=** (const DtDirectRtiAmbassador& orig) *assignment operator - not implemented*

Inherited from DtVrlRtiAmbassador:

Documentation

class DtDirectRtiAmbassador: Instances of DtDirectRtiAmbassador are for wrapping the calls to The RTI ambassador so that we may override them

`DtDirectRtiAmbassador()`

default constructor

`virtual ~DtDirectRtiAmbassador()`

destructor

`virtual RTI::RTIambassador* rtiAmb()`

Get the real RTI ambassador

`virtual DtBoolean setReflecting(DtBoolean onOff)`

Set/Get whether to reflect back local objects and interactions, if supported by the derived class. The set call's return value indicates the value of the flag as a result of the call.

`virtual DtBoolean reflecting() const`

`virtual void createFederationExecution( const char *executionName, const char *FED)`

Calls down to the RTI function of the same name

`virtual void destroyFederationExecution( const char *executionName)`

Calls down to the RTI function of the same name

`virtual RTI::FederateHandle joinFederationExecution( const char* yourName, const char* executionName, RTI::FederateAmbassadorPtr federateAmbassadorReference)`

Calls down to the RTI function of the same name

`virtual void resignFederationExecution( RTI::ResignAction theAction)`

Calls down to the RTI function of the same name

`virtual void registerFederationSynchronizationPoint( const char* label, const char* theTag)`

Calls down to the RTI function of the same name

`virtual void registerFederationSynchronizationPoint( const char* label, const char* theTag, const RTI::FederateHandleSet& syncSet)`

Calls down to the RTI function of the same name

- **virtual void synchronizationPointAchieved(const char* label)**
  Calls down to the RTI function of the same name
- **virtual void requestFederationSave(const char* label, const RTI::FedTime& theTime)**
  Calls down to the RTI function of the same name
- **virtual void requestFederationSave(const char *label)**
  Calls down to the RTI function of the same name
- **virtual void federateSaveBegun()**
  Calls down to the RTI function of the same name
- **virtual void federateSaveComplete()**
  Calls down to the RTI function of the same name
- **virtual void federateSaveNotComplete()**
  Calls down to the RTI function of the same name
- **virtual void requestFederationRestore(const char* label)**
  Calls down to the RTI function of the same name
- **virtual void federateRestoreComplete()**
  Calls down to the RTI function of the same name
- **virtual void federateRestoreNotComplete()**
  Calls down to the RTI function of the same name
- **virtual void publishObjectClass(RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& attributeList)**
  Calls down to the RTI function of the same name
- **virtual void unpublishObjectClass(RTI::ObjectClassHandle theClass)**
  Calls down to the RTI function of the same name
- **virtual void publishInteractionClass(RTI::InteractionClassHandle theInteraction)**
  Calls down to the RTI function of the same name
- **virtual void unpublishInteractionClass(RTI::InteractionClassHandle theInteraction)**
  Calls down to the RTI function of the same name
- **virtual void subscribeObjectClassAttributes(RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& attributeList, RTI::Boolean active = RTI::RTI_TRUE)**
  Calls down to the RTI function of the same name
- **virtual void unsubscribeObjectClass(RTI::ObjectClassHandle theClass)**
  Calls down to the RTI function of the same name
- **virtual void subscribeInteractionClass(RTI::InteractionClassHandle theClass, RTI::Boolean active = RTI::RTI_TRUE)**
  Calls down to the RTI function of the same name
- **virtual void unsubscribeInteractionClass(RTI::InteractionClassHandle theClass)**
  Calls down to the RTI function of the same name
- **virtual RTI::ObjectHandle registerObjectInstance(RTI::ObjectClassHandle theClass,**

const char* theObject)

Calls down to the RTI function of the same name

- **virtual RTI::ObjectHandle registerObjectInstance(RTI::ObjectClassHandle theClass)**

  Calls down to the RTI function of the same name

- **virtual RTI::EventRetractionHandle updateAttributeValues(RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const RTI::FedTime& theTime, const char* theTag)**

  Calls down to the RTI function of the same name

- **virtual void updateAttributeValues(RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const char* theTag)**

  Calls down to the RTI function of the same name

- **virtual RTI::EventRetractionHandle sendInteraction(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const RTI::FedTime& theTime, const char* theTag)**

  Calls down to the RTI function of the same name

- **virtual void sendInteraction(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const char* theTag)**

  Calls down to the RTI function of the same name

- **virtual RTI::EventRetractionHandle deleteObjectInstance(RTI::ObjectHandle theObject, const RTI::FedTime& theTime, const char* theTag)**

  Calls down to the RTI function of the same name

- **virtual void deleteObjectInstance(RTI::ObjectHandle theObject, const char* theTag)**

  Calls down to the RTI function of the same name

- **virtual void localDeleteObjectInstance(RTI::ObjectHandle theObject)**

  Calls down to the RTI function of the same name

- **virtual void changeAttributeTransportationType(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, RTI::TransportationHandle theType)**

  Calls down to the RTI function of the same name

- **virtual void changeInteractionTransportationType(RTI::InteractionClassHandle theClass, RTI::TransportationHandle theType)**

  Calls down to the RTI function of the same name

- **virtual void requestObjectAttributeValueUpdate(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)**

  Calls down to the RTI function of the same name

- **virtual void requestClassAttributeValueUpdate(RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& theAttributes)**

  Calls down to the RTI function of the same name

- **virtual void unconditionalAttributeOwnershipDivestiture(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)**

  Calls down to the RTI function of the same name

- **virtual void negotiatedAttributeOwnershipDivestiture(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, const char* theTag)**

Calls down to the RTI function of the same name

virtual void attributeOwnershipAcquisition(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& desiredAttributes, const char* theTag)

Calls down to the RTI function of the same name

virtual void attributeOwnershipAcquisitionIfAvailable(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& desiredAttributes)

Calls down to the RTI function of the same name

virtual RTI::AttributeHandleSet* attributeOwnershipReleaseResponse(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)

Calls down to the RTI function of the same name

virtual void cancelNegotiatedAttributeOwnershipDivestiture(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)

Calls down to the RTI function of the same name

virtual void cancelAttributeOwnershipAcquisition(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)

Calls down to the RTI function of the same name

virtual void queryAttributeOwnership(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute)

Calls down to the RTI function of the same name

virtual RTI::Boolean isAttributeOwnedByFederate(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute)

Calls down to the RTI function of the same name

virtual void enableTimeRegulation(const RTI::FedTime& theFederateTime, const RTI::FedTime& theLookahead)

Calls down to the RTI function of the same name

virtual void disableTimeRegulation()

Calls down to the RTI function of the same name

virtual void enableTimeConstrained()

Calls down to the RTI function of the same name

virtual void disableTimeConstrained()

Calls down to the RTI function of the same name

virtual void timeAdvanceRequest(const RTI::FedTime& theTime)

Calls down to the RTI function of the same name

virtual void timeAdvanceRequestAvailable(const RTI::FedTime& theTime)

Calls down to the RTI function of the same name

virtual void nextEventRequest(const RTI::FedTime& theTime)

Calls down to the RTI function of the same name

virtual void nextEventRequestAvailable(const RTI::FedTime& theTime)

Calls down to the RTI function of the same name

virtual void flushQueueRequest(const RTI::FedTime& theTime)

Calls down to the RTI function of the same name

- **`virtual void enableAsynchronousDelivery()`**

  Calls down to the RTI function of the same name

- **`virtual void disableAsynchronousDelivery()`**

  Calls down to the RTI function of the same name

- **`virtual void queryLBTS(RTI::FedTime& theTime)`**

  Calls down to the RTI function of the same name.

- **`virtual void queryFederateTime(RTI::FedTime& theTime)`**

  Calls down to the RTI function of the same name

- **`virtual void queryMinNextEventTime( RTI::FedTime& theTime)`**

  Calls down to the RTI function of the same name

- **`virtual void modifyLookahead( const RTI::FedTime& theLookahead)`**

  Calls down to the RTI function of the same name

- **`virtual void queryLookahead( RTI::FedTime& theTime)`**

  Calls down to the RTI function of the same name

- **`virtual void retract( RTI::EventRetractionHandle theHandle)`**

  Calls down to the RTI function of the same name

- **`virtual void changeAttributeOrderType( RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes, RTI::OrderingHandle theType)`**

  Calls down to the RTI function of the same name

- **`virtual void changeInteractionOrderType( RTI::InteractionClassHandle theClass, RTI::OrderingHandle theType)`**

  Calls down to the RTI function of the same name

- **`virtual void createRegion( RTI::SpaceHandle theSpace, RTI::ULong numberOfExtents)`**

  Calls down to the RTI function of the same name

- **`virtual void notifyAboutRegionModification( RTI::Region &theRegion)`**

  Calls down to the RTI function of the same name

- **`virtual void deleteRegion(RTI::Region *theRegion)`**

  Calls down to the RTI function of the same name

- **`virtual RTI::ObjectHandle registerObjectInstanceWithRegion( RTI::ObjectClassHandle theClass, const char* theObject, RTI::AttributeHandle theAttributes[], RTI::Region* theRegions[], RTI::ULong theNumberOfHandles)`**

  Calls down to the RTI function of the same name

- **`virtual RTI::ObjectHandle registerObjectInstanceWithRegion( RTI::ObjectClassHandle theClass, RTI::AttributeHandle theAttributes[], RTI::Region* theRegions[], RTI::ULong theNumberOfHandles)`**

  Calls down to the RTI function of the same name

- **`virtual void associateRegionForUpdates( RTI::Region& theRegion, RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes)`**

  Calls down to the RTI function of the same name

virtual void unassociateRegionForUpdates(RTI::Region& theRegion, RTI::ObjectHandle theObject)

Calls down to the RTI function of the same name

virtual void subscribeObjectClassAttributesWithRegion(RTI::ObjectClassHandle theClass, RTI::Region& theRegion, const RTI::AttributeHandleSet &attributeList, RTI::Boolean active = RTI::RTI_TRUE)

Calls down to the RTI function of the same name

virtual void unsubscribeObjectClassWithRegion(RTI::ObjectClassHandle theClass, RTI::Region& theRegion)

Calls down to the RTI function of the same name

virtual void subscribeInteractionClassWithRegion(RTI::InteractionClassHandle theClass, RTI::Region& theRegion, RTI::Boolean active = RTI::RTI_TRUE)

Calls down to the RTI function of the same name

virtual void unsubscribeInteractionClassWithRegion(RTI::InteractionClassHandle theClass, RTI::Region& theRegion)

Calls down to the RTI function of the same name

virtual RTI::EventRetractionHandle sendInteractionWithRegion(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet &theParameters, const RTI::FedTime& theTime, const char* theTag, const RTI::Region& theRegion)

Calls down to the RTI function of the same name

virtual void sendInteractionWithRegion(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet &theParameters, const char* theTag, const RTI::Region& theRegion)

Calls down to the RTI function of the same name

virtual void requestClassAttributeValueUpdateWithRegion(RTI::ObjectClassHandle theClass, const RTI::AttributeHandleSet& theAttributes, const RTI::Region& theRegion)

Calls down to the RTI function of the same name

virtual RTI::ObjectClassHandle getObjectClassHandle(const char* theName)

Calls down to the RTI function of the same name

virtual char* getObjectClassName(RTI::ObjectClassHandle theHandle)

Calls down to the RTI function of the same name

virtual RTI::AttributeHandle getAttributeHandle(const char* theName, RTI::ObjectClassHandle whichClass)

Calls down to the RTI function of the same name

virtual char* getAttributeName(RTI::AttributeHandle theHandle, RTI::ObjectClassHandle whichClass)

Calls down to the RTI function of the same name

virtual RTI::InteractionClassHandle getInteractionClassHandle(const char* theName)

Calls down to the RTI function of the same name

virtual char* getInteractionClassName(RTI::InteractionClassHandle theHandle)

Calls down to the RTI function of the same name

- **virtual RTI::ParameterHandle getParameterHandle(const char* theName, RTI::InteractionClassHandle whichClass)**

  Calls down to the RTI function of the same name

- **virtual char* getParameterName(RTI::ParameterHandle theHandle, RTI::InteractionClassHandle whichClass)**

  Calls down to the RTI function of the same name

- **virtual RTI::ObjectHandle getObjectInstanceHandle(const char* theName)**

  Calls down to the RTI function of the same name

- **virtual char* getObjectInstanceName(RTI::ObjectHandle theHandle)**

  Calls down to the RTI function of the same name

- **virtual RTI::SpaceHandle getRoutingSpaceHandle(const char* theName)**

  Calls down to the RTI function of the same name

- **virtual const char* getRoutingSpaceName(const RTI::SpaceHandle theHandle)**

  Calls down to the RTI function of the same name

- **virtual RTI::DimensionHandle getDimensionHandle(const char* theName, RTI::SpaceHandle whichSpace)**

  Calls down to the RTI function of the same name

- **virtual char* getDimensionName(RTI::DimensionHandle theHandle, RTI::SpaceHandle whichSpace)**

  Calls down to the RTI function of the same name

- **virtual RTI::SpaceHandle getAttributeRoutingSpaceHandle(RTI::AttributeHandle theHandle, RTI::ObjectClassHandle whichClass)**

  Calls down to the RTI function of the same name

- **virtual RTI::ObjectClassHandle getObjectClass(RTI::ObjectHandle theObject)**

  Calls down to the RTI function of the same name

- **virtual RTI::SpaceHandle getInteractionRoutingSpaceHandle(RTI::InteractionClassHandle theHandle)**

  Calls down to the RTI function of the same name

- **virtual RTI::TransportationHandle getTransportationHandle(const char *theName)**

  Calls down to the RTI function of the same name

- **virtual char* getTransportationName(RTI::TransportationHandle theHandle)**

  Calls down to the RTI function of the same name

- **virtual RTI::OrderingHandle getOrderingHandle(const char *theName)**

  Calls down to the RTI function of the same name

- **virtual char* getOrderingName(RTI::OrderingHandle theHandle)**

  Calls down to the RTI function of the same name

- **virtual void enableClassRelevanceAdvisorySwitch()**

  Calls down to the RTI function of the same name

- **virtual void disableClassRelevanceAdvisorySwitch()**

Calls down to the RTI function of the same name

- **virtual void enableAttributeRelevanceAdvisorySwitch()**
  Calls down to the RTI function of the same name
- **virtual void disableAttributeRelevanceAdvisorySwitch()**
  Calls down to the RTI function of the same name
- **virtual void enableAttributeScopeAdvisorySwitch()**
  Calls down to the RTI function of the same name
- **virtual void disableAttributeScopeAdvisorySwitch()**
  Calls down to the RTI function of the same name
- **virtual void enableInteractionRelevanceAdvisorySwitch()**
  Calls down to the RTI function of the same name
- **virtual void disableInteractionRelevanceAdvisorySwitch()**
  Calls down to the RTI function of the same name
- **virtual RTI::Boolean tick()**
  Calls down to the RTI function of the same name
- **virtual RTI::Boolean tick(RTI::TickTime minimum, RTI::TickTime maximum)**
  Calls down to the RTI function of the same name
- **virtual RTI::RegionToken getRegionToken(RTI::Region*)**
  Calls down to the RTI function of the same name
- **virtual RTI::Region* getRegion(RTI::RegionToken)**
  Calls down to the RTI function of the same name
- **static DtVrlRtiAmbassador* create()**
  returns a new DtDirectRtiAmbassador
- **DtDirectRtiAmbassador(const DtDirectRtiAmbassador& orig)**
  copy constructor - not implemented
- **DtDirectRtiAmbassador& operator=(const DtDirectRtiAmbassador& orig)**
  assignment operator - not implemented
- **RTI::RTIambassador* myRtiAmb**
- **RTI::FederateAmbassadorPtr myFedAmb**
- **DtBoolean myReflecting**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file nullfedamb.h:

class DtNullFederateAmbassador: public RTI FederateAmbassador

---

Inheritance:

DtNullFederateAmbassador

Public Methods

	DtNullFederateAmbassador ()
virtual	**~DtNullFederateAmbassador** ()
* virtual void	**synchronizationPointRegistrationSucceeded** (const char *label) throw(RTI::FederateInternalError) //////////////////////////
virtual void	**synchronizationPointRegistrationFailed** (const char *label) throw(RTI::FederateInternalError)
virtual void	**announceSynchronizationPoint** (const char *label, const char *tag) throw(RTI::FederateInternalError)
virtual void	**federationSynchronized** (const char *label) throw(RTI::FederateInternalError)
virtual void	**initiateFederateSave** (const char *label) throw(RTI::UnableToPerformSave, RTI::FederateInternalError)
virtual void	**federationSaved** () throw(RTI::FederateInternalError)
virtual void	**federationNotSaved** () throw(RTI::FederateInternalError)
virtual void	**requestFederationRestoreSucceeded** (const char *label) throw(RTI::FederateInternalError)
virtual void	**requestFederationRestoreFailed** (const char *label, const char *reason) throw(RTI::FederateInternalError)
virtual void	**federationRestoreBegun** () throw(RTI::FederateInternalError)
virtual void	**initiateFederateRestore** (const char *label, RTI::FederateHandle handle) throw(RTI::SpecifiedSaveLabelDoesNotExist, RTI::CouldNotRestore, RTI::FederateInternalError)
virtual void	**federationRestored** () throw(RTI::FederateInternalError)
virtual void	**federationNotRestored** () throw(RTI::FederateInternalError)
* virtual void	**startRegistrationForObjectClass** (RTI::ObjectClassHandle theClass) throw(RTI::ObjectClassNotPublished, RTI::FederateInternalError) //////////////////////////
virtual void	**stopRegistrationForObjectClass** (RTI::ObjectClassHandle theClass) throw(RTI::ObjectClassNotPublished, RTI::FederateInternalError)
virtual void	**turnInteractionsOn** (RTI::InteractionClassHandle theHandle) throw(RTI::InteractionClassNotPublished, RTI::FederateInternalError)

- virtual void **turnInteractionsOff** (RTI::InteractionClassHandle theHandle) throw(RTI::InteractionClassNotPublished, RTI::FederateInternalError)
- * virtual void **discoverObjectInstance** (RTI::ObjectHandle theObject, RTI::ObjectClassHandle theObjectClass, const char* theObjectName) throw(RTI::CouldNotDiscover, RTI::ObjectClassNotKnown, RTI::FederateInternalError)
  ////////////////////////
- virtual void **reflectAttributeValues** (RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const RTI::FedTime& theTime, const char *theTag, RTI::EventRetractionHandle theHandle) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateOwnsAttributes, RTI::InvalidFederationTime, RTI::FederateInternalError)
- virtual void **reflectAttributeValues** (RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const char *theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateOwnsAttributes, RTI::FederateInternalError)
- virtual void **receiveInteraction** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const RTI::FedTime& theTime, const char *theTag, RTI::EventRetractionHandle theHandle) throw(RTI::InteractionClassNotKnown, RTI::InteractionParameterNotKnown, RTI::InvalidFederationTime, RTI::FederateInternalError)
  4
- virtual void **receiveInteraction** (RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const char *theTag) throw(RTI::InteractionClassNotKnown, RTI::InteractionParameterNotKnown, RTI::FederateInternalError)
- virtual void **removeObjectInstance** (RTI::ObjectHandle theObject, const RTI::FedTime& theTime, const char *theTag, RTI::EventRetractionHandle theHandle) throw(RTI::ObjectNotKnown, RTI::InvalidFederationTime, RTI::FederateInternalError)
- virtual void **removeObjectInstance** (RTI::ObjectHandle theObject, const char *theTag) throw(RTI::ObjectNotKnown, RTI::FederateInternalError)
- virtual void **attributesInScope** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)
- virtual void **attributesOutOfScope** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)
- virtual void **provideAttributeValueUpdate** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)
- virtual void **turnUpdatesOnForObjectInstance** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)
- virtual void **turnUpdatesOffForObjectInstance** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)
- * virtual void **requestAttributeOwnershipAssumption** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& offeredAttributes, const char *theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned, RTI::AttributeNotPublished, RTI::FederateInternalError)
  ////////////////////////////
- virtual void **attributeOwnershipDivestitureNotification** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& releasedAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::AttributeDivestitureWasNotRequested, RTI::FederateInternalError)
- virtual void **attributeOwnershipAcquisitionNotification** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& securedAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAcquisitionWasNotRequested, RTI::AttributeAlreadyOwned, RTI::AttributeNotPublished, RTI::FederateInternalError)
- virtual void **attributeOwnershipUnavailable** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned, RTI::AttributeAcquisitionWasNotRequested, RTI::FederateInternalError)
- virtual void **requestAttributeOwnershipRelease** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& candidateAttributes, const char *theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)

- virtual void **confirmAttributeOwnershipAcquisitionCancellation** (RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned, RTI::AttributeAcquisitionWasNotCanceled, RTI::FederateInternalError)
- virtual void **informAttributeOwnership** (RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute, RTI::FederateHandle theOwner) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)
- virtual void **attributeIsNotOwned** (RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)
- virtual void **attributeOwnedByRTI** (RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)
- * virtual void **timeRegulationEnabled** (const RTI::FedTime& theFederateTime) throw(RTI::InvalidFederationTime, RTI::EnableTimeRegulationWasNotPending, RTI::FederateInternalError)
  ////////////////////////
- virtual void **timeConstrainedEnabled** (const RTI::FedTime& theFederateTime) throw(RTI::InvalidFederationTime, RTI::EnableTimeConstrainedWasNotPending, RTI::FederateInternalError)
- virtual void **timeAdvanceGrant** (const RTI::FedTime& theTime) throw(RTI::InvalidFederationTime, RTI::TimeAdvanceWasNotInProgress, RTI::FederateInternalError)
- virtual void **requestRetraction** (RTI::EventRetractionHandle theHandle) throw(RTI::EventNotKnown, RTI::FederateInternalError)

Public

- * Federation Management **Services***
  ////////////////////////////
- * Declaration Management **Services***
  ////////////////////////////
- * Object Management **Services***
  ////////////////////////
- * Ownership Management **Services***
  ////////////////////////////
- * Time Management **Services***
  ////////////////////

Documentation

- **`DtNullFederateAmbassador()`**
- **`virtual ~DtNullFederateAmbassador()`**
- **`* Federation Management Services*`**
  ////////////////////////////
- **`* virtual void synchronizationPointRegistrationSucceeded( const char *label) throw( RTI::FederateInternalError)`**
  ////////////////////////////
- **`virtual void synchronizationPointRegistrationFailed( const char *label) throw( RTI::FederateInternalError)`**
- **`virtual void announceSynchronizationPoint( const char *label, const char *tag) throw( RTI::FederateInternalError)`**

- virtual void federationSynchronized(const char *label) throw(RTI::FederateInternalError)
- virtual void initiateFederateSave(const char *label) throw(RTI::UnableToPerformSave, RTI::FederateInternalError)
- virtual void federationSaved() throw(RTI::FederateInternalError)
- virtual void federationNotSaved() throw(RTI::FederateInternalError)
- virtual void requestFederationRestoreSucceeded(const char *label) throw(RTI::FederateInternalError)
- virtual void requestFederationRestoreFailed(const char *label, const char *reason) throw(RTI::FederateInternalError)
- virtual void federationRestoreBegun() throw(RTI::FederateInternalError)
- virtual void initiateFederateRestore(const char *label, RTI::FederateHandle handle) throw(RTI::SpecifiedSaveLabelDoesNotExist, RTI::CouldNotRestore, RTI::FederateInternalError)
- virtual void federationRestored() throw(RTI::FederateInternalError)
- virtual void federationNotRestored() throw(RTI::FederateInternalError)
- * Declaration Management Services*

  ////////////////////////////
- * virtual void startRegistrationForObjectClass(RTI::ObjectClassHandle theClass) throw(RTI::ObjectClassNotPublished, RTI::FederateInternalError)

  ////////////////////////////
- virtual void stopRegistrationForObjectClass(RTI::ObjectClassHandle theClass) throw(RTI::ObjectClassNotPublished, RTI::FederateInternalError)
- virtual void turnInteractionsOn(RTI::InteractionClassHandle theHandle) throw(RTI::InteractionClassNotPublished, RTI::FederateInternalError)
- virtual void turnInteractionsOff(RTI::InteractionClassHandle theHandle) throw(RTI::InteractionClassNotPublished, RTI::FederateInternalError)
- * Object Management Services*

  ////////////////////////
- * virtual void discoverObjectInstance(RTI::ObjectHandle theObject, RTI::ObjectClassHandle theObjectClass, const char* theObjectName) throw(RTI::CouldNotDiscover, RTI::ObjectClassNotKnown, RTI::FederateInternalError)

  ////////////////////////
- virtual void reflectAttributeValues(RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const RTI::FedTime& theTime, const char *theTag, RTI::EventRetractionHandle theHandle) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateOwnsAttributes, RTI::InvalidFederationTime, RTI::FederateInternalError)
- virtual void reflectAttributeValues(RTI::ObjectHandle theObject, const RTI::AttributeHandleValuePairSet& theAttributes, const char *theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateOwnsAttributes, RTI::FederateInternalError)

- **virtual void receiveInteraction(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const RTI::FedTime& theTime, const char *theTag, RTI::EventRetractionHandle theHandle) throw(RTI::InteractionClassNotKnown, RTI::InteractionParameterNotKnown, RTI::InvalidFederationTime, RTI::FederateInternalError)**

  4

- **virtual void receiveInteraction(RTI::InteractionClassHandle theInteraction, const RTI::ParameterHandleValuePairSet& theParameters, const char *theTag) throw(RTI::InteractionClassNotKnown, RTI::InteractionParameterNotKnown, RTI::FederateInternalError)**

- **virtual void removeObjectInstance(RTI::ObjectHandle theObject, const RTI::FedTime& theTime, const char *theTag, RTI::EventRetractionHandle theHandle) throw(RTI::ObjectNotKnown, RTI::InvalidFederationTime, RTI::FederateInternalError)**

- **virtual void removeObjectInstance(RTI::ObjectHandle theObject, const char *theTag) throw(RTI::ObjectNotKnown, RTI::FederateInternalError)**

- **virtual void attributesInScope(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)**

- **virtual void attributesOutOfScope(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)**

- **virtual void provideAttributeValueUpdate(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)**

- **virtual void turnUpdatesOnForObjectInstance(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)**

- **virtual void turnUpdatesOffForObjectInstance(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)**

- *** Ownership Management Services***

  ////////////////////////////

- *** virtual void requestAttributeOwnershipAssumption(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& offeredAttributes, const char *theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned, RTI::AttributeNotPublished, RTI::FederateInternalError)**

  ////////////////////////////

- **virtual void attributeOwnershipDivestitureNotification(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& releasedAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::AttributeDivestitureWasNotRequested, RTI::FederateInternalError)**

- **virtual void attributeOwnershipAcquisitionNotification(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& securedAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAcquisitionWasNotRequested, RTI::AttributeAlreadyOwned, RTI::AttributeNotPublished, RTI::FederateInternalError)**

- **virtual void attributeOwnershipUnavailable(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown,**

RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned,
RTI::AttributeAcquisitionWasNotRequested, RTI::FederateInternalError)

- virtual void requestAttributeOwnershipRelease(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& candidateAttributes, const char *theTag) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeNotOwned, RTI::FederateInternalError)

- virtual void confirmAttributeOwnershipAcquisitionCancellation(RTI::ObjectHandle theObject, const RTI::AttributeHandleSet& theAttributes) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::AttributeAlreadyOwned, RTI::AttributeAcquisitionWasNotCanceled, RTI::FederateInternalError)

- virtual void informAttributeOwnership(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute, RTI::FederateHandle theOwner) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)

- virtual void attributeIsNotOwned(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)

- virtual void attributeOwnedByRTI(RTI::ObjectHandle theObject, RTI::AttributeHandle theAttribute) throw(RTI::ObjectNotKnown, RTI::AttributeNotKnown, RTI::FederateInternalError)

- * Time Management Services*

  //////////////////////

- * virtual void timeRegulationEnabled(const RTI::FedTime& theFederateTime) throw(RTI::InvalidFederationTime, RTI::EnableTimeRegulationWasNotPending, RTI::FederateInternalError)

  //////////////////////

- virtual void timeConstrainedEnabled(const RTI::FedTime& theFederateTime) throw(RTI::InvalidFederationTime, RTI::EnableTimeConstrainedWasNotPending, RTI::FederateInternalError)

- virtual void timeAdvanceGrant(const RTI::FedTime& theTime) throw(RTI::InvalidFederationTime, RTI::TimeAdvanceWasNotInProgress, RTI::FederateInternalError)

- virtual void requestRetraction(RTI::EventRetractionHandle theHandle) throw(RTI::EventNotKnown, RTI::FederateInternalError)

Direct child classes:

DtVrlFederateAmbassador

Alphabetic index *Hierarchy of classes*

In file refattrcblist.h:

class DtReflectAttributeValuesCbList

Public Methods

	DtReflectAttributeValuesCbList () *default constructor*
virtual	**~DtReflectAttributeValuesCbList** () *destructor*
	DtReflectAttributeValuesCbList (const DtReflectAttributeValuesCbList& orig) *copy constructor*
DtReflectAttributeValuesCbList&	**operator=** (const DtReflectAttributeValuesCbList& orig) *assignment operator*
virtual int	**add** (RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr) *Add/Remove a reflect attribute callback to/from the list*
virtual int	**remove** (RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr)
virtual void	**clear** () *Remove all callbacks from the list*
virtual void	**apply** (const DtStateMsg &msg) *State messages are for a specific object id*

Protected Fields

- DtList **myList**

Protected Methods

- void **dtorClear** ()
  Remove all callbacks from the list
- void **ctorCopyList** (const DtReflectAttributeValuesCbList& orig)
  Copy another callback list into this object's callback list

---

Documentation

- **`DtReflectAttributeValuesCbList()`**

  default constructor

- **`virtual ~DtReflectAttributeValuesCbList()`**

  destructor

- **`DtReflectAttributeValuesCbList( const DtReflectAttributeValuesCbList& orig)`**

  copy constructor

- **`DtReflectAttributeValuesCbList& operator=( const DtReflectAttributeValuesCbList& orig)`**

  assignment operator

- **`virtual int add(RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr)`**

  Add/Remove a reflect attribute callback to/from the list. Returns whether add/remove took place.

- **`virtual int remove(RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr)`**

- **`virtual void clear()`**

  Remove all callbacks from the list

- **`virtual void apply(const DtStateMsg &msg)`**

  State messages are for a specific object id. Callbacks are for a particular object id. Call any callbacks in the list for the object id specified in the state message.

- **`void dtorClear()`**

  Remove all callbacks from the list. Non-virtual since meant to be called from dtor, otherwise should call clear().

- **`void ctorCopyList(const DtReflectAttributeValuesCbList& orig)`**

  Copy another callback list into this object's callback list. Non-virtual since meant to be called from ctor.

- **`DtList myList`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file stintcblist.h:

class DtStartStopInterCbList

Public Methods

	DtStartStopInterCbList () *default constructor*
virtual	**~DtStartStopInterCbList** () *destructor*
	DtStartStopInterCbList (const DtStartStopInterCbList& orig) *copy constructor*
DtStartStopInterCbList&	**operator=** (const DtStartStopInterCbList& orig) *assignment operator*
virtual int	**add** (RTI::InteractionClassHandle hand, DtStartStopInterCb cb, void* usr) *Add/Remove a start updates callback to/from the list*
virtual int	**remove** (RTI::InteractionClassHandle hand, DtStartStopInterCb cb, void* usr)
virtual void	**clear** () *Remove all callbacks from the list*
virtual void	**apply** (RTI::InteractionClassHandle interClass) *Callbacks are for a particular inter class handle*

Protected Fields

DtList **myList**

Protected Methods

void	**dtorClear** () *Remove all callbacks from the list*
void	**ctorCopyList** (const DtStartStopInterCbList& orig) *Copy another callback list into this inter's callback list*

Documentation

`DtStartStopInterCbList()`

default constructor

`virtual ~DtStartStopInterCbList()`

destructor

- **`DtStartStopInterCbList( const DtStartStopInterCbList& orig)`**

  copy constructor

- **`DtStartStopInterCbList& operator=( const DtStartStopInterCbList& orig)`**

  assignment operator

- **`virtual int add(RTI::InteractionClassHandle hand, DtStartStopInterCb cb, void* usr)`**

  Add/Remove a start updates callback to/from the list. Returns whether add/remove took place.

- **`virtual int remove(RTI::InteractionClassHandle hand, DtStartStopInterCb cb, void* usr)`**

- **`virtual void clear()`**

  Remove all callbacks from the list

- **`virtual void apply(RTI::InteractionClassHandle interClass)`**

  Callbacks are for a particular inter class handle. Call any callbacks in the list for the inter class handle for the start updates inter.

- **`void dtorClear()`**

  Remove all callbacks from the list. Non-virtual since meant to be called from dtor, otherwise should call clear().

- **`void ctorCopyList(const DtStartStopInterCbList& orig)`**

  Copy another callback list into this inter's callback list. Non-virtual since meant to be called from ctor.

- **`DtList myList`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file fedtarget.h:

class DtFedTarget

class DtFedTarget: DtFedTarget is an abstract class representing some representation of FED file data that can be filled in by a DtFedReader

---

Inheritance:

DtFedTarget

Public Methods

	DtFedTarget () *Constructor*
virtual	**~DtFedTarget** () *Destructor*
	DtFedTarget (const DtFedTarget& orig) *copy constructor*
DtFedTarget&	**operator=** (const DtFedTarget& orig) *assignment operator*
virtual void	**addInterClass** (const char* name, const char* baseName, DtTransportType transport, DtOrderType order) *Define a class*
virtual void	**addObjClass** (const char* name, const char* baseName) *Define a class*
virtual void	**addParameter** (const char* className, const char* paramName) *Define a parameter of a class*
virtual void	**addAttribute** (const char* className, const char* attributeName, DtTransportType transport, DtOrderType order) *Define an attribute of a class*

Documentation

class DtFedTarget: DtFedTarget is an abstract class representing some representation of FED file data that can be filled in by a DtFedReader

`DtFedTarget()`

Constructor

`virtual ~DtFedTarget()`

Destructor

- **`DtFedTarget(const DtFedTarget& orig)`**

  copy constructor

- **`DtFedTarget& operator=(const DtFedTarget& orig)`**

  assignment operator

- **`virtual void addInterClass(const char* name, const char* baseName, DtTransportType transport, DtOrderType order)`**

  Define a class. Used when initializing from FED file.

- **`virtual void addObjClass(const char* name, const char* baseName)`**

  Define a class. Used when initializing from FED file.

- **`virtual void addParameter(const char* className, const char* paramName)`**

  Define a parameter of a class. Used when initializing from FED file.

- **`virtual void addAttribute(const char* className, const char* attributeName, DtTransportType transport, DtOrderType order)`**

  Define an attribute of a class. Used when initializing from FED file.

Direct child classes:

DtFom

Alphabetic index *Hierarchy of classes*

In file fedtarget.h:

enum DtTransportType

- **DtBest_Effort**
- **DtReliable**

Documentation

- `DtBest_Effort`
- `DtReliable`

Alphabetic index *Hierarchy of classes*

In file fedtarget.h:

enum DtOrderType

- **DtReceive**
- **DtTimestamp**

Documentation

- `DtReceive`
- `DtTimestamp`

In file rprinterfact.h:

class DtRprInteractionFactory: public DtInteractionFactory

Inheritance:

DtRprInteractionFactory - DtInteractionFactory

Public Methods

	DtRprInteractionFactory (DtExerciseConn* exConn) *constructor Adds creators for the RPR FOM interactions*
virtual	**~DtRprInteractionFactory** () *destructor*
	DtRprInteractionFactory (const DtRprInteractionFactory& orig) *copy constructor*
DtRprInteractionFactory&	**operator=** (const DtRprInteractionFactory& orig) *assignment operator*

Inherited from DtInteractionFactory:

Public Methods

- typedef DtInteraction***(*DtInteractionCreator)**()
- virtual DtInteraction* **createInteraction**(RTI::InteractionClassHandle handle, const RTI::ParameterHandleValuePairSet& pairSet)
- virtual DtInteractionCreator **addCreator**(RTI::InteractionClassHandle handle, DtInteractionCreator creator)
- virtual DtInteractionCreator **addCreator**(const char* className, DtInteractionCreator creator)

Protected Fields

- DtInteractionCreator* **myCreators**
- DtExerciseConn* **myExConn**

Documentation

- **`DtRprInteractionFactory(DtExerciseConn* exConn)`**

  constructor Adds creators for the RPR FOM interactions

- **`virtual ~DtRprInteractionFactory()`**

  destructor

- **`DtRprInteractionFactory(const DtRprInteractionFactory& orig)`**

  copy constructor

- **`DtRprInteractionFactory& operator=(const DtRprInteractionFactory& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file rprobjfact.h:

class DtRprHlaObjectFactory: public DtHlaObjectFactory

class DtRprHlaObjectFactory: Instances of DtRprHlaObjectFactory are used to create RPR (Real-time Platform Reference) FOM HLA objects

---

Inheritance:

DtRprHlaObjectFactory - DtHlaObjectFactory

Public Methods

	DtRprHlaObjectFactory (DtExerciseConn* conn) *constructor*
virtual	**~DtRprHlaObjectFactory** () *destructor*
	DtRprHlaObjectFactory (const DtRprHlaObjectFactory& orig) *copy constructor*
DtRprHlaObjectFactory&	**operator=** (const DtRprHlaObjectFactory& orig) *assignment operator*

Inherited from DtHlaObjectFactory:

Public Methods

- typedef DtHlaObject***(*DtHlaObjectCreator)**(DtExerciseConn* conn, RTI::ObjectHandle id, DtObjClassDesc* cDesc)
- typedef DtHlaObject***(*DtHlaObjectCreator)**(DtExerciseConn* conn, RTI::ObjectHandle id, const char* name, DtObjClassDesc* cDesc)
- virtual DtHlaObject* **createHlaObject**(RTI::ObjectClassHandle handle, RTI::ObjectHandle id)
- virtual DtHlaObject* **createHlaObject**(RTI::ObjectClassHandle handle, RTI::ObjectHandle id, const char* name)
- virtual DtHlaObjectCreator **addCreator**(RTI::ObjectClassHandle handle, DtHlaObjectCreator creator)

Protected Fields

- DtHlaObjectCreator* **myCreators**
- DtExerciseConn* **myConn**

Documentation

class DtRprHlaObjectFactory: Instances of DtRprHlaObjectFactory are used to create RPR (Real-time Platform Reference) FOM HLA objects

- **`DtRprHlaObjectFactory(DtExerciseConn* conn)`**

  constructor

- **`virtual ~DtRprHlaObjectFactory()`**

  destructor

- **`DtRprHlaObjectFactory(const DtRprHlaObjectFactory& orig)`**

  copy constructor

- **`DtRprHlaObjectFactory& operator=(const DtRprHlaObjectFactory& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file objidhashlst.h:

class DtObjectIdHashlist

class DtObjectIdHashlist: Instances of DtObjectIdHashlist are hashlists, where you can store associations between DtObjectIds and arbitrary pieces of data

---

Public Methods

	DtObjectIdHashlist (DtBoolean intrusive = DtFalse) *default constructor*
virtual	**~DtObjectIdHashlist** () *destructor*
virtual DtListItem*	**add** (const DtObjectId& key, void* data) *Add an association between key and data to the list*
virtual void*	**remove** (const DtObjectId& key) *Remove the current association involving key from the list*
virtual void*	**lookup** (const DtObjectId& key) const *Lookup the data associated with key*
virtual void*	**operator[]** (const DtObjectId& key) const
virtual const DtList*	**list** () const *Get a DtList containing all of the void* data that you have listed, regardless of keys*
inline void*	**operator[]** (const DtObjectId& key) const

Public

- *In the event that an item with key already exists, then add will replace the data and move it to the location in list*

Protected Fields

DtList*	**myList**
DtHashlist*	**myHashlist**

Protected Methods

	DtObjectIdHashlist (const DtObjectIdHashlist& orig) *copy constructor*
DtObjectIdHashlist&	**operator=** (const DtObjectIdHashlist& orig) *assignment operator*

Documentation

class DtObjectIdHashlist: Instances of DtObjectIdHashlist are hashlists, where you can store associations between DtObjectIds and arbitrary pieces of data

- **DtObjectIdHashlist(DtBoolean intrusive = DtFalse)**

  default constructor. Pass DtTrue only if the data that you are listing are instances of a class derived from DtListedObject.

- **virtual ~DtObjectIdHashlist()**

  destructor

-

  In the event that an item with key already exists, then add will replace the data and move it to the location in list. If your data was alloced, do not add unless you know key is unique, or a memory leak will result.

- **virtual DtListItem* add(const DtObjectId& key, void* data)**

  Add an association between key and data to the list

- **virtual void* remove(const DtObjectId& key)**

  Remove the current association involving key from the list. Returns the data that was associated with key.

- **virtual void* lookup(const DtObjectId& key) const**

  Lookup the data associated with key

- **virtual void* operator[](const DtObjectId& key) const**

- **virtual const DtList* list() const**

  Get a DtList containing all of the void* data that you have listed, regardless of keys

- **DtObjectIdHashlist(const DtObjectIdHashlist& orig)**

  copy constructor

- **DtObjectIdHashlist& operator=(const DtObjectIdHashlist& orig)**

  assignment operator

- **DtList* myList**

- **DtHashlist* myHashlist**

- **inline void* operator[](const DtObjectId& key) const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file staterep.h:

class DtStateRepository

class DtStateRepository: DtStateRepository is an abstract base class for maintaining state for an object

---

Inheritance:

DtStateRepository

Public Methods

virtual	**~DtStateRepository** () *destructor*
DtStateRepository&	**operator=** (const DtStateRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
virtual void	**printData** () const *Print state data*
virtual void	**setGlobalId** (const DtGlobalObjectDesignator& id) *Get/Set global identifier, or name*
virtual const DtGlobalObjectDesignator&	**globalId** () const
virtual void	**setLocalId** (const DtObjectId& id) *Get/Set local identifier, or handle*
virtual const DtObjectId&	**localId** () const
	DtStateRepository () *default constructor*

Protected Fields

DtGlobalObjectDesignator	**myGlobalId**
DtObjectId	**myLocalId**

Protected Methods

DtStateRepository (const DtStateRepository& orig)
copy constructor

Protected

- *Constructors are protected since this is an abstract base class*

Documentation

class DtStateRepository:

DtStateRepository is an abstract base class for maintaining state for an object. These are used to decode data into the repository from the network and encode data from the repository to the network.

- **`virtual ~DtStateRepository()`**
  destructor
- **`DtStateRepository& operator=(const DtStateRepository& orig)`**
  assignment operator
- **`virtual DtStateRepository* clone() const`**
  return an empty object of the same type as this
- **`virtual void printData() const`**
  Print state data
- **`virtual void setGlobalId(const DtGlobalObjectDesignator& id)`**
  Get/Set global identifier, or name. Set by VR-Link for reflected objects. Not necessary to set for local objects.
- **`virtual const DtGlobalObjectDesignator& globalId() const`**
- **`virtual void setLocalId(const DtObjectId& id)`**
  Get/Set local identifier, or handle. Set by VR-Link for reflected objects. Not necessary to set for local objects.
- **`virtual const DtObjectId& localId() const`**
-
  Constructors are protected since this is an abstract base class
- **`DtStateRepository()`**
  default constructor
- **`DtStateRepository(const DtStateRepository& orig)`**
  copy constructor
- **`DtGlobalObjectDesignator myGlobalId`**
- **`DtObjectId myLocalId`**

Direct child classes:

DtGenericStateRepository
DtEmitterBeamRepository
DtEmbeddedSystemRepository
DtBaseEntityStateRepository

In file hstateencoder.h:

class DtHlaStateEncoder

class DtHlaStateEncoder: DtHlaStateEncoder is used to provide encoding of a state repository into a state message in an HLA (as opposed to DIS) environment

---

Inheritance:

DtHlaStateEncoder

Public Methods

	DtHlaStateEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtHlaStateEncoder** () *destructor*
	DtHlaStateEncoder (const DtHlaStateEncoder& orig) *copy constructor*
DtHlaStateEncoder&	**operator=** (const DtHlaStateEncoder& orig) *assignment operator*
typedef void	**(*DtAttributeEncoder)** (const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle) *Attribute encoding function declaration*
typedef DtBoolean	**(*DtAttributeChecker)** (const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote) *Attribute checking function declaration*
DtAttributeEncoder	**addEncoder** (RTI::AttributeHandle hand, DtAttributeEncoder func) *Add/Remove an encoding function for the attribute specified by handle*
DtAttributeEncoder	**removeEncoder** (RTI::AttributeHandle hand)
DtAttributeEncoder	**addEncoder** (char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue) *Add/Remove an encoding function for the attribute specified by name*
DtAttributeEncoder	**removeEncoder** (char* name, DtBoolean complain = DtTrue)
DtAttributeChecker	**addChecker** (RTI::AttributeHandle hand, DtAttributeChecker func) *Add/Remove a checker function for the attribute specified by handle*
DtAttributeChecker	**removeChecker** (RTI::AttributeHandle hand)
DtAttributeChecker	**addChecker** (char* name, DtAttributeChecker func, DtBoolean complain = DtTrue) *Add/Remove a checker function for the attribute specified by name*
DtAttributeChecker	**removeChecker** (char* name, DtBoolean complain = DtTrue)

virtual DtAttributeEncoder	**encoder** (int handle) const *Get the attribute encoder for the specified handle*
virtual DtAttributeChecker	**checker** (int handle) const *Get the attribute checker for the specified handle*
virtual void	**encode** (const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg) *Encode necessary values from the state repository*
virtual RTI::ObjectClassHandle	**objectClassHandle** () *Get the object's class handle*
virtual DtHlaStateEncoder*	**clone** () const *Returns a new-ed clone of this object*
virtual void	**setClassDesc** (DtObjClassDesc* desc) *Set the class descriptor of the encoder*
static DtBoolean	**trueChecker** (const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote) *A checker function that always returns 1, meaning that the attribute for which it is being used will be deemed necessary to send on every tick*
static DtBoolean	**falseChecker** (const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote) *A checker function that always returns 0, meaning that the attribute for which it is being used will never be deemed necessary to send*
inline DtHlaStateEncoder::DtAttributeChecker	**checker** (int hand) const
inline DtHlaStateEncoder::DtAttributeEncoder	**encoder** (int hand) const
inline RTI::ObjectClassHandle	**objectClassHandle** ()
inline void	**setClassDesc** (DtObjClassDesc* desc)

Protected Fields

DtExerciseConn*	**myExConn**
DtObjClassDesc*	**myClassDesc**
DtVrlRtiAmbassador*	**myRtiAmb**
RTI::InteractionClassHandle	**myHandle**
DtAttributeEncoder*	**myEncoderList** *myEncoderList and myCheckerList are arrays of theMaxParametersPerInteraction items*
DtAttributeChecker*	**myCheckerList**
static const int	**theMaxAttributesPerObject**

Documentation

class DtHlaStateEncoder: DtHlaStateEncoder is used to provide encoding of a state repository into a state message in an HLA (as opposed to DIS) environment

- **`DtHlaStateEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

constructor

- **virtual ~DtHlaStateEncoder()**

  destructor

- **DtHlaStateEncoder(const DtHlaStateEncoder& orig)**

  copy constructor

- **DtHlaStateEncoder& operator=(const DtHlaStateEncoder& orig)**

  assignment operator

- **typedef void (*DtAttributeEncoder)(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)**

  Attribute encoding function declaration

- **typedef DtBoolean (*DtAttributeChecker)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)**

  Attribute checking function declaration

- **DtAttributeEncoder addEncoder(RTI::AttributeHandle hand, DtAttributeEncoder func)**

  Add/Remove an encoding function for the attribute specified by handle. Returns the encoding function which is replaced by the add/remove. Note: This is non-virtual since it is typically called by constructors.

- **DtAttributeEncoder removeEncoder(RTI::AttributeHandle hand)**

- **DtAttributeEncoder addEncoder(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)**

  Add/Remove an encoding function for the attribute specified by name. Complain flag determines whether to print a warning if the attribute does not exist in the FOM. Returns the encoding function which is replaced by the add/remove. Note: This is non-virtual since it is typically called by constructors.

- **DtAttributeEncoder removeEncoder(char* name, DtBoolean complain = DtTrue)**

- **DtAttributeChecker addChecker(RTI::AttributeHandle hand, DtAttributeChecker func)**

  Add/Remove a checker function for the attribute specified by handle. Returns the checker function which is replaced by the add/remove. Note: This is non-virtual since it is typically called by constructors.

- **DtAttributeChecker removeChecker(RTI::AttributeHandle hand)**

- **DtAttributeChecker addChecker(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)**

  Add/Remove a checker function for the attribute specified by name. Complain flag determines whether to print a warning if the attribute does not exist in the FOM. Returns the checker function which is replaced by the add/remove. Note: This is non-virtual since it is typically called by constructors.

- **DtAttributeChecker removeChecker(char* name, DtBoolean complain = DtTrue)**

- **virtual DtAttributeEncoder encoder(int handle) const**

  Get the attribute encoder for the specified handle

- **virtual DtAttributeChecker checker(int handle) const**

  Get the attribute checker for the specified handle

- **virtual void encode(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)**

Encode necessary values from the state repository. The second state repository represents what is seen by remote simulators and is used to determine when things need to be sent in the state message. includeThese is a set of attributes that should be included even if their thresholds have not been exceeded.

- **virtual RTI::ObjectClassHandle objectClassHandle()**

  Get the object's class handle

- **virtual DtHlaStateEncoder* clone() const**

  Returns a new-ed clone of this object

- **virtual void setClassDesc(DtObjClassDesc* desc)**

  Set the class descriptor of the encoder

- **static DtBoolean trueChecker(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)**

  A checker function that always returns 1, meaning that the attribute for which it is being used will be deemed necessary to send on every tick.

- **static DtBoolean falseChecker(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)**

  A checker function that always returns 0, meaning that the attribute for which it is being used will never be deemed necessary to send

- **DtExerciseConn* myExConn**

- **DtObjClassDesc* myClassDesc**

- **DtVrlRtiAmbassador* myRtiAmb**

- **RTI::InteractionClassHandle myHandle**

- **DtAttributeEncoder* myEncoderList**

  myEncoderList and myCheckerList are arrays of theMaxParametersPerInteraction items

- **DtAttributeChecker* myCheckerList**

- **static const int theMaxAttributesPerObject**

- **inline DtHlaStateEncoder::DtAttributeChecker checker(int hand) const**

- **inline DtHlaStateEncoder::DtAttributeEncoder encoder(int hand) const**

- **inline RTI::ObjectClassHandle objectClassHandle()**

- **inline void setClassDesc(DtObjClassDesc* desc)**

Direct child classes:

DtGenericObjectEncoder
DtBaseEntityEncoder
DtBaseEmitterBeamEncoder
DtBaseEmbeddedSystemEncoder

Alphabetic index *Hierarchy of classes*

In file hstatedecoder.h:

class DtHlaStateDecoder

class DtHlaStateDecoder: DtHlaStateDecoder is used to provide decoding of a state message into a state repository in an HLA (as opposed to DIS) environment

---

Inheritance:

DtHlaStateDecoder

Public Methods

	DtHlaStateDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtHlaStateDecoder** () *destructor*
	DtHlaStateDecoder (const DtHlaStateDecoder& orig) *copy constructor*
DtHlaStateDecoder&	**operator=** (const DtHlaStateDecoder& orig) *assignment operator*
typedef void	**(*DtAttributeDecoder)** (DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex) *Attribute decoding function declaration*
DtAttributeDecoder	**addDecoder** (RTI::AttributeHandle h, DtAttributeDecoder func) *Add/Remove a decoding function for the attribute specified by handle*
DtAttributeDecoder	**removeDecoder** (RTI::AttributeHandle h)
DtAttributeDecoder	**addDecoder** (char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue) *Add/Remove a decoding function for the attribute specified by name*
DtAttributeDecoder	**removeDecoder** (char* name, DtBoolean complain = DtTrue)
virtual DtAttributeDecoder	**decoder** (int handle) const *Get the attribute decoder for the specified handle*
virtual void	**decode** (const DtStateMsg& msg, DtStateRepository* stateRep) const *Decode the state message into the state repository*
virtual RTI::ObjectClassHandle	**objectClassHandle** () const *Get the object's class handle*
virtual DtHlaStateDecoder*	**clone** () const *Returns a new-ed clone of this object*
virtual void	**setClassDesc** (DtObjClassDesc* desc) *Set the class descriptor of the encoder*
static char*	**decodingBuffer** () *Returns a pointer to a large static buffer that can be used to copy data obtained from a phvps*

- inline char* **decodingBuffer** ()
- inline RTI::ObjectClassHandle **objectClassHandle** () const
- inline void **setClassDesc** (DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
  myDecoderList is an array of theMaxAttributesPerObject items
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer** [2000]

Protected Methods

- virtual void **decodeAttribute** (DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const
  Decode the attribute at the specified index in the attribute handle pair set

---

Documentation

class DtHlaStateDecoder: DtHlaStateDecoder is used to provide decoding of a state message into a state repository in an HLA (as opposed to DIS) environment

- **`DtHlaStateDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
  constructor
- **`virtual ~DtHlaStateDecoder()`**
  destructor
- **`DtHlaStateDecoder(const DtHlaStateDecoder& orig)`**
  copy constructor
- **`DtHlaStateDecoder& operator=(const DtHlaStateDecoder& orig)`**
  assignment operator
- **`typedef void (*DtAttributeDecoder)( DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)`**
  Attribute decoding function declaration
- **`DtAttributeDecoder addDecoder(RTI::AttributeHandle h, DtAttributeDecoder func)`**
  Add/Remove a decoding function for the attribute specified by handle. Returns the decoding function which is replaced by the add/remove. Note: This is non-virtual since it is typically called by constructors.

- **DtAttributeDecoder removeDecoder(RTI::AttributeHandle h)**
- **DtAttributeDecoder addDecoder(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)**

  Add/Remove a decoding function for the attribute specified by name. Complain flag determines whether to print a warning if the attribute does not exist in the FOM. Returns the decoding function which is replaced by the add/remove. Note: This is non-virtual since it is typically called by constructors.
- **DtAttributeDecoder removeDecoder(char* name, DtBoolean complain = DtTrue)**
- **virtual DtAttributeDecoder decoder(int handle) const**

  Get the attribute decoder for the specified handle
- **virtual void decode(const DtStateMsg& msg, DtStateRepository* stateRep) const**

  Decode the state message into the state repository
- **virtual RTI::ObjectClassHandle objectClassHandle() const**

  Get the object's class handle
- **virtual DtHlaStateDecoder* clone() const**

  Returns a new-ed clone of this object
- **virtual void setClassDesc(DtObjClassDesc* desc)**

  Set the class descriptor of the encoder
- **static char* decodingBuffer()**

  Returns a pointer to a large static buffer that can be used to copy data obtained from a phvps.
- **virtual void decodeAttribute(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const**

  Decode the attribute at the specified index in the attribute handle pair set
- **DtExerciseConn* myExConn**
- **DtObjClassDesc* myClassDesc**
- **DtVrlRtiAmbassador* myRtiAmb**
- **RTI::ObjectClassHandle myHandle**
- **DtAttributeDecoder* myDecoderList**

  myDecoderList is an array of theMaxAttributesPerObject items
- **static const unsigned int theMaxAttributesPerObject**
- **static char theDecBuffer[2000]**
- **inline char* decodingBuffer()**
- **inline RTI::ObjectClassHandle objectClassHandle() const**
- **inline void setClassDesc(DtObjClassDesc* desc)**

Direct child classes:

DtGenericObjectDecoder

DtBaseEntityDecoder
DtBaseEmitterBeamDecoder
DtBaseEmbeddedSystemDecoder

Alphabetic index *Hierarchy of classes*

In file hvcenc.h:

class DtViewControlEncoder: public DtInteractionEncoder

Inheritance:

DtViewControlEncoder - DtInteractionEncoder

Public Methods

	DtViewControlEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtViewControlEncoder** ()
	DtViewControlEncoder (const DtViewControlEncoder& orig)
DtViewControlEncoder&	**operator=** (const DtViewControlEncoder& orig)

Protected Methods

- **DtDECLARE_VIEW_CONTROL_PARAM_ENCODER** (ViewControlPdu)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const

- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **`DtViewControlEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtViewControlEncoder()`**
- **`DtViewControlEncoder(const DtViewControlEncoder& orig)`**
- **`DtViewControlEncoder& operator=(const DtViewControlEncoder& orig)`**
- **`DtDECLARE_VIEW_CONTROL_PARAM_ENCODER(ViewControlPdu)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hstopenc.h:

class DtStopFreezeEncoder: public DtInteractionEncoder

Inheritance:

DtStopFreezeEncoder - DtInteractionEncoder

Public Methods

	DtStopFreezeEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtStopFreezeEncoder** ()
	DtStopFreezeEncoder (const DtStopFreezeEncoder& orig)
DtStopFreezeEncoder&	**operator=** (const DtStopFreezeEncoder& orig)

Protected Methods

- **DtDECLARE_STOP_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_STOP_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_STOP_PARAM_ENCODER** (RequestID)
- **DtDECLARE_STOP_PARAM_ENCODER** (RealWorldTime)
- **DtDECLARE_STOP_PARAM_ENCODER** (Reason)
- **DtDECLARE_STOP_PARAM_ENCODER** (ReflectValues)
- **DtDECLARE_STOP_PARAM_ENCODER** (RunInternalSimulationClock)
- **DtDECLARE_STOP_PARAM_ENCODER** (UpdateAttributes)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **DtStopFreezeEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**
- **virtual ~DtStopFreezeEncoder()**
- **DtStopFreezeEncoder(const DtStopFreezeEncoder& orig)**
- **DtStopFreezeEncoder& operator=(const DtStopFreezeEncoder& orig)**
- **DtDECLARE_STOP_PARAM_ENCODER(OriginatingEntity)**
- **DtDECLARE_STOP_PARAM_ENCODER(ReceivingEntity)**
- **DtDECLARE_STOP_PARAM_ENCODER(RequestID)**
- **DtDECLARE_STOP_PARAM_ENCODER(RealWorldTime)**
- **DtDECLARE_STOP_PARAM_ENCODER(Reason)**
- **DtDECLARE_STOP_PARAM_ENCODER(ReflectValues)**
- **DtDECLARE_STOP_PARAM_ENCODER(RunInternalSimulationClock)**

DtDECLARE_STOP_PARAM_ENCODER(UpdateAttributes)

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hstartenc.h:

class DtStartResumeEncoder: public DtInteractionEncoder

Inheritance:

DtStartResumeEncoder - DtInteractionEncoder

Public Methods

- **DtStartResumeEncoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtStartResumeEncoder** ()
- **DtStartResumeEncoder** (const DtStartResumeEncoder& orig)
- DtStartResumeEncoder& **operator=** (const DtStartResumeEncoder& orig)

Protected Methods

- **DtDECLARE_START_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_START_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_START_PARAM_ENCODER** (RequestID)
- **DtDECLARE_START_PARAM_ENCODER** (RealWorldTime)
- **DtDECLARE_START_PARAM_ENCODER** (SimulationTime)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void(***DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const

- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtStartResumeEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtStartResumeEncoder()`
- `DtStartResumeEncoder(const DtStartResumeEncoder& orig)`
- `DtStartResumeEncoder& operator=(const DtStartResumeEncoder& orig)`
- `DtDECLARE_START_PARAM_ENCODER(OriginatingEntity)`
- `DtDECLARE_START_PARAM_ENCODER(ReceivingEntity)`
- `DtDECLARE_START_PARAM_ENCODER(RequestID)`
- `DtDECLARE_START_PARAM_ENCODER(RealWorldTime)`
- `DtDECLARE_START_PARAM_ENCODER(SimulationTime)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hsetdataenc.h:

class DtSetDataEncoder: public DtInteractionEncoder

Inheritance:

DtSetDataEncoder - DtInteractionEncoder

Public Methods

	DtSetDataEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtSetDataEncoder** ()
	DtSetDataEncoder (const DtSetDataEncoder& orig)
DtSetDataEncoder&	**operator=** (const DtSetDataEncoder& orig)

Protected Methods

- **DtDECLARE_SETDATA_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_SETDATA_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_SETDATA_PARAM_ENCODER** (RequestID)
- **DtDECLARE_SETDATA_PARAM_ENCODER** (FixedDatums)
- **DtDECLARE_SETDATA_PARAM_ENCODER** (VariableDatumSet)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const

- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtSetDataEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtSetDataEncoder()`
- `DtSetDataEncoder(const DtSetDataEncoder& orig)`
- `DtSetDataEncoder& operator=(const DtSetDataEncoder& orig)`
- `DtDECLARE_SETDATA_PARAM_ENCODER(OriginatingEntity)`
- `DtDECLARE_SETDATA_PARAM_ENCODER(ReceivingEntity)`
- `DtDECLARE_SETDATA_PARAM_ENCODER(RequestID)`
- `DtDECLARE_SETDATA_PARAM_ENCODER(FixedDatums)`
- `DtDECLARE_SETDATA_PARAM_ENCODER(VariableDatumSet)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hservreqenc.h:

class DtServiceRequestEncoder: public DtInteractionEncoder

Inheritance:

DtServiceRequestEncoder - DtInteractionEncoder

Public Methods

	DtServiceRequestEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtServiceRequestEncoder** ()
	DtServiceRequestEncoder (const DtServiceRequestEncoder& orig)
DtServiceRequestEncoder&	**operator=** (const DtServiceRequestEncoder& orig)

Protected Methods

- **DtDECLARE_SERVREQ_PARAM_ENCODER** (RequestingObject)
- **DtDECLARE_SERVREQ_PARAM_ENCODER** (ServicingObject)
- **DtDECLARE_SERVREQ_PARAM_ENCODER** (ServiceType)
- **DtDECLARE_SERVREQ_PARAM_ENCODER** (Supplies)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

virtual DtParameterEncoder **encoder**(int handle) const

virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)

- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtServiceRequestEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtServiceRequestEncoder()`
- `DtServiceRequestEncoder(const DtServiceRequestEncoder& orig)`
- `DtServiceRequestEncoder& operator=(const DtServiceRequestEncoder& orig)`
- `DtDECLARE_SERVREQ_PARAM_ENCODER(RequestingObject)`
- `DtDECLARE_SERVREQ_PARAM_ENCODER(ServicingObject)`
- `DtDECLARE_SERVREQ_PARAM_ENCODER(ServiceType)`
- `DtDECLARE_SERVREQ_PARAM_ENCODER(Supplies)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hresrecenc.h:

class DtResupplyReceivedEncoder: public DtInteractionEncoder

Inheritance:

DtResupplyReceivedEncoder - DtInteractionEncoder

Public Methods

	DtResupplyReceivedEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtResupplyReceivedEncoder** ()
	DtResupplyReceivedEncoder (const DtResupplyReceivedEncoder& orig)
DtResupplyReceivedEncoder&	**operator=** (const DtResupplyReceivedEncoder& orig)

Protected Methods

- **DtDECLARE_RESREC_PARAM_ENCODER** (ReceivingObject)
- **DtDECLARE_RESREC_PARAM_ENCODER** (SupplyingObject)
- **DtDECLARE_RESREC_PARAM_ENCODER** (Supplies)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)

- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtResupplyReceivedEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtResupplyReceivedEncoder()`
- `DtResupplyReceivedEncoder(const DtResupplyReceivedEncoder& orig)`
- `DtResupplyReceivedEncoder& operator=(const DtResupplyReceivedEncoder& orig)`
- `DtDECLARE_RESREC_PARAM_ENCODER(ReceivingObject)`
- `DtDECLARE_RESREC_PARAM_ENCODER(SupplyingObject)`
- `DtDECLARE_RESREC_PARAM_ENCODER(Supplies)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hresoffenc.h:

class DtResupplyOfferEncoder: public DtInteractionEncoder

Inheritance:

DtResupplyOfferEncoder - DtInteractionEncoder

Public Methods

	DtResupplyOfferEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtResupplyOfferEncoder** ()
	DtResupplyOfferEncoder (const DtResupplyOfferEncoder& orig)
DtResupplyOfferEncoder&	**operator=** (const DtResupplyOfferEncoder& orig)

Protected Methods

- **DtDECLARE_RESOFF_PARAM_ENCODER** (ReceivingObject)
- **DtDECLARE_RESOFF_PARAM_ENCODER** (SupplyingObject)
- **DtDECLARE_RESOFF_PARAM_ENCODER** (Supplies)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtResupplyOfferEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtResupplyOfferEncoder()`
- `DtResupplyOfferEncoder(const DtResupplyOfferEncoder& orig)`
- `DtResupplyOfferEncoder& operator=(const DtResupplyOfferEncoder& orig)`
- `DtDECLARE_RESOFF_PARAM_ENCODER(ReceivingObject)`
- `DtDECLARE_RESOFF_PARAM_ENCODER(SupplyingObject)`
- `DtDECLARE_RESOFF_PARAM_ENCODER(Supplies)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrescancenc.h:

class DtResupplyCancelEncoder: public DtInteractionEncoder

Inheritance:

DtResupplyCancelEncoder - DtInteractionEncoder

Public Methods

- **DtResupplyCancelEncoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtResupplyCancelEncoder** ()
- **DtResupplyCancelEncoder** (const DtResupplyCancelEncoder& orig)
- DtResupplyCancelEncoder& **operator=** (const DtResupplyCancelEncoder& orig)

Protected Methods

- **DtDECLARE_RESCANC_PARAM_ENCODER** (ReceivingObject)
- **DtDECLARE_RESCANC_PARAM_ENCODER** (SupplyingObject)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const

- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtResupplyCancelEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtResupplyCancelEncoder()`
- `DtResupplyCancelEncoder(const DtResupplyCancelEncoder& orig)`
- `DtResupplyCancelEncoder& operator=(const DtResupplyCancelEncoder& orig)`
- `DtDECLARE_RESCANC_PARAM_ENCODER(ReceivingObject)`
- `DtDECLARE_RESCANC_PARAM_ENCODER(SupplyingObject)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrepresp enc.h:

class DtRepairResponseEncoder: public DtInteractionEncoder

Inheritance:

DtRepairResponseEncoder - DtInteractionEncoder

Public Methods

	DtRepairResponseEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRepairResponseEncoder** ()
	DtRepairResponseEncoder (const DtRepairResponseEncoder& orig)
DtRepairResponseEncoder&	**operator=** (const DtRepairResponseEncoder& orig)

Protected Methods

- **DtDECLARE_REPRESP_PARAM_ENCODER** (ReceivingObject)
- **DtDECLARE_REPRESP_PARAM_ENCODER** (RepairingObject)
- **DtDECLARE_REPRESP_PARAM_ENCODER** (RepairResult)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtRepairResponseEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRepairResponseEncoder()`
- `DtRepairResponseEncoder(const DtRepairResponseEncoder& orig)`
- `DtRepairResponseEncoder& operator=(const DtRepairResponseEncoder& orig)`
- `DtDECLARE_REPRESP_PARAM_ENCODER(ReceivingObject)`
- `DtDECLARE_REPRESP_PARAM_ENCODER(RepairingObject)`
- `DtDECLARE_REPRESP_PARAM_ENCODER(RepairResult)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrepcompenc.h:

class DtRepairCompleteEncoder: public DtInteractionEncoder

Inheritance:

DtRepairCompleteEncoder - DtInteractionEncoder

Public Methods

	DtRepairCompleteEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRepairCompleteEncoder** ()
	DtRepairCompleteEncoder (const DtRepairCompleteEncoder& orig)
DtRepairCompleteEncoder&	**operator=** (const DtRepairCompleteEncoder& orig)

Protected Methods

- **DtDECLARE_REPCOMP_PARAM_ENCODER** (ReceivingObject)
- **DtDECLARE_REPCOMP_PARAM_ENCODER** (RepairingObject)
- **DtDECLARE_REPCOMP_PARAM_ENCODER** (Repair)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtRepairCompleteEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRepairCompleteEncoder()`
- `DtRepairCompleteEncoder(const DtRepairCompleteEncoder& orig)`
- `DtRepairCompleteEncoder& operator=(const DtRepairCompleteEncoder& orig)`
- `DtDECLARE_REPCOMP_PARAM_ENCODER(ReceivingObject)`
- `DtDECLARE_REPCOMP_PARAM_ENCODER(RepairingObject)`
- `DtDECLARE_REPCOMP_PARAM_ENCODER(Repair)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremobjresenc.h:

class DtRemoveObjectResultEncoder: public DtInteractionEncoder

---

Inheritance:

DtRemoveObjectResultEncoder - DtInteractionEncoder

Public Methods

	DtRemoveObjectResultEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRemoveObjectResultEncoder** ()
	DtRemoveObjectResultEncoder (const DtRemoveObjectResultEncoder& orig)
DtRemoveObjectResultEncoder&	**operator=** (const DtRemoveObjectResultEncoder& orig)

Protected Methods

- **DtDECLARE_REMOBJRES_PARAM_ENCODER** (RemoveObjectResult)
- **DtDECLARE_REMOBJRES_PARAM_ENCODER** (RequestID)
- **DtDECLARE_REMOBJRES_PARAM_ENCODER** (RemoveObjectResultPre08)

Inherited from DtInteractionEncoder:

Public Methods

typedef void(***DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

virtual DtParameterEncoder **encoder**(int handle) const

virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)

- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtRemoveObjectResultEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRemoveObjectResultEncoder()`
- `DtRemoveObjectResultEncoder(const DtRemoveObjectResultEncoder& orig)`
- `DtRemoveObjectResultEncoder& operator=( const DtRemoveObjectResultEncoder& orig)`
- `DtDECLARE_REMOBJRES_PARAM_ENCODER(RemoveObjectResult)`
- `DtDECLARE_REMOBJRES_PARAM_ENCODER(RequestID)`
- `DtDECLARE_REMOBJRES_PARAM_ENCODER(RemoveObjectResultPre08)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremobjreqenc.h:

class DtRemoveObjectRequestEncoder: public DtInteractionEncoder

---

Inheritance:

DtRemoveObjectRequestEncoder - DtInteractionEncoder

Public Methods

	DtRemoveObjectRequestEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRemoveObjectRequestEncoder** ()
	DtRemoveObjectRequestEncoder (const DtRemoveObjectRequestEncoder& orig)
DtRemoveObjectRequestEncoder&	**operator=** (const DtRemoveObjectRequestEncoder& orig)

Protected Methods

- **DtDECLARE_REMOBJREQ_PARAM_ENCODER** (ObjectIDs)
- **DtDECLARE_REMOBJREQ_PARAM_ENCODER** (RequestID)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

virtual DtParameterEncoder **encoder**(int handle) const

virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)

virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtRemoveObjectRequestEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRemoveObjectRequestEncoder()`
- `DtRemoveObjectRequestEncoder(const DtRemoveObjectRequestEncoder& orig)`
- `DtRemoveObjectRequestEncoder& operator=( const DtRemoveObjectRequestEncoder& orig)`
- `DtDECLARE_REMOBJREQ_PARAM_ENCODER(ObjectIDs)`
- `DtDECLARE_REMOBJREQ_PARAM_ENCODER(RequestID)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremoventenc.h:

class DtRemoveEntityEncoder: public DtInteractionEncoder

Inheritance:

DtRemoveEntityEncoder - DtInteractionEncoder

Public Methods

	DtRemoveEntityEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRemoveEntityEncoder** ()
	DtRemoveEntityEncoder (const DtRemoveEntityEncoder& orig)
DtRemoveEntityEncoder&	**operator=** (const DtRemoveEntityEncoder& orig)

Protected Methods

- **DtDECLARE_REMOVENT_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_REMOVENT_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_REMOVENT_PARAM_ENCODER** (RequestID)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtRemoveEntityEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRemoveEntityEncoder()`
- `DtRemoveEntityEncoder(const DtRemoveEntityEncoder& orig)`
- `DtRemoveEntityEncoder& operator=(const DtRemoveEntityEncoder& orig)`
- `DtDECLARE_REMOVENT_PARAM_ENCODER(OriginatingEntity)`
- `DtDECLARE_REMOVENT_PARAM_ENCODER(ReceivingEntity)`
- `DtDECLARE_REMOVENT_PARAM_ENCODER(RequestID)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrawbinrsenc.h:

class DtRawBinaryRadioSignalEncoder: public DtInteractionEncoder

---

Inheritance:

DtRawBinaryRadioSignalEncoder - DtInteractionEncoder

Public Methods

	DtRawBinaryRadioSignalEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRawBinaryRadioSignalEncoder** ()
	DtRawBinaryRadioSignalEncoder (const DtRawBinaryRadioSignalEncoder& orig)
DtRawBinaryRadioSignalEncoder&	**operator=** (const DtRawBinaryRadioSignalEncoder& orig)

Protected Methods

- **DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER** (HostRadioID)
- **DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER** (DataRate)
- **DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER** (SignalDataLength)
- **DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER** (SignalData)
- **DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER** (TacticalDataLinkType)
- **DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER** (TDLMessageCount)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtRawBinaryRadioSignalEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRawBinaryRadioSignalEncoder()`
- `DtRawBinaryRadioSignalEncoder(const DtRawBinaryRadioSignalEncoder& orig)`
- `DtRawBinaryRadioSignalEncoder& operator=( const DtRawBinaryRadioSignalEncoder& orig)`
- `DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER(HostRadioID)`
- `DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER(DataRate)`
- `DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER(SignalDataLength)`
- `DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER(SignalData)`
- `DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER(TacticalDataLinkType)`
- `DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER(TDLMessageCount)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hlcenc.h:

class DtLgrControlEncoder: public DtInteractionEncoder

Inheritance:

DtLgrControlEncoder - DtInteractionEncoder

Public Methods

	DtLgrControlEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtLgrControlEncoder** ()
	DtLgrControlEncoder (const DtLgrControlEncoder& orig)
DtLgrControlEncoder&	**operator=** (const DtLgrControlEncoder& orig)

Protected Methods

- **DtDECLARE_LGR_CONTROL_PARAM_ENCODER** (LgrControlPdu)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const

- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtLgrControlEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtLgrControlEncoder()`
- `DtLgrControlEncoder(const DtLgrControlEncoder& orig)`
- `DtLgrControlEncoder& operator=(const DtLgrControlEncoder& orig)`
- `DtDECLARE_LGR_CONTROL_PARAM_ENCODER(LgrControlPdu)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hfireenc.h:

class DtFireEncoder: public DtInteractionEncoder

Inheritance:

DtFireEncoder - DtInteractionEncoder

Public Methods

	DtFireEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtFireEncoder** ()
	DtFireEncoder (const DtFireEncoder& orig)
DtFireEncoder&	**operator=** (const DtFireEncoder& orig)

Protected Methods

- **DtDECLARE_FIRE_PARAM_ENCODER** (EventID)
- **DtDECLARE_FIRE_PARAM_ENCODER** (FireControlSolutionRange)
- **DtDECLARE_FIRE_PARAM_ENCODER** (FireMissionIndex)
- **DtDECLARE_FIRE_PARAM_ENCODER** (FiringLocation)
- **DtDECLARE_FIRE_PARAM_ENCODER** (FiringObjectID)
- **DtDECLARE_FIRE_PARAM_ENCODER** (FuseType)
- **DtDECLARE_FIRE_PARAM_ENCODER** (InitialVelocityVector)
- **DtDECLARE_FIRE_PARAM_ENCODER** (MunitionObjectID)
- **DtDECLARE_FIRE_PARAM_ENCODER** (MunitionType)
- **DtDECLARE_FIRE_PARAM_ENCODER** (QuantityFired)
- **DtDECLARE_FIRE_PARAM_ENCODER** (RateOfFire)
- **DtDECLARE_FIRE_PARAM_ENCODER** (TargetObjectID)
- **DtDECLARE_FIRE_PARAM_ENCODER** (WarheadType)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void(***DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **`DtFireEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtFireEncoder()`**
- **`DtFireEncoder(const DtFireEncoder& orig)`**
- **`DtFireEncoder& operator=(const DtFireEncoder& orig)`**
- **`DtDECLARE_FIRE_PARAM_ENCODER(EventID)`**
- **`DtDECLARE_FIRE_PARAM_ENCODER(FireControlSolutionRange)`**
- **`DtDECLARE_FIRE_PARAM_ENCODER(FireMissionIndex)`**

- **DtDECLARE_FIRE_PARAM_ENCODER(FiringLocation)**
- **DtDECLARE_FIRE_PARAM_ENCODER(FiringObjectID)**
- **DtDECLARE_FIRE_PARAM_ENCODER(FuseType)**
- **DtDECLARE_FIRE_PARAM_ENCODER(InitialVelocityVector)**
- **DtDECLARE_FIRE_PARAM_ENCODER(MunitionObjectID)**
- **DtDECLARE_FIRE_PARAM_ENCODER(MunitionType)**
- **DtDECLARE_FIRE_PARAM_ENCODER(QuantityFired)**
- **DtDECLARE_FIRE_PARAM_ENCODER(RateOfFire)**
- **DtDECLARE_FIRE_PARAM_ENCODER(TargetObjectID)**
- **DtDECLARE_FIRE_PARAM_ENCODER(WarheadType)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file heventrepenc.h:

class DtEventReportEncoder: public DtInteractionEncoder

Inheritance:

DtEventReportEncoder - DtInteractionEncoder

Public Methods

- **DtEventReportEncoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtEventReportEncoder** ()
- **DtEventReportEncoder** (const DtEventReportEncoder& orig)
- DtEventReportEncoder& **operator=** (const DtEventReportEncoder& orig)

Protected Methods

- **DtDECLARE_EVENTREP_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_EVENTREP_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_EVENTREP_PARAM_ENCODER** (EventType)
- **DtDECLARE_EVENTREP_PARAM_ENCODER** (FixedDatums)
- **DtDECLARE_EVENTREP_PARAM_ENCODER** (VariableDatumSet)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void(***DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const

- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **`DtEventReportEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtEventReportEncoder()`**
- **`DtEventReportEncoder(const DtEventReportEncoder& orig)`**
- **`DtEventReportEncoder& operator=(const DtEventReportEncoder& orig)`**
- **`DtDECLARE_EVENTREP_PARAM_ENCODER(OriginatingEntity)`**
- **`DtDECLARE_EVENTREP_PARAM_ENCODER(ReceivingEntity)`**
- **`DtDECLARE_EVENTREP_PARAM_ENCODER(EventType)`**
- **`DtDECLARE_EVENTREP_PARAM_ENCODER(FixedDatums)`**
- **`DtDECLARE_EVENTREP_PARAM_ENCODER(VariableDatumSet)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hencarsigenc.h:

class DtEncodedAudioRadioSignalEncoder: public DtInteractionEncoder

---

Inheritance:

DtEncodedAudioRadioSignalEncoder - DtInteractionEncoder

Public Methods

	DtEncodedAudioRadioSignalEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtEncodedAudioRadioSignalEncoder** ()
	DtEncodedAudioRadioSignalEncoder (const DtEncodedAudioRadioSignalEncoder& orig)
DtEncodedAudioRadioSignalEncoder&	**operator=** (const DtEncodedAudioRadioSignalEncoder& orig)

Protected Methods

- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER** (HostRadioID)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER** (EncodingType)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER** (SampleRate)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER** (SampleCount)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER** (SignalDataLength)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER** (SignalData)

Inherited from DtInteractionEncoder:

Public Methods

typedef void(***DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **DtEncodedAudioRadioSignalEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**
- **virtual ~DtEncodedAudioRadioSignalEncoder()**
- **DtEncodedAudioRadioSignalEncoder(const DtEncodedAudioRadioSignalEncoder& orig)**
- **DtEncodedAudioRadioSignalEncoder& operator=(const DtEncodedAudioRadioSignalEncoder& orig)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER(HostRadioID)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER(EncodingType)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER(SampleRate)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER(SampleCount)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER(SignalDataLength)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER(SignalData)**

This class has no child classes.

In file hdetenc.h:

class DtDetonationEncoder: public DtInteractionEncoder

Inheritance:

DtDetonationEncoder - DtInteractionEncoder

Public Methods

- **DtDetonationEncoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtDetonationEncoder** ()
- **DtDetonationEncoder** (const DtDetonationEncoder& orig)
- DtDetonationEncoder& **operator=** (const DtDetonationEncoder& orig)

Protected Methods

- **DtDECLARE_DETONATION_PARAM_ENCODER** (ArticulatedPartsArray)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (ArticulatedParametersArray)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (ArticulatedParametersCount)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (DetonationLocation)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (DetonationResult)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (EventID)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (FiringObjectID)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (FinalVelocityVector)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (FuseType)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (MunitionObjectID)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (MunitionType)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (QuantityFired)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (RateOfFire)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (RelativeDetonationLocation)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (TargetObjectID)
- **DtDECLARE_DETONATION_PARAM_ENCODER** (WarheadType)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void(***DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **`DtDetonationEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtDetonationEncoder()`**
- **`DtDetonationEncoder(const DtDetonationEncoder& orig)`**
- **`DtDetonationEncoder& operator=(const DtDetonationEncoder& orig)`**
- **`DtDECLARE_DETONATION_PARAM_ENCODER(ArticulatedPartsArray)`**
- **`DtDECLARE_DETONATION_PARAM_ENCODER(ArticulatedParametersArray)`**

- **DtDECLARE_DETONATION_PARAM_ENCODER(ArticulatedParametersCount)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(DetonationLocation)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(DetonationResult)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(EventID)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(FiringObjectID)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(FinalVelocityVector)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(FuseType)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(MunitionObjectID)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(MunitionType)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(QuantityFired)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(RateOfFire)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(RelativeDetonationLocation)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(TargetObjectID)**
- **DtDECLARE_DETONATION_PARAM_ENCODER(WarheadType)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdbiradsigenc.h:

class DtDatabaseIndexRadioSignalEncoder: public DtInteractionEncoder

---

Inheritance:

DtDatabaseIndexRadioSignalEncoder - DtInteractionEncoder

Public Methods

	DtDatabaseIndexRadioSignalEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtDatabaseIndexRadioSignalEncoder** ()
	DtDatabaseIndexRadioSignalEncoder (const DtDatabaseIndexRadioSignalEncoder& orig)
DtDatabaseIndexRadioSignalEncoder&	**operator=** (const DtDatabaseIndexRadioSignalEncoder& orig)

Protected Methods

- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER** (HostRadioID)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER** (DatabaseIndex)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER** (Duration)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER** (StartOffset)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER** (TacticalDataLinkType)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER** (TDLMessageCount)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **DtDatabaseIndexRadioSignalEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**
- **virtual ~DtDatabaseIndexRadioSignalEncoder()**
- **DtDatabaseIndexRadioSignalEncoder(const DtDatabaseIndexRadioSignalEncoder& orig)**
- **DtDatabaseIndexRadioSignalEncoder& operator=(const DtDatabaseIndexRadioSignalEncoder& orig)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER(HostRadioID)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER(DatabaseIndex)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER(Duration)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER(StartOffset)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER(TacticalDataLinkType)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER(TDLMessageCount)**

This class has no child classes.

In file hdataqenc.h:

class DtDataQueryEncoder: public DtInteractionEncoder

Inheritance:

DtDataQueryEncoder - DtInteractionEncoder

Public Methods

	DtDataQueryEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtDataQueryEncoder** ()
	DtDataQueryEncoder (const DtDataQueryEncoder& orig)
DtDataQueryEncoder&	**operator=** (const DtDataQueryEncoder& orig)

Protected Methods

- **DtDECLARE_DATAQ_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_DATAQ_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_DATAQ_PARAM_ENCODER** (RequestID)
- **DtDECLARE_DATAQ_PARAM_ENCODER** (TimeInterval)
- **DtDECLARE_DATAQ_PARAM_ENCODER** (FixedDatumIDs)
- **DtDECLARE_DATAQ_PARAM_ENCODER** (VariableDatumIDs)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

virtual DtParameterEncoder **encoder**(int handle) const

- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **`DtDataQueryEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtDataQueryEncoder()`**
- **`DtDataQueryEncoder(const DtDataQueryEncoder& orig)`**
- **`DtDataQueryEncoder& operator=(const DtDataQueryEncoder& orig)`**
- **`DtDECLARE_DATAQ_PARAM_ENCODER(OriginatingEntity)`**
- **`DtDECLARE_DATAQ_PARAM_ENCODER(ReceivingEntity)`**
- **`DtDECLARE_DATAQ_PARAM_ENCODER(RequestID)`**
- **`DtDECLARE_DATAQ_PARAM_ENCODER(TimeInterval)`**
- **`DtDECLARE_DATAQ_PARAM_ENCODER(FixedDatumIDs)`**
- **`DtDECLARE_DATAQ_PARAM_ENCODER(VariableDatumIDs)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdataenc.h:

class DtDataEncoder: public DtInteractionEncoder

Inheritance:

DtDataEncoder - DtInteractionEncoder

Public Methods

- **DtDataEncoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtDataEncoder** ()
- **DtDataEncoder** (const DtDataEncoder& orig)
- DtDataEncoder& **operator=** (const DtDataEncoder& orig)

Protected Methods

- **DtDECLARE_DATA_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_DATA_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_DATA_PARAM_ENCODER** (RequestID)
- **DtDECLARE_DATA_PARAM_ENCODER** (FixedDatums)
- **DtDECLARE_DATA_PARAM_ENCODER** (VariableDatumSet)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtDataEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtDataEncoder()`
- `DtDataEncoder(const DtDataEncoder& orig)`
- `DtDataEncoder& operator=(const DtDataEncoder& orig)`
- `DtDECLARE_DATA_PARAM_ENCODER(OriginatingEntity)`
- `DtDECLARE_DATA_PARAM_ENCODER(ReceivingEntity)`
- `DtDECLARE_DATA_PARAM_ENCODER(RequestID)`
- `DtDECLARE_DATA_PARAM_ENCODER(FixedDatums)`
- `DtDECLARE_DATA_PARAM_ENCODER(VariableDatumSet)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcrtobjresenc.h:

class DtCreateObjectResultEncoder: public DtInteractionEncoder

---

Inheritance:

DtCreateObjectResultEncoder - DtInteractionEncoder

Public Methods

	DtCreateObjectResultEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtCreateObjectResultEncoder** ()
	DtCreateObjectResultEncoder (const DtCreateObjectResultEncoder& orig)
DtCreateObjectResultEncoder&	**operator=** (const DtCreateObjectResultEncoder& orig)

Protected Methods

- **DtDECLARE_CRTOBJRES_PARAM_ENCODER** (CreateObjectResult)
- **DtDECLARE_CRTOBJRES_PARAM_ENCODER** (RequestID)
- **DtDECLARE_CRTOBJRES_PARAM_ENCODER** (CreateObjectResultPre08)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void(***DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)

- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtCreateObjectResultEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCreateObjectResultEncoder()`
- `DtCreateObjectResultEncoder(const DtCreateObjectResultEncoder& orig)`
- `DtCreateObjectResultEncoder& operator=( const DtCreateObjectResultEncoder& orig)`
- `DtDECLARE_CRTOBJRES_PARAM_ENCODER(CreateObjectResult)`
- `DtDECLARE_CRTOBJRES_PARAM_ENCODER(RequestID)`
- `DtDECLARE_CRTOBJRES_PARAM_ENCODER(CreateObjectResultPre08)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcrtobjreqenc.h:

class DtCreateObjReqEncoder: public DtInteractionEncoder

Inheritance:

DtCreateObjReqEncoder - DtInteractionEncoder

Public Methods

	DtCreateObjReqEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtCreateObjReqEncoder** ()
	DtCreateObjReqEncoder (const DtCreateObjReqEncoder& orig)
DtCreateObjReqEncoder&	**operator=** (const DtCreateObjReqEncoder& orig)

Protected Methods

- **DtDECLARE_CRTOBJREQ_PARAM_ENCODER** (ObjectClass)
- **DtDECLARE_CRTOBJREQ_PARAM_ENCODER** (AttributeValueSet)
- **DtDECLARE_CRTOBJREQ_PARAM_ENCODER** (RequestID)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtCreateObjReqEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCreateObjReqEncoder()`
- `DtCreateObjReqEncoder(const DtCreateObjReqEncoder& orig)`
- `DtCreateObjReqEncoder& operator=(const DtCreateObjReqEncoder& orig)`
- `DtDECLARE_CRTOBJREQ_PARAM_ENCODER(ObjectClass)`
- `DtDECLARE_CRTOBJREQ_PARAM_ENCODER(AttributeValueSet)`
- `DtDECLARE_CRTOBJREQ_PARAM_ENCODER(RequestID)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcreatentenc.h:

class DtCreateEntityEncoder: public DtInteractionEncoder

Inheritance:

DtCreateEntityEncoder - DtInteractionEncoder

Public Methods

	DtCreateEntityEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtCreateEntityEncoder** ()
	DtCreateEntityEncoder (const DtCreateEntityEncoder& orig)
DtCreateEntityEncoder&	**operator=** (const DtCreateEntityEncoder& orig)

Protected Methods

- **DtDECLARE_CREATENT_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_CREATENT_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_CREATENT_PARAM_ENCODER** (RequestID)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtCreateEntityEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCreateEntityEncoder()`
- `DtCreateEntityEncoder(const DtCreateEntityEncoder& orig)`
- `DtCreateEntityEncoder& operator=(const DtCreateEntityEncoder& orig)`
- `DtDECLARE_CREATENT_PARAM_ENCODER(OriginatingEntity)`
- `DtDECLARE_CREATENT_PARAM_ENCODER(ReceivingEntity)`
- `DtDECLARE_CREATENT_PARAM_ENCODER(RequestID)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcommentenc.h:

class DtCommentEncoder: public DtInteractionEncoder

Inheritance:

DtCommentEncoder - DtInteractionEncoder

Public Methods

	DtCommentEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtCommentEncoder** ()
	DtCommentEncoder (const DtCommentEncoder& orig)
DtCommentEncoder&	**operator=** (const DtCommentEncoder& orig)

Protected Methods

- **DtDECLARE_COMMENT_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_COMMENT_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_COMMENT_PARAM_ENCODER** (VariableDatumSet)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtCommentEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCommentEncoder()`
- `DtCommentEncoder(const DtCommentEncoder& orig)`
- `DtCommentEncoder& operator=(const DtCommentEncoder& orig)`
- `DtDECLARE_COMMENT_PARAM_ENCODER(OriginatingEntity)`
- `DtDECLARE_COMMENT_PARAM_ENCODER(ReceivingEntity)`
- `DtDECLARE_COMMENT_PARAM_ENCODER(VariableDatumSet)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcollideenc.h:

class DtCollisionEncoder: public DtInteractionEncoder

Inheritance:

DtCollisionEncoder - DtInteractionEncoder

Public Methods

- **DtCollisionEncoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtCollisionEncoder** ()
- **DtCollisionEncoder** (const DtCollisionEncoder& orig)
- DtCollisionEncoder& **operator=** (const DtCollisionEncoder& orig)

Protected Methods

- **DtDECLARE_COLLISION_PARAM_ENCODER** (CollidingObjectID)
- **DtDECLARE_COLLISION_PARAM_ENCODER** (IssuingObjectMass)
- **DtDECLARE_COLLISION_PARAM_ENCODER** (IssuingObjectVelocity)
- **DtDECLARE_COLLISION_PARAM_ENCODER** (CollisionType)
- **DtDECLARE_COLLISION_PARAM_ENCODER** (CollisionLocation)
- **DtDECLARE_COLLISION_PARAM_ENCODER** (EventID)
- **DtDECLARE_COLLISION_PARAM_ENCODER** (IssuingObjectID)
- **DtDECLARE_COLLISION_PARAM_ENCODER** (CollidingObjectMass)
- **DtDECLARE_COLLISION_PARAM_ENCODER** (CollidingObjectVelocity)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void(***DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtCollisionEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCollisionEncoder()`
- `DtCollisionEncoder(const DtCollisionEncoder& orig)`
- `DtCollisionEncoder& operator=(const DtCollisionEncoder& orig)`
- `DtDECLARE_COLLISION_PARAM_ENCODER(CollidingObjectID)`
- `DtDECLARE_COLLISION_PARAM_ENCODER(IssuingObjectMass)`
- `DtDECLARE_COLLISION_PARAM_ENCODER(IssuingObjectVelocity)`
- `DtDECLARE_COLLISION_PARAM_ENCODER(CollisionType)`
- `DtDECLARE_COLLISION_PARAM_ENCODER(CollisionLocation)`
- `DtDECLARE_COLLISION_PARAM_ENCODER(EventID)`

- `DtDECLARE_COLLISION_PARAM_ENCODER(IssuingObjectID)`
- `DtDECLARE_COLLISION_PARAM_ENCODER(CollidingObjectMass)`
- `DtDECLARE_COLLISION_PARAM_ENCODER(CollidingObjectVelocity)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hattchgresenc.h:

class DtAttributeChangeResEnc: public DtInteractionEncoder

Inheritance:

DtAttributeChangeResEnc - DtInteractionEncoder

Public Methods

- **DtAttributeChangeResEnc** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtAttributeChangeResEnc** ()
- **DtAttributeChangeResEnc** (const DtAttributeChangeResEnc& orig)
- DtAttributeChangeResEnc& **operator=** (const DtAttributeChangeResEnc& orig)

Protected Methods

- **DtDECLARE_ATTCHGRES_PARAM_ENCODER** (ObjectID)
- **DtDECLARE_ATTCHGRES_PARAM_ENCODER** (AttributeChangeResult)
- **DtDECLARE_ATTCHGRES_PARAM_ENCODER** (AttributeValueSet)
- **DtDECLARE_ATTCHGRES_PARAM_ENCODER** (AttributeChangeResultPre08)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)

- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtAttributeChangeResEnc(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtAttributeChangeResEnc()`
- `DtAttributeChangeResEnc(const DtAttributeChangeResEnc& orig)`
- `DtAttributeChangeResEnc& operator=(const DtAttributeChangeResEnc& orig)`
- `DtDECLARE_ATTCHGRES_PARAM_ENCODER(ObjectID)`
- `DtDECLARE_ATTCHGRES_PARAM_ENCODER(AttributeChangeResult)`
- `DtDECLARE_ATTCHGRES_PARAM_ENCODER(AttributeValueSet)`
- `DtDECLARE_ATTCHGRES_PARAM_ENCODER(AttributeChangeResultPre08)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hattchgreqenc.h:

class DtAttributeChangeReqEnc: public DtInteractionEncoder

Inheritance:

DtAttributeChangeReqEnc - DtInteractionEncoder

Public Methods

- **DtAttributeChangeReqEnc** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtAttributeChangeReqEnc** ()
- **DtAttributeChangeReqEnc** (const DtAttributeChangeReqEnc& orig)
- DtAttributeChangeReqEnc& **operator=** (const DtAttributeChangeReqEnc& orig)

Protected Methods

- **DtDECLARE_ATTCHGREQ_PARAM_ENCODER** (ObjectIDs)
- **DtDECLARE_ATTCHGREQ_PARAM_ENCODER** (AttributeValueSet)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const

- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **`DtAttributeChangeReqEnc(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtAttributeChangeReqEnc()`**
- **`DtAttributeChangeReqEnc(const DtAttributeChangeReqEnc& orig)`**
- **`DtAttributeChangeReqEnc& operator=(const DtAttributeChangeReqEnc& orig)`**
- **`DtDECLARE_ATTCHGREQ_PARAM_ENCODER(ObjectIDs)`**
- **`DtDECLARE_ATTCHGREQ_PARAM_ENCODER(AttributeValueSet)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file happradsigenc.h:

class DtAppSpecificRadioSignalEncoder: public DtInteractionEncoder

---

Inheritance:

DtAppSpecificRadioSignalEncoder - DtInteractionEncoder

Public Methods

	DtAppSpecificRadioSignalEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtAppSpecificRadioSignalEncoder** ()
	DtAppSpecificRadioSignalEncoder (const DtAppSpecificRadioSignalEncoder& orig)
DtAppSpecificRadioSignalEncoder&	**operator=** (const DtAppSpecificRadioSignalEncoder& orig)

Protected Methods

- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER** (HostRadioID)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER** (DataRate)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER** (SignalDataLength)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER** (SignalData)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER** (TacticalDataLinkType)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER** (TDLMessageCount)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER** (UserProtocolID)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **DtAppSpecificRadioSignalEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**
- **virtual ~DtAppSpecificRadioSignalEncoder()**
- **DtAppSpecificRadioSignalEncoder(const DtAppSpecificRadioSignalEncoder& orig)**
- **DtAppSpecificRadioSignalEncoder& operator=(const DtAppSpecificRadioSignalEncoder& orig)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER(HostRadioID)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER(DataRate)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER(SignalDataLength)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER(SignalData)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER(TacticalDataLinkType)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER(TDLMessageCount)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER(UserProtocolID)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactrspobjenc.h:

class DtActionResponseFromObjEncoder: public DtInteractionEncoder

---

Inheritance:

DtActionResponseFromObjEncoder - DtInteractionEncoder

Public Methods

	DtActionResponseFromObjEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtActionResponseFromObjEncoder** ()
	DtActionResponseFromObjEncoder (const DtActionResponseFromObjEncoder& orig)
DtActionResponseFromObjEncoder&	**operator=** (const DtActionResponseFromObjEncoder& orig)

Protected Methods

- **DtDECLARE_ACTRSPOBJ_PARAM_ENCODER** (ActionResult)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtActionResponseFromObjEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtActionResponseFromObjEncoder()`
- `DtActionResponseFromObjEncoder(const DtActionResponseFromObjEncoder& orig)`
- `DtActionResponseFromObjEncoder& operator=( const DtActionResponseFromObjEncoder& orig)`
- `DtDECLARE_ACTRSPOBJ_PARAM_ENCODER(ActionResult)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactrespenc.h:

class DtActionResponseEncoder: public DtInteractionEncoder

Inheritance:

DtActionResponseEncoder - DtInteractionEncoder

Public Methods

	DtActionResponseEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtActionResponseEncoder** ()
	DtActionResponseEncoder (const DtActionResponseEncoder& orig)
DtActionResponseEncoder&	**operator=** (const DtActionResponseEncoder& orig)

Protected Methods

- **DtDECLARE_ACTRESP_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_ACTRESP_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_ACTRESP_PARAM_ENCODER** (RequestID)
- **DtDECLARE_ACTRESP_PARAM_ENCODER** (RequestStatus)
- **DtDECLARE_ACTRESP_PARAM_ENCODER** (FixedDatums)
- **DtDECLARE_ACTRESP_PARAM_ENCODER** (VariableDatumSet)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

virtual DtParameterEncoder **encoder**(int handle) const

- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **`DtActionResponseEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtActionResponseEncoder()`**
- **`DtActionResponseEncoder(const DtActionResponseEncoder& orig)`**
- **`DtActionResponseEncoder& operator=(const DtActionResponseEncoder& orig)`**
- **`DtDECLARE_ACTRESP_PARAM_ENCODER(OriginatingEntity)`**
- **`DtDECLARE_ACTRESP_PARAM_ENCODER(ReceivingEntity)`**
- **`DtDECLARE_ACTRESP_PARAM_ENCODER(RequestID)`**
- **`DtDECLARE_ACTRESP_PARAM_ENCODER(RequestStatus)`**
- **`DtDECLARE_ACTRESP_PARAM_ENCODER(FixedDatums)`**
- **`DtDECLARE_ACTRESP_PARAM_ENCODER(VariableDatumSet)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactreqobjenc.h:

class DtActionRequestToObjEncoder: public DtInteractionEncoder

---

Inheritance:

DtActionRequestToObjEncoder - DtInteractionEncoder

Public Methods

	DtActionRequestToObjEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtActionRequestToObjEncoder** ()
	DtActionRequestToObjEncoder (const DtActionRequestToObjEncoder& orig)
DtActionRequestToObjEncoder&	**operator=** (const DtActionRequestToObjEncoder& orig)

Protected Methods

- **DtDECLARE_ACTREQOBJ_PARAM_ENCODER** (ObjectIDs)
- **DtDECLARE_ACTREQOBJ_PARAM_ENCODER** (Action)

Inherited from DtInteractionEncoder:

Public Methods

- typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)
- virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)
- virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)
- virtual DtParameterEncoder **encoder**(int handle) const
- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()

- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtActionRequestToObjEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtActionRequestToObjEncoder()`
- `DtActionRequestToObjEncoder(const DtActionRequestToObjEncoder& orig)`
- `DtActionRequestToObjEncoder& operator=( const DtActionRequestToObjEncoder& orig)`
- `DtDECLARE_ACTREQOBJ_PARAM_ENCODER(ObjectIDs)`
- `DtDECLARE_ACTREQOBJ_PARAM_ENCODER(Action)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactreqenc.h:

class DtActionRequestEncoder: public DtInteractionEncoder

Inheritance:

DtActionRequestEncoder - DtInteractionEncoder

Public Methods

	DtActionRequestEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtActionRequestEncoder** ()
	DtActionRequestEncoder (const DtActionRequestEncoder& orig)
DtActionRequestEncoder&	**operator=** (const DtActionRequestEncoder& orig)

Protected Methods

- **DtDECLARE_ACTREQ_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_ACTREQ_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_ACTREQ_PARAM_ENCODER** (RequestID)
- **DtDECLARE_ACTREQ_PARAM_ENCODER** (Action)
- **DtDECLARE_ACTREQ_PARAM_ENCODER** (FixedDatums)
- **DtDECLARE_ACTREQ_PARAM_ENCODER** (VariableDatumSet)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

virtual DtParameterEncoder **encoder**(int handle) const

- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- **`DtActionRequestEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtActionRequestEncoder()`**
- **`DtActionRequestEncoder(const DtActionRequestEncoder& orig)`**
- **`DtActionRequestEncoder& operator=(const DtActionRequestEncoder& orig)`**
- **`DtDECLARE_ACTREQ_PARAM_ENCODER(OriginatingEntity)`**
- **`DtDECLARE_ACTREQ_PARAM_ENCODER(ReceivingEntity)`**
- **`DtDECLARE_ACTREQ_PARAM_ENCODER(RequestID)`**
- **`DtDECLARE_ACTREQ_PARAM_ENCODER(Action)`**
- **`DtDECLARE_ACTREQ_PARAM_ENCODER(FixedDatums)`**
- **`DtDECLARE_ACTREQ_PARAM_ENCODER(VariableDatumSet)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hackenc.h:

class DtAcknowledgeEncoder: public DtInteractionEncoder

Inheritance:

DtAcknowledgeEncoder - DtInteractionEncoder

Public Methods

	DtAcknowledgeEncoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtAcknowledgeEncoder** ()
	DtAcknowledgeEncoder (const DtAcknowledgeEncoder& orig)
DtAcknowledgeEncoder&	**operator=** (const DtAcknowledgeEncoder& orig)

Protected Methods

- **DtDECLARE_ACK_PARAM_ENCODER** (OriginatingEntity)
- **DtDECLARE_ACK_PARAM_ENCODER** (ReceivingEntity)
- **DtDECLARE_ACK_PARAM_ENCODER** (RequestID)
- **DtDECLARE_ACK_PARAM_ENCODER** (AcknowledgeFlag)
- **DtDECLARE_ACK_PARAM_ENCODER** (ResponseFlag)

Inherited from DtInteractionEncoder:

Public Methods

typedef void**(*DtParameterEncoder)**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* avlist, RTI::ParameterHandle handle)

virtual DtParameterEncoder **addEncoder**(RTI::ParameterHandle h, DtParameterEncoder func)

virtual DtParameterEncoder **addEncoder**(char* name, DtParameterEncoder proc)

virtual DtParameterEncoder **encoder**(int handle) const

- virtual void **encode**(const DtInteractionRep& inter, RTI::ParameterHandleValuePairSet* pvList)
- virtual RTI::InteractionClassHandle **interClassHandle**()
- virtual DtInteractionEncoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- inline DtInteractionEncoder::DtParameterEncoder **encoder**(int h) const
- inline RTI::InteractionClassHandle **interClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtParameterEncoder* **myList**
- static const int **theMaxParametersPerInteraction**

Documentation

- `DtAcknowledgeEncoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtAcknowledgeEncoder()`
- `DtAcknowledgeEncoder(const DtAcknowledgeEncoder& orig)`
- `DtAcknowledgeEncoder& operator=(const DtAcknowledgeEncoder& orig)`
- `DtDECLARE_ACK_PARAM_ENCODER(OriginatingEntity)`
- `DtDECLARE_ACK_PARAM_ENCODER(ReceivingEntity)`
- `DtDECLARE_ACK_PARAM_ENCODER(RequestID)`
- `DtDECLARE_ACK_PARAM_ENCODER(AcknowledgeFlag)`
- `DtDECLARE_ACK_PARAM_ENCODER(ResponseFlag)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hvcdec.h:

class DtViewControlDecoder: public DtInteractionDecoder

Inheritance:

DtViewControlDecoder - DtInteractionDecoder

Public Methods

	DtViewControlDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtViewControlDecoder** ()
	DtViewControlDecoder (const DtViewControlDecoder& orig)
DtViewControlDecoder&	**operator=** (const DtViewControlDecoder& orig)

Protected Methods

- **DtDECLARE_VIEW_CONTROL_PARAM_DECODER** (ViewControlPdu)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)

- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtViewControlDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtViewControlDecoder()`
- `DtViewControlDecoder(const DtViewControlDecoder& orig)`
- `DtViewControlDecoder& operator=(const DtViewControlDecoder& orig)`
- `DtDECLARE_VIEW_CONTROL_PARAM_DECODER(ViewControlPdu)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hstopdec.h:

class DtStopFreezeDecoder: public DtInteractionDecoder

Inheritance:

DtStopFreezeDecoder - DtInteractionDecoder

Public Methods

	DtStopFreezeDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtStopFreezeDecoder** ()
	DtStopFreezeDecoder (const DtStopFreezeDecoder& orig)
DtStopFreezeDecoder&	**operator=** (const DtStopFreezeDecoder& orig)

Protected Methods

- **DtDECLARE_STOP_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_STOP_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_STOP_PARAM_DECODER** (RequestID)
- **DtDECLARE_STOP_PARAM_DECODER** (RealWorldTime)
- **DtDECLARE_STOP_PARAM_DECODER** (Reason)
- **DtDECLARE_STOP_PARAM_DECODER** (ReflectValues)
- **DtDECLARE_STOP_PARAM_DECODER** (RunInternalSimulationClock)
- **DtDECLARE_STOP_PARAM_DECODER** (UpdateAttributes)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)

- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtStopFreezeDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtStopFreezeDecoder()`**
- **`DtStopFreezeDecoder(const DtStopFreezeDecoder& orig)`**
- **`DtStopFreezeDecoder& operator=(const DtStopFreezeDecoder& orig)`**
- **`DtDECLARE_STOP_PARAM_DECODER(OriginatingEntity)`**
- **`DtDECLARE_STOP_PARAM_DECODER(ReceivingEntity)`**
- **`DtDECLARE_STOP_PARAM_DECODER(RequestID)`**

- `DtDECLARE_STOP_PARAM_DECODER(RealWorldTime)`
- `DtDECLARE_STOP_PARAM_DECODER(Reason)`
- `DtDECLARE_STOP_PARAM_DECODER(ReflectValues)`
- `DtDECLARE_STOP_PARAM_DECODER(RunInternalSimulationClock)`
- `DtDECLARE_STOP_PARAM_DECODER(UpdateAttributes)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hstartdec.h:

class DtStartResumeDecoder: public DtInteractionDecoder

Inheritance:

DtStartResumeDecoder - DtInteractionDecoder

Public Methods

- **DtStartResumeDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtStartResumeDecoder** ()
- **DtStartResumeDecoder** (const DtStartResumeDecoder& orig)
- DtStartResumeDecoder& **operator=** (const DtStartResumeDecoder& orig)

Protected Methods

- **DtDECLARE_START_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_START_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_START_PARAM_DECODER** (RequestID)
- **DtDECLARE_START_PARAM_DECODER** (RealWorldTime)
- **DtDECLARE_START_PARAM_DECODER** (SimulationTime)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)

- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtStartResumeDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtStartResumeDecoder()`**
- **`DtStartResumeDecoder(const DtStartResumeDecoder& orig)`**
- **`DtStartResumeDecoder& operator=(const DtStartResumeDecoder& orig)`**
- **`DtDECLARE_START_PARAM_DECODER(OriginatingEntity)`**
- **`DtDECLARE_START_PARAM_DECODER(ReceivingEntity)`**
- **`DtDECLARE_START_PARAM_DECODER(RequestID)`**
- **`DtDECLARE_START_PARAM_DECODER(RealWorldTime)`**
- **`DtDECLARE_START_PARAM_DECODER(SimulationTime)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hsetdatadec.h:

class DtSetDataDecoder: public DtInteractionDecoder

Inheritance:

DtSetDataDecoder - DtInteractionDecoder

Public Methods

	DtSetDataDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtSetDataDecoder** ()
	DtSetDataDecoder (const DtSetDataDecoder& orig)
DtSetDataDecoder&	**operator=** (const DtSetDataDecoder& orig)

Protected Methods

- **DtDECLARE_SETDATA_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_SETDATA_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_SETDATA_PARAM_DECODER** (RequestID)
- **DtDECLARE_SETDATA_PARAM_DECODER** (FixedDatums)
- **DtDECLARE_SETDATA_PARAM_DECODER** (VariableDatumSet)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)

- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtSetDataDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtSetDataDecoder()`**
- **`DtSetDataDecoder(const DtSetDataDecoder& orig)`**
- **`DtSetDataDecoder& operator=(const DtSetDataDecoder& orig)`**
- **`DtDECLARE_SETDATA_PARAM_DECODER(OriginatingEntity)`**
- **`DtDECLARE_SETDATA_PARAM_DECODER(ReceivingEntity)`**
- **`DtDECLARE_SETDATA_PARAM_DECODER(RequestID)`**
- **`DtDECLARE_SETDATA_PARAM_DECODER(FixedDatums)`**
- **`DtDECLARE_SETDATA_PARAM_DECODER(VariableDatumSet)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hservreqdec.h:

class DtServiceRequestDecoder: public DtInteractionDecoder

Inheritance:

DtServiceRequestDecoder - DtInteractionDecoder

Public Methods

	DtServiceRequestDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtServiceRequestDecoder** ()
	DtServiceRequestDecoder (const DtServiceRequestDecoder& orig)
DtServiceRequestDecoder&	**operator=** (const DtServiceRequestDecoder& orig)

Protected Methods

- **DtDECLARE_SERVREQ_PARAM_DECODER** (RequestingObject)
- **DtDECLARE_SERVREQ_PARAM_DECODER** (ServicingObject)
- **DtDECLARE_SERVREQ_PARAM_DECODER** (ServiceType)
- **DtDECLARE_SERVREQ_PARAM_DECODER** (Supplies)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtServiceRequestDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtServiceRequestDecoder()`
- `DtServiceRequestDecoder(const DtServiceRequestDecoder& orig)`
- `DtServiceRequestDecoder& operator=(const DtServiceRequestDecoder& orig)`
- `DtDECLARE_SERVREQ_PARAM_DECODER(RequestingObject)`
- `DtDECLARE_SERVREQ_PARAM_DECODER(ServicingObject)`
- `DtDECLARE_SERVREQ_PARAM_DECODER(ServiceType)`
- `DtDECLARE_SERVREQ_PARAM_DECODER(Supplies)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hresrecdec.h:

class DtResupplyReceivedDecoder: public DtInteractionDecoder

Inheritance:

DtResupplyReceivedDecoder - DtInteractionDecoder

Public Methods

	DtResupplyReceivedDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtResupplyReceivedDecoder** ()
	DtResupplyReceivedDecoder (const DtResupplyReceivedDecoder& orig)
DtResupplyReceivedDecoder&	**operator=** (const DtResupplyReceivedDecoder& orig)

Protected Methods

- **DtDECLARE_RESREC_PARAM_DECODER** (ReceivingObject)
- **DtDECLARE_RESREC_PARAM_DECODER** (SupplyingObject)
- **DtDECLARE_RESREC_PARAM_DECODER** (Supplies)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtResupplyReceivedDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtResupplyReceivedDecoder()`**
- **`DtResupplyReceivedDecoder(const DtResupplyReceivedDecoder& orig)`**
- **`DtResupplyReceivedDecoder& operator=(const DtResupplyReceivedDecoder& orig)`**
- **`DtDECLARE_RESREC_PARAM_DECODER(ReceivingObject)`**
- **`DtDECLARE_RESREC_PARAM_DECODER(SupplyingObject)`**
- **`DtDECLARE_RESREC_PARAM_DECODER(Supplies)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hresoffdec.h:

class DtResupplyOfferDecoder: public DtInteractionDecoder

Inheritance:

DtResupplyOfferDecoder - DtInteractionDecoder

Public Methods

- **DtResupplyOfferDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtResupplyOfferDecoder** ()
- **DtResupplyOfferDecoder** (const DtResupplyOfferDecoder& orig)
- DtResupplyOfferDecoder& **operator=** (const DtResupplyOfferDecoder& orig)

Protected Methods

- **DtDECLARE_RESOFF_PARAM_DECODER** (ReceivingObject)
- **DtDECLARE_RESOFF_PARAM_DECODER** (SupplyingObject)
- **DtDECLARE_RESOFF_PARAM_DECODER** (Supplies)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtResupplyOfferDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtResupplyOfferDecoder()`
- `DtResupplyOfferDecoder(const DtResupplyOfferDecoder& orig)`
- `DtResupplyOfferDecoder& operator=(const DtResupplyOfferDecoder& orig)`
- `DtDECLARE_RESOFF_PARAM_DECODER(ReceivingObject)`
- `DtDECLARE_RESOFF_PARAM_DECODER(SupplyingObject)`
- `DtDECLARE_RESOFF_PARAM_DECODER(Supplies)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrescancdec.h:

class DtResupplyCancelDecoder: public DtInteractionDecoder

Inheritance:

DtResupplyCancelDecoder - DtInteractionDecoder

Public Methods

	DtResupplyCancelDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtResupplyCancelDecoder** ()
	DtResupplyCancelDecoder (const DtResupplyCancelDecoder& orig)
DtResupplyCancelDecoder&	**operator=** (const DtResupplyCancelDecoder& orig)

Protected Methods

- **DtDECLARE_RESCANC_PARAM_DECODER** (ReceivingObject)
- **DtDECLARE_RESCANC_PARAM_DECODER** (SupplyingObject)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const

- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtResupplyCancelDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtResupplyCancelDecoder()`
- `DtResupplyCancelDecoder(const DtResupplyCancelDecoder& orig)`
- `DtResupplyCancelDecoder& operator=(const DtResupplyCancelDecoder& orig)`
- `DtDECLARE_RESCANC_PARAM_DECODER(ReceivingObject)`
- `DtDECLARE_RESCANC_PARAM_DECODER(SupplyingObject)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrepresp dec.h:

class DtRepairResponseDecoder: public DtInteractionDecoder

Inheritance:

DtRepairResponseDecoder - DtInteractionDecoder

Public Methods

	DtRepairResponseDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRepairResponseDecoder** ()
	DtRepairResponseDecoder (const DtRepairResponseDecoder& orig)
DtRepairResponseDecoder&	**operator=** (const DtRepairResponseDecoder& orig)

Protected Methods

- **DtDECLARE_REPRESP_PARAM_DECODER** (ReceivingObject)
- **DtDECLARE_REPRESP_PARAM_DECODER** (RepairingObject)
- **DtDECLARE_REPRESP_PARAM_DECODER** (RepairResult)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtRepairResponseDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRepairResponseDecoder()`
- `DtRepairResponseDecoder(const DtRepairResponseDecoder& orig)`
- `DtRepairResponseDecoder& operator=(const DtRepairResponseDecoder& orig)`
- `DtDECLARE_REPRESP_PARAM_DECODER(ReceivingObject)`
- `DtDECLARE_REPRESP_PARAM_DECODER(RepairingObject)`
- `DtDECLARE_REPRESP_PARAM_DECODER(RepairResult)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrepcompdec.h:

class DtRepairCompleteDecoder: public DtInteractionDecoder

Inheritance:

DtRepairCompleteDecoder - DtInteractionDecoder

Public Methods

- **DtRepairCompleteDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtRepairCompleteDecoder** ()
- **DtRepairCompleteDecoder** (const DtRepairCompleteDecoder& orig)
- DtRepairCompleteDecoder& **operator=** (const DtRepairCompleteDecoder& orig)

Protected Methods

- **DtDECLARE_REPCOMP_PARAM_DECODER** (ReceivingObject)
- **DtDECLARE_REPCOMP_PARAM_DECODER** (RepairingObject)
- **DtDECLARE_REPCOMP_PARAM_DECODER** (Repair)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtRepairCompleteDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRepairCompleteDecoder()`
- `DtRepairCompleteDecoder(const DtRepairCompleteDecoder& orig)`
- `DtRepairCompleteDecoder& operator=(const DtRepairCompleteDecoder& orig)`
- `DtDECLARE_REPCOMP_PARAM_DECODER(ReceivingObject)`
- `DtDECLARE_REPCOMP_PARAM_DECODER(RepairingObject)`
- `DtDECLARE_REPCOMP_PARAM_DECODER(Repair)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremobjresdec.h:

class DtRemoveObjectResultDecoder: public DtInteractionDecoder

---

Inheritance:

DtRemoveObjectResultDecoder - DtInteractionDecoder

Public Methods

	DtRemoveObjectResultDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRemoveObjectResultDecoder** ()
	DtRemoveObjectResultDecoder (const DtRemoveObjectResultDecoder& orig)
DtRemoveObjectResultDecoder&	**operator=** (const DtRemoveObjectResultDecoder& orig)

Protected Methods

- **DtDECLARE_REMOBJRES_PARAM_DECODER** (RemoveObjectResult)
- **DtDECLARE_REMOBJRES_PARAM_DECODER** (RequestID)
- **DtDECLARE_REMOBJRES_PARAM_DECODER** (RemoveObjectResultPre08)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtRemoveObjectResultDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRemoveObjectResultDecoder()`
- `DtRemoveObjectResultDecoder(const DtRemoveObjectResultDecoder& orig)`
- `DtRemoveObjectResultDecoder& operator=( const DtRemoveObjectResultDecoder& orig)`
- `DtDECLARE_REMOBJRES_PARAM_DECODER(RemoveObjectResult)`
- `DtDECLARE_REMOBJRES_PARAM_DECODER(RequestID)`
- `DtDECLARE_REMOBJRES_PARAM_DECODER(RemoveObjectResultPre08)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremobjreqdec.h:

class DtRemoveObjectRequestDecoder: public DtInteractionDecoder

---

Inheritance:

DtRemoveObjectRequestDecoder - DtInteractionDecoder

Public Methods

	DtRemoveObjectRequestDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRemoveObjectRequestDecoder** ()
	DtRemoveObjectRequestDecoder (const DtRemoveObjectRequestDecoder& orig)
DtRemoveObjectRequestDecoder&	**operator=** (const DtRemoveObjectRequestDecoder& orig)

Protected Methods

- **DtDECLARE_REMOBJREQ_PARAM_DECODER** (ObjectIDs)
- **DtDECLARE_REMOBJREQ_PARAM_DECODER** (RequestID)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtRemoveObjectRequestDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRemoveObjectRequestDecoder()`
- `DtRemoveObjectRequestDecoder(const DtRemoveObjectRequestDecoder& orig)`
- `DtRemoveObjectRequestDecoder& operator=( const DtRemoveObjectRequestDecoder& orig)`
- `DtDECLARE_REMOBJREQ_PARAM_DECODER(ObjectIDs)`
- `DtDECLARE_REMOBJREQ_PARAM_DECODER(RequestID)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hremoventdec.h:

class DtRemoveEntityDecoder: public DtInteractionDecoder

Inheritance:

DtRemoveEntityDecoder - DtInteractionDecoder

Public Methods

	DtRemoveEntityDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRemoveEntityDecoder** ()
	DtRemoveEntityDecoder (const DtRemoveEntityDecoder& orig)
DtRemoveEntityDecoder&	**operator=** (const DtRemoveEntityDecoder& orig)

Protected Methods

- **DtDECLARE_REMOVENT_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_REMOVENT_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_REMOVENT_PARAM_DECODER** (RequestID)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtRemoveEntityDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtRemoveEntityDecoder()`
- `DtRemoveEntityDecoder(const DtRemoveEntityDecoder& orig)`
- `DtRemoveEntityDecoder& operator=(const DtRemoveEntityDecoder& orig)`
- `DtDECLARE_REMOVENT_PARAM_DECODER(OriginatingEntity)`
- `DtDECLARE_REMOVENT_PARAM_DECODER(ReceivingEntity)`
- `DtDECLARE_REMOVENT_PARAM_DECODER(RequestID)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrawbinrsdec.h:

class DtRawBinaryRadioSignalDecoder: public DtInteractionDecoder

---

Inheritance:

DtRawBinaryRadioSignalDecoder - DtInteractionDecoder

Public Methods

	DtRawBinaryRadioSignalDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtRawBinaryRadioSignalDecoder** ()
	DtRawBinaryRadioSignalDecoder (const DtRawBinaryRadioSignalDecoder& orig)
DtRawBinaryRadioSignalDecoder&	**operator=** (const DtRawBinaryRadioSignalDecoder& orig)

Protected Methods

- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER** (HostRadioID)
- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER** (DataRate)
- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER** (SignalDataLength)
- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER** (SignalData)
- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER** (TacticalDataLinkType)
- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER** (TDLMessageCount)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)

- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtRawBinaryRadioSignalDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtRawBinaryRadioSignalDecoder()`**
- **`DtRawBinaryRadioSignalDecoder( const DtRawBinaryRadioSignalDecoder& orig)`**
- **`DtRawBinaryRadioSignalDecoder& operator=( const DtRawBinaryRadioSignalDecoder& orig)`**
- **`DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER(HostRadioID)`**
- **`DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER(DataRate)`**
- **`DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER(SignalDataLength)`**

- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER(SignalData)**
- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER(TacticalDataLinkType)**
- **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER(TDLMessageCount)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hlcdec.h:

class DtLgrControlDecoder: public DtInteractionDecoder

Inheritance:

DtLgrControlDecoder - DtInteractionDecoder

Public Methods

- **DtLgrControlDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtLgrControlDecoder** ()
- **DtLgrControlDecoder** (const DtLgrControlDecoder& orig)
- DtLgrControlDecoder& **operator=** (const DtLgrControlDecoder& orig)

Protected Methods

- **DtDECLARE_LGR_CONTROL_PARAM_DECODER** (LgrControlPdu)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)

- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtLgrControlDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtLgrControlDecoder()`
- `DtLgrControlDecoder(const DtLgrControlDecoder& orig)`
- `DtLgrControlDecoder& operator=(const DtLgrControlDecoder& orig)`
- `DtDECLARE_LGR_CONTROL_PARAM_DECODER(LgrControlPdu)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hfiredec.h:

class DtFireDecoder: public DtInteractionDecoder

Inheritance:

DtFireDecoder - DtInteractionDecoder

Public Methods

- **DtFireDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtFireDecoder** ()
- **DtFireDecoder** (const DtFireDecoder& orig)
- DtFireDecoder& **operator=** (const DtFireDecoder& orig)

Protected Methods

- **DtDECLARE_FIRE_PARAM_DECODER** (EventID)
- **DtDECLARE_FIRE_PARAM_DECODER** (FireControlSolutionRange)
- **DtDECLARE_FIRE_PARAM_DECODER** (FireMissionIndex)
- **DtDECLARE_FIRE_PARAM_DECODER** (FiringLocation)
- **DtDECLARE_FIRE_PARAM_DECODER** (FiringObjectID)
- **DtDECLARE_FIRE_PARAM_DECODER** (FuseType)
- **DtDECLARE_FIRE_PARAM_DECODER** (InitialVelocityVector)
- **DtDECLARE_FIRE_PARAM_DECODER** (MunitionObjectID)
- **DtDECLARE_FIRE_PARAM_DECODER** (MunitionType)
- **DtDECLARE_FIRE_PARAM_DECODER** (QuantityFired)
- **DtDECLARE_FIRE_PARAM_DECODER** (RateOfFire)
- **DtDECLARE_FIRE_PARAM_DECODER** (TargetObjectID)
- **DtDECLARE_FIRE_PARAM_DECODER** (WarheadType)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtFireDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtFireDecoder()`**
- **`DtFireDecoder(const DtFireDecoder& orig)`**

- DtFireDecoder& operator=(const DtFireDecoder& orig)
- DtDECLARE_FIRE_PARAM_DECODER(EventID)
- DtDECLARE_FIRE_PARAM_DECODER(FireControlSolutionRange)
- DtDECLARE_FIRE_PARAM_DECODER(FireMissionIndex)
- DtDECLARE_FIRE_PARAM_DECODER(FiringLocation)
- DtDECLARE_FIRE_PARAM_DECODER(FiringObjectID)
- DtDECLARE_FIRE_PARAM_DECODER(FuseType)
- DtDECLARE_FIRE_PARAM_DECODER(InitialVelocityVector)
- DtDECLARE_FIRE_PARAM_DECODER(MunitionObjectID)
- DtDECLARE_FIRE_PARAM_DECODER(MunitionType)
- DtDECLARE_FIRE_PARAM_DECODER(QuantityFired)
- DtDECLARE_FIRE_PARAM_DECODER(RateOfFire)
- DtDECLARE_FIRE_PARAM_DECODER(TargetObjectID)
- DtDECLARE_FIRE_PARAM_DECODER(WarheadType)

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file heventrepdec.h:

class DtEventReportDecoder: public DtInteractionDecoder

Inheritance:

DtEventReportDecoder - DtInteractionDecoder

Public Methods

	DtEventReportDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtEventReportDecoder** ()
	DtEventReportDecoder (const DtEventReportDecoder& orig)
DtEventReportDecoder&	**operator=** (const DtEventReportDecoder& orig)

Protected Methods

- **DtDECLARE_EVENTREP_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_EVENTREP_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_EVENTREP_PARAM_DECODER** (EventType)
- **DtDECLARE_EVENTREP_PARAM_DECODER** (FixedDatums)
- **DtDECLARE_EVENTREP_PARAM_DECODER** (VariableDatumSet)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)

- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtEventReportDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtEventReportDecoder()`**
- **`DtEventReportDecoder(const DtEventReportDecoder& orig)`**
- **`DtEventReportDecoder& operator=(const DtEventReportDecoder& orig)`**
- **`DtDECLARE_EVENTREP_PARAM_DECODER(OriginatingEntity)`**
- **`DtDECLARE_EVENTREP_PARAM_DECODER(ReceivingEntity)`**
- **`DtDECLARE_EVENTREP_PARAM_DECODER(EventType)`**
- **`DtDECLARE_EVENTREP_PARAM_DECODER(FixedDatums)`**
- **`DtDECLARE_EVENTREP_PARAM_DECODER(VariableDatumSet)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hencarsigdec.h:

class DtEncodedAudioRadioSignalDecoder: public DtInteractionDecoder

---

Inheritance:

DtEncodedAudioRadioSignalDecoder - DtInteractionDecoder

Public Methods

	DtEncodedAudioRadioSignalDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtEncodedAudioRadioSignalDecoder** ()
	DtEncodedAudioRadioSignalDecoder (const DtEncodedAudioRadioSignalDecoder& orig)
DtEncodedAudioRadioSignalDecoder&	**operator=** (const DtEncodedAudioRadioSignalDecoder& orig)

Protected Methods

- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER** (HostRadioID)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER** (EncodingType)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER** (SampleRate)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER** (SampleCount)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER** (SignalDataLength)
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER** (SignalData)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)

- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **DtEncodedAudioRadioSignalDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**
- **virtual ~DtEncodedAudioRadioSignalDecoder()**
- **DtEncodedAudioRadioSignalDecoder(const DtEncodedAudioRadioSignalDecoder& orig)**
- **DtEncodedAudioRadioSignalDecoder& operator=(const DtEncodedAudioRadioSignalDecoder& orig)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER(HostRadioID)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER(EncodingType)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER(SampleRate)**

- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER(SampleCount)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER(SignalDataLength)**
- **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER(SignalData)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdetdec.h:

class DtDetonationDecoder: public DtInteractionDecoder

Inheritance:

DtDetonationDecoder - DtInteractionDecoder

Public Methods

	DtDetonationDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtDetonationDecoder** ()
	DtDetonationDecoder (const DtDetonationDecoder& orig)
DtDetonationDecoder&	**operator=** (const DtDetonationDecoder& orig)

Protected Methods

- **DtDECLARE_DETONATION_PARAM_DECODER** (ArticulatedPartsArray)
- **DtDECLARE_DETONATION_PARAM_DECODER** (ArticulatedParametersArray)
- **DtDECLARE_DETONATION_PARAM_DECODER** (ArticulatedParametersCount)
- **DtDECLARE_DETONATION_PARAM_DECODER** (DetonationLocation)
- **DtDECLARE_DETONATION_PARAM_DECODER** (DetonationResult)
- **DtDECLARE_DETONATION_PARAM_DECODER** (EventID)
- **DtDECLARE_DETONATION_PARAM_DECODER** (FiringObjectID)
- **DtDECLARE_DETONATION_PARAM_DECODER** (FinalVelocityVector)
- **DtDECLARE_DETONATION_PARAM_DECODER** (FuseType)
- **DtDECLARE_DETONATION_PARAM_DECODER** (MunitionObjectID)
- **DtDECLARE_DETONATION_PARAM_DECODER** (MunitionType)
- **DtDECLARE_DETONATION_PARAM_DECODER** (QuantityFired)
- **DtDECLARE_DETONATION_PARAM_DECODER** (RateOfFire)
- **DtDECLARE_DETONATION_PARAM_DECODER** (RelativeDetonationLocation)
- **DtDECLARE_DETONATION_PARAM_DECODER** (TargetObjectID)
- **DtDECLARE_DETONATION_PARAM_DECODER** (WarheadType)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void(***DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- DtDetonationDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual ~DtDetonationDecoder()
- DtDetonationDecoder(const DtDetonationDecoder& orig)
- DtDetonationDecoder& operator=(const DtDetonationDecoder& orig)
- DtDECLARE_DETONATION_PARAM_DECODER(ArticulatedPartsArray)
- DtDECLARE_DETONATION_PARAM_DECODER(ArticulatedParametersArray)
- DtDECLARE_DETONATION_PARAM_DECODER(ArticulatedParametersCount)
- DtDECLARE_DETONATION_PARAM_DECODER(DetonationLocation)
- DtDECLARE_DETONATION_PARAM_DECODER(DetonationResult)
- DtDECLARE_DETONATION_PARAM_DECODER(EventID)
- DtDECLARE_DETONATION_PARAM_DECODER(FiringObjectID)
- DtDECLARE_DETONATION_PARAM_DECODER(FinalVelocityVector)
- DtDECLARE_DETONATION_PARAM_DECODER(FuseType)
- DtDECLARE_DETONATION_PARAM_DECODER(MunitionObjectID)
- DtDECLARE_DETONATION_PARAM_DECODER(MunitionType)
- DtDECLARE_DETONATION_PARAM_DECODER(QuantityFired)
- DtDECLARE_DETONATION_PARAM_DECODER(RateOfFire)
- DtDECLARE_DETONATION_PARAM_DECODER(RelativeDetonationLocation)
- DtDECLARE_DETONATION_PARAM_DECODER(TargetObjectID)
- DtDECLARE_DETONATION_PARAM_DECODER(WarheadType)

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdbiradsigdec.h:

class DtDatabaseIndexRadioSignalDecoder: public DtInteractionDecoder

---

Inheritance:

DtDatabaseIndexRadioSignalDecoder - DtInteractionDecoder

Public Methods

	DtDatabaseIndexRadioSignalDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtDatabaseIndexRadioSignalDecoder** ()
	DtDatabaseIndexRadioSignalDecoder (const DtDatabaseIndexRadioSignalDecoder& orig)
DtDatabaseIndexRadioSignalDecoder&	**operator=** (const DtDatabaseIndexRadioSignalDecoder& orig)

Protected Methods

- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER** (HostRadioID)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER** (DatabaseIndex)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER** (Duration)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER** (StartOffset)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER** (TacticalDataLinkType)
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER** (TDLMessageCount)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)

- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **DtDatabaseIndexRadioSignalDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**
- **virtual ~DtDatabaseIndexRadioSignalDecoder()**
- **DtDatabaseIndexRadioSignalDecoder(const DtDatabaseIndexRadioSignalDecoder& orig)**
- **DtDatabaseIndexRadioSignalDecoder& operator=(const DtDatabaseIndexRadioSignalDecoder& orig)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER(HostRadioID)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER(DatabaseIndex)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER(Duration)**

- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER(StartOffset)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER(TacticalDataLinkType)**
- **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER(TDLMessageCount)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdataqdec.h:

class DtDataQueryDecoder: public DtInteractionDecoder

Inheritance:

DtDataQueryDecoder - DtInteractionDecoder

Public Methods

- **DtDataQueryDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtDataQueryDecoder** ()
- **DtDataQueryDecoder** (const DtDataQueryDecoder& orig)
- DtDataQueryDecoder& **operator=** (const DtDataQueryDecoder& orig)

Protected Methods

- **DtDECLARE_DATAQ_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_DATAQ_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_DATAQ_PARAM_DECODER** (RequestID)
- **DtDECLARE_DATAQ_PARAM_DECODER** (TimeInterval)
- **DtDECLARE_DATAQ_PARAM_DECODER** (FixedDatumIDs)
- **DtDECLARE_DATAQ_PARAM_DECODER** (VariableDatumIDs)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const

- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtDataQueryDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtDataQueryDecoder()`
- `DtDataQueryDecoder(const DtDataQueryDecoder& orig)`
- `DtDataQueryDecoder& operator=(const DtDataQueryDecoder& orig)`
- `DtDECLARE_DATAQ_PARAM_DECODER(OriginatingEntity)`
- `DtDECLARE_DATAQ_PARAM_DECODER(ReceivingEntity)`
- `DtDECLARE_DATAQ_PARAM_DECODER(RequestID)`
- `DtDECLARE_DATAQ_PARAM_DECODER(TimeInterval)`

- `DtDECLARE_DATAQ_PARAM_DECODER(FixedDatumIDs)`
- `DtDECLARE_DATAQ_PARAM_DECODER(VariableDatumIDs)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdatadec.h:

class DtDataDecoder: public DtInteractionDecoder

Inheritance:

DtDataDecoder - DtInteractionDecoder

Public Methods

- **DtDataDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtDataDecoder** ()
- **DtDataDecoder** (const DtDataDecoder& orig)
- DtDataDecoder& **operator=** (const DtDataDecoder& orig)

Protected Methods

- **DtDECLARE_DATA_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_DATA_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_DATA_PARAM_DECODER** (RequestID)
- **DtDECLARE_DATA_PARAM_DECODER** (FixedDatums)
- **DtDECLARE_DATA_PARAM_DECODER** (VariableDatumSet)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtDataDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtDataDecoder()`
- `DtDataDecoder(const DtDataDecoder& orig)`
- `DtDataDecoder& operator=(const DtDataDecoder& orig)`
- `DtDECLARE_DATA_PARAM_DECODER(OriginatingEntity)`
- `DtDECLARE_DATA_PARAM_DECODER(ReceivingEntity)`
- `DtDECLARE_DATA_PARAM_DECODER(RequestID)`
- `DtDECLARE_DATA_PARAM_DECODER(FixedDatums)`
- `DtDECLARE_DATA_PARAM_DECODER(VariableDatumSet)`

This class has no child classes.

In file hcrtobjresdec.h:

class DtCreateObjectResultDecoder: public DtInteractionDecoder

---

Inheritance:

DtCreateObjectResultDecoder - DtInteractionDecoder

Public Methods

	DtCreateObjectResultDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtCreateObjectResultDecoder** ()
	DtCreateObjectResultDecoder (const DtCreateObjectResultDecoder& orig)
DtCreateObjectResultDecoder&	**operator=** (const DtCreateObjectResultDecoder& orig)

Protected Methods

- **DtDECLARE_CREATOBJRES_PARAM_DECODER** (CreateObjectResult)
- **DtDECLARE_CREATOBJRES_PARAM_DECODER** (RequestID)
- **DtDECLARE_CREATOBJRES_PARAM_DECODER** (CreateObjectResultPre08)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtCreateObjectResultDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCreateObjectResultDecoder()`
- `DtCreateObjectResultDecoder(const DtCreateObjectResultDecoder& orig)`
- `DtCreateObjectResultDecoder& operator=( const DtCreateObjectResultDecoder& orig)`
- `DtDECLARE_CREATOBJRES_PARAM_DECODER(CreateObjectResult)`
- `DtDECLARE_CREATOBJRES_PARAM_DECODER(RequestID)`
- `DtDECLARE_CREATOBJRES_PARAM_DECODER(CreateObjectResultPre08)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcrtobjreqdec.h:

class DtCreateObjReqDecoder: public DtInteractionDecoder

Inheritance:

DtCreateObjReqDecoder - DtInteractionDecoder

Public Methods

- **DtCreateObjReqDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtCreateObjReqDecoder** ()
- **DtCreateObjReqDecoder** (const DtCreateObjReqDecoder& orig)
- DtCreateObjReqDecoder& **operator=** (const DtCreateObjReqDecoder& orig)

Protected Methods

- **DtDECLARE_CRTOBJREQ_PARAM_DECODER** (ObjectClass)
- **DtDECLARE_CRTOBJREQ_PARAM_DECODER** (AttributeValueSet)
- **DtDECLARE_CRTOBJREQ_PARAM_DECODER** (RequestID)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtCreateObjReqDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCreateObjReqDecoder()`
- `DtCreateObjReqDecoder(const DtCreateObjReqDecoder& orig)`
- `DtCreateObjReqDecoder& operator=(const DtCreateObjReqDecoder& orig)`
- `DtDECLARE_CRTOBJREQ_PARAM_DECODER(ObjectClass)`
- `DtDECLARE_CRTOBJREQ_PARAM_DECODER(AttributeValueSet)`
- `DtDECLARE_CRTOBJREQ_PARAM_DECODER(RequestID)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcreatentdec.h:

class DtCreateEntityDecoder: public DtInteractionDecoder

Inheritance:

DtCreateEntityDecoder - DtInteractionDecoder

Public Methods

	DtCreateEntityDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtCreateEntityDecoder** ()
	DtCreateEntityDecoder (const DtCreateEntityDecoder& orig)
DtCreateEntityDecoder&	**operator=** (const DtCreateEntityDecoder& orig)

Protected Methods

- **DtDECLARE_CREATENT_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_CREATENT_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_CREATENT_PARAM_DECODER** (RequestID)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtCreateEntityDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCreateEntityDecoder()`
- `DtCreateEntityDecoder(const DtCreateEntityDecoder& orig)`
- `DtCreateEntityDecoder& operator=(const DtCreateEntityDecoder& orig)`
- `DtDECLARE_CREATENT_PARAM_DECODER(OriginatingEntity)`
- `DtDECLARE_CREATENT_PARAM_DECODER(ReceivingEntity)`
- `DtDECLARE_CREATENT_PARAM_DECODER(RequestID)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcommentdec.h:

class DtCommentDecoder: public DtInteractionDecoder

Inheritance:

DtCommentDecoder - DtInteractionDecoder

Public Methods

	DtCommentDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtCommentDecoder** ()
	DtCommentDecoder (const DtCommentDecoder& orig)
DtCommentDecoder&	**operator=** (const DtCommentDecoder& orig)

Protected Methods

- **DtDECLARE_COMMENT_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_COMMENT_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_COMMENT_PARAM_DECODER** (VariableDatumSet)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtCommentDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCommentDecoder()`
- `DtCommentDecoder(const DtCommentDecoder& orig)`
- `DtCommentDecoder& operator=(const DtCommentDecoder& orig)`
- `DtDECLARE_COMMENT_PARAM_DECODER(OriginatingEntity)`
- `DtDECLARE_COMMENT_PARAM_DECODER(ReceivingEntity)`
- `DtDECLARE_COMMENT_PARAM_DECODER(VariableDatumSet)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcollidedec.h:

class DtCollisionDecoder: public DtInteractionDecoder

Inheritance:

DtCollisionDecoder - DtInteractionDecoder

Public Methods

- **DtCollisionDecoder** (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
- virtual **~DtCollisionDecoder** ()
- **DtCollisionDecoder** (const DtCollisionDecoder& orig)
- DtCollisionDecoder& **operator=** (const DtCollisionDecoder& orig)

Protected Methods

- **DtDECLARE_COLLISION_PARAM_DECODER** (CollidingObjectID)
- **DtDECLARE_COLLISION_PARAM_DECODER** (IssuingObjectMass)
- **DtDECLARE_COLLISION_PARAM_DECODER** (IssuingObjectVelocity)
- **DtDECLARE_COLLISION_PARAM_DECODER** (CollisionType)
- **DtDECLARE_COLLISION_PARAM_DECODER** (CollisionLocation)
- **DtDECLARE_COLLISION_PARAM_DECODER** (EventID)
- **DtDECLARE_COLLISION_PARAM_DECODER** (IssuingObjectID)
- **DtDECLARE_COLLISION_PARAM_DECODER** (CollidingObjectMass)
- **DtDECLARE_COLLISION_PARAM_DECODER** (CollidingObjectVelocity)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void(***DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)

- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtCollisionDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtCollisionDecoder()`
- `DtCollisionDecoder(const DtCollisionDecoder& orig)`
- `DtCollisionDecoder& operator=(const DtCollisionDecoder& orig)`
- `DtDECLARE_COLLISION_PARAM_DECODER(CollidingObjectID)`
- `DtDECLARE_COLLISION_PARAM_DECODER(IssuingObjectMass)`

- **DtDECLARE_COLLISION_PARAM_DECODER(IssuingObjectVelocity)**
- **DtDECLARE_COLLISION_PARAM_DECODER(CollisionType)**
- **DtDECLARE_COLLISION_PARAM_DECODER(CollisionLocation)**
- **DtDECLARE_COLLISION_PARAM_DECODER(EventID)**
- **DtDECLARE_COLLISION_PARAM_DECODER(IssuingObjectID)**
- **DtDECLARE_COLLISION_PARAM_DECODER(CollidingObjectMass)**
- **DtDECLARE_COLLISION_PARAM_DECODER(CollidingObjectVelocity)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hattchgresdec.h:

class DtAttributeChangeResDec: public DtInteractionDecoder

Inheritance:

DtAttributeChangeResDec - DtInteractionDecoder

Public Methods

	DtAttributeChangeResDec (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtAttributeChangeResDec** ()
	DtAttributeChangeResDec (const DtAttributeChangeResDec& orig)
DtAttributeChangeResDec&	**operator=** (const DtAttributeChangeResDec& orig)

Protected Methods

- **DtDECLARE_ATTCHGRES_PARAM_DECODER** (ObjectID)
- **DtDECLARE_ATTCHGRES_PARAM_DECODER** (AttributeChangeResult)
- **DtDECLARE_ATTCHGRES_PARAM_DECODER** (AttributeValueSet)
- **DtDECLARE_ATTCHGRES_PARAM_DECODER** (AttributeChangeResultPre08)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()

- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtAttributeChangeResDec(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtAttributeChangeResDec()`
- `DtAttributeChangeResDec(const DtAttributeChangeResDec& orig)`
- `DtAttributeChangeResDec& operator=( const DtAttributeChangeResDec& orig)`
- `DtDECLARE_ATTCHGRES_PARAM_DECODER(ObjectID)`
- `DtDECLARE_ATTCHGRES_PARAM_DECODER(AttributeChangeResult)`
- `DtDECLARE_ATTCHGRES_PARAM_DECODER(AttributeValueSet)`
- `DtDECLARE_ATTCHGRES_PARAM_DECODER(AttributeChangeResultPre08)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hattchgreqdec.h:

class DtAttributeChangeReqDec: public DtInteractionDecoder

Inheritance:

DtAttributeChangeReqDec - DtInteractionDecoder

Public Methods

	DtAttributeChangeReqDec (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtAttributeChangeReqDec** ()
	DtAttributeChangeReqDec (const DtAttributeChangeReqDec& orig)
DtAttributeChangeReqDec&	**operator=** (const DtAttributeChangeReqDec& orig)

Protected Methods

- **DtDECLARE_ATTCHGREQ_PARAM_DECODER** (ObjectIDs)
- **DtDECLARE_ATTCHGREQ_PARAM_DECODER** (AttributeValueSet)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const

- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtAttributeChangeReqDec(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtAttributeChangeReqDec()`
- `DtAttributeChangeReqDec(const DtAttributeChangeReqDec& orig)`
- `DtAttributeChangeReqDec& operator=(const DtAttributeChangeReqDec& orig)`
- `DtDECLARE_ATTCHGREQ_PARAM_DECODER(ObjectIDs)`
- `DtDECLARE_ATTCHGREQ_PARAM_DECODER(AttributeValueSet)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file happradsigdec.h:

class DtAppSpecificRadioSignalDecoder: public DtInteractionDecoder

---

Inheritance:

DtAppSpecificRadioSignalDecoder - DtInteractionDecoder

Public Methods

	DtAppSpecificRadioSignalDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtAppSpecificRadioSignalDecoder** ()
	DtAppSpecificRadioSignalDecoder (const DtAppSpecificRadioSignalDecoder& orig)
DtAppSpecificRadioSignalDecoder&	**operator=** (const DtAppSpecificRadioSignalDecoder& orig)

Protected Methods

- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER** (HostRadioID)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER** (DataRate)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER** (SignalDataLength)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER** (SignalData)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER** (TacticalDataLinkType)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER** (TDLMessageCount)
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER** (UserProtocolID)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)

- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtAppSpecificRadioSignalDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtAppSpecificRadioSignalDecoder()`**
- **`DtAppSpecificRadioSignalDecoder( const DtAppSpecificRadioSignalDecoder& orig)`**
- **`DtAppSpecificRadioSignalDecoder& operator=( const DtAppSpecificRadioSignalDecoder& orig)`**
- **`DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER(HostRadioID)`**
- **`DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER(DataRate)`**

- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER(SignalDataLength)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER(SignalData)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER(TacticalDataLinkType)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER(TDLMessageCount)**
- **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER(UserProtocolID)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactrspobjdec.h:

class DtActionResponseFromObjDecoder: public DtInteractionDecoder

---

Inheritance:

DtActionResponseFromObjDecoder - DtInteractionDecoder

Public Methods

	DtActionResponseFromObjDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtActionResponseFromObjDecoder** ()
	DtActionResponseFromObjDecoder (const DtActionResponseFromObjDecoder& orig)
DtActionResponseFromObjDecoder&	**operator=** (const DtActionResponseFromObjDecoder& orig)

Protected Methods

- **DtDECLARE_ACTRESPOBJ_PARAM_DECODER** (ActionResult)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const

- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtActionResponseFromObjDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtActionResponseFromObjDecoder()`**
- **`DtActionResponseFromObjDecoder(const DtActionResponseFromObjDecoder& orig)`**
- **`DtActionResponseFromObjDecoder& operator=( const DtActionResponseFromObjDecoder& orig)`**
- **`DtDECLARE_ACTRESPOBJ_PARAM_DECODER(ActionResult)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactrespdec.h:

class DtActionResponseDecoder: public DtInteractionDecoder

Inheritance:

DtActionResponseDecoder - DtInteractionDecoder

Public Methods

	DtActionResponseDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtActionResponseDecoder** ()
	DtActionResponseDecoder (const DtActionResponseDecoder& orig)
DtActionResponseDecoder&	**operator=** (const DtActionResponseDecoder& orig)

Protected Methods

- **DtDECLARE_ACTRESP_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_ACTRESP_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_ACTRESP_PARAM_DECODER** (RequestID)
- **DtDECLARE_ACTRESP_PARAM_DECODER** (RequestStatus)
- **DtDECLARE_ACTRESP_PARAM_DECODER** (FixedDatums)
- **DtDECLARE_ACTRESP_PARAM_DECODER** (VariableDatumSet)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const

- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **DtActionResponseDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**
- **virtual ~DtActionResponseDecoder()**
- **DtActionResponseDecoder(const DtActionResponseDecoder& orig)**
- **DtActionResponseDecoder& operator=(const DtActionResponseDecoder& orig)**
- **DtDECLARE_ACTRESP_PARAM_DECODER(OriginatingEntity)**
- **DtDECLARE_ACTRESP_PARAM_DECODER(ReceivingEntity)**
- **DtDECLARE_ACTRESP_PARAM_DECODER(RequestID)**
- **DtDECLARE_ACTRESP_PARAM_DECODER(RequestStatus)**

- `DtDECLARE_ACTRESP_PARAM_DECODER(FixedDatums)`
- `DtDECLARE_ACTRESP_PARAM_DECODER(VariableDatumSet)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactreqobjdec.h:

class DtActionRequestToObjDecoder: public DtInteractionDecoder

---

Inheritance:

DtActionRequestToObjDecoder - DtInteractionDecoder

Public Methods

	DtActionRequestToObjDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtActionRequestToObjDecoder** ()
	DtActionRequestToObjDecoder (const DtActionRequestToObjDecoder& orig)
DtActionRequestToObjDecoder&	**operator=** (const DtActionRequestToObjDecoder& orig)

Protected Methods

- **DtDECLARE_ACTREQOBJ_PARAM_DECODER** (ObjectIDs)
- **DtDECLARE_ACTREQOBJ_PARAM_DECODER** (Action)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const

- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- `DtActionRequestToObjDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`
- `virtual ~DtActionRequestToObjDecoder()`
- `DtActionRequestToObjDecoder(const DtActionRequestToObjDecoder& orig)`
- `DtActionRequestToObjDecoder& operator=( const DtActionRequestToObjDecoder& orig)`
- `DtDECLARE_ACTREQOBJ_PARAM_DECODER(ObjectIDs)`
- `DtDECLARE_ACTREQOBJ_PARAM_DECODER(Action)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactreqdec.h:

class DtActionRequestDecoder: public DtInteractionDecoder

Inheritance:

DtActionRequestDecoder - DtInteractionDecoder

Public Methods

	DtActionRequestDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtActionRequestDecoder** ()
	DtActionRequestDecoder (const DtActionRequestDecoder& orig)
DtActionRequestDecoder&	**operator=** (const DtActionRequestDecoder& orig)

Protected Methods

- **DtDECLARE_ACTREQ_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_ACTREQ_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_ACTREQ_PARAM_DECODER** (RequestID)
- **DtDECLARE_ACTREQ_PARAM_DECODER** (Action)
- **DtDECLARE_ACTREQ_PARAM_DECODER** (FixedDatums)
- **DtDECLARE_ACTREQ_PARAM_DECODER** (VariableDatumSet)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const

- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)
- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **DtActionRequestDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)**
- **virtual ~DtActionRequestDecoder()**
- **DtActionRequestDecoder(const DtActionRequestDecoder& orig)**
- **DtActionRequestDecoder& operator=(const DtActionRequestDecoder& orig)**
- **DtDECLARE_ACTREQ_PARAM_DECODER(OriginatingEntity)**
- **DtDECLARE_ACTREQ_PARAM_DECODER(ReceivingEntity)**
- **DtDECLARE_ACTREQ_PARAM_DECODER(RequestID)**
- **DtDECLARE_ACTREQ_PARAM_DECODER(Action)**

- `DtDECLARE_ACTREQ_PARAM_DECODER(FixedDatums)`
- `DtDECLARE_ACTREQ_PARAM_DECODER(VariableDatumSet)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hackdec.h:

class DtAcknowledgeDecoder: public DtInteractionDecoder

Inheritance:

DtAcknowledgeDecoder - DtInteractionDecoder

Public Methods

	DtAcknowledgeDecoder (DtExerciseConn* exConn, DtInterClassDesc* classDesc)
virtual	**~DtAcknowledgeDecoder** ()
	DtAcknowledgeDecoder (const DtAcknowledgeDecoder& orig)
DtAcknowledgeDecoder&	**operator=** (const DtAcknowledgeDecoder& orig)

Protected Methods

- **DtDECLARE_ACK_PARAM_DECODER** (OriginatingEntity)
- **DtDECLARE_ACK_PARAM_DECODER** (ReceivingEntity)
- **DtDECLARE_ACK_PARAM_DECODER** (RequestID)
- **DtDECLARE_ACK_PARAM_DECODER** (AcknowledgeFlag)
- **DtDECLARE_ACK_PARAM_DECODER** (ResponseFlag)

Inherited from DtInteractionDecoder:

Public Methods

- typedef void**(*DtParameterDecoder)**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvlist, int i)
- DtParameterDecoder **addDecoder**(RTI::ParameterHandle handle, DtParameterDecoder proc)
- DtParameterDecoder **addDecoder**(char* name, DtParameterDecoder proc)
- virtual DtParameterDecoder **decoder**(int handle) const
- virtual void **decode**(DtInteractionRep* interaction, const RTI::ParameterHandleValuePairSet& pvList)

- virtual RTI::InteractionClassHandle **interactionClassHandle**()
- virtual DtInteractionDecoder* **clone**() const
- virtual void **setClassDesc**(DtInterClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::InteractionClassHandle **interactionClassHandle**()
- inline void **setClassDesc**(DtInterClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtInterClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myClassHandle**
- DtParameterDecoder* **myList**
- static const int **theMaxParametersPerInteraction**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeParameter**(DtInteractionRep* inter, const RTI::ParameterHandleValuePairSet& pvList, int i)

Documentation

- **`DtAcknowledgeDecoder(DtExerciseConn* exConn, DtInterClassDesc* classDesc)`**
- **`virtual ~DtAcknowledgeDecoder()`**
- **`DtAcknowledgeDecoder(const DtAcknowledgeDecoder& orig)`**
- **`DtAcknowledgeDecoder& operator=(const DtAcknowledgeDecoder& orig)`**
- **`DtDECLARE_ACK_PARAM_DECODER(OriginatingEntity)`**
- **`DtDECLARE_ACK_PARAM_DECODER(ReceivingEntity)`**
- **`DtDECLARE_ACK_PARAM_DECODER(RequestID)`**
- **`DtDECLARE_ACK_PARAM_DECODER(AcknowledgeFlag)`**
- **`DtDECLARE_ACK_PARAM_DECODER(ResponseFlag)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hoststructs.h:

class DtEntityIdentifier

Public Fields

DtSimulationAddress	**DtSimulator**
DtU16	**DtEntity**

Public Methods

	DtEntityIdentifier (const char* str)
	DtEntityIdentifier (int site, int host, int entityNum)
	DtEntityIdentifier (DtNetEntityIdentifier netId)
	DtEntityIdentifier ()
void	**init** (int site, int host, int entityNum)
void	**setFromNet** (DtNetEntityIdentifier netId)
	operator DtNetEntityIdentifier () const
int	**site** () const
int	**host** () const
int	**entityNum** () const
void	**setSite** (int site)
void	**setHost** (int host)
void	**setEntityNum** (int entityNum)
int	**sameHost** (const DtEntityIdentifier &otherGuy) const
int	**sameEntity** (const DtEntityIdentifier &otherGuy) const
int	**matchPattern** (const DtEntityIdentifier &pat) const
int	**operator==** (const DtEntityIdentifier &otherGuy) const
int	**operator!=** (const DtEntityIdentifier &otherGuy) const
char*	**string** () const
static const DtEntityIdentifier&	**nullId** ()

Documentation

- DtEntityIdentifier(const char* str)
- DtEntityIdentifier(int site, int host, int entityNum)
- DtEntityIdentifier(DtNetEntityIdentifier netId)
- DtEntityIdentifier()
- void init(int site, int host, int entityNum)
- void setFromNet(DtNetEntityIdentifier netId)
- operator DtNetEntityIdentifier () const
- int site() const
- int host() const
- int entityNum() const
- void setSite(int site)
- void setHost(int host)
- void setEntityNum(int entityNum)
- int sameHost(const DtEntityIdentifier &otherGuy) const
- int sameEntity(const DtEntityIdentifier &otherGuy) const
- int matchPattern(const DtEntityIdentifier &pat) const
- int operator==(const DtEntityIdentifier &otherGuy) const
- int operator!=(const DtEntityIdentifier &otherGuy) const
- char* string() const
- static const DtEntityIdentifier& nullId()
- DtSimulationAddress DtSimulator
- DtU16 DtEntity

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtStopReason

- **DtStopReasonOther**
- **DtStopReasonRecess**
- **DtStopReasonTermination**
- **DtStopReasonSystemFailure**
- **DtStopReasonSecurityViolation**
- **DtStopReasonEntityReconstitution**

Documentation

- `DtStopReasonOther`
- `DtStopReasonRecess`
- `DtStopReasonTermination`
- `DtStopReasonSystemFailure`
- `DtStopReasonSecurityViolation`
- `DtStopReasonEntityReconstitution`

In file disenums.h:

enum DtTacticalDataLinkType

- **DtTacticalDataLinkOther**
- **DtAbbreviated_Command_And_Control**
- **DtTdlGC3**

Documentation

- `DtTacticalDataLinkOther`
- `DtAbbreviated_Command_And_Control`
- `DtTdlGC3`

In file disenums.h:

enum DtEncodingType

- **DtEncodingTypeOther**
- **DtEncoding_8_bit_mu_law**
- **DtCVSD_per_MIL_STD_188_113**
- **DtADPCM_per_CCITT_G721**
- **DtEncoding_16_bit_linear_PCM**
- **DtEncoding_8_bit_linear_PCM**
- **DtVQ_Vector_Quantization**

Documentation

- `DtEncodingTypeOther`
- `DtEncoding_8_bit_mu_law`
- `DtCVSD_per_MIL_STD_188_113`
- `DtADPCM_per_CCITT_G721`
- `DtEncoding_16_bit_linear_PCM`
- `DtEncoding_8_bit_linear_PCM`
- `DtVQ_Vector_Quantization`

In file disenums.h:

enum DtEncodingClass

- **DtSignalClassEncodedVoice**
- **DtSignalClassRawBinary**
- **DtSignalClassApplicationSpecific**
- **DtSignalClassPreRecorded**

Documentation

- `DtSignalClassEncodedVoice`
- `DtSignalClassRawBinary`
- `DtSignalClassApplicationSpecific`
- `DtSignalClassPreRecorded`

In file disenums.h:

enum DtUserProtocol

- **DtUserProtocolOther**
- **DtCCSIL**
- **DtA2ATD_SINCGARS_ERF**
- **DtA2ATD_CAC2**
- **DtBattle_Command**
- **DtModSAF_Text_Radio**
- **DtCCTT_SINCGARS_ERF_LOCKOUT**
- **DtCCTT_SINCGARS_ERF_HOPSET**
- **DtCCTT_SINCGARS_OTAR**
- **DtCCTT_SINCGARS_DATA**
- **DtModSAF_FWA_Forward_Air_Controller**
- **DtModSAF_Threat_ADA_C3**
- **DtModSAF_Artillery_Fire_Control**
- **DtAGTS**
- **DtUserGC3**
- **DtWNCP_data**

Documentation

- `DtUserProtocolOther`
- `DtCCSIL`
- `DtA2ATD_SINCGARS_ERF`
- `DtA2ATD_CAC2`
- `DtBattle_Command`
- `DtModSAF_Text_Radio`
- `DtCCTT_SINCGARS_ERF_LOCKOUT`

- **DtCCTT_SINCGARS_ERF_HOPSET**
- **DtCCTT_SINCGARS_OTAR**
- **DtCCTT_SINCGARS_DATA**
- **DtModSAF_FWA_Forward_Air_Controller**
- **DtModSAF_Threat_ADA_C3**
- **DtModSAF_Artillery_Fire_Control**
- **DtAGTS**
- **DtUserGC3**
- **DtWNCP_data**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtServiceType

- **DtServiceTypeOther**
- **DtServiceTypeResupply**
- **DtServiceTypeRepair**

Documentation

- **`DtServiceTypeOther`**
- **`DtServiceTypeResupply`**
- **`DtServiceTypeRepair`**

In file disenums.h:

enum DtRepairResult

- **DtRepairResultOther**
- **DtRepairResultEnded**
- **DtRepairResultInvalid**
- **DtRepairResultInterrupted**
- **DtRepairResultCancelled**

Documentation

- `DtRepairResultOther`
- `DtRepairResultEnded`
- `DtRepairResultInvalid`
- `DtRepairResultInterrupted`
- `DtRepairResultCancelled`

In file disenums.h:

enum DtRepairType

Section 2

- **DtNoRepairsPerformed**
- **DtRepairEverything**
- **DtDriveTrainMotor**
- **DtDriveTrainStarter**
- **DtDriveTrainAlternator**
- **DtDriveTrainGenerator**
- **DtDriveTrainBattery**
- **DtDriveTrainEngineCoolantLeak**
- **DtDriveTrainFuelFilter**
- **DtDriveTrainTranmissionOilLeak**
- **DtDriveTrainEngineOilLeak**
- **DtDriveTrainPumps**
- **DtDriveTrainFilters**
- **DtBodyHull**
- **DtBodyAirframe**
- **DtInterfacePropeller**
- **DtDriveTrainTransmission**
- **DtDriveTrainBrakes**
- **DtInterfaceFilters**
- **DtWeaponGunElevationDrive**
- **DtWeaponGunStabilizationSys**
- **DtWeaponGunnersPrimarySight**
- **DtWeaponCommandersGPSExtension**
- **DtWeaponLoadingMecahnism**
- **DtFuelTransferPump**
- **DtFuelLines**

- **DtFuelGauges**
- **DtElectronicWarfareSys**
- **DtElectronicDetectionSys**
- **DtElectronicRangeFinder**
- **DtElectronicSys**
- **DtElectronicRadios**
- **DtElectronicIntercoms**
- **DtElectronicLasers**
- **DtElectronicCoders**
- **DtElectronicDecoders**
- **DtElectronicComputers**
- **DtLifeSupportAirSupply**
- **DtLifeSupportFilters**
- **DtLifeSupportWaterSupply**
- **DtLifeSupportRefrigerationSys**
- **DtLifeSupportCBRProtection**
- **DtLifeSupportWaterWashSys**
- **DtLifeSupportDecontaminationSys**
- **DtActuatorWaterSupply**
- **DtActuatorCoolingSys**
- **DtActuatorWinches**
- **DtActuatorCatapults**
- **DtActuatorCranes**
- **DtActuatorLaunchers**
- **DtAuxCraftLifeBoat**
- **DtAuxCraftLandingCraft**
- **DtAuxCraftEjectionSeat**

Documentation

Section 2

- `DtNoRepairsPerformed`

- **DtRepairEverything**
- **DtDriveTrainMotor**
- **DtDriveTrainStarter**
- **DtDriveTrainAlternator**
- **DtDriveTrainGenerator**
- **DtDriveTrainBattery**
- **DtDriveTrainEngineCoolantLeak**
- **DtDriveTrainFuelFilter**
- **DtDriveTrainTranmissionOilLeak**
- **DtDriveTrainEngineOilLeak**
- **DtDriveTrainPumps**
- **DtDriveTrainFilters**
- **DtBodyHull**
- **DtBodyAirframe**
- **DtInterfacePropeller**
- **DtDriveTrainTransmission**
- **DtDriveTrainBrakes**
- **DtInterfaceFilters**
- **DtWeaponGunElevationDrive**
- **DtWeaponGunStabilizationSys**
- **DtWeaponGunnersPrimarySight**
- **DtWeaponCommandersGPSExtension**
- **DtWeaponLoadingMecahnism**
- **DtFuelTransferPump**
- **DtFuelLines**
- **DtFuelGauges**
- **DtElectronicWarfareSys**
- **DtElectronicDetectionSys**
- **DtElectronicRangeFinder**
- **DtElectronicSys**
- **DtElectronicRadios**
- **DtElectronicIntercoms**

- **DtElectronicLasers**
- **DtElectronicCoders**
- **DtElectronicDecoders**
- **DtElectronicComputers**
- **DtLifeSupportAirSupply**
- **DtLifeSupportFilters**
- **DtLifeSupportWaterSupply**
- **DtLifeSupportRefrigerationSys**
- **DtLifeSupportCBRProtection**
- **DtLifeSupportWaterWashSys**
- **DtLifeSupportDecontaminationSys**
- **DtActuatorWaterSupply**
- **DtActuatorCoolingSys**
- **DtActuatorWinches**
- **DtActuatorCatapults**
- **DtActuatorCranes**
- **DtActuatorLaunchers**
- **DtAuxCraftLifeBoat**
- **DtAuxCraftLandingCraft**
- **DtAuxCraftEjectionSeat**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtRemoveObjectResult

DtRemoveObjectResultOther

Documentation

`DtRemoveObjectResultOther`

In file objdeslist.h:

class DtGlobalObjectDesignatorList

Public Methods

	DtGlobalObjectDesignatorList () *default constructor*
virtual	**~DtGlobalObjectDesignatorList** () *destructor*
	DtGlobalObjectDesignatorList (const DtGlobalObjectDesignatorList& orig) *copy constructor*
DtGlobalObjectDesignatorList&	**operator=** (const DtGlobalObjectDesignatorList& orig) *assignment operator*
int	**operator==** (const DtGlobalObjectDesignatorList& other) const *equivlance operators*
int	**operator!=** (const DtGlobalObjectDesignatorList& other) const
virtual void	**setNumObjects** (int num) *Set/Get number of objects*
virtual int	**numObjects** () const
virtual DtBoolean	**setObject** (int index, const DtGlobalObjectDesignator& des) *Set index'th object designator*
virtual const DtGlobalObjectDesignator&	**object** (int index, DtBoolean* valid = NULL) const *Get index'th object designator*
virtual void	**print** () const *Print data*
virtual void	**setFromNet** (const void* netRep, int length) *Set from network representation*
virtual const void*	**netRep** (int* size = NULL) const *Returns network representation, fills size with size of netRep*
virtual int	**netSize** () const *Returns size of network representation*

Protected Fields

int	**myNum**
DtGlobalObjectDesignator*	**myObjs**

Documentation

- **DtGlobalObjectDesignatorList()**

  default constructor

- **virtual ~DtGlobalObjectDesignatorList()**

  destructor

- **DtGlobalObjectDesignatorList(const DtGlobalObjectDesignatorList& orig)**

  copy constructor

- **DtGlobalObjectDesignatorList& operator=(const DtGlobalObjectDesignatorList& orig)**

  assignment operator

- **int operator==(const DtGlobalObjectDesignatorList& other) const**

  equivlance operators

- **int operator!=(const DtGlobalObjectDesignatorList& other) const**

- **virtual void setNumObjects(int num)**

  Set/Get number of objects. Setting to a number lower than the current number will remove trailing objects from the list.

- **virtual int numObjects() const**

- **virtual DtBoolean setObject(int index, const DtGlobalObjectDesignator& des)**

  Set index'th object designator

- **virtual const DtGlobalObjectDesignator& object(int index, DtBoolean* valid = NULL) const**

  Get index'th object designator. Set valid to DtFalse if bad index, DtTrue otherwise.

- **virtual void print() const**

  Print data

- **virtual void setFromNet(const void* netRep, int length)**

  Set from network representation

- **virtual const void* netRep(int* size = NULL) const**

  Returns network representation, fills size with size of netRep. Return value is pointer to internal object with same lifetime as this. Do not delete.

- **virtual int netSize() const**

  Returns size of network representation

- **int myNum**

- **DtGlobalObjectDesignator* myObjs**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcpdu.h:

class DtLgrControlPdu: public DtPdu

class DtLgrControlPdu: Instances of DtLgrControlPdu are a family of messgae used to control a remote MaK-Logger over the network

---

Inheritance:

DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrControlPdu () *default constructor*
static DtPdu*	**create** (const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *Create a PDU representation from a network packet*
static DtLgrControlPdu*	**create** (DtLgrCommand command) *Create a PDU from DtLgrCommand;*
virtual	**~DtLgrControlPdu** () *destructor*
	DtLgrControlPdu (const DtLgrControlPdu& orig) *copy constructor*
DtLgrControlPdu&	**operator=** (const DtLgrControlPdu& orig) *assignment operator*
virtual void	**setReceiverId** (const DtGlobalObjectDesignator& id) *set/get Object Identifier of intended recipient (logger)*
const DtGlobalObjectDesignator&	**receiverId** () const
virtual void	**setSenderId** (const DtGlobalObjectDesignator& id) *set/get Object Identifier of sender, if applicable*
virtual const DtGlobalObjectDesignator&	**senderId** () const
DtLgrCommand	**lgrCommand** () const *get message-type for proper casting to a subclass*
virtual void	**printData** () const *print contents*
inline DtPduKind	**internalGetPduKind** () const

Protected Methods

- DtPduKind **internalGetPduKind** () const
  identity for use by toolkit
- void **setLgrCommand** (DtLgrCommand com)
  set is done by way of constructing the proper subclass

Private Fields

- DtGlobalObjectDesignator **mySenderId**
  parameter
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const

- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrControlPdu:

Instances of DtLgrControlPdu are a family of messgae used to control a remote MaK-Logger over the network. See lc*.h

- **`DtLgrControlPdu()`**

  default constructor

- **`static DtPdu* create(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  Create a PDU representation from a network packet

- **`static DtLgrControlPdu* create(DtLgrCommand command)`**

  Create a PDU from DtLgrCommand;

- **`virtual ~DtLgrControlPdu()`**

  destructor

- **`DtLgrControlPdu(const DtLgrControlPdu& orig)`**

  copy constructor

- **`DtLgrControlPdu& operator=(const DtLgrControlPdu& orig)`**

assignment operator

- **`virtual void setReceiverId(const DtGlobalObjectDesignator& id)`**
  set/get Object Identifier of intended recipient (logger)
- **`const DtGlobalObjectDesignator& receiverId() const`**
- **`virtual void setSenderId(const DtGlobalObjectDesignator& id)`**
  set/get Object Identifier of sender, if applicable
- **`virtual const DtGlobalObjectDesignator& senderId() const`**
- **`DtLgrCommand lgrCommand() const`**
  get message-type for proper casting to a subclass
- **`virtual void printData() const`**
  print contents
- **`DtPduKind internalGetPduKind() const`**
  identity for use by toolkit
- **`void setLgrCommand(DtLgrCommand com)`**
  set is done by way of constructing the proper subclass
- **`DtGlobalObjectDesignator mySenderId`**
  parameter
- **`DtGlobalObjectDesignator myReceiverId`**
- **`inline DtPduKind internalGetPduKind() const`**

Direct child classes:

DtLgrTimeoutPdu
DtLgrTimeScalePdu
DtLgrStopPdu
DtLgrStartPdu
DtLgrSkipTimePdu
DtLgrSetTimePdu
DtLgrRecordPdu
DtLgrRecordActionPdu
DtLgrRecFilePdu
DtLgrQuitPdu
DtLgrPlayPdu
DtLgrPlayActionPdu
DtLgrPbFilePdu
DtLgrPausePdu
DtLgrHeartbeatPdu
DtLgrEofActionPdu
DtLgrEndPdu

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtDatumParam

- **DtDatumMass**
- **DtDatumForceId**
- **DtDatumKind**
- **DtDatumDomain**
- **DtDatumCountry**
- **DtDatumCategory**
- **DtDatumSubCategory**
- **DtDatumSpecific**
- **DtDatumExtra**
- **DtDatumAltKind**
- **DtDatumAltDomain**
- **DtDatumAltCountry**
- **DtDatumAltCategory**
- **DtDatumAltSubCategory**
- **DtDatumAltSpecific**
- **DtDatumAltExtra**
- **DtDatumXpos**
- **DtDatumYpos**
- **DtDatumZpos**
- **DtDatumXvel**
- **DtDatumYvel**
- **DtDatumZvel**
- **DtDatumPhi**
- **DtDatumTheta**
- **DtDatumPsi**
- **DtDatumDrAlgorithm**

- **DtDatumXaccel**
- **DtDatumYaccel**
- **DtDatumZaccel**
- **DtDatumXangvel**
- **DtDatumYangvel**
- **DtDatumZangvel**
- **DtDatumAppearance**
- **DtDatumMarkingCS**
- **DtDatumMarking**
- **DtDatumNumArtParam**
- **DtDatumArtParamId**
- **DtDatumArtParamType**
- **DtDatumArtParamValue**
- **DtDatumAmountAmmoSupplies**
- **DtDatumAmmoQuantity**
- **DtDatumFuelQuantity**
- **DtDatumRadarSysStatus**
- **DtDatumRadioCommSysStatus**
- **DtDatumStationaryRadioTime**
- **DtDatumMovingRadioTime**
- **DtDatumBodyPartDamagedRatio**
- **DtDatumTerrainDatabase**
- **DtDatumLoggerFile**
- **DtDatumCapability**

Documentation

- `DtDatumMass`
- `DtDatumForceId`
- `DtDatumKind`
- `DtDatumDomain`
- `DtDatumCountry`

- **DtDatumCategory**
- **DtDatumSubCategory**
- **DtDatumSpecific**
- **DtDatumExtra**
- **DtDatumAltKind**
- **DtDatumAltDomain**
- **DtDatumAltCountry**
- **DtDatumAltCategory**
- **DtDatumAltSubCategory**
- **DtDatumAltSpecific**
- **DtDatumAltExtra**
- **DtDatumXpos**
- **DtDatumYpos**
- **DtDatumZpos**
- **DtDatumXvel**
- **DtDatumYvel**
- **DtDatumZvel**
- **DtDatumPhi**
- **DtDatumTheta**
- **DtDatumPsi**
- **DtDatumDrAlgorithm**
- **DtDatumXaccel**
- **DtDatumYaccel**
- **DtDatumZaccel**
- **DtDatumXangvel**
- **DtDatumYangvel**
- **DtDatumZangvel**
- **DtDatumAppearance**
- **DtDatumMarkingCS**
- **DtDatumMarking**
- **DtDatumNumArtParam**
- **DtDatumArtParamId**

- **DtDatumArtParamType**
- **DtDatumArtParamValue**
- **DtDatumAmountAmmoSupplies**
- **DtDatumAmmoQuantity**
- **DtDatumFuelQuantity**
- **DtDatumRadarSysStatus**
- **DtDatumRadioCommSysStatus**
- **DtDatumStationaryRadioTime**
- **DtDatumMovingRadioTime**
- **DtDatumBodyPartDamagedRatio**
- **DtDatumTerrainDatabase**
- **DtDatumLoggerFile**
- **DtDatumCapability**

Alphabetic index *Hierarchy of classes*

In file datumrecset.h:

enum DtDatumRecordType

- **DtFixed**
- **DtVar**

Documentation

- **`DtFixed`**
- **`DtVar`**

In file datumrecset.h:

enum DtDatumRetCode

- **DtDatumValid**
- **DtDatumBadLength**
- **DtDatumBadId**
- **DtDatumBadIndex**

Documentation

- `DtDatumValid`
- `DtDatumBadLength`
- `DtDatumBadId`
- `DtDatumBadIndex`

In file datumrecset.h:

class DtVarDatumSpec

This class has public data members for backwards compatibility with pre-VR-Link3

Public Fields

- DtDatumParam **param**
- int **length**

Documentation

This class has public data members for backwards compatibility with pre-VR-Link3.3 versions.

- **`DtDatumParam param`**
- **`int length`**

This class has no child classes.

In file datumrecset.h:

class DtDatumRecordSet: public DtBaseMsg

Inheritance:

DtDatumRecordSet - DtBaseMsg

Public Methods

	DtDatumRecordSet (DtBufferPtr bufferPtr = DtUSE_INTERNAL_BUFFER) *Constructor*
	DtDatumRecordSet (const void* netRep, int size, DtBufferPtr bufferPtr = DtUSE_INTERNAL_BUFFER) *Constructor*
virtual	**~DtDatumRecordSet** () *Destructor*
	DtDatumRecordSet (const DtDatumRecordSet& orig) *Copy ctor*
DtDatumRecordSet&	**operator=** (const DtDatumRecordSet& orig) *Assignment operator*
virtual int	**numFixedFields** () const *returns number of fixed fields*
virtual int	**numVarFields** () const *returns number of variable length fields*
virtual DtDatumParam	**varDatumId** (int index) const *Returns the datum param found in the index'th variable datum record*
virtual DtDatumParam	**fixedDatumId** (int index) const *Returns the datum param found in the index'th fixed datum record*
virtual DtDatumParam	**datumId** (DtDatumRecordType fixedOrVar, int index) const *Returns the datum param found in the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
virtual int	**datumSizeBits** (DtDatumRecordType fixedOrVar, int index) const *Returns the number of bits of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
virtual int	**datumSizeBytes** (DtDatumRecordType fixedOrVar, int index) const *Returns the number of bytes of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar*
virtual void	**getIndex** (DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const *Looks through all fixed and variable records for one containing param as the parameter ID*

- virtual DtFloat32 — **datumValFloat32** (DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
  ******* *INSPECTING DATUM VALUES BY INDEX ********* If you know the data type of the value contained in a particular record, use one of the following to return it*
- virtual DtFloat64 — **datumValFloat64** (DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long — **datumValUnsigned32** (DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual long — **datumValInt32** (DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual const char* — **datumValByteArray** (DtDatumRecordType fixedOrVar, int index) const
- virtual DtFloat32 — **datumValFloat32** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
  ******* *INSPECTING DATUM VALUES BY PARAM ID ********* If you know the data type of the value contained in a particular record, use one of the following to return it*
- virtual DtFloat64 — **datumValFloat64** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long — **datumValUnsigned32** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual long — **datumValInt32** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual const char* — **datumValByteArray** (DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual void — **init** (const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)
  Initialize to reflect the fixed parameters and variable parameters and sizes indicated
- virtual void — **setNumFixedFields** (int newNum)
  Sets the number of fixed or variable fields to newNum
- virtual void — **setNumVarFields** (int newNum)
- virtual void — **setDatumParam** (DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)
  Set the datum ID in the index'th fixed or variable datum record to param, depending on whether fixedOrVar is DtFixed or DtVar
- virtual void — **setVarDataBytes** (int index, int length, DtDatumRetCode* retcode = NULL)
  Allocates length bytes for the index'th variable datum record
- virtual void — **setDatumValFloat64** (DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)
  ******* *SETTING DATA VALUES BY INDEX ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data*
- virtual void — **setDatumValFloat32** (DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void — **setDatumValInt32** (DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)
- virtual void — **setDatumValUnsigned32** (DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void — **setDatumValByteArray** (DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void — **setDatumValFloat64** (DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)
  ****** *SETTING DATA VALUES BY PARAM ID ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data*
- virtual void — **setDatumValFloat32** (DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void — **setDatumValInt32** (DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)

- virtual void **setDatumValUnsigned32** (DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray** (DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void **printData** () const
  prints the data
- virtual DtNetDataPduTail* **netDataRep** () const
  Returns a pointer to the network representation, as might appear in a data PDU
- virtual DtNetFixDatum* **nthFixedDataRecord** (int index) const
  Return a pointer to the index'th fixed data record in the network representation
- virtual DtNetVarDatum* **nthVarDataRecord** (int index) const
  Return a pointer to the index'th variable data record in the network representation
- virtual void **setFromNetFixed** (const char* fixedFields, int numFixedFields)
  Sets the fixed data from a network representation of a fixed datum record array
- virtual void **setFromNetVar** (const char* varFields, int numVarFields, int totalVarFieldsLength)
  Sets the variable data from a network representation consisting of the variable length records concatentated together
- virtual void **sizeChanged** (int size)
  Really, should not be pure virtual in DtBaseMsg, but since it is, we need a null definition here
- inline DtNetDataPduTail* **netDataRep** () const

Public

- ********** *MUTATOR FUNCTIONS* **********

Protected Methods

- virtual void* **datumValPtr** (DtDatumRecordType fixedOrVar, int index) const
  Returns a pointer to the index'th fixed or variable record in the network representation, depending on whether fixedOrVar is DtFixed or DtVar

---

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**

- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- **`DtDatumRecordSet(DtBufferPtr bufferPtr = DtUSE_INTERNAL_BUFFER)`**

  Constructor

- **`DtDatumRecordSet(const void* netRep, int size, DtBufferPtr bufferPtr = DtUSE_INTERNAL_BUFFER)`**

  Constructor

- **`virtual ~DtDatumRecordSet()`**

  Destructor

- **`DtDatumRecordSet(const DtDatumRecordSet& orig)`**

  Copy ctor

- **`DtDatumRecordSet& operator=(const DtDatumRecordSet& orig)`**

  Assignment operator

- **`virtual int numFixedFields() const`**

  returns number of fixed fields

- **`virtual int numVarFields() const`**

  returns number of variable length fields

- **`virtual DtDatumParam varDatumId(int index) const`**

  Returns the datum param found in the index'th variable datum record. Index is between 1 and numRecords. Returns DtDatumParam(0) if index is out of bounds.

- **`virtual DtDatumParam fixedDatumId(int index) const`**

Returns the datum param found in the index'th fixed datum record. Index is between 1 and numRecords. Returns DtDatumParam(0) if index is out of bounds.

virtual DtDatumParam datumId(DtDatumRecordType fixedOrVar, int index) const

Returns the datum param found in the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar. Index is between 1 and numRecords. Returns DtDatumParam(0) if index is out of bounds.

virtual int datumSizeBits(DtDatumRecordType fixedOrVar, int index) const

Returns the number of bits of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar. If DtFixed, the answer is always 32.

virtual int datumSizeBytes(DtDatumRecordType fixedOrVar, int index) const

Returns the number of bytes of data contained in the value fields of the index'th fixed or variable datum record, depending on whether fixedOrVar is DtFixed or DtVar. If DtFixed, the answer is always 4.

virtual void getIndex(DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const

Looks through all fixed and variable records for one containing param as the parameter ID. If it finds one, fills supplied fixedOrVar pointer with DtFixed or DtVar, and index with the index to indicate which record was found, and fills retcode with DtDatumValid. If param was not found, retcode is filled with DtDatumBadId, and index and fixedOrVar are invalid.

virtual DtFloat32 datumValFloat32(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const

******* INSPECTING DATUM VALUES BY INDEX ********* If you know the data type of the value contained in a particular record, use one of the following to return it. fixedOrVar (either DtFixed or DtVar) and index (between 1 and numRecords) indicate which record you are inspecting. datumValBytesArray is used when the data type is a character string, or something other than a long, unsigned long, float or double. It returns a pointer to a character string containing the exact bytes in the network representation that represents the value being inspected. The supplied retcode pointer will be filled with DtDatumValid if the value was successfully inspected, and DtDatumBadLength if either the index was invalid, or the size of the data indicated that it could not be interpreted as the chosen data type.

virtual DtFloat64 datumValFloat64(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const

virtual unsigned long datumValUnsigned32(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const

virtual long datumValInt32(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const

virtual const char* datumValByteArray(DtDatumRecordType fixedOrVar, int index) const

virtual DtFloat32 datumValFloat32(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

******* INSPECTING DATUM VALUES BY PARAM ID ********* If you know the data type of the value contained in a particular record, use one of the following to return it. The particular record inspected will be the first one containing param as the datum ID. datumValByteArray is used when the data type is a character string, or something other than a long, unsigned long, float or double. It returns a pointer to a character string containing the exact bytes in the network representation that represents the value being inspected. The supplied retcode pointer will be filled with DtDatumValid if the value was successfully inspected, and DtDatumBadLength if either param was not found, or the size of the data indicated that it could not be interpreted as the chosen data type.

virtual DtFloat64 datumValFloat64(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

virtual unsigned long datumValUnsigned32(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

virtual long datumValInt32(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

virtual const char* datumValByteArray(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

********** MUTATOR FUNCTIONS **********

virtual void init(const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)

Initialize to reflect the fixed parameters and variable parameters and sizes indicated. fixedParams is a NULL-terminated array of DtDatumParams. specList is an array of DtVarDatumSpecs terminated with the DtNullVarDatumSpec constant. This call can be made instead of calls to setNumVarDatumRecords, setNumFixedDatumRecord, and repeated calls to setDatumParam and setVarDataBytes.

virtual void setNumFixedFields(int newNum)

Sets the number of fixed or variable fields to newNum. Previously existing fixed fields with indices greater than newNum are deleted. If newNum exceeds the number of existing fields, trailing new fields are created with zeros as datum params, sizes (for var only) and values.

virtual void setNumVarFields(int newNum)

virtual void setDatumParam(DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)

Set the datum ID in the index'th fixed or variable datum record to param, depending on whether fixedOrVar is DtFixed or DtVar. Supplied retcode pointer is filled with DtDatumValid if successful, or DtDatumBadIndex if the index is not valid.

virtual void setVarDataBytes(int index, int length, DtDatumRetCode* retcode = NULL)

Allocates length bytes for the index'th variable datum record. Should be called before using one of the setDatumVal functions to actually set the data. Supplied retcode pointer is set to DtDatumValid if successful, or DtDatumBadIndex if the index is not valid.

virtual void setDatumValFloat64(DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)

******* SETTING DATA VALUES BY INDEX ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data. fixedOrVar (either DtFixed or DtVar) and index (between 1 and numRecords) indicate which record you are modifying. setDatumValBytesArray is used when the data type is a character string, or something other than a long, unsigned long, float or double. It takes a pointer to a character string containing the exact bytes that should be used in the network representation. The supplied retcode pointer will be filled with DtDatumValid if the value was successfully set, DtDatumBadIndex if the index is invalid, and DtDatumBadLength if the size of the data indicated that it could not be set from as the chosen data type.

virtual void setDatumValFloat32(DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValInt32(DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValUnsigned32(DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)

virtual void setDatumValByteArray(DtDatumRecordType fixedOrVar, int index, const char*

val, DtDatumRetCode* retcode = NULL)

- **virtual void setDatumValFloat64(DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)**

  ****** SETTING DATA VALUES BY PARAM ID ********* Use one of the following to set the data value in a fixed or variable datum record, depending on the data type of your data. The particular record modified will be the first one containing param as its parameter ID. setDatumValBytesArray is used when the data type is a character string, or something other than a long, unsigned long, float or double. It takes a pointer to a character string containing the exact bytes that should be used in the network representation. The supplied retcode pointer will be filled with DtDatumValid if the value was successfully set, DtDatumBadId if param is invalid, and DtDatumBadLength if the size of the data indicated that it could not be set from the chosen data type.

- **virtual void setDatumValFloat32(DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)**

- **virtual void setDatumValInt32(DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)**

- **virtual void setDatumValUnsigned32(DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)**

- **virtual void setDatumValByteArray(DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)**

- **virtual void printData() const**

  prints the data

- **virtual DtNetDataPduTail* netDataRep() const**

  Returns a pointer to the network representation, as might appear in a data PDU. Consists of numberOfFixedFields, numberOfVarFields, then the fixed and variable fields.

- **virtual DtNetFixDatum* nthFixedDataRecord(int index) const**

  Return a pointer to the index'th fixed data record in the network representation. Index is between 1 and numRecords.

- **virtual DtNetVarDatum* nthVarDataRecord(int index) const**

  Return a pointer to the index'th variable data record in the network representation. Index is between 1 and numRecords.

- **virtual void setFromNetFixed(const char* fixedFields, int numFixedFields)**

  Sets the fixed data from a network representation of a fixed datum record array. Also pass the number of fixed fields.

- **virtual void setFromNetVar(const char* varFields, int numVarFields, int totalVarFieldsLength)**

  Sets the variable data from a network representation consisting of the variable length records concatentated together. Also pass the number of fields, and the total length of the variable data records.

- **virtual void sizeChanged(int size)**

  Really, should not be pure virtual in DtBaseMsg, but since it is, we need a null definition here

- **virtual void* datumValPtr(DtDatumRecordType fixedOrVar, int index) const**

  Returns a pointer to the index'th fixed or variable record in the network representation, depending on whether fixedOrVar is DtFixed or DtVar

- **inline DtNetDataPduTail* netDataRep() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hsetdatainter.h:

class DtSetDataInteraction: public DtInteractionWithData

Inheritance:

DtSetDataInteraction - DtInteractionWithData - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtSetDataInteraction () *default constructor*
virtual	**~DtSetDataInteraction** () *destructor*
	DtSetDataInteraction (const DtSetDataInteraction& orig) *copy constructor*
DtSetDataInteraction&	**operator=** (const DtSetDataInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRequestId** (unsigned long id) *Set/Get request ID*
virtual unsigned long	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtSetDataInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtSetDataInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtSetDataInteractionCb cb, void* usr)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- unsigned long **myRequestId**

Inherited from DtInteractionWithData:

Public Methods

- virtual int **numFixedFields**() const
- virtual int **numVarFields**() const
- virtual DtDatumParam **varDatumId**(int index) const
- virtual DtDatumParam **fixedDatumId**(int index) const
- virtual DtDatumParam **datumId**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBits**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBytes**(DtDatumRecordType fixedOrVar, int index) const
- virtual void **getIndex**(DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat32 **datumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumRecordType fixedOrVar, int index) const
- virtual DtFloat32 **datumValFloat32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual void **initDataSection**(const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)
- virtual void **setNumFixedFields**(int newNum)

- virtual void **setNumVarFields**(int newNum)
- virtual void **setDatumParam**(DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)
- virtual void **setVarDataBytes**(int index, int length, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)
- virtual DtDatumRecordSet* **datumRecSet**()
- virtual const DtDatumRecordSet* **datumRecSet**() const

Public

-

Protected Fields

- DtDatumRecordSet **myRecSet**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const

- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const

- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtSetDataInteraction()`**
  default constructor
- **`virtual ~DtSetDataInteraction()`**
  destructor
- **`DtSetDataInteraction(const DtSetDataInteraction& orig)`**
  copy constructor
- **`DtSetDataInteraction& operator=( const DtSetDataInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setSenderId(const DtEntityIdentifier& id)`**
  Set/Get sender ID
- **`virtual const DtEntityIdentifier& senderId() const`**
- **`virtual void setReceiverId(const DtEntityIdentifier& id)`**
  Set/Get receiver ID
- **`virtual const DtEntityIdentifier& receiverId() const`**
- **`virtual void setRequestId(unsigned long id)`**
  Set/Get request ID
- **`virtual unsigned long requestId() const`**
- **`virtual const char* name() const`**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **`static DtInteraction* create()`**
  Create a DtSetDataInteraction. Caller is reponsible for deletion.
- **`static void addCallback(DtExerciseConn* conn, DtSetDataInteractionCb cb, void* usr)`**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **`static void removeCallback(DtExerciseConn* conn, DtSetDataInteractionCb cb, void* usr)`**

- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **unsigned long myRequestId**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file heventrepint.h:

class DtEventReportInteraction: public DtInteractionWithData

Inheritance:

DtEventReportInteraction - DtInteractionWithData - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtEventReportInteraction () *default constructor*
virtual	**~DtEventReportInteraction** () *destructor*
	DtEventReportInteraction (const DtEventReportInteraction& orig) *copy constructor*
DtEventReportInteraction&	**operator=** (const DtEventReportInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setEventType** (DtEventType type) *Set/Get event Type*
virtual DtEventType	**eventType** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtEventReportInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtEventReportInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtEventReportInteractionCb cb, void* usr)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- DtEventType **myEventType**

Inherited from DtInteractionWithData:

Public Methods

- virtual int **numFixedFields**() const
- virtual int **numVarFields**() const
- virtual DtDatumParam **varDatumId**(int index) const
- virtual DtDatumParam **fixedDatumId**(int index) const
- virtual DtDatumParam **datumId**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBits**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBytes**(DtDatumRecordType fixedOrVar, int index) const
- virtual void **getIndex**(DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat32 **datumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumRecordType fixedOrVar, int index) const
- virtual DtFloat32 **datumValFloat32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual void **initDataSection**(const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)

- virtual void **setNumFixedFields**(int newNum)
- virtual void **setNumVarFields**(int newNum)
- virtual void **setDatumParam**(DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)
- virtual void **setVarDataBytes**(int index, int length, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)
- virtual DtDatumRecordSet* **datumRecSet**()
- virtual const DtDatumRecordSet* **datumRecSet**() const

Public

-

Protected Fields

- DtDatumRecordSet **myRecSet**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const

- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**

- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtEventReportInteraction()`**
  default constructor
- **`virtual ~DtEventReportInteraction()`**
  destructor
- **`DtEventReportInteraction(const DtEventReportInteraction& orig)`**
  copy constructor
- **`DtEventReportInteraction& operator=( const DtEventReportInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setSenderId(const DtEntityIdentifier& id)`**
  Set/Get sender ID
- **`virtual const DtEntityIdentifier& senderId() const`**
- **`virtual void setReceiverId(const DtEntityIdentifier& id)`**
  Set/Get receiver ID
- **`virtual const DtEntityIdentifier& receiverId() const`**
- **`virtual void setEventType(DtEventType type)`**
  Set/Get event Type
- **`virtual DtEventType eventType() const`**
- **`virtual const char* name() const`**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **`static DtInteraction* create()`**
  Create a DtEventReportInteraction. Caller is reponsible for deletion.
- **`static void addCallback(DtExerciseConn* conn, DtEventReportInteractionCb cb, void* usr)`**
  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtEventReportInteractionCb cb, void* usr)**
- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **DtEventType myEventType**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdatainter.h:

class DtDataInteraction: public DtInteractionWithData

Inheritance:

DtDataInteraction - DtInteractionWithData - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtDataInteraction () *default constructor*
virtual	**~DtDataInteraction** () *destructor*
	DtDataInteraction (const DtDataInteraction& orig) *copy constructor*
DtDataInteraction&	**operator=** (const DtDataInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *`Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRequestId** (unsigned long id) *Set/Get request ID*
virtual unsigned long	**requestId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtDataInteraction*
static void	**addCallback** (DtExerciseConn* conn, DtDataInteractionCb cb, void* usr) *Add (register)/Remove (deregister) function to be called when interaction occurs*
static void	**removeCallback** (DtExerciseConn* conn, DtDataInteractionCb cb, void* usr)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- int **myRequestId**

Inherited from DtInteractionWithData:

Public Methods

- virtual int **numFixedFields**() const
- virtual int **numVarFields**() const
- virtual DtDatumParam **varDatumId**(int index) const
- virtual DtDatumParam **fixedDatumId**(int index) const
- virtual DtDatumParam **datumId**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBits**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBytes**(DtDatumRecordType fixedOrVar, int index) const
- virtual void **getIndex**(DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat32 **datumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumRecordType fixedOrVar, int index) const
- virtual DtFloat32 **datumValFloat32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual void **initDataSection**(const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)
- virtual void **setNumFixedFields**(int newNum)

- virtual void **setNumVarFields**(int newNum)
- virtual void **setDatumParam**(DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)
- virtual void **setVarDataBytes**(int index, int length, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)
- virtual DtDatumRecordSet* **datumRecSet**()
- virtual const DtDatumRecordSet* **datumRecSet**() const

Public

-

Protected Fields

- DtDatumRecordSet **myRecSet**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const

- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const

- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtDataInteraction()`**
  default constructor
- **`virtual ~DtDataInteraction()`**
  destructor
- **`DtDataInteraction(const DtDataInteraction& orig)`**
  copy constructor
- **`DtDataInteraction& operator=( const DtDataInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setSenderId(const DtEntityIdentifier& id)`**
  `Set/Get sender ID
- **`virtual const DtEntityIdentifier& senderId() const`**
- **`virtual void setReceiverId(const DtEntityIdentifier& id)`**
  Set/Get receiver ID
- **`virtual const DtEntityIdentifier& receiverId() const`**
- **`virtual void setRequestId(unsigned long id)`**
  Set/Get request ID
- **`virtual unsigned long requestId() const`**
- **`virtual const char* name() const`**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **`static DtInteraction* create()`**
  Create a DtDataInteraction. Caller is reponsible for deletion.
- **`static void addCallback(DtExerciseConn* conn, DtDataInteractionCb cb, void* usr)`**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **`static void removeCallback(DtExerciseConn* conn, DtDataInteractionCb cb, void* usr)`**

- **DtEntityIdentifier myOrigEnt**
- **DtEntityIdentifier myReceiveEnt**
- **int myRequestId**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hcommentinter.h:

class DtCommentInteraction: public DtInteractionWithData

Inheritance:

DtCommentInteraction - DtInteractionWithData - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtCommentInteraction () *default constructor*
virtual	**~DtCommentInteraction** () *destructor*
	DtCommentInteraction (const DtCommentInteraction& orig) *copy constructor*

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**

Inherited from DtInteractionWithData:

Public Methods

- virtual int **numFixedFields**() const
- virtual int **numVarFields**() const
- virtual DtDatumParam **varDatumId**(int index) const
- virtual DtDatumParam **fixedDatumId**(int index) const
- virtual DtDatumParam **datumId**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBits**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBytes**(DtDatumRecordType fixedOrVar, int index) const
- virtual void **getIndex**(DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat32 **datumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumRecordType fixedOrVar, int index) const
- virtual DtFloat32 **datumValFloat32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual void **initDataSection**(const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)
- virtual void **setNumFixedFields**(int newNum)
- virtual void **setNumVarFields**(int newNum)

- virtual void **setDatumParam**(DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)
- virtual void **setVarDataBytes**(int index, int length, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)
- virtual DtDatumRecordSet* **datumRecSet**()
- virtual const DtDatumRecordSet* **datumRecSet**() const

Public

-

Protected Fields

- DtDatumRecordSet **myRecSet**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)

- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const
- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const

- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**
- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const

DtInteraction* **self**() const

Documentation

- **`DtCommentInteraction()`**
  default constructor
- **`virtual ~DtCommentInteraction()`**
  destructor
- **`DtCommentInteraction(const DtCommentInteraction& orig)`**
  copy constructor
- **`DtCommentInteraction& operator=( const DtCommentInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setSenderId(const DtEntityIdentifier& id)`**
  `Set/Get sender ID
- **`virtual const DtEntityIdentifier& senderId() const`**
- **`virtual void setReceiverId(const DtEntityIdentifier& id)`**
  Set/Get receiver ID
- **`virtual const DtEntityIdentifier& receiverId() const`**
- **`virtual void setComment(const char *message, int size)`**
  Set/Get the comment; warning: inspector returns a readable text string. Use DtInteractionWithData member functions to get non-string comments.
- **`virtual const char* comment() const`**
- **`virtual const char* name() const`**
  Name of the interaction (Not nec. the ClassName from the FOM).
- **`static DtInteraction* create()`**
  Create a DtCommentInteraction. Caller is reponsible for deletion.
- **`static void addCallback(DtExerciseConn* conn, DtCommentInteractionCb cb, void* usr)`**
  Add (register)/Remove (deregister) function to be called when interaction occurs
- **`static void removeCallback(DtExerciseConn* conn, DtCommentInteractionCb cb, void* usr)`**
- **`DtEntityIdentifier myOrigEnt`**
- **`DtEntityIdentifier myReceiveEnt`**

This class has no child classes.

In file hactrespinter.h:

class DtActionResponseInteraction: public DtInteractionWithData

---

Inheritance:

DtActionResponseInteraction - DtInteractionWithData - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtActionResponseInteraction () *default constructor*
virtual	**~DtActionResponseInteraction** () *destructor*
	DtActionResponseInteraction (const DtActionResponseInteraction& orig) *copy constructor*
DtActionResponseInteraction&	**operator=** (const DtActionResponseInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *`Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRequestId** (DtU32 id) *Set/Get request ID*
virtual DtU32	**requestId** () const
virtual void	**setRequestStatus** (DtRequestStatus rStatus) *Set/Get the request status*
virtual DtRequestStatus	**requestStatus** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtActionResponseInteraction*

- static void **addCallback** (DtExerciseConn* conn, DtActionResponseInteractionCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtActionResponseInteractionCb cb, void* usr)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- DtU32 **myRequestId**
- DtRequestStatus **myRequestStatus**

Inherited from DtInteractionWithData:

Public Methods

- virtual int **numFixedFields**() const
- virtual int **numVarFields**() const
- virtual DtDatumParam **varDatumId**(int index) const
- virtual DtDatumParam **fixedDatumId**(int index) const
- virtual DtDatumParam **datumId**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBits**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBytes**(DtDatumRecordType fixedOrVar, int index) const
- virtual void **getIndex**(DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat32 **datumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumRecordType fixedOrVar, int index) const
- virtual DtFloat32 **datumValFloat32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

- virtual long **datumValInt32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual void **initDataSection**(const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)
- virtual void **setNumFixedFields**(int newNum)
- virtual void **setNumVarFields**(int newNum)
- virtual void **setDatumParam**(DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)
- virtual void **setVarDataBytes**(int index, int length, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)
- virtual DtDatumRecordSet* **datumRecSet**()
- virtual const DtDatumRecordSet* **datumRecSet**() const

Public

-

Protected Fields

- DtDatumRecordSet **myRecSet**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const

- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**

- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtActionResponseInteraction()`**
  default constructor
- **`virtual ~DtActionResponseInteraction()`**
  destructor
- **`DtActionResponseInteraction(const DtActionResponseInteraction& orig)`**
  copy constructor
- **`DtActionResponseInteraction& operator=( const DtActionResponseInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setSenderId(const DtEntityIdentifier& id)`**
  `Set/Get sender ID
- **`virtual const DtEntityIdentifier& senderId() const`**
- **`virtual void setReceiverId(const DtEntityIdentifier& id)`**
  Set/Get receiver ID
- **`virtual const DtEntityIdentifier& receiverId() const`**
- **`virtual void setRequestId(DtU32 id)`**
  Set/Get request ID
- **`virtual DtU32 requestId() const`**
- **`virtual void setRequestStatus(DtRequestStatus rStatus)`**
  Set/Get the request status
- **`virtual DtRequestStatus requestStatus() const`**
- **`virtual const char* name() const`**

Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtActionResponseInteraction. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtActionResponseInteractionCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtActionResponseInteractionCb cb, void* usr)**

- **DtEntityIdentifier myOrigEnt**

- **DtEntityIdentifier myReceiveEnt**

- **DtU32 myRequestId**

- **DtRequestStatus myRequestStatus**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hactreqinter.h:

class DtActionRequestInteraction: public DtInteractionWithData

---

Inheritance:

DtActionRequestInteraction - DtInteractionWithData - DtInteractionWithEncDec - DtInteraction

---

Public Methods

	DtActionRequestInteraction () *default constructor*
virtual	**~DtActionRequestInteraction** () *destructor*
	DtActionRequestInteraction (const DtActionRequestInteraction& orig) *copy constructor*
DtActionRequestInteraction&	**operator=** (const DtActionRequestInteraction& orig) *assignment operator*
virtual void	**printData** () const *Print the interaction's data*
virtual void	**setSenderId** (const DtEntityIdentifier& id) *`Set/Get sender ID*
virtual const DtEntityIdentifier&	**senderId** () const
virtual void	**setReceiverId** (const DtEntityIdentifier& id) *Set/Get receiver ID*
virtual const DtEntityIdentifier&	**receiverId** () const
virtual void	**setRequestId** (int id) *Set/Get request ID*
virtual int	**requestId** () const
virtual void	**setActionId** (DtActionId id) *Set/Get action ID*
virtual DtActionId	**actionId** () const
virtual const char*	**name** () const *Name of the interaction (Not nec*
static DtInteraction*	**create** () *Create a DtActionRequestInteraction*

- static void **addCallback** (DtExerciseConn* conn, DtActionRequestInteractionCb cb, void* usr)
  Add (register)/Remove (deregister) function to be called when interaction occurs
- static void **removeCallback** (DtExerciseConn* conn, DtActionRequestInteractionCb cb, void* usr)

Protected Fields

- DtEntityIdentifier **myOrigEnt**
- DtEntityIdentifier **myReceiveEnt**
- int **myRequestId**
- DtActionId **myActionId**

Inherited from DtInteractionWithData:

Public Methods

- virtual int **numFixedFields**() const
- virtual int **numVarFields**() const
- virtual DtDatumParam **varDatumId**(int index) const
- virtual DtDatumParam **fixedDatumId**(int index) const
- virtual DtDatumParam **datumId**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBits**(DtDatumRecordType fixedOrVar, int index) const
- virtual int **datumSizeBytes**(DtDatumRecordType fixedOrVar, int index) const
- virtual void **getIndex**(DtDatumParam param, DtDatumRecordType* fixedOrVar, int* index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat32 **datumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual long **datumValInt32**(DtDatumRecordType fixedOrVar, int index, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumRecordType fixedOrVar, int index) const
- virtual DtFloat32 **datumValFloat32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual DtFloat64 **datumValFloat64**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual unsigned long **datumValUnsigned32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const

- virtual long **datumValInt32**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual const char* **datumValByteArray**(DtDatumParam param, DtDatumRetCode* retcode = NULL) const
- virtual void **initDataSection**(const DtDatumParam* fixedParams, const DtVarDatumSpec* specList)
- virtual void **setNumFixedFields**(int newNum)
- virtual void **setNumVarFields**(int newNum)
- virtual void **setDatumParam**(DtDatumRecordType fixedOrVar, int index, DtDatumParam param, DtDatumRetCode* retcode = NULL)
- virtual void **setVarDataBytes**(int index, int length, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumRecordType fixedOrVar, int index, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumRecordType fixedOrVar, int index, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumRecordType fixedOrVar, int index, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumRecordType fixedOrVar, int index, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumRecordType fixedOrVar, int index, const char* val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat64**(DtDatumParam param, DtFloat64 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValFloat32**(DtDatumParam param, DtFloat32 val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValInt32**(DtDatumParam param, long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValUnsigned32**(DtDatumParam param, unsigned long val, DtDatumRetCode* retcode = NULL)
- virtual void **setDatumValByteArray**(DtDatumParam param, const char* val, DtDatumRetCode* retcode = NULL)
- virtual DtDatumRecordSet* **datumRecSet**()
- virtual const DtDatumRecordSet* **datumRecSet**() const

Public

-

Protected Fields

- DtDatumRecordSet **myRecSet**

Inherited from DtInteractionWithEncDec:

Public Methods

- virtual void **setExConn**(DtExerciseConn* exConn, DtInterClassDesc* classDesc = NULL)
- virtual const RTI::ParameterHandleValuePairSet& **phvps**() const
- virtual void **setFromPhvps**(const RTI::ParameterHandleValuePairSet& pvList)
- virtual void **useThese**(DtInteractionEncoder* enc, DtInteractionDecoder* dec, DtInteractionRep* data)
- virtual DtInteractionEncoder* **encoder**() const
- virtual DtInteractionDecoder* **decoder**() const
- inline DtInteractionDecoder* **decoder**() const
- inline DtInteractionEncoder* **encoder**() const

Protected Fields

- DtInteractionDecoder* **myDecoder**
- DtInteractionEncoder* **myEncoder**
- DtInteractionDecoder* **myCreatedDecoder**
- DtInteractionEncoder* **myCreatedEncoder**
- DtInteractionRep* **myData**

Protected Methods

- virtual DtInteractionDecoder* **createDecoder**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **createEncoder**(DtExerciseConn* exConn) const
- virtual DtInteractionDecoder* **decoderToUse**(DtExerciseConn* exConn) const
- virtual DtInteractionEncoder* **encoderToUse**(DtExerciseConn* exConn) const

---

Inherited from DtInteraction:

Public Methods

- virtual RTI::InteractionClassHandle **interactionClassHandle**() const
- virtual char* **interactionClassName**() const
- virtual int **numParameters**() const

- virtual DtBoolean **neededByFederation**() const
- virtual void **print**() const
- virtual void **printHeader**() const
- virtual void **printParams**(DtBoolean withHex) const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual DtInterClassDesc* **classDesc**() const
- virtual void **setTag**(const DtString& tag)
- virtual const DtString& **tag**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- static void **setTagIsAsciiTime**(DtBoolean val)
- static DtBoolean **tagIsAsciiTime**()
- static int **maxInteractionClasses**()
- inline DtInterClassDesc* **classDesc**() const
- inline DtExerciseConn* **exerciseConn**() const
- inline RTI::InteractionClassHandle **interactionClassHandle**() const
- inline char* **interactionClassName**() const
- inline int **maxInteractionClasses**()
- inline int **numParameters**() const
- inline DtInteraction* **self**() const

Protected Fields

- DtString **myTag**
- DtInterClassDesc* **myClassDesc**
- DtExerciseConn* **myExConn**
- DtTime **myTime**
- DtTimeStampType **myTimeType**
- static DtBoolean **theTagIsAsciiTime**
- static RTI::ParameterHandleValuePairSet* **thePvList**

- static int **thePvListMaxSize**
- static const int **theMaxInteractionClasses**

Protected Methods

- virtual char* **interactionClassToUse**(DtExerciseConn* exConn) const
- virtual RTI::ParameterHandleValuePairSet* **privPvList**() const
- DtInteraction* **self**() const

Documentation

- **`DtActionRequestInteraction()`**
  default constructor
- **`virtual ~DtActionRequestInteraction()`**
  destructor
- **`DtActionRequestInteraction(const DtActionRequestInteraction& orig)`**
  copy constructor
- **`DtActionRequestInteraction& operator=( const DtActionRequestInteraction& orig)`**
  assignment operator
- **`virtual void printData() const`**
  Print the interaction's data
- **`virtual void setSenderId(const DtEntityIdentifier& id)`**
  `Set/Get sender ID
- **`virtual const DtEntityIdentifier& senderId() const`**
- **`virtual void setReceiverId(const DtEntityIdentifier& id)`**
  Set/Get receiver ID
- **`virtual const DtEntityIdentifier& receiverId() const`**
- **`virtual void setRequestId(int id)`**
  Set/Get request ID
- **`virtual int requestId() const`**
- **`virtual void setActionId(DtActionId id)`**
  Set/Get action ID
- **`virtual DtActionId actionId() const`**
- **`virtual const char* name() const`**

Name of the interaction (Not nec. the ClassName from the FOM).

- **static DtInteraction* create()**

  Create a DtActionRequestInteraction. Caller is reponsible for deletion.

- **static void addCallback(DtExerciseConn* conn, DtActionRequestInteractionCb cb, void* usr)**

  Add (register)/Remove (deregister) function to be called when interaction occurs

- **static void removeCallback(DtExerciseConn* conn, DtActionRequestInteractionCb cb, void* usr)**

- **DtEntityIdentifier myOrigEnt**

- **DtEntityIdentifier myReceiveEnt**

- **int myRequestId**

- **DtActionId myActionId**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtDetonatorFuze

- **DtFuzeOther**
- **DtFuzeContact**
- **DtFuzeContactInstant**
- **DtFuzeContactDelayed**
- **DtFuzeTimed**
- **DtFuzeProximity**
- **DtFuzeCommand**
- **DtFuzeAltitude**
- **DtFuzeDepth**
- **DtFuzeAcoustic**

Documentation

- `DtFuzeOther`
- `DtFuzeContact`
- `DtFuzeContactInstant`
- `DtFuzeContactDelayed`
- `DtFuzeTimed`
- `DtFuzeProximity`
- `DtFuzeCommand`
- `DtFuzeAltitude`
- `DtFuzeDepth`
- `DtFuzeAcoustic`

In file disenums.h:

enum DtWarheadType

- **DtWarheadOther**
- **DtWarheadHighExplosive**
- **DtWarheadHighExplosivePlastic**
- **DtWarheadHighExplosiveIncendiary**
- **DtWarheadHighExplosiveFragmentation**
- **DtWarheadHighExplosiveAntiTank**
- **DtWarheadHighExplosiveBomblets**
- **DtWarheadHighExplosiveShapedCharge**
- **DtWarheadSmoke**
- **DtWarheadIllumination**
- **DtWarheadPractice**
- **DtWarheadKinetic**
- **DtWarheadUnused**
- **DtWarheadNuclear**
- **DtWarheadChemicalGeneral**
- **DtWarheadChemicalBlisterAgent**
- **DtWarheadChemicalBloodAgent**
- **DtWarheadChemicalNerveAgent**
- **DtWarheadBiologicalGeneral**

Documentation

- `DtWarheadOther`
- `DtWarheadHighExplosive`
- `DtWarheadHighExplosivePlastic`
- `DtWarheadHighExplosiveIncendiary`

- **DtWarheadHighExplosiveFragmentation**
- **DtWarheadHighExplosiveAntiTank**
- **DtWarheadHighExplosiveBomblets**
- **DtWarheadHighExplosiveShapedCharge**
- **DtWarheadSmoke**
- **DtWarheadIllumination**
- **DtWarheadPractice**
- **DtWarheadKinetic**
- **DtWarheadUnused**
- **DtWarheadNuclear**
- **DtWarheadChemicalGeneral**
- **DtWarheadChemicalBlisterAgent**
- **DtWarheadChemicalBloodAgent**
- **DtWarheadChemicalNerveAgent**
- **DtWarheadBiologicalGeneral**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtDetonationResult

- **DtDetResOther**
- **DtDetResEntityImpact**
- **DtDetResEntityProximate**
- **DtDetResGroundImpact**
- **DtDetResGroundProximate**
- **DtDetResDetonation**
- **DtDetResNone**
- **DtDetResHE_HitSmall**
- **DtDetResHE_HitMedium**
- **DtDetResHE_HitLarge**
- **DtDetResArmorPiercingHit**
- **DtDetResDirtBlastSmall**
- **DtDetResDirtBlastMedium**
- **DtDetResDirtBlastLarge**
- **DtDetResWaterBlastSmall**
- **DtDetResWaterBlastMedium**
- **DtDetResWaterBlastLarge**
- **DtDetResAirHit**
- **DtDetResBuildingHitSmall**
- **DtDetResBuildingHitMedium**
- **DtDetResBuildingHitLarge**
- **DtDetResMineClearingLineCharge**
- **DtDetResEnvironmentObjectImpact**
- **DtDetResEnvironmentObjectProximate**
- **DtDetResWaterImpact**
- **DtDetResAirBurst**

Documentation

- **DtDetResOther**
- **DtDetResEntityImpact**
- **DtDetResEntityProximate**
- **DtDetResGroundImpact**
- **DtDetResGroundProximate**
- **DtDetResDetonation**
- **DtDetResNone**
- **DtDetResHE_HitSmall**
- **DtDetResHE_HitMedium**
- **DtDetResHE_HitLarge**
- **DtDetResArmorPiercingHit**
- **DtDetResDirtBlastSmall**
- **DtDetResDirtBlastMedium**
- **DtDetResDirtBlastLarge**
- **DtDetResWaterBlastSmall**
- **DtDetResWaterBlastMedium**
- **DtDetResWaterBlastLarge**
- **DtDetResAirHit**
- **DtDetResBuildingHitSmall**
- **DtDetResBuildingHitMedium**
- **DtDetResBuildingHitLarge**
- **DtDetResMineClearingLineCharge**
- **DtDetResEnvironmentObjectImpact**
- **DtDetResEnvironmentObjectProximate**
- **DtDetResWaterImpact**
- **DtDetResAirBurst**

Alphabetic index *Hierarchy of classes*

In file nettimestamp.h:

class DtNetTimeStamp

class DtNetTimeStamp: Instances of DtNetTimeStamp are network representations of a DIS time stamp

---

Public Methods

	DtNetTimeStamp () *default constructor - creates an instance with a time of zero and timestamp type of relative*
	DtNetTimeStamp (DtTime time, DtTimeStampType type = DtTimeStampRelative) *Creates an instance with the time set to time (specified in seconds) modulo an hour, and the timestamp type set to type*
	~DtNetTimeStamp () *destructor*
	DtNetTimeStamp (const DtNetTimeStamp& orig) *copy constructor*
DtNetTimeStamp&	**operator=** (const DtNetTimeStamp& orig) *assignment operator*
	operator DtU32 () const *Cast to a DtU32 representing the whole timestamp Useful mainly for backwards compatibility for applications that used to cast DtNetTimeStamp to DtU32 before accessing it*
DtTime	**time** () const *Get the time contained in the timestamp (in seconds)*
void	**setTime** (DtTime time) *Get the time contained in the timestamp to time % an hour*
void	**setType** (DtTimeStampType type) *Set/Get timestamp type*
DtTimeStampType	**type** () const

Protected Fields

DtNetU32	**myRep**
static double	**theHourFracPerHour**
static double	**theHourPerHourFrac**

Documentation

class DtNetTimeStamp:

Instances of DtNetTimeStamp are network representations of a DIS time stamp. This class has NO VIRTUAL FUNCTIONS, since it is a DtNetType - whose exact byte layout is relied on by PDU classes, etc.

- **`DtNetTimeStamp()`**

  default constructor - creates an instance with a time of zero and timestamp type of relative

- **`DtNetTimeStamp(DtTime time, DtTimeStampType type = DtTimeStampRelative)`**

  Creates an instance with the time set to time (specified in seconds) modulo an hour, and the timestamp type set to type

- **`~DtNetTimeStamp()`**

  destructor

- **`DtNetTimeStamp(const DtNetTimeStamp& orig)`**

  copy constructor

- **`DtNetTimeStamp& operator=(const DtNetTimeStamp& orig)`**

  assignment operator

- **`operator DtU32 () const`**

  Cast to a DtU32 representing the whole timestamp Useful mainly for backwards compatibility for applications that used to cast DtNetTimeStamp to DtU32 before accessing it

- **`DtTime time() const`**

  Get the time contained in the timestamp (in seconds). The value returned will be between 0.0 and 3600.0 seconds.

- **`void setTime(DtTime time)`**

  Get the time contained in the timestamp to time % an hour. time is specified in seconds.

- **`void setType(DtTimeStampType type)`**

  Set/Get timestamp type

- **`DtTimeStampType type() const`**

- **`DtNetU32 myRep`**

- **`static double theHourFracPerHour`**

- **`static double theHourPerHourFrac`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtCreateObjectResult

DtCreateObjectResultOther

Documentation

`DtCreateObjectResultOther`

In file disenums.h:

enum DtCollisionType

- **DtInelastic**
- **DtElastic**
- **DtCollisionTypeOther**

Documentation

- `DtInelastic`
- `DtElastic`
- `DtCollisionTypeOther`

In file disenums.h:

enum DtAttributeChangeResult

DtAttributeChangeResultOther

Documentation

`DtAttributeChangeResultOther`

In file disenums.h:

enum DtResponseFlag

- **DtNoWay**
- **DtSureThing**

Documentation

- `DtNoWay`
- `DtSureThing`

In file disenums.h:

enum DtActionId

- **DtActionOther**
- **DtActionLocalStorage**
- **DtActionOutOfAmmo**
- **DtActionKilledInAction**
- **DtActionDamage**
- **DtActionMobilityDisabled**
- **DtActionFireDisabled**
- **DtActionOutOfFuel**
- **DtActionRecallCheckpointData**
- **DtActionRecallInitialParameters**
- **DtActionInitiateTetherLead**
- **DtActionInitiateTetherFollow**
- **DtActionUntether**
- **DtActionInitiateServiceStationResupply**
- **DtActionInitiateTailgateResupply**
- **DtActionInitiateHitchLead**
- **DtActionInitiateHitchFollow**
- **DtActionUnhitch**
- **DtActionMount**
- **DtActionDismount**
- **DtActionStartDRC**
- **DtActionStopDRC**
- **DtActionDataQuery**
- **DtActionStatusRequest**
- **DtActionSendObjectStateData**
- **DtActionReconstitute**

- **DtActionLockSiteConfiguration**
- **DtActionUnlockSiteConfiguration**
- **DtActionUpdateSiteConfiguration**
- **DtActionQuerySiteConfiguration**
- **DtActionTetheringInformation**
- **DtActionMountIntent**
- **DtActionOther2**
- **DtActionAcceptSubscription**
- **DtActionUnsubscribe**
- **DtActionTeleportEntity**
- **DtActionChangeAggregateState**
- **DtActionRequestStartPDU**
- **DtActionWakeUpGetReadyForInitialization**
- **DtActionInitializeInternalParameters**
- **DtActionSendPlanData**
- **DtActionSynchronizeInternalClocks**
- **DtActionRun**
- **DtActionSaveInternalParameters**
- **DtActionSimulateMalfunction**
- **DtActionJoinExercise**
- **DtActionResignExercise**
- **DtActionTimeAdvance**

Documentation

- `DtActionOther`
- `DtActionLocalStorage`
- `DtActionOutOfAmmo`
- `DtActionKilledInAction`
- `DtActionDamage`
- `DtActionMobilityDisabled`
- `DtActionFireDisabled`

- **DtActionOutOfFuel**
- **DtActionRecallCheckpointData**
- **DtActionRecallInitialParameters**
- **DtActionInitiateTetherLead**
- **DtActionInitiateTetherFollow**
- **DtActionUntether**
- **DtActionInitiateServiceStationResupply**
- **DtActionInitiateTailgateResupply**
- **DtActionInitiateHitchLead**
- **DtActionInitiateHitchFollow**
- **DtActionUnhitch**
- **DtActionMount**
- **DtActionDismount**
- **DtActionStartDRC**
- **DtActionStopDRC**
- **DtActionDataQuery**
- **DtActionStatusRequest**
- **DtActionSendObjectStateData**
- **DtActionReconstitute**
- **DtActionLockSiteConfiguration**
- **DtActionUnlockSiteConfiguration**
- **DtActionUpdateSiteConfiguration**
- **DtActionQuerySiteConfiguration**
- **DtActionTetheringInformation**
- **DtActionMountIntent**
- **DtActionOther2**
- **DtActionAcceptSubscription**
- **DtActionUnsubscribe**
- **DtActionTeleportEntity**
- **DtActionChangeAggregateState**
- **DtActionRequestStartPDU**
- **DtActionWakeUpGetReadyForInitialization**

- **DtActionInitializeInternalParameters**
- **DtActionSendPlanData**
- **DtActionSynchronizeInternalClocks**
- **DtActionRun**
- **DtActionSaveInternalParameters**
- **DtActionSimulateMalfunction**
- **DtActionJoinExercise**
- **DtActionResignExercise**
- **DtActionTimeAdvance**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtAcknowledgeFlag

- **DtAckCreate**
- **DtAckRemove**
- **DtAckStart**
- **DtAckStop**

Documentation

- `DtAckCreate`
- `DtAckRemove`
- `DtAckStart`
- `DtAckStop`

In file pdu.h:

class DtPdu: public DtBaseMsg

Inheritance:

DtPdu - DtBaseMsg

Public Methods

virtual	**~DtPdu** () *Destructor*
	DtPdu (const DtPdu &pdu, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *Copy constructor*
DtPdu&	**operator=** (const DtPdu &pdu) *Assignment operator*
virtual void*	**packet** () const *Older function that returns a pointer to the network representation*
virtual int	**status** () const *Get PDU status*
virtual void	**setVersion** (int vers) *Set/Get protocol version from PDU header*
virtual int	**protocolVersion** () const
virtual void	**setExerciseId** (int id) *Set/Get exerciseID in PDU header*
virtual int	**exerciseId** () const
virtual DtPduKind	**kind** () const *Returns PDU kind from PDU header*
virtual DtProtocolFamily	**protocolFamily** () const *Returns protocol family from PDU header*
virtual void	**setTimeStamp** (DtTime time, DtTimeStampType type = DtTimeStampRelative) *Set the PDU header's time stamp to reflect time and type*
virtual DtTime	**timeStamp** () const *Return the time (in seconds past an hour) encoded in the PDU header's time stamp*
virtual DtTimeStampType	**timeStampType** () const *Return the type of the time stamp, as indicated in PDU header*
virtual DtTime	**guessTimeValid** (DtTime referenceTime) const *Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past that hour*
virtual void	**print** () const *Print the contents of the PDU by calling printHeader and printData*

virtual void	**printHeader** () const *Print the PDU's header*
virtual void	**printData** () const *Print the PDU's data (not including header)*
inline int	**exerciseId** () const
inline DtPduKind	**kind** () const
inline DtNetPduHeader*	**netHeader** () const
inline void*	**packet** () const
inline DtProtocolFamily	**protocolFamily** () const
inline int	**protocolVersion** () const
inline void	**setExerciseId** (int id)
inline void	**setStatus** (int okStatus)
inline void	**setTimeStamp** (DtTime time, DtTimeStampType relOrAbs)
inline void	**setVersion** (int vers)
inline int	**status** () const
inline DtTime	**timeStamp** () const
inline DtTimeStampType	**timeStampType** () const

Protected Fields

int **myOKStatus**

Protected Methods

	DtPdu () *Constructor*
virtual void	**setStatus** (int okStatus) *Sets the PDU's status to okStatus*
virtual DtNetPduHeader*	**netHeader** () const *Returns a pointer to the PDU's header*
virtual void	**initPdu** (int size, DtBufferPtr buffer) *Must be called after default ctor before PDU is completely constructed*
virtual void	**initPdu** (const void* initial, DtBufferPtr buffer) *Called by from-net-rep ctor*
virtual DtPduKind	**internalGetPduKind** () const *Returns the PDU kind associated with the derived PDU class*
virtual DtProtocolFamily	**kindToFamily** (DtPduKind kind) const *Maps PDU kind to ProtocolFamily*
virtual void	**sizeChanged** (int size) *Called by changeSize*
virtual void	**clearData** () *zeros out all bytes after the PDU header*
virtual void	**initHeader** (DtPduKind kind, int size) *Initializes PDU header*

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- **`virtual ~DtPdu()`**
  Destructor
- **`DtPdu(const DtPdu &pdu, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**
  Copy constructor
- **`DtPdu& operator=(const DtPdu &pdu)`**

Assignment operator

- **virtual void* packet() const**

  Older function that returns a pointer to the network representation. The newer DtBaseMsg::netRep should generally be used instead.

- **virtual int status() const**

  Get PDU status. DtSTATUS_OK means it's good. Other values indicate a bad PDU. Return codes are derived PDU-dependent.

- **virtual void setVersion(int vers)**

  Set/Get protocol version from PDU header

- **virtual int protocolVersion() const**

- **virtual void setExerciseId(int id)**

  Set/Get exerciseID in PDU header

- **virtual int exerciseId() const**

- **virtual DtPduKind kind() const**

  Returns PDU kind from PDU header

- **virtual DtProtocolFamily protocolFamily() const**

  Returns protocol family from PDU header

- **virtual void setTimeStamp(DtTime time, DtTimeStampType type = DtTimeStampRelative)**

  Set the PDU header's time stamp to reflect time and type. Time is in seconds, but only the time modulo an hour actually gets into the time stamp.

- **virtual DtTime timeStamp() const**

  Return the time (in seconds past an hour) encoded in the PDU header's time stamp

- **virtual DtTimeStampType timeStampType() const**

  Return the type of the time stamp, as indicated in PDU header

- **virtual DtTime guessTimeValid(DtTime referenceTime) const**

  Since timestamp only represents number of seconds past the hour, guessTimeValid chooses the hour that would make the timestamp closest to reference time, and returns a time that is timestamp seconds past that hour

- **virtual void print() const**

  Print the contents of the PDU by calling printHeader and printData

- **virtual void printHeader() const**

  Print the PDU's header

- **virtual void printData() const**

  Print the PDU's data (not including header). Base DtPdu version just prints hex.

- **DtPdu()**

  Constructor. PDU is not fully constructed until initPdu is called.

- **virtual void setStatus(int okStatus)**

  Sets the PDU's status to okStatus

- **virtual DtNetPduHeader* netHeader() const**

Returns a pointer to the PDU's header

- **virtual void initPdu(int size, DtBufferPtr buffer)**

  Must be called after default ctor before PDU is completely constructed. Fills in PDU header, and initializes DtBaseMsg.

- **virtual void initPdu(const void* initial, DtBufferPtr buffer)**

  Called by from-net-rep ctor. Initializes DtBaseMsg.

- **virtual DtPduKind internalGetPduKind() const**

  Returns the PDU kind associated with the derived PDU class

- **virtual DtProtocolFamily kindToFamily(DtPduKind kind) const**

  Maps PDU kind to ProtocolFamily

- **virtual void sizeChanged(int size)**

  Called by changeSize. Sets the size in the header to size.

- **virtual void clearData()**

  zeros out all bytes after the PDU header

- **virtual void initHeader(DtPduKind kind, int size)**

  Initializes PDU header. Called by initPdu.

- **int myOKStatus**
- **inline int exerciseId() const**
- **inline DtPduKind kind() const**
- **inline DtNetPduHeader* netHeader() const**
- **inline void* packet() const**
- **inline DtProtocolFamily protocolFamily() const**
- **inline int protocolVersion() const**
- **inline void setExerciseId(int id)**
- **inline void setStatus(int okStatus)**
- **inline void setTimeStamp(DtTime time, DtTimeStampType relOrAbs)**
- **inline void setVersion(int vers)**
- **inline int status() const**
- **inline DtTime timeStamp() const**
- **inline DtTimeStampType timeStampType() const**

Direct child classes:

DtViewControlPdu
DtLgrControlPdu

Alphabetic index *Hierarchy of classes*

In file basemsg.h:

class DtBaseMsg

Inheritance:

DtBaseMsg

Public Methods

virtual	**~DtBaseMsg** () *Destructor*
	DtBaseMsg (const DtBaseMsg& msg, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *Copy constructor*
DtBaseMsg&	**operator=** (const DtBaseMsg& msg) *Assignment*
virtual const char*	**netRep** () const *Get network representation*
virtual int	**length** () const *Get size of message*
inline const char*	**netRep** () const

Protected Fields

void*	**myNetMsg**
void*	**myInternalBuff**
int	**myIntBuffSize**
int	**myMsgSize**

Protected Methods

	DtBaseMsg () *Constructor Object is not fully constructed until initMsg is called*
void	**initMsg** (int size, DtBufferPtr bufferPtr)
void	**initMsg** (const void *initial, int size, DtBufferPtr bufferPtr)
virtual void	**changeSize** (int size) *Changes the size of the message*

- virtual void **insertBytes** (int offset, int numBytes)
  Inserts numBytes zeroed-out bytes at the indicated offset into the net rep
- virtual void **insertBytes** (void *address, int numBytes)
  Inserts numBytes zeroed-out bytes at the indicated address
- virtual void **deleteBytes** (int offset, int numBytes)
  Deletes numBytes bytes at the indicated offset in the net rep
- virtual void **deleteBytes** (void *address, int numBytes)
  Deletes numBytes bytes starting at the indicated address
- virtual void **clearAll** ()
  zeros out all bytes in the network representation
- virtual void* **newInternalBuffer** (int size)
  Sets myInternalBuff to point to a malloc'ed buffer of size bytes
- virtual void **sizeChanged** (int size)
  Called by changeSize

Documentation

- **`virtual ~DtBaseMsg()`**

  Destructor

- **`DtBaseMsg(const DtBaseMsg& msg, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  Copy constructor

- **`DtBaseMsg& operator=(const DtBaseMsg& msg)`**

  Assignment

- **`virtual const char* netRep() const`**

  Get network representation

- **`virtual int length() const`**

  Get size of message

- **`DtBaseMsg()`**

  Constructor Object is not fully constructed until initMsg is called

- **`void initMsg(int size, DtBufferPtr bufferPtr)`**

- **`void initMsg(const void *initial, int size, DtBufferPtr bufferPtr)`**

- **`virtual void changeSize(int size)`**

  Changes the size of the message. If we are using an external buffer, assumes that the full buffer size is big enough to accommodate the new msg size. If using an internal buffer, the call may result in allocating a new internal buffer, and copying data from the old to the new buffer.

- **`virtual void insertBytes(int offset, int numBytes)`**

  Inserts numBytes zeroed-out bytes at the indicated offset into the net rep. Uses changeSize to increase the size of the buffer.

- **`virtual void insertBytes(void *address, int numBytes)`**

  Inserts numBytes zeroed-out bytes at the indicated address. Assumes that address falls within the net rep. Uses changeSize to increase the size of the buffer.

- **`virtual void deleteBytes(int offset, int numBytes)`**

  Deletes numBytes bytes at the indicated offset in the net rep. Uses change size to decrease the size of the buffer.

- **`virtual void deleteBytes(void *address, int numBytes)`**

  Deletes numBytes bytes starting at the indicated address. Assumes that address falls within the net rep. Uses changeSize to decrease the size of the buffer.

- **`virtual void clearAll()`**

  zeros out all bytes in the network representation

- **`virtual void* newInternalBuffer(int size)`**

  Sets myInternalBuff to point to a malloc'ed buffer of size bytes. Sets myIntBuffSize to size. Returns myInternalBuff. Size must be a positive number.

- **`virtual void sizeChanged(int size)`**

  Called by changeSize. Derived versions can provide definitions that set the size field in network rep.

- **`void* myNetMsg`**

- **`void* myInternalBuff`**

- **`int myIntBuffSize`**

- **`int myMsgSize`**

- **`inline const char* netRep() const`**

Direct child classes:

DtPdu
DtDatumRecordSet

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtPduKind

- **DtOtherPduKind**
- **DtEntityStatePduKind**
- **DtFirePduKind**
- **DtDetonationPduKind**
- **DtCollisionPduKind**
- **DtServiceRequestPduKind**
- **DtResupplyOfferPduKind**
- **DtResupplyReceivedPduKind**
- **DtResupplyCancelPduKind**
- **DtRepairCompletePduKind**
- **DtRepairResponsePduKind**
- **DtCreateEntityPduKind**
- **DtRemoveEntityPduKind**
- **DtStartResumePduKind**
- **DtStopFreezePduKind**
- **DtAcknowledgePduKind**
- **DtActionRequestPduKind**
- **DtActionResponsePduKind**
- **DtDataQueryPduKind**
- **DtSetDataPduKind**
- **DtDataPduKind**
- **DtEventReportPduKind**
- **DtCommentPduKind**
- **DtMessagePduKind**
- **DtElectromagneticEmissionPduKind**
- **DtDesignatorPduKind**

- DtTransmitterPduKind
- DtSignalPduKind
- DtReceiverPduKind
- DtIffAtcNavaidsPduKind
- DtUnderwaterAcousticPduKind
- DtSuppEmisEntStatePduKind
- DtIntercomSignalPduKind
- DtIntercomControlPduKind
- DtAggregateStatePduKind
- DtIsGroupOfPduKind
- DtTransferControlRequestPduKind
- DtIsPartOfPduKind
- DtMinefieldStatePduKind
- DtMinefieldQueryPduKind
- DtMinefieldDataPduKind
- DtMinefieldResponseNackPduKind
- DtEnvironmentalProcessPduKind
- DtGriddedDataPduKind
- DtPointObjectStatePduKind
- DtLinearObjectStatePduKind
- DtArealObjectStatePduKind
- DtLeTspiPduKind
- DtLeAppearancePduKind
- DtLeArticulatedPartsPduKind
- DtLeFirePduKind
- DtLeDetonationPduKind
- DtCreateEntityRPduKind
- DtRemoveEntityRPduKind
- DtStartResumeRPduKind
- DtStopFreezeRPduKind
- DtAcknowledgeRPduKind
- DtActionRequestRPduKind

- **DtActionResponseRPduKind**
- **DtDataQueryRPduKind**
- **DtSetDataRPduKind**
- **DtDataRPduKind**
- **DtEventReptRPduKind**
- **DtCommentRPduKind**
- **DtRecordQueryPduKind**
- **DtSetRecordPduKind**
- **DtRecordPduKind**
- **DtCollisionElasticPduKind**
- **DtEntityStateUpdatePduKind**

Documentation

- `DtOtherPduKind`
- `DtEntityStatePduKind`
- `DtFirePduKind`
- `DtDetonationPduKind`
- `DtCollisionPduKind`
- `DtServiceRequestPduKind`
- `DtResupplyOfferPduKind`
- `DtResupplyReceivedPduKind`
- `DtResupplyCancelPduKind`
- `DtRepairCompletePduKind`
- `DtRepairResponsePduKind`
- `DtCreateEntityPduKind`
- `DtRemoveEntityPduKind`
- `DtStartResumePduKind`
- `DtStopFreezePduKind`
- `DtAcknowledgePduKind`
- `DtActionRequestPduKind`
- `DtActionResponsePduKind`

- **DtDataQueryPduKind**
- **DtSetDataPduKind**
- **DtDataPduKind**
- **DtEventReportPduKind**
- **DtCommentPduKind**
- **DtMessagePduKind**
- **DtElectromagneticEmissionPduKind**
- **DtDesignatorPduKind**
- **DtTransmitterPduKind**
- **DtSignalPduKind**
- **DtReceiverPduKind**
- **DtIffAtcNavaidsPduKind**
- **DtUnderwaterAcousticPduKind**
- **DtSuppEmisEntStatePduKind**
- **DtIntercomSignalPduKind**
- **DtIntercomControlPduKind**
- **DtAggregateStatePduKind**
- **DtIsGroupOfPduKind**
- **DtTransferControlRequestPduKind**
- **DtIsPartOfPduKind**
- **DtMinefieldStatePduKind**
- **DtMinefieldQueryPduKind**
- **DtMinefieldDataPduKind**
- **DtMinefieldResponseNackPduKind**
- **DtEnvironmentalProcessPduKind**
- **DtGriddedDataPduKind**
- **DtPointObjectStatePduKind**
- **DtLinearObjectStatePduKind**
- **DtArealObjectStatePduKind**
- **DtLeTspiPduKind**
- **DtLeAppearancePduKind**
- **DtLeArticulatedPartsPduKind**

- **DtLeFirePduKind**
- **DtLeDetonationPduKind**
- **DtCreateEntityRPduKind**
- **DtRemoveEntityRPduKind**
- **DtStartResumeRPduKind**
- **DtStopFreezeRPduKind**
- **DtAcknowledgeRPduKind**
- **DtActionRequestRPduKind**
- **DtActionResponseRPduKind**
- **DtDataQueryRPduKind**
- **DtSetDataRPduKind**
- **DtDataRPduKind**
- **DtEventReptRPduKind**
- **DtCommentRPduKind**
- **DtRecordQueryPduKind**
- **DtSetRecordPduKind**
- **DtRecordPduKind**
- **DtCollisionElasticPduKind**
- **DtEntityStateUpdatePduKind**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtProtocolFamily

- **DtFamilyOther**
- **DtFamilyEntityInteraction**
- **DtFamilyWarfare**
- **DtFamilyLogistics**
- **DtFamilyRadioCommunication**
- **DtFamilySimulationManagement**
- **DtFamilyDistributedEmissionRegeneration**
- **DtFamilyEntityManagement**
- **DtFamilyMinefield**
- **DtFamilySyntheticEnvironment**
- **DtFamilySimulationManagementWithReliability**
- **DtFamilyLiveEntity**
- **DtFamilyNonRealTime**

Documentation

- `DtFamilyOther`
- `DtFamilyEntityInteraction`
- `DtFamilyWarfare`
- `DtFamilyLogistics`
- `DtFamilyRadioCommunication`
- `DtFamilySimulationManagement`
- `DtFamilyDistributedEmissionRegeneration`
- `DtFamilyEntityManagement`
- `DtFamilyMinefield`
- `DtFamilySyntheticEnvironment`

- **DtFamilySimulationManagementWithReliability**
- **DtFamilyLiveEntity**
- **DtFamilyNonRealTime**

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtUnknownVCPdu: public DtViewControlPdu

Inheritance:

DtUnknownVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

- **DtUnknownVCPdu** (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)
- virtual void **setFromString**(const char *)
- virtual char* **string**() const
- void **printData**() const

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)

- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- **`DtUnknownVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtTrackVCPdu: public DtViewControlPdu

Inheritance:

DtTrackVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtTrackVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtTrackVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
DtConstVector	**worldLocation** () const
const DtGlobalObjectDesignator&	**trackId** () const
void	**setWorldLocation** (DtConstVector loc)
void	**setTrackId** (const DtGlobalObjectDesignator& id)
void	**setFromString** (const char *str)
char*	**string** () const
void	**printData** () const

Protected Fields

DtVector	**theVec**
DtGlobalObjectDesignator	**theId**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const

- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const

- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- **`DtTrackVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**
- **`DtTrackVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**
- **`DtConstVector worldLocation() const`**
- **`const DtGlobalObjectDesignator& trackId() const`**
- **`void setWorldLocation(DtConstVector loc)`**
- **`void setTrackId(const DtGlobalObjectDesignator& id)`**
- **`void setFromString(const char *str)`**

- `char* string() const`
- `void printData() const`
- `DtVector theVec`
- `DtGlobalObjectDesignator theId`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtTetherVCPdu: public DtViewControlPdu

Inheritance:

DtTetherVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtTetherVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtTetherVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
DtConstVector	**topoUpOffset** () const
const DtTaitBryan&	**orientation** () const
const DtGlobalObjectDesignator&	**attachId** () const
void	**setTopoUpOffset** (DtConstVector offset)
void	**setOrientation** (const DtTaitBryan &ori)
void	**setAttachId** (const DtGlobalObjectDesignator& id)
void	**setFromString** (const char *str)
char*	**string** () const
void	**printData** () const

Protected Fields

DtVector	**theVec**
DtTaitBryan	**theAngles**
DtGlobalObjectDesignator	**theId**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const

- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- `DtTetherVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtTetherVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtConstVector topoUpOffset() const`
- `const DtTaitBryan& orientation() const`
- `const DtGlobalObjectDesignator& attachId() const`

- `void setTopoUpOffset(DtConstVector offset)`
- `void setOrientation(const DtTaitBryan &ori)`
- `void setAttachId(const DtGlobalObjectDesignator& id)`
- `void setFromString(const char *str)`
- `char* string() const`
- `void printData() const`
- `DtVector theVec`
- `DtTaitBryan theAngles`
- `DtGlobalObjectDesignator theId`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtTetherTrackVCPdu: public DtViewControlPdu

Inheritance:

DtTetherTrackVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtTetherTrackVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtTetherTrackVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
DtConstVector	**offset** () const
const DtGlobalObjectDesignator&	**attachId** () const
const DtGlobalObjectDesignator&	**trackId** () const
void	**setOffset** (DtConstVector loc)
void	**setAttachId** (const DtGlobalObjectDesignator& id)
void	**setTrackId** (const DtGlobalObjectDesignator& id)
void	**setFromString** (const char *str)
char*	**string** () const
void	**printData** () const

Protected Fields

DtVector	**theVec**
DtGlobalObjectDesignator	**theId**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const

- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()

- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- `DtTetherTrackVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtTetherTrackVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtConstVector offset() const`

- `const DtGlobalObjectDesignator& attachId() const`
- `const DtGlobalObjectDesignator& trackId() const`
- `void setOffset(DtConstVector loc)`
- `void setAttachId(const DtGlobalObjectDesignator& id)`
- `void setTrackId(const DtGlobalObjectDesignator& id)`
- `void setFromString(const char *str)`
- `char* string() const`
- `void printData() const`
- `DtVector theVec`
- `DtGlobalObjectDesignator theId`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtOrbitVCPdu: public DtViewControlPdu

Inheritance:

DtOrbitVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtOrbitVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtOrbitVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
float	**radius** () const
float	**azimuthRate** () const
float	**heading** () const
float	**pitch** () const
float	**headingOffset** () const
float	**pitchOffset** () const
float	**roll** () const
const DtGlobalObjectDesignator&	**attachId** () const
void	**setRadius** (float r)
void	**setAzimuthRate** (float a)
void	**setHeading** (float h)
void	**setPitch** (float p)
void	**setHeadingOffset** (float o)
void	**setPitchOffset** (float p)
void	**setRoll** (float r)
void	**setAttachId** (const DtGlobalObjectDesignator& id)
void	**setFromString** (const char *str)
char*	**string** () const
void	**printData** () const

Protected Fields

- DtGlobalObjectDesignator **theId**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)

- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)

- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- `DtOrbitVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtOrbitVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `float radius() const`
- `float azimuthRate() const`
- `float heading() const`
- `float pitch() const`
- `float headingOffset() const`
- `float pitchOffset() const`
- `float roll() const`
- `const DtGlobalObjectDesignator& attachId() const`
- `void setRadius(float r)`
- `void setAzimuthRate(float a)`
- `void setHeading(float h)`
- `void setPitch(float p)`
- `void setHeadingOffset(float o)`
- `void setPitchOffset(float p)`
- `void setRoll(float r)`
- `void setAttachId(const DtGlobalObjectDesignator& id)`
- `void setFromString(const char *str)`
- `char* string() const`
- `void printData() const`
- `DtGlobalObjectDesignator theId`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtModelessVCPdu: public DtViewControlPdu

Inheritance:

DtModelessVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtModelessVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtModelessVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
const DtGlobalObjectDesignator&	**primary** () const
const DtGlobalObjectDesignator&	**secondary** () const
int	**numEntitiesToRemove** () const
int	**numEntitiesToAdd** () const
const DtGlobalObjectDesignator&	**entityToRemove** (int n) const
const DtGlobalObjectDesignator&	**entityToAdd** (int n) const
void	**setPrimary** (const DtGlobalObjectDesignator& id)
void	**setSecondary** (const DtGlobalObjectDesignator& id)
void	**setNumEntitiesToRemove** (int n)
void	**setNumEntitiesToAdd** (int n)
void	**setEntityToRemove** (int n, const DtGlobalObjectDesignator& id)
void	**setEntityToAdd** (int n, const DtGlobalObjectDesignator& id)
void	**printData** () const

Protected Fields

DtGlobalObjectDesignator	**theId**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)
- virtual void **setFromString**(const char *)
- virtual char* **string**() const

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const

- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const

- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- `DtModelessVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtModelessVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer =`

DtUSE_INTERNAL_BUFFER)

- const DtGlobalObjectDesignator& primary() const
- const DtGlobalObjectDesignator& secondary() const
- int numEntitiesToRemove() const
- int numEntitiesToAdd() const
- const DtGlobalObjectDesignator& entityToRemove(int n) const
- const DtGlobalObjectDesignator& entityToAdd(int n) const
- void setPrimary(const DtGlobalObjectDesignator& id)
- void setSecondary(const DtGlobalObjectDesignator& id)
- void setNumEntitiesToRemove(int n)
- void setNumEntitiesToAdd(int n)
- void setEntityToRemove(int n, const DtGlobalObjectDesignator& id)
- void setEntityToAdd(int n, const DtGlobalObjectDesignator& id)
- void printData() const
- DtGlobalObjectDesignator theId

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtMimicVCPdu: public DtViewControlPdu

Inheritance:

DtMimicVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtMimicVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtMimicVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
DtConstVector	**offset** () const
const DtTaitBryan&	**bodyOrientation** () const
const DtGlobalObjectDesignator&	**attachId** () const
void	**setOffset** (DtConstVector offset)
void	**setBodyOrientation** (const DtTaitBryan &ori)
void	**setAttachId** (const DtGlobalObjectDesignator& id)
void	**setFromString** (const char *str)
char*	**string** () const
void	**printData** () const

Protected Fields

DtGlobalObjectDesignator	**theId**
DtVector	**theVec**
DtTaitBryan	**theAngles**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const

- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- `DtMimicVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtMimicVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtConstVector offset() const`
- `const DtTaitBryan& bodyOrientation() const`
- `const DtGlobalObjectDesignator& attachId() const`

- `void setOffset(DtConstVector offset)`
- `void setBodyOrientation(const DtTaitBryan &ori)`
- `void setAttachId(const DtGlobalObjectDesignator& id)`
- `void setFromString(const char *str)`
- `char* string() const`
- `void printData() const`
- `DtGlobalObjectDesignator theId`
- `DtVector theVec`
- `DtTaitBryan theAngles`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtMimicTrackVCPdu: public DtViewControlPdu

Inheritance:

DtMimicTrackVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtMimicTrackVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtMimicTrackVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
DtConstVector	**offset** () const
const DtGlobalObjectDesignator&	**attachId** () const
const DtGlobalObjectDesignator&	**trackId** () const
void	**setOffset** (DtConstVector loc)
void	**setAttachId** (const DtGlobalObjectDesignator& id)
void	**setTrackId** (const DtGlobalObjectDesignator& id)
void	**setFromString** (const char *str)
char*	**string** () const
void	**printData** () const

Protected Fields

DtVector	**theVec**
DtGlobalObjectDesignator	**theId**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const

- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()

- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- `DtMimicTrackVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtMimicTrackVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtConstVector offset() const`

- `const DtGlobalObjectDesignator& attachId() const`
- `const DtGlobalObjectDesignator& trackId() const`
- `void setOffset(DtConstVector loc)`
- `void setAttachId(const DtGlobalObjectDesignator& id)`
- `void setTrackId(const DtGlobalObjectDesignator& id)`
- `void setFromString(const char *str)`
- `char* string() const`
- `void printData() const`
- `DtVector theVec`
- `DtGlobalObjectDesignator theId`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtGroupVCPdu: public DtViewControlPdu

Inheritance:

DtGroupVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtGroupVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtGroupVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
float	**radialOffset** () const
float	**heading** () const
float	**pitch** () const
float	**headingOffset** () const
float	**pitchOffset** () const
int	**groupCmd** () const
int	**numEntitiesToRemove** () const
int	**numEntitiesToAdd** () const
const DtGlobalObjectDesignator&	**entityToRemove** (int n) const
const DtGlobalObjectDesignator&	**entityToAdd** (int n) const
void	**setRadialOffset** (float o)
void	**setHeading** (float h)
void	**setPitch** (float p)
void	**setHeadingOffset** (float o)
void	**setPitchOffset** (float o)
void	**setGroupCmd** (int c)
void	**setNumEntitiesToRemove** (int n)
void	**setNumEntitiesToAdd** (int n)
void	**setEntityToRemove** (int n, const DtGlobalObjectDesignator& id)
void	**setEntityToAdd** (int n, const DtGlobalObjectDesignator& id)

- void **setFromString** (const char *str)
- char* **string** () const
- void **printData** () const

Protected Fields

- DtGlobalObjectDesignator **theId**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const

- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)

- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- **DtGroupVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**
- **DtGroupVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**
- **float radialOffset() const**
- **float heading() const**
- **float pitch() const**
- **float headingOffset() const**
- **float pitchOffset() const**
- **int groupCmd() const**
- **int numEntitiesToRemove() const**
- **int numEntitiesToAdd() const**
- **const DtGlobalObjectDesignator& entityToRemove(int n) const**
- **const DtGlobalObjectDesignator& entityToAdd(int n) const**
- **void setRadialOffset(float o)**
- **void setHeading(float h)**
- **void setPitch(float p)**
- **void setHeadingOffset(float o)**
- **void setPitchOffset(float o)**
- **void setGroupCmd(int c)**
- **void setNumEntitiesToRemove(int n)**
- **void setNumEntitiesToAdd(int n)**
- **void setEntityToRemove(int n, const DtGlobalObjectDesignator& id)**
- **void setEntityToAdd(int n, const DtGlobalObjectDesignator& id)**
- **void setFromString(const char *str)**
- **char* string() const**
- **void printData() const**
- **DtGlobalObjectDesignator theId**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtCompassVCPdu: public DtViewControlPdu

Inheritance:

DtCompassVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtCompassVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtCompassVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
float	**radius** () const
float	**bodyHeading** () const
float	**pitch** () const
float	**headingOffset** () const
float	**pitchOffset** () const
float	**roll** () const
const DtGlobalObjectDesignator&	**attachId** () const
void	**setRadius** (float r)
void	**setBodyHeading** (float h)
void	**setPitch** (float p)
void	**setHeadingOffset** (float o)
void	**setPitchOffset** (float p)
void	**setRoll** (float r)
void	**setAttachId** (const DtGlobalObjectDesignator& id)
void	**setFromString** (const char *str)
char*	**string** () const
void	**printData** () const

Protected Fields

- DtGlobalObjectDesignator **theId**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const
- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)

- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)

- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- `DtCompassVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtCompassVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `float radius() const`
- `float bodyHeading() const`
- `float pitch() const`
- `float headingOffset() const`
- `float pitchOffset() const`
- `float roll() const`
- `const DtGlobalObjectDesignator& attachId() const`
- `void setRadius(float r)`
- `void setBodyHeading(float h)`
- `void setPitch(float p)`
- `void setHeadingOffset(float o)`
- `void setPitchOffset(float p)`
- `void setRoll(float r)`
- `void setAttachId(const DtGlobalObjectDesignator& id)`
- `void setFromString(const char *str)`
- `char* string() const`
- `void printData() const`
- `DtGlobalObjectDesignator theId`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vcpdu.h:

class DtAbsoluteVCPdu: public DtViewControlPdu

Inheritance:

DtAbsoluteVCPdu - DtViewControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtAbsoluteVCPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
	DtAbsoluteVCPdu (const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
DtConstVector	**worldLocation** () const
const DtTaitBryan&	**orientation** () const
void	**setWorldLocation** (DtConstVector loc)
void	**setOrientation** (const DtTaitBryan &ori)
void	**setFromString** (const char *str)
char*	**string** () const
void	**printData** () const

Protected Fields

DtVector	**theVec**
DtTaitBryan	**theAngles**

Inherited from DtViewControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- const DtGlobalObjectDesignator& **viewerId**() const
- const DtGlobalObjectDesignator& **senderId**() const

- DtViewerCommand **viewerCommand**() const
- unsigned long **ignoreFlags**() const
- void **setViewerId**(const DtGlobalObjectDesignator& id)
- void **setSenderId**(const DtGlobalObjectDesignator& id)
- void **setIgnoreFlags**(unsigned long flags)

Protected Fields

- DtGlobalObjectDesignator **retId**

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setViewerCommand**(DtViewerCommand com)

Inherited from DtPdu:

Public Methods

- DtPdu& **operator=**(const DtPdu &pdu)
- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const

- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

- `DtAbsoluteVCPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtAbsoluteVCPdu(const DtNetViewControlPdu *initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`
- `DtConstVector worldLocation() const`
- `const DtTaitBryan& orientation() const`
- `void setWorldLocation(DtConstVector loc)`
- `void setOrientation(const DtTaitBryan &ori)`
- `void setFromString(const char *str)`

- `char* string() const`
- `void printData() const`
- `DtVector theVec`
- `DtTaitBryan theAngles`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file rprdecfact.h:

class DtRprStateDecoderFactory: public DtStateDecoderFactory

Inheritance:

DtRprStateDecoderFactory - DtStateDecoderFactory

Public Methods

	DtRprStateDecoderFactory (DtExerciseConn* exConn) *constructor*
virtual	**~DtRprStateDecoderFactory** () *destructor*
	DtRprStateDecoderFactory (const DtRprStateDecoderFactory& orig) *copy constructor*
DtRprStateDecoderFactory&	**operator=** (const DtRprStateDecoderFactory& orig) *assignment operator*

Inherited from DtStateDecoderFactory:

Public Methods

- virtual DtHlaStateDecoder* **createDecoder**(RTI::ObjectClassHandle hand)
- virtual DtHlaStateDecoder* **addDecoder**(RTI::ObjectClassHandle hand, DtHlaStateDecoder* decoder)
- virtual void **addAttributeDecoder**(RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand, DtHlaStateDecoder::DtAttributeDecoder decoder, DtBoolean applyToSubclasses = DtTrue)
- void **addAttributeDecoder**(char* className, RTI::AttributeHandle attrHand, DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue)
- void **addAttributeDecoder**(RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue)
- void **addAttributeDecoder**(char* className, char* attrName, DtHlaStateDecoder::DtAttributeDecoder func, DtBoolean applyToSubclasses = DtTrue)
- virtual DtHlaStateDecoder* **decoder**(RTI::ObjectClassHandle hand, DtBoolean defaultToBase = DtFalse)

Protected Fields

- DtHlaStateDecoder** **myProtos**
- DtExerciseConn* **myExConn**

Documentation

- **`DtRprStateDecoderFactory(DtExerciseConn* exConn)`**
  constructor
- **`virtual ~DtRprStateDecoderFactory()`**
  destructor
- **`DtRprStateDecoderFactory(const DtRprStateDecoderFactory& orig)`**
  copy constructor
- **`DtRprStateDecoderFactory& operator=(const DtRprStateDecoderFactory& orig)`**
  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file rprencfact.h:

class DtRprStateEncoderFactory: public DtStateEncoderFactory

Inheritance:

DtRprStateEncoderFactory - DtStateEncoderFactory

Public Methods

	DtRprStateEncoderFactory (DtExerciseConn* exConn) *constructor Adds RPR FOM specific encoders*
virtual	**~DtRprStateEncoderFactory** () *destructor*
	DtRprStateEncoderFactory (const DtRprStateEncoderFactory& orig) *copy constructor*
DtRprStateEncoderFactory&	**operator=** (const DtRprStateEncoderFactory& orig) *assignment operator*

Inherited from DtStateEncoderFactory:

Public Methods

virtual DtHlaStateEncoder* **createEncoder**(RTI::ObjectClassHandle hand)

virtual DtHlaStateEncoder* **addEncoder**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder)

virtual void **addAttributeEncoder**(RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeEncoder encoder, DtBoolean applyToSubclasses = DtTrue)

virtual void **addAttributeEncoder**(char* className, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue)

virtual void **addAttributeEncoder**(RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue)

virtual void **addAttributeEncoder**(char* className, char* attrName, DtHlaStateEncoder::DtAttributeEncoder func, DtBoolean applyToSubclasses = DtTrue)

virtual void **addAttributeChecker**(RTI::ObjectClassHandle classHand, RTI::AttributeHandle attrHand,

DtHlaStateEncoder::DtAttributeChecker encoder, DtBoolean applyToSubclasses = DtTrue)

- virtual void **addAttributeChecker**(char* className, RTI::AttributeHandle attrHand, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)

- virtual void **addAttributeChecker**(RTI::ObjectClassHandle classHand, char* attrName, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)

- virtual void **addAttributeChecker**(char* className, char* attrName, DtHlaStateEncoder::DtAttributeChecker func, DtBoolean applyToSubclasses = DtTrue)

- virtual DtHlaStateEncoder* **encoder**(RTI::ObjectClassHandle hand, DtBoolean defaultToBase = DtFalse)

Protected Fields

- DtHlaStateEncoder** **myProtos**

- DtExerciseConn* **myExConn**

Documentation

- **`DtRprStateEncoderFactory(DtExerciseConn* exConn)`**

  constructor Adds RPR FOM specific encoders

- **`virtual ~DtRprStateEncoderFactory()`**

  destructor

- **`DtRprStateEncoderFactory(const DtRprStateEncoderFactory& orig)`**

  copy constructor

- **`DtRprStateEncoderFactory& operator=( const DtRprStateEncoderFactory& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file rpridecfact.h:

class DtRprInteractionDecoderFactory: public DtInteractionDecoderFactory

---

Inheritance:

DtRprInteractionDecoderFactory - DtInteractionDecoderFactory

Public Methods

	DtRprInteractionDecoderFactory (DtExerciseConn* exConn) *constructor*
virtual	**~DtRprInteractionDecoderFactory** () *destructor*
	DtRprInteractionDecoderFactory (const DtRprInteractionDecoderFactory& orig) *copy constructor*
DtRprInteractionDecoderFactory&	**operator=** (const DtRprInteractionDecoderFactory& orig) *assignment operator*

Inherited from DtInteractionDecoderFactory:

Public Methods

- virtual DtInteractionDecoder* **createDecoder**(RTI::InteractionClassHandle hand)
- virtual DtInteractionDecoder* **addDecoder**(RTI::InteractionClassHandle hand, DtInteractionDecoder* decoder)
- virtual void **addParameterDecoder**(RTI::InteractionClassHandle classHand, RTI::ParameterHandle paramHand, DtInteractionDecoder::DtParameterDecoder decoder, DtBoolean applyToSubclasses = DtTrue)
- void **addParameterDecoder**(char* className, RTI::ParameterHandle paramHand, DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue)
- void **addParameterDecoder**(RTI::InteractionClassHandle classHand, char* paramName, DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue)
- void **addParameterDecoder**(char* className, char* paramName, DtInteractionDecoder::DtParameterDecoder func, DtBoolean applyToSubclasses = DtTrue)
- virtual DtInteractionDecoder* **decoder**(RTI::InteractionClassHandle hand)

Protected Fields

- DtInteractionDecoder** **myProtos**
- DtExerciseConn* **myExConn**

Documentation

- **`DtRprInteractionDecoderFactory(DtExerciseConn* exConn)`**
  constructor
- **`virtual ~DtRprInteractionDecoderFactory()`**
  destructor
- **`DtRprInteractionDecoderFactory( const DtRprInteractionDecoderFactory& orig)`**
  copy constructor
- **`DtRprInteractionDecoderFactory& operator=( const DtRprInteractionDecoderFactory& orig)`**
  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file rpriencfact.h:

class DtRprInteractionEncoderFactory: public DtInteractionEncoderFactory

---

Inheritance:

DtRprInteractionEncoderFactory - DtInteractionEncoderFactory

Public Methods

	DtRprInteractionEncoderFactory (DtExerciseConn* exConn) *constructor Adds RPR FOM specific encoders*
virtual	**~DtRprInteractionEncoderFactory** () *destructor*
	DtRprInteractionEncoderFactory (const DtRprInteractionEncoderFactory& orig) *copy constructor*
DtRprInteractionEncoderFactory&	**operator=** (const DtRprInteractionEncoderFactory& orig) *assignment operator*

Inherited from DtInteractionEncoderFactory:

Public Methods

virtual DtInteractionEncoder* **createEncoder**(RTI::InteractionClassHandle hand)

virtual DtInteractionEncoder* **addEncoder**(RTI::InteractionClassHandle hand, DtInteractionEncoder* encoder)

virtual void **addParameterEncoder**(RTI::InteractionClassHandle classHand, RTI::ParameterHandle paramHand, DtInteractionEncoder::DtParameterEncoder encoder, DtBoolean applyToSubclasses = DtTrue)

void **addParameterEncoder**(char* className, RTI::ParameterHandle paramHand, DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue)

void **addParameterEncoder**(RTI::InteractionClassHandle classHand, char* paramName, DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue)

void **addParameterEncoder**(char* className, char* paramName, DtInteractionEncoder::DtParameterEncoder func, DtBoolean applyToSubclasses = DtTrue)

virtual DtInteractionEncoder* **encoder**(RTI::InteractionClassHandle hand)

Protected Fields

- DtInteractionEncoder** **myProtos**
- DtExerciseConn* **myExConn**

Documentation

- **`DtRprInteractionEncoderFactory(DtExerciseConn* exConn)`**
  constructor Adds RPR FOM specific encoders
- **`virtual ~DtRprInteractionEncoderFactory()`**
  destructor
- **`DtRprInteractionEncoderFactory( const DtRprInteractionEncoderFactory& orig)`**
  copy constructor
- **`DtRprInteractionEncoderFactory& operator=( const DtRprInteractionEncoderFactory& orig)`**
  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file basehashkey.h:

class DtBaseHashKey

class DtBaseHashKey: DtBaseHashKey is an abstract base class for hashkeys

---

Inheritance:

DtBaseHashKey

Public Methods

	DtBaseHashKey () *default constructor*
virtual	**~DtBaseHashKey** () *destructor*
	DtBaseHashKey (const DtBaseHashKey& orig) *copy constructor*
DtBaseHashKey&	**operator=** (const DtBaseHashKey& orig) *assignment operator*
virtual DtBoolean	**operator==** (const DtBaseHashKey& other) const *equality operator*
virtual DtBoolean	**operator!=** (const DtBaseHashKey& other) const *equality operator*
virtual DtU32	**hashValue** () const *get the hashed value for this key*
virtual DtBaseHashKey*	**clone** () const *make a copy of this hash key*
virtual const DtString&	**stringRep** () const *get the string representation of this key*

Protected Fields

DtBoolean	**myHashValueIsComputed**
DtU32	**myHashValue**

Protected Methods

virtual DtU32	**computeHashValue** () *compute the hashed value for this key*

Documentation

class DtBaseHashKey: DtBaseHashKey is an abstract base class for hashkeys

- **`DtBaseHashKey()`**
  default constructor
- **`virtual ~DtBaseHashKey()`**
  destructor
- **`DtBaseHashKey(const DtBaseHashKey& orig)`**
  copy constructor
- **`DtBaseHashKey& operator=(const DtBaseHashKey& orig)`**
  assignment operator
- **`virtual DtBoolean operator==(const DtBaseHashKey& other) const`**
  equality operator
- **`virtual DtBoolean operator!=(const DtBaseHashKey& other) const`**
  equality operator
- **`virtual DtU32 hashValue() const`**
  get the hashed value for this key
- **`virtual DtBaseHashKey* clone() const`**
  make a copy of this hash key
- **`virtual const DtString& stringRep() const`**
  get the string representation of this key
- **`virtual DtU32 computeHashValue()`**
  compute the hashed value for this key
- **`DtBoolean myHashValueIsComputed`**
- **`DtU32 myHashValue`**

Direct child classes:

DtSystemHashKey
DtStringHashKey
DtObjectIdHashKey
DtIntHashKey

Alphabetic index *Hierarchy of classes*

In file rprfommap.h:

class DtRprFomMapper: public DtEmptyFomMapper

Inheritance:

DtRprFomMapper - DtEmptyFomMapper - DtFomMapper

Public Methods

	DtRprFomMapper (double vers = 0.8) *default constructor*
virtual	**~DtRprFomMapper** () *destructor*
virtual void	**init** (DtExerciseConn* conn) *Virtual function override*
static DtFomMapper*	**create** (void* usr) *Creator function - returns new instance of DtRprFomMapper*

Protected Methods

	DtRprFomMapper (const DtRprFomMapper& orig) *copy constructor*
DtRprFomMapper&	**operator=** (const DtRprFomMapper& orig) *assignment operator*
virtual void	**initInterClasses** () *Set myInterClasses to contain default mappings*
virtual void	**initObjClasses** () *Set myObjClassesClasses to contain default mappings*
virtual void	**initObjChoosers** () *Setup the object class choosers*
virtual void	**initInterChoosers** () *Setup the interaction class choosers*
static char*	**dfltEntityClassChooser** (const char* vrlinkClassName, DtExerciseConn* exConn, void* usr) *The default entity class chooser function*
static char*	**entityClassChooserPre08** (const char* vrlinkClassName, DtExerciseConn* exConn, void* usr) *The entity class chooser function for older RPR FOM versions*
static char*	**dfltEmitterBeamClassChooser** (const char* vrlinkClassName, DtExerciseConn* exConn, void* usr) *The default emitter beam class chooser function*

- static char* **emitterBeamClassChooserPre08** (const char* vrlinkClassName, DtExerciseConn* exConn, void* usr)
  The default emitter beam class chooser function
- static char* **dfltSignalClassChooser** (DtExerciseConn* exConn, const DtInteraction& inter)
  The default signal class chooser function

Inherited from DtEmptyFomMapper:

Public Methods

- DtStateDecoderFactory* **dfltStateDecoderFactory**()
- DtStateEncoderFactory* **dfltStateEncoderFactory**()
- DtInteractionDecoderFactory* **dfltInteractionDecoderFactory**()
- DtInteractionEncoderFactory* **dfltInteractionEncoderFactory**()
- DtHlaObjectFactory* **dfltHlaObjectFactory**()
- DtInteractionFactory* **dfltInteractionFactory**()

Protected Fields

- DtStateDecoderFactory* **myDfltDecFact**
- DtStateEncoderFactory* **myDfltEncFact**
- DtInteractionDecoderFactory* **myDfltInterDecFact**
- DtInteractionEncoderFactory* **myDfltInterEncFact**
- DtHlaObjectFactory* **myDfltObjFact**
- DtInteractionFactory* **myDfltInterFact**

Inherited from DtFomMapper:

Public Methods

- typedef char***(*DtInteractionClassChooser)**(DtExerciseConn* exConn, const DtInteraction& inter)
- typedef char***(*DtObjectClassChooser)**(const char* vrlinkClassName, DtExerciseConn* conn, void* usr)
- virtual DtStateDecoderFactory* **setStateDecoderFactory**(DtStateDecoderFactory* fact)
- virtual DtStateDecoderFactory* **stateDecoderFactory**()
- virtual DtStateEncoderFactory* **setStateEncoderFactory**(DtStateEncoderFactory* fact)

- virtual DtStateEncoderFactory* **stateEncoderFactory**()
- virtual DtInteractionDecoderFactory* **setInteractionDecoderFactory**(DtInteractionDecoderFactory* fact)
- virtual DtInteractionDecoderFactory* **interactionDecoderFactory**()
- virtual DtInteractionEncoderFactory* **setInteractionEncoderFactory**(DtInteractionEncoderFactory* fact)
- virtual DtInteractionEncoderFactory* **interactionEncoderFactory**()
- virtual DtHlaObjectFactory* **setHlaObjectFactory**(DtHlaObjectFactory* fact)
- virtual DtHlaObjectFactory* **hlaObjectFactory**()
- virtual DtInteractionFactory* **setInteractionFactory**(DtInteractionFactory* fact)
- virtual DtInteractionFactory* **interactionFactory**()
- virtual void **setObjectClassToChoose**(const char* vrlinkClassName, const char* fomClassName)
- virtual void **setObjectClassChooser**(const char* vrlinkClassName, DtObjectClassChooser chooser)
- virtual char* **chooseObjectClass**(const char* vrlinkClassName, void* usr = NULL) const
- virtual void **setInteractionClassToChoose**(const char* vrlinkClassName, const char* fomClassName)
- virtual void **setInteractionClassChooser**(const char* vrlinkClassName, DtInteractionClassChooser chooser)
- virtual char* **chooseInteractionClass**(const DtInteraction& inter) const
- virtual void **setInteractionClasses**(const char* vrlinkClassName, const DtList& classes)
- virtual DtList* **interactionClasses**(const char* vrlinkClassName)
- virtual void **setInteractionClass**(const char* vrlinkClassName, const char* fomClassName)
- virtual void **setObjectClassNames**(const char* vrlinkClassName, const DtList& classes)
- virtual DtList* **objectClassNames**(const char* vrlinkClassName)
- virtual DtList* **objectClassHandles**(const char* vrlinkClassName)
- virtual void **setObjectClass**(const char* vrlinkClassName, const char* fomClassName)
- virtual void **setVersion**(double vers)
- virtual double **version**() const

Protected Fields

- DtExerciseConn* **myExConn**
- DtStateDecoderFactory* **myDecFact**
- DtStateEncoderFactory* **myEncFact**
- DtInteractionDecoderFactory* **myInterDecFact**

- DtInteractionEncoderFactory* **myInterEncFact**
- DtHlaObjectFactory* **myObjFact**
- DtInteractionFactory* **myInterFact**
- DtHashlist **myIntMapList**
- DtHashlist **myObjMapList**
- DtHashlist **myObjChooserList**
- DtHashlist **mySimpleObjChooserList**
- DtHashlist **myInterChooserList**
- DtHashlist **mySimpleInterChooserList**
- DtList **myObjHandleList**
- DtList **myEmptyList**
- double **myVersion**

Protected Methods

- virtual DtObjectClassChooser **objectClassChooser**(const char* vrlinkClassName) const
- virtual DtInteractionClassChooser **interactionClassChooser**(const char* vrlinkClassName) const

---

Documentation

- **`DtRprFomMapper(double vers = 0.8)`**

  default constructor

- **`virtual ~DtRprFomMapper()`**

  destructor

- **`virtual void init(DtExerciseConn* conn)`**

  Virtual function override. init actually initializes most data. It is called from DtExerciseConn constructor.

- **`static DtFomMapper* create(void* usr)`**

  Creator function - returns new instance of DtRprFomMapper. usr is cast to a pointer to double. If the pointer is NULL, or the value pointed to is 0.0, the default version is used in the DtRprFomMapper constructor. Otherwise the value is passed to the DtRprFomMapper constructor as the version number.

- **`DtRprFomMapper(const DtRprFomMapper& orig)`**

  copy constructor

- **`DtRprFomMapper& operator=(const DtRprFomMapper& orig)`**

  assignment operator

virtual void initInterClasses()

Set myInterClasses to contain default mappings

virtual void initObjClasses()

Set myObjClassesClasses to contain default mappings

virtual void initObjChoosers()

Setup the object class choosers

virtual void initInterChoosers()

Setup the interaction class choosers

static char* dfltEntityClassChooser(const char* vrlinkClassName, DtExerciseConn* exConn, void* usr)

The default entity class chooser function

static char* entityClassChooserPre08(const char* vrlinkClassName, DtExerciseConn* exConn, void* usr)

The entity class chooser function for older RPR FOM versions

static char* dfltEmitterBeamClassChooser(const char* vrlinkClassName, DtExerciseConn* exConn, void* usr)

The default emitter beam class chooser function

static char* emitterBeamClassChooserPre08(const char* vrlinkClassName, DtExerciseConn* exConn, void* usr)

The default emitter beam class chooser function

static char* dfltSignalClassChooser(DtExerciseConn* exConn, const DtInteraction& inter)

The default signal class chooser function

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hoststructs.h:

class DtObjectId

Public Methods

	DtObjectId ()
	DtObjectId (RTI::ObjectHandle id)
	DtObjectId (const char* str)
DtObjectId&	**operator=** (const DtObjectId& orig)
	DtObjectId (const DtObjectId& orig)
	DtObjectId (DtNetObjectId netId) *On big endian machines, this is the same as DtObjectId(RTI::ObjectHandle id); defined above*
char*	**string** () const
static const DtObjectId&	**nullId** ()

Protected Fields

RTI::ObjectHandle	**myId**

Documentation

DtObjectId()

DtObjectId(RTI::ObjectHandle id)

DtObjectId(const char* str)

DtObjectId& operator=(const DtObjectId& orig)

DtObjectId(const DtObjectId& orig)

DtObjectId(DtNetObjectId netId)

On big endian machines, this is the same as DtObjectId(RTI::ObjectHandle id); defined above

char* string() const

static const DtObjectId& nullId()

RTI::ObjectHandle myId

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file genericdec.h:

class DtGenericObjectDecoder: public DtHlaStateDecoder

The DtGenericObjectDecoder class:

Inheritance:

DtGenericObjectDecoder - DtHlaStateDecoder

Public Methods

	DtGenericObjectDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *default constructor*
virtual	**~DtGenericObjectDecoder** () *destructor*
virtual void	**decodeAttribute** (DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const *Virtual function override*

Inherited from DtHlaStateDecoder:

Public Methods

- DtHlaStateDecoder& **operator=**(const DtHlaStateDecoder& orig)
- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const

- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Documentation

The DtGenericObjectDecoder class:

- **`DtGenericObjectDecoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
  default constructor
- **`virtual ~DtGenericObjectDecoder()`**
  destructor
- **`virtual void decodeAttribute( DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const`**
  Virtual function override

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file baseentdec.h:

class DtBaseEntityDecoder: public DtHlaStateDecoder

Inheritance:

DtBaseEntityDecoder - DtHlaStateDecoder

Public Methods

	DtBaseEntityDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtBaseEntityDecoder** ()
	DtBaseEntityDecoder (const DtBaseEntityDecoder& orig)
DtBaseEntityDecoder&	**operator=** (const DtBaseEntityDecoder& orig)
virtual DtHlaStateDecoder*	**clone** () const
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

void	**addMyDecoders** ()
virtual void	**decode** (const DtStateMsg& msg, DtStateRepository* stateRep) const
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (AccelerationVector)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (AngularVelocityVector)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (DRAlgorithm)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (EntityType)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (EntityID)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (FederateID)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (IsFrozen)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (Orientation)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (Position)
	DtDECLARE_BASE_ENTITY_ATTR_DECODER (VelocityVector)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void(***DtAttributeDecoder**)(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- DtBaseEntityDecoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)
- virtual ~DtBaseEntityDecoder()
- DtBaseEntityDecoder(const DtBaseEntityDecoder& orig)
- DtBaseEntityDecoder& operator=(const DtBaseEntityDecoder& orig)
- virtual DtHlaStateDecoder* clone() const
- static DtHlaStateDecoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)
- void addMyDecoders()
- virtual void decode(const DtStateMsg& msg, DtStateRepository* stateRep) const
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(AccelerationVector)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(AngularVelocityVector)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(DRAlgorithm)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(EntityType)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(EntityID)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(FederateID)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(IsFrozen)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(Orientation)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(Position)
- DtDECLARE_BASE_ENTITY_ATTR_DECODER(VelocityVector)

Direct child classes:

DtPhysicalEntityDecoder
DtEnvironmentEntityDecoder
DtAggregateEntityDecoder

Alphabetic index *Hierarchy of classes*

In file baseemittrdec.h:

class DtBaseEmitterBeamDecoder: public DtHlaStateDecoder

Inheritance:

DtBaseEmitterBeamDecoder - DtHlaStateDecoder

Public Methods

	DtBaseEmitterBeamDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtBaseEmitterBeamDecoder** ()
	DtBaseEmitterBeamDecoder (const DtBaseEmitterBeamDecoder& orig)
DtBaseEmitterBeamDecoder&	**operator=** (const DtBaseEmitterBeamDecoder& orig)
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- void **addMyDecoders** ()
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (AzimuthCenter)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (AzimuthSweep)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (BeamFunction)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (BeamID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (BeamParameterIndex)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (ElevationCenter)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (ElevationSweep)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (EmittingSystemID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (ERP)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (EntityID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (EventID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (Frequency)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (FrequencyRange)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (PRF)

- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (PulseWidth)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER** (SweepSynch)

Inherited from DtHlaStateDecoder:

Public Methods

- DtHlaStateDecoder& **operator=**(const DtHlaStateDecoder& orig)
- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- **DtBaseEmitterBeamDecoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtBaseEmitterBeamDecoder()**
- **DtBaseEmitterBeamDecoder(const DtBaseEmitterBeamDecoder& orig)**
- **DtBaseEmitterBeamDecoder& operator= (const DtBaseEmitterBeamDecoder& orig)**
- **static DtHlaStateDecoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **void addMyDecoders()**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(AzimuthCenter)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(AzimuthSweep)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(BeamFunction)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(BeamID)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(BeamParameterIndex)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(ElevationCenter)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(ElevationSweep)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(EmittingSystemID)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(ERP)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(EntityID)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(EventID)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(Frequency)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(FrequencyRange)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(PRF)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(PulseWidth)**
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER(SweepSynch)**

Direct child classes:

DtTrackJamBeamDecoder
DtRadarBeamDecoder
DtJammerBeamDecoder

In file baseembeddec.h:

class DtBaseEmbeddedSystemDecoder: public DtHlaStateDecoder

Inheritance:

DtBaseEmbeddedSystemDecoder - DtHlaStateDecoder

Public Methods

	DtBaseEmbeddedSystemDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtBaseEmbeddedSystemDecoder** ()
	DtBaseEmbeddedSystemDecoder (const DtBaseEmbeddedSystemDecoder& orig)
DtBaseEmbeddedSystemDecoder&	**operator=** (const DtBaseEmbeddedSystemDecoder& orig)
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- void **addMyDecoders** ()
- **DtDECLARE_EMBEDSYS_ATTR_DECODER** (EntityID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER** (HostObjectID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER** (RelativePosition)

Inherited from DtHlaStateDecoder:

Public Methods

- DtHlaStateDecoder& **operator=**(const DtHlaStateDecoder& orig)
- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)

- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- **`DtBaseEmbeddedSystemDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
- **`virtual ~DtBaseEmbeddedSystemDecoder()`**
- **`DtBaseEmbeddedSystemDecoder(const DtBaseEmbeddedSystemDecoder& orig)`**

- `DtBaseEmbeddedSystemDecoder& operator= (const DtBaseEmbeddedSystemDecoder& orig)`
- `static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `void addMyDecoders()`
- `DtDECLARE_EMBEDSYS_ATTR_DECODER(EntityID)`
- `DtDECLARE_EMBEDSYS_ATTR_DECODER(HostObjectID)`
- `DtDECLARE_EMBEDSYS_ATTR_DECODER(RelativePosition)`

Direct child classes:

DtRadioTransmitterDecoder
DtRadioReceiverDecoder
DtEmitterSystemDecoder
DtDesignatorDecoder

Alphabetic index *Hierarchy of classes*

In file genericenc.h:

class DtGenericObjectEncoder: public DtHlaStateEncoder

The DtGenericObjectEncoder class - the encoder used by DtGenericObjectPublisher

---

Inheritance:

DtGenericObjectEncoder - DtHlaStateEncoder

Public Methods

	DtGenericObjectEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *default constructor*
virtual	**~DtGenericObjectEncoder** () *destructor*
virtual void	**encode** (const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg) *Encode necessary values from the state repository*

Protected Methods

virtual DtBoolean	**valueDiffers** (const DtGenericStateRepository& stateRep, const DtGenericStateRepository& asSeenByRemote, RTI::AttributeHandle handle)

Inherited from DtHlaStateEncoder:

Public Methods

DtHlaStateEncoder& **operator=**(const DtHlaStateEncoder& orig)

typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)

typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)

DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)

- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

The DtGenericObjectEncoder class - the encoder used by DtGenericObjectPublisher

- **DtGenericObjectEncoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**

  default constructor

- **virtual ~DtGenericObjectEncoder()**

  destructor

- **virtual void encode(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)**

  Encode necessary values from the state repository. The second state repository represents what is seen by remote simulators and is used to determine when things need to be sent in the state message. includeThese is a set of attributes that should be included even if their thresholds have not been exceeded. This DtGenericObjectEncoder assumes an update condition of ÖnChange" for all attributes.

- **virtual DtBoolean valueDiffers(const DtGenericStateRepository& stateRep, const DtGenericStateRepository& asSeenByRemote, RTI::AttributeHandle handle)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file baseentenc.h:

class DtBaseEntityEncoder: public DtHlaStateEncoder

Inheritance:

DtBaseEntityEncoder - DtHlaStateEncoder

Public Methods

	DtBaseEntityEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtBaseEntityEncoder** ()
	DtBaseEntityEncoder (const DtBaseEntityEncoder& orig)
DtBaseEntityEncoder&	**operator=** (const DtBaseEntityEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
static DtBoolean	**tpsiChanged** (const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)

Protected Methods

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (Position)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER** (VelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (EntityType)

- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (Position)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER** (VelocityVector)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const

- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- **DtBaseEntityEncoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtBaseEntityEncoder()**
- **DtBaseEntityEncoder(const DtBaseEntityEncoder& orig)**
- **DtBaseEntityEncoder& operator=(const DtBaseEntityEncoder& orig)**
- **static DtHlaStateEncoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **static DtBoolean tpsiChanged(const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(AccelerationVector)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(AngularVelocityVector)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(DRAlgorithm)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(EntityType)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(EntityID)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(FederateID)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(IsFrozen)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(Orientation)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(Position)**
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER(VelocityVector)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(AccelerationVector)**

- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(AngularVelocityVector)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(DRAlgorithm)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(EntityType)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(EntityID)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(FederateID)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(IsFrozen)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(Orientation)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(Position)**
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER(VelocityVector)**

Direct child classes:

DtPhysicalEntityEncoder
DtEnvironmentEntityEncoder
DtAggregateEntityEncoder

Alphabetic index *Hierarchy of classes*

In file baseemittrenc.h:

class DtBaseEmitterBeamEncoder: public DtHlaStateEncoder

Inheritance:

DtBaseEmitterBeamEncoder - DtHlaStateEncoder

Public Methods

	DtBaseEmitterBeamEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtBaseEmitterBeamEncoder** ()
	DtBaseEmitterBeamEncoder (const DtBaseEmitterBeamEncoder& orig)
DtBaseEmitterBeamEncoder&	**operator=** (const DtBaseEmitterBeamEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (AzimuthCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (AzimuthSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (BeamFunction)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (BeamID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (BeamParameterIndex)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (ElevationCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (ElevationSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (EmittingSystemID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (ERP)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (EntityID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (EventID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (Frequency)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (FrequencyRange)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (PRF)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (PulseWidth)

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER** (SweepSynch)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (AzimuthCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (AzimuthSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (BeamFunction)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (BeamID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (BeamParameterIndex)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (ElevationCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (ElevationSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (EmittingSystemID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (ERP)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (EntityID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (EventID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (Frequency)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (FrequencyRange)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (PRF)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (PulseWidth)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER** (SweepSynch)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)

- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- **DtBaseEmitterBeamEncoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtBaseEmitterBeamEncoder()**
- **DtBaseEmitterBeamEncoder(const DtBaseEmitterBeamEncoder& orig)**
- **DtBaseEmitterBeamEncoder& operator=(const DtBaseEmitterBeamEncoder& orig)**
- **static DtHlaStateEncoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(AzimuthCenter)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(AzimuthSweep)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(BeamFunction)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(BeamID)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(BeamParameterIndex)**

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(ElevationCenter)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(ElevationSweep)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(EmittingSystemID)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(ERP)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(EntityID)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(EventID)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(Frequency)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(FrequencyRange)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(PRF)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(PulseWidth)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER(SweepSynch)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(AzimuthCenter)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(AzimuthSweep)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(BeamFunction)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(BeamID)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(BeamParameterIndex)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(ElevationCenter)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(ElevationSweep)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(EmittingSystemID)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(ERP)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(EntityID)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(EventID)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(Frequency)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(FrequencyRange)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(PRF)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(PulseWidth)**
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER(SweepSynch)**

Direct child classes:

DtTrackJamBeamEncoder
DtRadarBeamEncoder
DtJammerBeamEncoder

Alphabetic index *Hierarchy of classes*

In file baseembedenc.h:

class DtBaseEmbeddedSystemEncoder: public DtHlaStateEncoder

Inheritance:

DtBaseEmbeddedSystemEncoder - DtHlaStateEncoder

Public Methods

	DtBaseEmbeddedSystemEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtBaseEmbeddedSystemEncoder** ()
	DtBaseEmbeddedSystemEncoder (const DtBaseEmbeddedSystemEncoder& orig)
DtBaseEmbeddedSystemEncoder&	**operator=** (const DtBaseEmbeddedSystemEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER** (EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER** (HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER** (RelativePosition)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER** (EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER** (HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER** (RelativePosition)

Inherited from DtHlaStateEncoder:

Public Methods

typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)

- typedef DtBoolean(***DtAttributeChecker**)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- `DtBaseEmbeddedSystemEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `virtual ~DtBaseEmbeddedSystemEncoder()`
- `DtBaseEmbeddedSystemEncoder(const DtBaseEmbeddedSystemEncoder& orig)`
- `DtBaseEmbeddedSystemEncoder& operator=(const DtBaseEmbeddedSystemEncoder& orig)`
- `static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER(EntityID)`
- `DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER(HostObjectID)`
- `DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER(RelativePosition)`
- `DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER(EntityID)`
- `DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER(HostObjectID)`
- `DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER(RelativePosition)`

Direct child classes:

DtRadioTransmitterEncoder
DtRadioReceiverEncoder
DtEmitterSystemEncoder
DtDesignatorEncoder

Alphabetic index *Hierarchy of classes*

In file syshashkey.h:

class DtSystemHashKey: public DtBaseHashKey

class DtSystemHashKey: Instances of DtSystemHashKey are used as hash keys for an emitter system

---

Inheritance:

DtSystemHashKey - DtBaseHashKey

Public Methods

	DtSystemHashKey (const DtGlobalObjectDesignator& id, int systemId) *default constructor*
virtual	**~DtSystemHashKey** () *destructor*
	DtSystemHashKey (const DtSystemHashKey& orig) *copy constructor*
DtSystemHashKey&	**operator=** (const DtSystemHashKey& orig) *assignment operator*
virtual DtBaseHashKey*	**clone** () const *make a copy of this hash key*

Protected Fields

DtStringHashKey	**myDesHashKey**
DtString	**myStr**
int	**mySystemId**

Protected Methods

virtual const DtString&	**stringRep** () const *get the string representation of this key*
virtual DtU32	**computeHashValue** () *compute the hashed value for this key*

Inherited from DtBaseHashKey:

Public Methods

- virtual DtBoolean **operator==**(const DtBaseHashKey& other) const
- virtual DtBoolean **operator!=**(const DtBaseHashKey& other) const
- virtual DtU32 **hashValue**() const

Protected Fields

- DtBoolean **myHashValueIsComputed**
- DtU32 **myHashValue**

Documentation

class DtSystemHashKey: Instances of DtSystemHashKey are used as hash keys for an emitter system

- **`DtSystemHashKey(const DtGlobalObjectDesignator& id, int systemId)`**
  default constructor
- **`virtual ~DtSystemHashKey()`**
  destructor
- **`DtSystemHashKey(const DtSystemHashKey& orig)`**
  copy constructor
- **`DtSystemHashKey& operator=(const DtSystemHashKey& orig)`**
  assignment operator
- **`virtual DtBaseHashKey* clone() const`**
  make a copy of this hash key
- **`virtual const DtString& stringRep() const`**
  get the string representation of this key
- **`virtual DtU32 computeHashValue()`**
  compute the hashed value for this key
- **`DtStringHashKey myDesHashKey`**
- **`DtString myStr`**
- **`int mySystemId`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file strhashkey.h:

class DtStringHashKey: public DtBaseHashKey

class DtStringHashKey: Instances of DtStringHashKey are used as a hash key made from a string

---

Inheritance:

DtStringHashKey - DtBaseHashKey

Public Methods

	DtStringHashKey (const DtString& str) *default constructor*
	DtStringHashKey (const char* str) *alternate constructor*
virtual	**~DtStringHashKey** () *destructor*
	DtStringHashKey (const DtStringHashKey& orig) *copy constructor*
DtStringHashKey&	**operator=** (const DtStringHashKey& orig) *assignment operator*
virtual DtBaseHashKey*	**clone** () const *make a copy of this hash key*

Protected Fields

DtString **myString**

Protected Methods

virtual const DtString&	**stringRep** () const *get the string representation of this key*
virtual DtU32	**computeHashValue** () *compute the hashed value for this key*

Inherited from DtBaseHashKey:

Public Methods

- virtual DtBoolean **operator==**(const DtBaseHashKey& other) const
- virtual DtBoolean **operator!=**(const DtBaseHashKey& other) const
- virtual DtU32 **hashValue**() const

Protected Fields

- DtBoolean **myHashValueIsComputed**
- DtU32 **myHashValue**

Documentation

class DtStringHashKey: Instances of DtStringHashKey are used as a hash key made from a string

- **`DtStringHashKey(const DtString& str)`**
  default constructor
- **`DtStringHashKey(const char* str)`**
  alternate constructor
- **`virtual ~DtStringHashKey()`**
  destructor
- **`DtStringHashKey(const DtStringHashKey& orig)`**
  copy constructor
- **`DtStringHashKey& operator=(const DtStringHashKey& orig)`**
  assignment operator
- **`virtual DtBaseHashKey* clone() const`**
  make a copy of this hash key
- **`virtual const DtString& stringRep() const`**
  get the string representation of this key
- **`virtual DtU32 computeHashValue()`**
  compute the hashed value for this key
- **`DtString myString`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file objidhashkey.h:

class DtObjectIdHashKey: public DtBaseHashKey

class DtObjectIdHashKey: Instances of DtObjectIdHashKey are hash keys made from DtObjectIds

---

Inheritance:

DtObjectIdHashKey - DtBaseHashKey

Public Methods

	DtObjectIdHashKey (const DtObjectId& id) *default constructor*
virtual	**~DtObjectIdHashKey** () *destructor*
	DtObjectIdHashKey (const DtObjectIdHashKey& orig) *copy constructor*
DtObjectIdHashKey&	**operator=** (const DtObjectIdHashKey& orig) *assignment operator*
virtual DtBaseHashKey*	**clone** () const *make a copy of this hash key*

Protected Fields

DtObjectId	**myId**
DtString	**myStr**

Protected Methods

virtual const DtString&	**stringRep** () const *get the string representation of this key*
virtual DtU32	**computeHashValue** () *compute the hashed value for this key*

Inherited from DtBaseHashKey:

Public Methods

- virtual DtBoolean **operator==**(const DtBaseHashKey& other) const
- virtual DtBoolean **operator!=**(const DtBaseHashKey& other) const
- virtual DtU32 **hashValue**() const

Protected Fields

- DtBoolean **myHashValueIsComputed**
- DtU32 **myHashValue**

Documentation

class DtObjectIdHashKey:

Instances of DtObjectIdHashKey are hash keys made from DtObjectIds. They can be used with DtBaseHashList to store arbitrary pieces of data hashed by DtObjectId.

- **`DtObjectIdHashKey(const DtObjectId& id)`**

  default constructor

- **`virtual ~DtObjectIdHashKey()`**

  destructor

- **`DtObjectIdHashKey(const DtObjectIdHashKey& orig)`**

  copy constructor

- **`DtObjectIdHashKey& operator=(const DtObjectIdHashKey& orig)`**

  assignment operator

- **`virtual DtBaseHashKey* clone() const`**

  make a copy of this hash key

- **`virtual const DtString& stringRep() const`**

  get the string representation of this key

- **`virtual DtU32 computeHashValue()`**

  compute the hashed value for this key

- **`DtObjectId myId`**

- **`DtString myStr`**

This class has no child classes.

In file inthashkey.h:

class DtIntHashKey: public DtBaseHashKey

class DtIntHashKey: Instances of DtIntHashKey are used as a hash key made from an integer

---

Inheritance:

DtIntHashKey - DtBaseHashKey

Public Methods

	DtIntHashKey (unsigned long num) *default constructor*
virtual	**~DtIntHashKey** () *destructor*
	DtIntHashKey (const DtIntHashKey& orig) *copy constructor*
DtIntHashKey&	**operator=** (const DtIntHashKey& orig) *assignment operator*
virtual DtBaseHashKey*	**clone** () const *make a copy of this hash key*

Protected Fields

unsigned long	**myNum**
DtString	**myStr**

Protected Methods

virtual const DtString&	**stringRep** () const *get the string representation of this key*
virtual DtU32	**computeHashValue** () *compute the hashed value for this key*

Inherited from DtBaseHashKey:

Public Methods

- virtual DtBoolean **operator==**(const DtBaseHashKey& other) const
- virtual DtBoolean **operator!=**(const DtBaseHashKey& other) const
- virtual DtU32 **hashValue**() const

Protected Fields

- DtBoolean **myHashValueIsComputed**
- DtU32 **myHashValue**

Documentation

class DtIntHashKey: Instances of DtIntHashKey are used as a hash key made from an integer

- **`DtIntHashKey(unsigned long num)`**
  default constructor
- **`virtual ~DtIntHashKey()`**
  destructor
- **`DtIntHashKey(const DtIntHashKey& orig)`**
  copy constructor
- **`DtIntHashKey& operator=(const DtIntHashKey& orig)`**
  assignment operator
- **`virtual DtBaseHashKey* clone() const`**
  make a copy of this hash key
- **`virtual const DtString& stringRep() const`**
  get the string representation of this key
- **`virtual DtU32 computeHashValue()`**
  compute the hashed value for this key
- **`unsigned long myNum`**
- **`DtString myStr`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtLgrCommand

- **DtLgrPbFile**
- **DtLgrRecFile**
- **DtLgrQuit**
- **DtLgrStart**
- **DtLgrEnd**
- **DtLgrPlay**
- **DtLgrPause**
- **DtLgrStop**
- **DtLgrRecord**
- **DtLgrSetTime**
- **DtLgrSkipTime**
- **DtLgrTimeScale**
- **DtLgrPlayAction**
- **DtLgrRecordAction**
- **DtLgrEofAction**
- **DtLgrHeartbeat**
- **DtLgrTimeout**

Documentation

- `DtLgrPbFile`
- `DtLgrRecFile`
- `DtLgrQuit`
- `DtLgrStart`
- `DtLgrEnd`
- `DtLgrPlay`

- **DtLgrPause**
- **DtLgrStop**
- **DtLgrRecord**
- **DtLgrSetTime**
- **DtLgrSkipTime**
- **DtLgrTimeScale**
- **DtLgrPlayAction**
- **DtLgrRecordAction**
- **DtLgrEofAction**
- **DtLgrHeartbeat**
- **DtLgrTimeout**

Alphabetic index *Hierarchy of classes*

In file lctimeout.h:

class DtLgrTimeoutPdu: public DtLgrControlPdu

class DtLgrTimeoutPdu: Instances of DtLgrTimeoutPdu are to tell remote logger how long to generate interim packets for entity before assuming no more packets in file

---

Inheritance:

DtLgrTimeoutPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrTimeoutPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrTimeoutPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrTimeoutPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtReal tmout, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrTimeoutPdu** () *destructor*
	DtLgrTimeoutPdu (const DtLgrTimeoutPdu& orig) *copy constructor*
DtLgrTimeoutPdu&	**operator=** (const DtLgrTimeoutPdu& orig) *assignment operator*
virtual void	**setTimeout** (DtReal val) *set/get timeout*
virtual DtReal	**timeout** ()
virtual DtLgrRealParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*

Private Fields

DtReal	**myTimeout** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const

- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)

- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrTimeoutPdu:

Instances of DtLgrTimeoutPdu are to tell remote logger how long to generate interim packets for entity before assuming no more packets in file. Useful during slow-motion playback.

- **`DtLgrTimeoutPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

default constructor

- **`DtLgrTimeoutPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrTimeoutPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtReal tmout, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrTimeoutPdu()`**

  destructor

- **`DtLgrTimeoutPdu(const DtLgrTimeoutPdu& orig)`**

  copy constructor

- **`DtLgrTimeoutPdu& operator=(const DtLgrTimeoutPdu& orig)`**

  assignment operator

- **`virtual void setTimeout( DtReal val)`**

  set/get timeout. see class description

- **`virtual DtReal timeout()`**

- **`virtual DtLgrRealParams* params()`**

  return correctly casted parameters pointer; see lgrControlP

- **`DtReal myTimeout`**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lctimescale.h:

class DtLgrTimeScalePdu: public DtLgrControlPdu

class DtLgrTimeScalePdu: Instances of DtLgrTimeScalePdu are to tell remote logger to slow down or speed up playback

---

Inheritance:

DtLgrTimeScalePdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrTimeScalePdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrTimeScalePdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrTimeScalePdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, double scale, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrTimeScalePdu** () *destructor*
	DtLgrTimeScalePdu (const DtLgrTimeScalePdu& orig) *copy constructor*
DtLgrTimeScalePdu&	**operator=** (const DtLgrTimeScalePdu& orig) *assignment operator*
virtual void	**setTimeScale** (DtReal time) *set/get playback speed*
virtual double	**timeScale** ()
virtual DtLgrRealParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*

Private Fields

DtReal	**myScale** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const

- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)

- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrTimeScalePdu:

Instances of DtLgrTimeScalePdu are to tell remote logger to slow down or speed up playback. 1.0 is normal speed

- **`DtLgrTimeScalePdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

default constructor

- **`DtLgrTimeScalePdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrTimeScalePdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, double scale, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrTimeScalePdu()`**

  destructor

- **`DtLgrTimeScalePdu(const DtLgrTimeScalePdu& orig)`**

  copy constructor

- **`DtLgrTimeScalePdu& operator=(const DtLgrTimeScalePdu& orig)`**

  assignment operator

- **`virtual void setTimeScale( DtReal time)`**

  set/get playback speed

- **`virtual double timeScale()`**

- **`virtual DtLgrRealParams* params()`**

  return correctly casted parameters pointer; see lgrControlP

- **`DtReal myScale`**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcstop.h:

class DtLgrStopPdu: public DtLgrControlPdu

class DtLgrStopPdu: Instances of DtLgrStopPdu are to tell remote logger to stop

---

Inheritance:

DtLgrStopPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrStopPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrStopPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrStopPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrStopPdu** () *destructor*
	DtLgrStopPdu (const DtLgrStopPdu& orig) *copy constructor*
DtLgrStopPdu&	**operator=** (const DtLgrStopPdu& orig) *assignment operator*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const

- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const

- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrStopPdu: Instances of DtLgrStopPdu are to tell remote logger to stop

- **`DtLgrStopPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrStopPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrStopPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrStopPdu()`**

  destructor

- **`DtLgrStopPdu(const DtLgrStopPdu& orig)`**

  copy constructor

- **`DtLgrStopPdu& operator=(const DtLgrStopPdu& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcstart.h:

class DtLgrStartPdu: public DtLgrControlPdu

class DtLgrStartPdu: Instances of DtLgrStartPdu are used to tell remote logger to start recording

---

Inheritance:

DtLgrStartPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrStartPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrStartPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrStartPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrStartPdu** () *destructor*
	DtLgrStartPdu (const DtLgrStartPdu& orig) *copy constructor*
DtLgrStartPdu&	**operator=** (const DtLgrStartPdu& orig) *assignment operator*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const

- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const

- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrStartPdu: Instances of DtLgrStartPdu are used to tell remote logger to start recording

- **`DtLgrStartPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrStartPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrStartPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrStartPdu()`**

  destructor

- **`DtLgrStartPdu(const DtLgrStartPdu& orig)`**

  copy constructor

- **`DtLgrStartPdu& operator=(const DtLgrStartPdu& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcskiptime.h:

class DtLgrSkipTimePdu: public DtLgrControlPdu

class DtLgrSkipTimePdu: Instances of DtLgrSetTimePdu are to tell remote logger to forward or backward HH:MM:SS

Inheritance:

DtLgrSkipTimePdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrSkipTimePdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrSkipTimePdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrSkipTimePdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtTime tm, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrSkipTimePdu** () *destructor*
	DtLgrSkipTimePdu (const DtLgrSkipTimePdu& orig) *copy constructor*
DtLgrSkipTimePdu&	**operator=** (const DtLgrSkipTimePdu& orig) *assignment operator*
virtual void	**setTime** (DtTime dt) *set/get amount of time to skip forward or back in playback file*
virtual DtTime	**time** ()
virtual DtLgrTimeParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*

Private Fields

DtTime	**myTime** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const

- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)

- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrSkipTimePdu: Instances of DtLgrSetTimePdu are to tell remote logger to forward or backward HH:MM:SS

- **`DtLgrSkipTimePdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrSkipTimePdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrSkipTimePdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtTime tm, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrSkipTimePdu()`**

  destructor

- **`DtLgrSkipTimePdu(const DtLgrSkipTimePdu& orig)`**

  copy constructor

- **`DtLgrSkipTimePdu& operator=(const DtLgrSkipTimePdu& orig)`**

  assignment operator

- **`virtual void setTime(DtTime dt)`**

  set/get amount of time to skip forward or back in playback file

- **`virtual DtTime time()`**

- **`virtual DtLgrTimeParams* params()`**

  return correctly casted parameters pointer; see lgrControlP

- **`DtTime myTime`**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcsettime.h:

class DtLgrSetTimePdu: public DtLgrControlPdu

class DtLgrSetTimePdu: Instances of DtLgrSetTimePdu are to tell remote logger to go to set position to HH:MM:SS

Inheritance:

DtLgrSetTimePdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrSetTimePdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrSetTimePdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrSetTimePdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtTime tm, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrSetTimePdu** () *destructor*
	DtLgrSetTimePdu (const DtLgrSetTimePdu& orig) *copy constructor*
DtLgrSetTimePdu&	**operator=** (const DtLgrSetTimePdu& orig) *assignment operator*
virtual void	**setTime** (const DtTime time) *set/get time to jump to within logger playback file*
virtual DtTime	**time** ()
virtual DtLgrTimeParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*

Private Fields

DtTime	**myTime** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const

- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)

- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrSetTimePdu: Instances of DtLgrSetTimePdu are to tell remote logger to go to set position to HH:MM:SS

- **`DtLgrSetTimePdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrSetTimePdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrSetTimePdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtTime tm, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrSetTimePdu()`**

  destructor

- **`DtLgrSetTimePdu(const DtLgrSetTimePdu& orig)`**

  copy constructor

- **`DtLgrSetTimePdu& operator=(const DtLgrSetTimePdu& orig)`**

  assignment operator

- **`virtual void setTime(const DtTime time)`**

  set/get time to jump to within logger playback file

- **`virtual DtTime time()`**

- **`virtual DtLgrTimeParams* params()`**

  return correctly casted parameters pointer; see lgrControlP

- **`DtTime myTime`**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcrecord.h:

class DtLgrRecordPdu: public DtLgrControlPdu

class DtLgrRecordPdu: Instances of DtLgrRecordPdu are to tell remote logger to begin recording

---

Inheritance:

DtLgrRecordPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrRecordPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrRecordPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrRecordPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrRecordPdu** () *destructor*
	DtLgrRecordPdu (const DtLgrRecordPdu& orig) *copy constructor*
DtLgrRecordPdu&	**operator=** (const DtLgrRecordPdu& orig) *assignment operator*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const

- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const

- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrRecordPdu: Instances of DtLgrRecordPdu are to tell remote logger to begin recording

- **`DtLgrRecordPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrRecordPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrRecordPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrRecordPdu()`**

  destructor

- **`DtLgrRecordPdu(const DtLgrRecordPdu& orig)`**

  copy constructor

- **`DtLgrRecordPdu& operator=(const DtLgrRecordPdu& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcrecaction.h:

class DtLgrRecordActionPdu: public DtLgrControlPdu

class DtLgrRecordActionPdu: Instances of DtLgrRecordActionPdu are to tell remote logger to stop or begin recording when entering record-mode

---

Inheritance:

DtLgrRecordActionPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrRecordActionPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrRecordActionPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrRecordActionPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtLgrMode mode, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrRecordActionPdu** () *destructor*
	DtLgrRecordActionPdu (const DtLgrRecordActionPdu& orig) *copy constructor*
DtLgrRecordActionPdu&	**operator=** (const DtLgrRecordActionPdu& orig) *assignment operator*
virtual void	**setAction** (DtLgrMode cmd) *set/get action to take when switching to record-mode*
virtual DtLgrMode	**action** ()
virtual DtLgrIntParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*

Private Fields

DtLgrMode	**myAction** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const

- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)

- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrRecordActionPdu:

Instances of DtLgrRecordActionPdu are to tell remote logger to stop or begin recording when entering record-mode.

- **`DtLgrRecordActionPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

default constructor

- **DtLgrRecordActionPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct from network representation

- **DtLgrRecordActionPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtLgrMode mode, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct and prepare for send

- **virtual ~DtLgrRecordActionPdu()**

  destructor

- **DtLgrRecordActionPdu(const DtLgrRecordActionPdu& orig)**

  copy constructor

- **DtLgrRecordActionPdu& operator=(const DtLgrRecordActionPdu& orig)**

  assignment operator

- **virtual void setAction(DtLgrMode cmd)**

  set/get action to take when switching to record-mode

- **virtual DtLgrMode action()**

- **virtual DtLgrIntParams* params()**

  return correctly casted parameters pointer; see lgrControlP

- **DtLgrMode myAction**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcpdu.h:

enum DtLgrMode

General operation modes

- **DtLGR_PLAY**
- **DtLGR_PAUSE**
- **DtLGR_RECORD**
- **DtLGR_STOP**
- **DtLGR_EMPTY**
- **DtLGR_LOOP**

Documentation

General operation modes

- `DtLGR_PLAY`
- `DtLGR_PAUSE`
- `DtLGR_RECORD`
- `DtLGR_STOP`
- `DtLGR_EMPTY`
- `DtLGR_LOOP`

In file lcrecfile.h:

class DtLgrRecFilePdu: public DtLgrControlPdu

class DtLgrRecFilePdu: Instances of DtLgrRecFilePdu are used to switching remote logger to recording-mode and to set the file into which to store captured simulation traffic

---

Inheritance:

DtLgrRecFilePdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrRecFilePdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrRecFilePdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrRecFilePdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, char* filename, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrRecFilePdu** () *destructor*
	DtLgrRecFilePdu (const DtLgrRecFilePdu& orig) *copy constructor*
DtLgrRecFilePdu&	**operator=** (const DtLgrRecFilePdu& orig) *assignment operator*
virtual void	**setFilename** (const char* path) *set/get the name of file into which traffic will be recorded*
virtual const char*	**filename** ()
virtual DtLgrStringParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*
virtual void	**printData** () const *diagnostic output*

Private Fields

char*	**myFilename** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const

- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()

- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrRecFilePdu: Instances of DtLgrRecFilePdu are used to switching remote logger to recording-mode and to set the file into which to store captured simulation traffic

- **`DtLgrRecFilePdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrRecFilePdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer =`**

DtUSE_INTERNAL_BUFFER)

construct from network representation

- **DtLgrRecFilePdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, char* filename, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct and prepare for send

- **virtual ~DtLgrRecFilePdu()**

  destructor

- **DtLgrRecFilePdu(const DtLgrRecFilePdu& orig)**

  copy constructor

- **DtLgrRecFilePdu& operator=(const DtLgrRecFilePdu& orig)**

  assignment operator

- **virtual void setFilename(const char* path)**

  set/get the name of file into which traffic will be recorded

- **virtual const char* filename()**

- **virtual DtLgrStringParams* params()**

  return correctly casted parameters pointer; see lgrControlP

- **virtual void printData() const**

  diagnostic output

- **char* myFilename**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcquit.h:

class DtLgrQuitPdu: public DtLgrControlPdu

class DtLgrQuitPdu: Instances of DtLgrQuitPdu are messages telling a remote logger to quit

---

Inheritance:

DtLgrQuitPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrQuitPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrQuitPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrQuitPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrQuitPdu** () *destructor*
	DtLgrQuitPdu (const DtLgrQuitPdu& orig) *copy constructor*
DtLgrQuitPdu&	**operator=** (const DtLgrQuitPdu& orig) *assignment operator*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const

- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const

- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrQuitPdu: Instances of DtLgrQuitPdu are messages telling a remote logger to quit

- **`DtLgrQuitPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrQuitPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrQuitPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrQuitPdu()`**

  destructor

- **`DtLgrQuitPdu(const DtLgrQuitPdu& orig)`**

  copy constructor

- **`DtLgrQuitPdu& operator=(const DtLgrQuitPdu& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcplay.h:

class DtLgrPlayPdu: public DtLgrControlPdu

class DtLgrPlayPdu: Instances of DtLgrPlayPdu are used to tell remote logger to begin recording

---

Inheritance:

DtLgrPlayPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrPlayPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrPlayPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrPlayPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrPlayPdu** () *destructor*
	DtLgrPlayPdu (const DtLgrPlayPdu& orig) *copy constructor*
DtLgrPlayPdu&	**operator=** (const DtLgrPlayPdu& orig) *assignment operator*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const

- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const

- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrPlayPdu: Instances of DtLgrPlayPdu are used to tell remote logger to begin recording

- **`DtLgrPlayPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrPlayPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrPlayPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrPlayPdu()`**

  destructor

- **`DtLgrPlayPdu(const DtLgrPlayPdu& orig)`**

  copy constructor

- **`DtLgrPlayPdu& operator=(const DtLgrPlayPdu& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcplayaction.h:

class DtLgrPlayActionPdu: public DtLgrControlPdu

class DtLgrPlayActionPdu: Instances of DtLgrPlayActionPdu are to tell remote logger what action to take initially when set to playback mode

---

Inheritance:

DtLgrPlayActionPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrPlayActionPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrPlayActionPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrPlayActionPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtLgrMode mode, DtBufferPtr buffer =DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrPlayActionPdu** () *destructor*
	DtLgrPlayActionPdu (const DtLgrPlayActionPdu& orig) *copy constructor*
DtLgrPlayActionPdu&	**operator=** (const DtLgrPlayActionPdu& orig) *assignment operator*
virtual void	**setAction** (DtLgrMode cmd) *set/get action to take place when logger switched to playback mode*
virtual DtLgrMode	**action** ()
virtual DtLgrIntParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*

Private Fields

DtLgrMode	**myAction** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const

- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)

- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrPlayActionPdu: Instances of DtLgrPlayActionPdu are to tell remote logger what action to take initially when set to playback mode

- **`DtLgrPlayActionPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **DtLgrPlayActionPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct from network representation

- **DtLgrPlayActionPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtLgrMode mode, DtBufferPtr buffer =DtUSE_INTERNAL_BUFFER)**

  construct and prepare for send

- **virtual ~DtLgrPlayActionPdu()**

  destructor

- **DtLgrPlayActionPdu(const DtLgrPlayActionPdu& orig)**

  copy constructor

- **DtLgrPlayActionPdu& operator=(const DtLgrPlayActionPdu& orig)**

  assignment operator

- **virtual void setAction(DtLgrMode cmd)**

  set/get action to take place when logger switched to playback mode

- **virtual DtLgrMode action()**

- **virtual DtLgrIntParams* params()**

  return correctly casted parameters pointer; see lgrControlP

- **DtLgrMode myAction**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcplayfile.h:

class DtLgrPbFilePdu: public DtLgrControlPdu

class DtLgrPbFilePdu: Instances of DtLgrPbFilePdu are used for switching to play-mode and defining which file to read from when sending traffic into the simulation

---

Inheritance:

DtLgrPbFilePdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrPbFilePdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrPbFilePdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrPbFilePdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, char* filename, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrPbFilePdu** () *destructor*
	DtLgrPbFilePdu (const DtLgrPbFilePdu& orig) *copy constructor*
DtLgrPbFilePdu&	**operator=** (const DtLgrPbFilePdu& orig) *assignment operator*
virtual void	**setFilename** (const char* path) *set/get file to open for playback*
virtual const char*	**filename** ()
virtual DtLgrStringParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*
virtual void	**printData** () const *diagnostic output*

Private Fields

char*	**myFilename** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const

- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()

- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrPbFilePdu: Instances of DtLgrPbFilePdu are used for switching to play-mode and defining which file to read from when sending traffic into the simulation

- **`DtLgrPbFilePdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrPbFilePdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer =`**

DtUSE_INTERNAL_BUFFER)

construct from network representation

- **DtLgrPbFilePdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, char* filename, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct and prepare for send

- **virtual ~DtLgrPbFilePdu()**

  destructor

- **DtLgrPbFilePdu(const DtLgrPbFilePdu& orig)**

  copy constructor

- **DtLgrPbFilePdu& operator=(const DtLgrPbFilePdu& orig)**

  assignment operator

- **virtual void setFilename(const char* path)**

  set/get file to open for playback. (Path relative to logger)

- **virtual const char* filename()**

- **virtual DtLgrStringParams* params()**

  return correctly casted parameters pointer; see lgrControlP

- **virtual void printData() const**

  diagnostic output

- **char* myFilename**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcpause.h:

class DtLgrPausePdu: public DtLgrControlPdu

class DtLgrPausePdu: Instances of DtLgrPausePdu are to tell remote logger to pause

---

Inheritance:

DtLgrPausePdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrPausePdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrPausePdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrPausePdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrPausePdu** () *destructor*
	DtLgrPausePdu (const DtLgrPausePdu& orig) *copy constructor*
DtLgrPausePdu&	**operator=** (const DtLgrPausePdu& orig) *assignment operator*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const

- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const

- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrPausePdu: Instances of DtLgrPausePdu are to tell remote logger to pause

- **`DtLgrPausePdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrPausePdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrPausePdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrPausePdu()`**

  destructor

- **`DtLgrPausePdu(const DtLgrPausePdu& orig)`**

  copy constructor

- **`DtLgrPausePdu& operator=(const DtLgrPausePdu& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcheartbeat.h:

class DtLgrHeartbeatPdu: public DtLgrControlPdu

class DtLgrHeartbeatPdu: Instances of DtLgrHeartbeatPdu are to tell remote logger to send packet for each playback entity at least every HEARTBEAT seconds

---

Inheritance:

DtLgrHeartbeatPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrHeartbeatPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrHeartbeatPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrHeartbeatPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtReal hbeat, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrHeartbeatPdu** () *destructor*
	DtLgrHeartbeatPdu (const DtLgrHeartbeatPdu& orig) *copy constructor*
DtLgrHeartbeatPdu&	**operator=** (const DtLgrHeartbeatPdu& orig) *assignment operator*
virtual void	**setHeartbeat** (DtReal val) *set/get the heartbeat*
virtual DtReal	**heartbeat** ()
virtual DtLgrRealParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*

Private Fields

DtReal	**myHeartbeat** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const

- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)

- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrHeartbeatPdu:

Instances of DtLgrHeartbeatPdu are to tell remote logger to send packet for each playback entity at least every HEARTBEAT seconds. Important when in slow-motion.

- **`DtLgrHeartbeatPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

default constructor

- **DtLgrHeartbeatPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct from network representation

- **DtLgrHeartbeatPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtReal hbeat, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct and prepare for send

- **virtual ~DtLgrHeartbeatPdu()**

  destructor

- **DtLgrHeartbeatPdu(const DtLgrHeartbeatPdu& orig)**

  copy constructor

- **DtLgrHeartbeatPdu& operator=(const DtLgrHeartbeatPdu& orig)**

  assignment operator

- **virtual void setHeartbeat(DtReal val)**

  set/get the heartbeat. see class description.

- **virtual DtReal heartbeat()**

- **virtual DtLgrRealParams* params()**

  return correctly casted parameters pointer; see lgrControlP

- **DtReal myHeartbeat**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lceofaction.h:

class DtLgrEofActionPdu: public DtLgrControlPdu

class DtLgrEofActionPdu: Instances of DtLgrEofActionPdu are to tell remote logger that it should pause, wrap, or stop at the end of playback

---

Inheritance:

DtLgrEofActionPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrEofActionPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrEofActionPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrEofActionPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtLgrMode mode, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrEofActionPdu** () *destructor*
	DtLgrEofActionPdu (const DtLgrEofActionPdu& orig) *copy constructor*
DtLgrEofActionPdu&	**operator=** (const DtLgrEofActionPdu& orig) *assignment operator*
virtual DtLgrMode	**action** () *Set/Get value for what to do when Logger reaches EOF*
virtual void	**setAction** (DtLgrMode cmd) *mutator functions*
virtual DtLgrIntParams*	**params** () *return correctly casted parameters pointer; see lgrControlP*

Private Fields

DtLgrMode	**myAction** *parameter*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const
- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const

- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const
- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)

- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrEofActionPdu: Instances of DtLgrEofActionPdu are to tell remote logger that it should pause, wrap, or stop at the end of playback

- **`DtLgrEofActionPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **DtLgrEofActionPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct from network representation

- **DtLgrEofActionPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtLgrMode mode, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)**

  construct and prepare for send

- **virtual ~DtLgrEofActionPdu()**

  destructor

- **DtLgrEofActionPdu(const DtLgrEofActionPdu& orig)**

  copy constructor

- **DtLgrEofActionPdu& operator=(const DtLgrEofActionPdu& orig)**

  assignment operator

- **virtual DtLgrMode action()**

  Set/Get value for what to do when Logger reaches EOF

- **virtual void setAction(DtLgrMode cmd)**

  mutator functions

- **virtual DtLgrIntParams* params()**

  return correctly casted parameters pointer; see lgrControlP

- **DtLgrMode myAction**

  parameter

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file lcend.h:

class DtLgrEndPdu: public DtLgrControlPdu

class DtLgrEndPdu: Instances of DtLgrEndPdu are to tell remote logger to skip to end of file during playback

---

Inheritance:

DtLgrEndPdu - DtLgrControlPdu - DtPdu - DtBaseMsg

Public Methods

	DtLgrEndPdu (DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *default constructor*
	DtLgrEndPdu (const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct from network representation*
	DtLgrEndPdu (DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER) *construct and prepare for send*
virtual	**~DtLgrEndPdu** () *destructor*
	DtLgrEndPdu (const DtLgrEndPdu& orig) *copy constructor*
DtLgrEndPdu&	**operator=** (const DtLgrEndPdu& orig) *assignment operator*

Inherited from DtLgrControlPdu:

Public Methods

- static DtPdu* **create**(const DtNetPacket* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)
- static DtLgrControlPdu* **create**(DtLgrCommand command)
- virtual void **setReceiverId**(const DtGlobalObjectDesignator& id)
- const DtGlobalObjectDesignator& **receiverId**() const
- virtual void **setSenderId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **senderId**() const

- DtLgrCommand **lgrCommand**() const
- virtual void **printData**() const
- inline DtPduKind **internalGetPduKind**() const

Protected Methods

- DtPduKind **internalGetPduKind**() const
- void **setLgrCommand**(DtLgrCommand com)

Private Fields

- DtGlobalObjectDesignator **mySenderId**
- DtGlobalObjectDesignator **myReceiverId**

Inherited from DtPdu:

Public Methods

- virtual void* **packet**() const
- virtual int **status**() const
- virtual void **setVersion**(int vers)
- virtual int **protocolVersion**() const
- virtual void **setExerciseId**(int id)
- virtual int **exerciseId**() const
- virtual DtPduKind **kind**() const
- virtual DtProtocolFamily **protocolFamily**() const
- virtual void **setTimeStamp**(DtTime time, DtTimeStampType type = DtTimeStampRelative)
- virtual DtTime **timeStamp**() const
- virtual DtTimeStampType **timeStampType**() const
- virtual DtTime **guessTimeValid**(DtTime referenceTime) const
- virtual void **print**() const
- virtual void **printHeader**() const
- inline int **exerciseId**() const
- inline DtPduKind **kind**() const

- inline DtNetPduHeader* **netHeader**() const
- inline void* **packet**() const
- inline DtProtocolFamily **protocolFamily**() const
- inline int **protocolVersion**() const
- inline void **setExerciseId**(int id)
- inline void **setStatus**(int okStatus)
- inline void **setTimeStamp**(DtTime time, DtTimeStampType relOrAbs)
- inline void **setVersion**(int vers)
- inline int **status**() const
- inline DtTime **timeStamp**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- int **myOKStatus**

Protected Methods

- virtual void **setStatus**(int okStatus)
- virtual DtNetPduHeader* **netHeader**() const
- virtual void **initPdu**(int size, DtBufferPtr buffer)
- virtual void **initPdu**(const void* initial, DtBufferPtr buffer)
- virtual DtProtocolFamily **kindToFamily**(DtPduKind kind) const
- virtual void **sizeChanged**(int size)
- virtual void **clearData**()
- virtual void **initHeader**(DtPduKind kind, int size)

Inherited from DtBaseMsg:

Public Methods

- virtual const char* **netRep**() const
- virtual int **length**() const
- inline const char* **netRep**() const

Protected Fields

- void* **myNetMsg**
- void* **myInternalBuff**
- int **myIntBuffSize**
- int **myMsgSize**

Protected Methods

- void **initMsg**(int size, DtBufferPtr bufferPtr)
- void **initMsg**(const void *initial, int size, DtBufferPtr bufferPtr)
- virtual void **changeSize**(int size)
- virtual void **insertBytes**(int offset, int numBytes)
- virtual void **insertBytes**(void *address, int numBytes)
- virtual void **deleteBytes**(int offset, int numBytes)
- virtual void **deleteBytes**(void *address, int numBytes)
- virtual void **clearAll**()
- virtual void* **newInternalBuffer**(int size)

Documentation

class DtLgrEndPdu: Instances of DtLgrEndPdu are to tell remote logger to skip to end of file during playback

- **`DtLgrEndPdu(DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  default constructor

- **`DtLgrEndPdu(const DtNetLgrControlPdu* initial, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct from network representation

- **`DtLgrEndPdu(DtGlobalObjectDesignator& sender, DtGlobalObjectDesignator& receiver, DtBufferPtr buffer = DtUSE_INTERNAL_BUFFER)`**

  construct and prepare for send

- **`virtual ~DtLgrEndPdu()`**

  destructor

- **`DtLgrEndPdu(const DtLgrEndPdu& orig)`**

  copy constructor

- **`DtLgrEndPdu& operator=(const DtLgrEndPdu& orig)`**

  assignment operator

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hoststructs.h:

class DtBurstDescriptor

Public Fields

DtEntityType	**DtMunition**
DtWarheadType	**DtWarhead**
DtDetonatorFuze	**DtFuze**
int	**DtQuantity**
int	**DtRate**

Public Methods

- **DtBurstDescriptor** ()
- **DtBurstDescriptor** (DtNetBurstDescriptor *netBurst)
- **DtBurstDescriptor** (DtEntityType *type, DtWarheadType warhead, DtDetonatorFuze fuze, DtU16 quantity, DtU16 rate)
- **operator DtNetBurstDescriptor** () const
- void **init** (DtEntityType *type, DtWarheadType warhead, DtDetonatorFuze fuze, int quantity, int rate)

Documentation

- **`DtBurstDescriptor()`**
- **`DtBurstDescriptor(DtNetBurstDescriptor *netBurst)`**
- **`DtBurstDescriptor(DtEntityType *type, DtWarheadType warhead, DtDetonatorFuze fuze, DtU16 quantity, DtU16 rate)`**
- **`operator DtNetBurstDescriptor () const`**
- **`void init(DtEntityType *type, DtWarheadType warhead, DtDetonatorFuze fuze, int quantity, int rate)`**
- **`DtEntityType DtMunition`**
- **`DtWarheadType DtWarhead`**
- **`DtDetonatorFuze DtFuze`**

- `int DtQuantity`
- `int DtRate`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file netstructs.h:

class DtNetBurstDescriptor

Public Fields

- DtNetEntityType **DtMunition**
- DtNetU16 **DtWarhead**
- DtNetU16 **DtFuze**
- DtNetU16 **DtQuantity**
- DtNetU16 **DtRate**

Public Methods

- DtBurstDescriptor **operator =** (DtBurstDescriptor)
- DtNetBurstDescriptor **operator =** (DtNetBurstDescriptor)

Documentation

- **`DtNetEntityType DtMunition`**
- **`DtNetU16 DtWarhead`**
- **`DtNetU16 DtFuze`**
- **`DtNetU16 DtQuantity`**
- **`DtNetU16 DtRate`**
- **`DtBurstDescriptor operator = (DtBurstDescriptor)`**
- **`DtNetBurstDescriptor operator = (DtNetBurstDescriptor)`**

This class has no child classes.

In file cartview.h:

class DtCartesianView

class DtCartesianView: Instances of DtCartesianView are used to create a Cartesian coordinate view of the geocentric coordinates in an Entity State Repository

---

Inheritance:

DtCartesianView

Public Methods

	DtCartesianView (DtBaseEntityStateRepository* ksr) *constructor*
	DtCartesianView (DtBaseEntityStateRepository* ksr, DtConstVector originInGeoc, DtConstDcm geocToLocalRot)
virtual	**~DtCartesianView** () *destructor*
	DtCartesianView (const DtCartesianView& orig) *copy constructor*
DtCartesianView&	**operator=** (const DtCartesianView& orig) *assignment operator*
virtual void	**setOffset** (DtConstVector originInGeoc) *set the origin of the local system in geocentric coords*
virtual void	**setRotation** (DtConstDcm geocToLocalRot) *set the rotation matrix that converts from geocentric to local coords*
virtual void	**setLocation** (DtConstVector Location) *Set/Get the location in local coordinates*
virtual DtConstVector	**location** () const
virtual void	**setVelocity** (DtConstVector Velocity) *Set/Get the velocity*
virtual DtConstVector	**velocity** () const
virtual void	**setOrientation** (const DtTaitBryan& Orient) *Set/Get the orientation*
virtual const DtTaitBryan&	**orientation** () const
virtual DtConstDcm	**bodyToLocal** () const *Get the body to local coordinate matrix*
virtual void	**setAcceleration** (DtConstVector Acceleration) *Set/Get the acceleration*
virtual DtConstVector	**acceleration** () const

virtual void	**setRotationalVelocity** (DtConstVector BodyRates) *Set/Get the rotational velocity*
virtual DtConstVector	**rotationalVelocity** () const
inline DtCartesianView*	**self** () const

Protected Fields

DtBaseEntityStateRepository*	**myESR**
DtCoordTransform	**myGeocToLocal**
DtCoordTransform	**myLocalToGeoc**
DtVector	**myLocalLoc** *used as return values by inspectors*
DtVector	**myLocalVel**
DtVector	**myLocalAccel**
DtTaitBryan	**myLocalOrient**
DtDcm	**myBodyToLocal**

Protected Methods

DtCartesianView*	**self** () const *Cast off const-ness for trivial internal change*

Documentation

class DtCartesianView: Instances of DtCartesianView are used to create a Cartesian coordinate view of the geocentric coordinates in an Entity State Repository

`DtCartesianView(DtBaseEntityStateRepository* ksr)`

constructor

`DtCartesianView(DtBaseEntityStateRepository* ksr, DtConstVector originInGeoc, DtConstDcm geocToLocalRot)`

`virtual ~DtCartesianView()`

destructor

`DtCartesianView(const DtCartesianView& orig)`

copy constructor

`DtCartesianView& operator=(const DtCartesianView& orig)`

assignment operator

`virtual void setOffset(DtConstVector originInGeoc)`

set the origin of the local system in geocentric coords

- **`virtual void setRotation(DtConstDcm geocToLocalRot)`**
  set the rotation matrix that converts from geocentric to local coords
- **`virtual void setLocation(DtConstVector Location)`**
  Set/Get the location in local coordinates
- **`virtual DtConstVector location() const`**
- **`virtual void setVelocity(DtConstVector Velocity)`**
  Set/Get the velocity
- **`virtual DtConstVector velocity() const`**
- **`virtual void setOrientation(const DtTaitBryan& Orient)`**
  Set/Get the orientation
- **`virtual const DtTaitBryan& orientation() const`**
- **`virtual DtConstDcm bodyToLocal() const`**
  Get the body to local coordinate matrix. This is just a matrix representation of the euler angles returned by orientation().
- **`virtual void setAcceleration(DtConstVector Acceleration)`**
  Set/Get the acceleration
- **`virtual DtConstVector acceleration() const`**
- **`virtual void setRotationalVelocity(DtConstVector BodyRates)`**
  Set/Get the rotational velocity
- **`virtual DtConstVector rotationalVelocity() const`**
- **`DtCartesianView* self() const`**
  Cast off const-ness for trivial internal change. Non-virtual since can't override on return type.
- **`DtBaseEntityStateRepository* myESR`**
- **`DtCoordTransform myGeocToLocal`**
- **`DtCoordTransform myLocalToGeoc`**
- **`DtVector myLocalLoc`**
  used as return values by inspectors. Do not necessarily hold current state.
- **`DtVector myLocalVel`**
- **`DtVector myLocalAccel`**
- **`DtTaitBryan myLocalOrient`**
- **`DtDcm myBodyToLocal`**
- **`inline DtCartesianView* self() const`**

Direct child classes:

DtTopoView

In file baseentitysr.h:

class DtBaseEntityStateRepository: public DtStateRepository

class DtBaseEntityStateRepository: Instances of DtBaseEntityStateRepository are used to maintain basic state information needed by all kinds of entities

---

Inheritance:

DtBaseEntityStateRepository - DtStateRepository

Public Methods

	DtBaseEntityStateRepository () *default constructor*
virtual	**~DtBaseEntityStateRepository** () *destructor*
	DtBaseEntityStateRepository (const DtBaseEntityStateRepository& orig) *copy constructor*
DtBaseEntityStateRepository&	**operator=** (const DtBaseEntityStateRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
void	**setTimeStampType** (DtTimeStampType type) *Set/Get flag indicating whether we're using absolute timestamps for dead reckoning*
DtTimeStampType	**timeStampType** () const
virtual void	**setLocation** (DtConstVector location, DtTime timeNow = DtGetSimTime()) *Set/Get the location of the center of mass in world coordinates*
virtual DtConstVector	**location** (DtTime timeNow = DtGetSimTime()) const
virtual DtConstVector	**lastSetLocation** () const *Return the value last passed to setLocation, regardless of whether we have an approximator*
virtual void	**setVelocity** (DtConstVector Velocity, DtTime timeNow = DtGetSimTime()) *Set/Get the velocity*
virtual DtConstVector	**velocity** (DtTime timeNow = DtGetSimTime()) const
virtual DtConstVector	**lastSetVelocity** () const *Return the value last passed to setVelocity, regardless of whether we have an approximator*
virtual void	**setAcceleration** (DtConstVector Acceleration, DtTime timeNow = DtGetSimTime()) *Set/Get the acceleration*

virtual DtConstVector	**acceleration** (DtTime timeNow = DtGetSimTime()) const
virtual DtConstVector	**lastSetAcceleration** () const *Return the value last passed to setAcceleration, regardless of whether we have an approximator*
virtual void	**setOrientation** (const DtTaitBryan& orient, DtTime timeNow = DtGetSimTime()) *Set/Get the orientation*
virtual const DtTaitBryan&	**orientation** (DtTime timeNow = DtGetSimTime()) const
virtual const DtTaitBryan&	**lastSetOrientation** () const *Return the value last passed to setOrientation, regardless of whether we have an approximator*
virtual void	**setRotationalVelocity** (DtConstVector BodyRates, DtTime timeNow = DtGetSimTime()) *Set/Get the rotational velocity*
virtual DtConstVector	**rotationalVelocity** (DtTime timeNow = DtGetSimTime()) const
virtual DtConstVector	**lastSetRotationalVelocity** () const *Return the value last passed to setRotationalVelocity, regardless of whether we have an approximator*
virtual DtConstDcm	**bodyToGeoc** (DtTime timeNow = DtGetSimTime()) const *Get the body to geocentric coordinate matrix*
virtual void	**setAlgorithm** (DtDeadReckonTypes alg) *Set/Get the dead reckoning algorithm*
virtual DtDeadReckonTypes	**algorithm** () const
virtual void	**setEntityId** (const DtEntityIdentifier& id) *Set/Get the entity's identifier*
virtual const DtEntityIdentifier&	**entityId** () const
virtual void	**setEntityType** (const DtEntityType& type) *Set/Get the entity type*
virtual const DtEntityType&	**entityType** () const
virtual void	**setFederateId** (const DtFederateId& id) *Set/Get the federate ID (part of entity ID)*
virtual const DtFederateId&	**federateId** () const
virtual void	**setFrozen** (DtBoolean onOff) *Set/Get whether frozen or not*
virtual DtBoolean	**frozen** () const
virtual void	**setApproximator** (DtApproximator* approx) *Get and set the approximator used to provide approximated position and orientation information for the entity*
virtual DtApproximator*	**approximator** () const
virtual void	**useDeadReckoner** (DtDeadReckoner* dr) *Tells the ESR to use dr as its approximator, and to notify it when DR algorithm changes*
virtual void	**useDeadReckoner** () *Create and use a new (private) dead-reckoner as our approximator*
virtual DtDeadReckoner*	**usingDeadReckoner** () *If the ESR is using a dead reckoner, returns it*
virtual void	**useSmoother** (DtSmoother* sm) *Tells the ESR to use sm as its approximator, and to notify it when DR algorithm changes*

virtual void	**useSmoother** () *Create and use a new (private) smoother as our approximator*
virtual DtSmoother*	**usingSmoother** () *If the ESR is using a smoother, returns it*
virtual DtThresholder*	**thresholder** () *Returns the SR's thresholder, which contains thresholds needed for encoding*
virtual const DtThresholder&	**thresholder** () const *Const version*
virtual void	**printData** () const *Print the state data*
virtual void	**setTimeForSets** (DtTime time)
inline const DtEntityIdentifier&	**entityId** () const
inline const DtEntityType&	**entityType** () const
inline const DtFederateId&	**federateId** () const
inline DtConstVector	**lastSetAcceleration** ( ) const
inline DtConstVector	**lastSetLocation** () const
inline const DtTaitBryan&	**lastSetOrientation** ( ) const
inline DtConstVector	**lastSetRotationalVelocity** ( ) const
inline DtConstVector	**lastSetVelocity** () const
inline void	**setEntityId** (const DtEntityIdentifier& id)
inline void	**setEntityType** (const DtEntityType& type)
inline void	**setFederateId** (const DtFederateId& id)
inline void	**setTimeStampType** (DtTimeStampType type)
inline DtThresholder*	**thresholder** ()
inline const DtThresholder&	**thresholder** () const
inline DtTimeStampType	**timeStampType** () const

Protected Fields

DtApproximator*	**myApproximator** *holds what's currently being used as our approximator*
DtDeadReckoner*	**myDeadReckoner** *holds what was most recently passed to useDeadReckoner*
DtSmoother*	**mySmoother** *holds what was most recently passed to useSmoother*
DtDeadReckoner*	**myDfltDeadReckoner** *Created and deleted by this ESR if needed*
DtSmoother*	**myDfltSmoother**
DtVector	**myLocation**
DtVector	**myVelocity**
DtVector	**myAcceleration**

- DtTaitBryan **myOrientation**
- DtVector **myRotationalVelocity**
- DtDeadReckonTypes **myAlgorithm**
- DtDcm **myBodyToGeoc**
- DtBoolean **myBodyToGeocDirty**
- DtTimeStampType **myTimeStampType**
- DtEntityIdentifier **myEntityId**
- DtFederateId **myFederateId**
- DtEntityType **myEntityType**
- DtBoolean **myFrozen**
- DtThresholder **myThresholder**
- DtTime **myTimeForSets**

Protected Methods

- DtBaseEntityStateRepository* **self** () const
  Cast off const-ness for trivial internal change
- virtual DtTime **timeForSets** () const

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

Documentation

class DtBaseEntityStateRepository: Instances of DtBaseEntityStateRepository are used to maintain basic state information needed by all kinds of entities

- **`DtBaseEntityStateRepository()`**

  default constructor

- **`virtual ~DtBaseEntityStateRepository()`**

  destructor

- **`DtBaseEntityStateRepository(const DtBaseEntityStateRepository& orig)`**

  copy constructor

- **`DtBaseEntityStateRepository& operator=( const DtBaseEntityStateRepository& orig)`**

  assignment operator

- **`virtual DtStateRepository* clone() const`**

  return an empty object of the same type as this

- **`void setTimeStampType(DtTimeStampType type)`**

  Set/Get flag indicating whether we're using absolute timestamps for dead reckoning. This flag will be set to DtAbsolute by the decoders only if incoming state messages indicate that the sender is using absolute time stamps, AND our local exercise connection indicates that we are using absolute timestamping.

- **`DtTimeStampType timeStampType() const`**

- **`virtual void setLocation(DtConstVector location, DtTime timeNow = DtGetSimTime())`**

  Set/Get the location of the center of mass in world coordinates

- **`virtual DtConstVector location(DtTime timeNow = DtGetSimTime()) const`**

- **`virtual DtConstVector lastSetLocation() const`**

  Return the value last passed to setLocation, regardless of whether we have an approximator

- **`virtual void setVelocity(DtConstVector Velocity, DtTime timeNow = DtGetSimTime())`**

  Set/Get the velocity

- **`virtual DtConstVector velocity( DtTime timeNow = DtGetSimTime()) const`**

- **`virtual DtConstVector lastSetVelocity() const`**

  Return the value last passed to setVelocity, regardless of whether we have an approximator

- **`virtual void setAcceleration(DtConstVector Acceleration, DtTime timeNow = DtGetSimTime())`**

  Set/Get the acceleration

- **`virtual DtConstVector acceleration( DtTime timeNow = DtGetSimTime()) const`**

virtual DtConstVector lastSetAcceleration() const

Return the value last passed to setAcceleration, regardless of whether we have an approximator

virtual void setOrientation(const DtTaitBryan& orient, DtTime timeNow = DtGetSimTime())

Set/Get the orientation

virtual const DtTaitBryan& orientation(DtTime timeNow = DtGetSimTime()) const

virtual const DtTaitBryan& lastSetOrientation() const

Return the value last passed to setOrientation, regardless of whether we have an approximator

virtual void setRotationalVelocity(DtConstVector BodyRates, DtTime timeNow = DtGetSimTime())

Set/Get the rotational velocity

virtual DtConstVector rotationalVelocity(DtTime timeNow = DtGetSimTime()) const

virtual DtConstVector lastSetRotationalVelocity() const

Return the value last passed to setRotationalVelocity, regardless of whether we have an approximator

virtual DtConstDcm bodyToGeoc(DtTime timeNow = DtGetSimTime()) const

Get the body to geocentric coordinate matrix

virtual void setAlgorithm(DtDeadReckonTypes alg)

Set/Get the dead reckoning algorithm

virtual DtDeadReckonTypes algorithm() const

virtual void setEntityId(const DtEntityIdentifier& id)

Set/Get the entity's identifier

virtual const DtEntityIdentifier& entityId() const

virtual void setEntityType(const DtEntityType& type)

Set/Get the entity type

virtual const DtEntityType& entityType() const

virtual void setFederateId(const DtFederateId& id)

Set/Get the federate ID (part of entity ID)

virtual const DtFederateId& federateId() const

virtual void setFrozen(DtBoolean onOff)

Set/Get whether frozen or not

virtual DtBoolean frozen() const

virtual void setApproximator(DtApproximator* approx)

Get and set the approximator used to provide approximated position and orientation information for the entity. If your approximator is a DtDeadReckoner, use useDeadReckoner instead of setApproximator. If it's a smoother, use useSmoother. Caller of setApproximator has the responsibility to delete approx after the StateRep no longer needs it.

virtual DtApproximator* approximator() const

- **virtual void useDeadReckoner(DtDeadReckoner* dr)**

  Tells the ESR to use dr as its approximator, and to notify it when DR algorithm changes. Caller is responsible for deleting dr, but only after the StateRep no longer needs it.

- **virtual void useDeadReckoner()**

  Create and use a new (private) dead-reckoner as our approximator

- **virtual DtDeadReckoner* usingDeadReckoner()**

  If the ESR is using a dead reckoner, returns it. Else returns NULL.

- **virtual void useSmoother(DtSmoother* sm)**

  Tells the ESR to use sm as its approximator, and to notify it when DR algorithm changes. Caller is responsible for deleting sm, but only after the StateRep no longer needs it.

- **virtual void useSmoother()**

  Create and use a new (private) smoother as our approximator

- **virtual DtSmoother* usingSmoother()**

  If the ESR is using a smoother, returns it. Else returns NULL.

- **virtual DtThresholder* thresholder()**

  Returns the SR's thresholder, which contains thresholds needed for encoding

- **virtual const DtThresholder& thresholder() const**

  Const version

- **virtual void printData() const**

  Print the state data

- **DtBaseEntityStateRepository* self() const**

  Cast off const-ness for trivial internal change. Non-virtual since can't override on return type.

- **DtApproximator* myApproximator**

  holds what's currently being used as our approximator

- **DtDeadReckoner* myDeadReckoner**

  holds what was most recently passed to useDeadReckoner

- **DtSmoother* mySmoother**

  holds what was most recently passed to useSmoother

- **DtDeadReckoner* myDfltDeadReckoner**

  Created and deleted by this ESR if needed

- **DtSmoother* myDfltSmoother**

- **DtVector myLocation**

- **DtVector myVelocity**

- **DtVector myAcceleration**

- **DtTaitBryan myOrientation**

- DtVector myRotationalVelocity
- DtDeadReckonTypes myAlgorithm
- DtDcm myBodyToGeoc
- DtBoolean myBodyToGeocDirty
- DtTimeStampType myTimeStampType
- DtEntityIdentifier myEntityId
- DtFederateId myFederateId
- DtEntityType myEntityType
- DtBoolean myFrozen
- DtThresholder myThresholder
- virtual void setTimeForSets(DtTime time)
- virtual DtTime timeForSets() const
- DtTime myTimeForSets
- inline const DtEntityIdentifier& entityId() const
- inline const DtEntityType& entityType() const
- inline const DtFederateId& federateId() const
- inline DtConstVector lastSetAcceleration( ) const
- inline DtConstVector lastSetLocation() const
- inline const DtTaitBryan& lastSetOrientation( ) const
- inline DtConstVector lastSetRotationalVelocity( ) const
- inline DtConstVector lastSetVelocity() const
- inline void setEntityId(const DtEntityIdentifier& id)
- inline void setEntityType(const DtEntityType& type)
- inline void setFederateId(const DtFederateId& id)
- inline void setTimeStampType(DtTimeStampType type)
- inline DtThresholder* thresholder()
- inline const DtThresholder& thresholder() const
- inline DtTimeStampType timeStampType() const

Direct child classes:

DtEntityStateRepository

DtAggregateStateRepository

Alphabetic index *Hierarchy of classes*

In file hoststructs.h:

class DtFederateId

Public Methods

	DtFederateId ()
	DtFederateId (const char* str)
	DtFederateId (int site, int app)
	DtFederateId (const DtNetFederateId& netId)
const DtFederateId&	**operator=** (const DtFederateId &netId)
const DtNetFederateId&	**operator=** (const DtNetFederateId &netId)
int	**operator==** (const DtFederateId &otherGuy) const
int	**operator!=** (const DtFederateId &otherGuy) const
void	**init** (int site, int app)
	operator DtNetFederateId () const
int	**siteId** () const
void	**setSiteId** (int siteId)
int	**applicationId** () const
void	**setApplicationId** (int id)
char*	**string** () const
static const DtFederateId&	**nullId** ()
inline	**DtFederateId** ()
inline	**DtFederateId** (int site, int app)
inline	**DtFederateId** (const DtNetFederateId& netId)
inline int	**applicationId** () const
inline void	**init** (int site, int app)
inline	**operator DtNetFederateId** () const
inline int	**operator!=** (const DtFederateId &otherGuy) const
inline const DtFederateId&	**operator=** (const DtFederateId &id)
inline const DtNetFederateId&	**operator=** (const DtNetFederateId &netId)
inline int	**operator==** (const DtFederateId &otherGuy) const
inline void	**setApplicationId** (int id)

- inline void **setSiteId** (int siteId)
- inline int **siteId** () const

Protected Fields

- int **mySiteId**
- int **myAppId**

Documentation

- `DtFederateId()`
- `DtFederateId(const char* str)`
- `DtFederateId(int site, int app)`
- `DtFederateId(const DtNetFederateId& netId)`
- `const DtFederateId& operator=(const DtFederateId &netId)`
- `const DtNetFederateId& operator=(const DtNetFederateId &netId)`
- `int operator==(const DtFederateId &otherGuy) const`
- `int operator!=(const DtFederateId &otherGuy) const`
- `void init(int site, int app)`
- `operator DtNetFederateId () const`
- `int siteId() const`
- `void setSiteId(int siteId)`
- `int applicationId() const`
- `void setApplicationId(int id)`
- `char* string() const`
- `static const DtFederateId& nullId()`
- `int mySiteId`
- `int myAppId`
- `inline DtFederateId()`
- `inline DtFederateId(int site, int app)`
- `inline DtFederateId(const DtNetFederateId& netId)`
- `inline int applicationId() const`

- `inline void init(int site, int app)`
- `inline operator DtNetFederateId () const`
- `inline int operator!=(const DtFederateId &otherGuy) const`
- `inline const DtFederateId& operator=(const DtFederateId &id)`
- `inline const DtNetFederateId& operator=( const DtNetFederateId &netId)`
- `inline int operator==(const DtFederateId &otherGuy) const`
- `inline void setApplicationId(int id)`
- `inline void setSiteId(int siteId)`
- `inline int siteId() const`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file thresholder.h:

class DtThresholder

class DtThresholder: Instances of DtThresholder store thresholding information used to determine what attributes of an entity need to be updated

---

Public Methods

	DtThresholder () *default constructor*
virtual	**~DtThresholder** () *destructor*
	DtThresholder (const DtThresholder& orig) *copy constructor*
DtThresholder&	**operator=** (const DtThresholder& orig) *assignment operator*
virtual void	**setRotationThreshold** (DtReal thresh) *Set/Get rotation threshold in radians*
virtual double	**rotationThreshold** () const
virtual double	**rotationTraceThreshold** () const *Returns the "RotationTrace" threshold*
virtual void	**setTranslationThreshold** (DtReal thresh) *Set/Get translation threshold in meters*
virtual double	**translationThreshold** () const
virtual void	**setAggDimensionThreshold** (DtReal thresh) *Set/Get aggregate dimension threshold*
virtual double	**aggDimensionThreshold** () const
virtual DtArtPartThresholder*	**artPartThresholder** () *Returns a pointer to an object that maintains thresholds for articulated parts*
virtual const DtArtPartThresholder&	**artPartThresholder** () const
static void	**setDfltTranslationThreshold** (DtReal thresh) *Set/Get default translation threshold in radians*
static double	**dfltTranslationThreshold** ()
static void	**setDfltRotationThreshold** (DtReal thresh) *Set/Get default rotation threshold in radians*
static double	**dfltRotationThreshold** ()
static double	**dfltRotationTraceThreshold** () *Returns the "RotationTrace" threshold*
static void	**setDfltAggDimensionThreshold** (DtReal thresh) *Set/Get aggregate dimension threshold*
static double	**dfltAggDimensionThreshold** ()
inline DtArtPartThresholder*	**artPartThresholder** ()

- inline const DtArtPartThresholder& **artPartThresholder** () const
- inline double **dfltAggDimensionThreshold** ()
- inline double **dfltRotationThreshold** ()
- inline double **dfltRotationTraceThreshold** ()
- inline double **dfltTranslationThreshold** ()
- inline void **setDfltAggDimensionThreshold** (DtReal thresh)
- inline void **setDfltRotationThreshold** (DtReal thresh)
- inline void **setDfltTranslationThreshold** (DtReal thresh)

Protected Fields

- double **myTransThresh**
- double **myRotThresh**
- double **myTraceRotThresh**
- double **myAggDimThresh**
- DtBoolean **myUsingDfltTrans**
- DtBoolean **myUsingDfltRot**
- DtBoolean **myUsingDfltAggDim**
- DtArtPartThresholder **myArtPartThresh**
- static double **theDfltTransThresh**
- static double **theDfltRotThresh**
- static double **theDfltTraceRotThresh**
- static double **theDfltAggDimThresh**

Documentation

class DtThresholder: Instances of DtThresholder store thresholding information used to determine what attributes of an entity need to be updated

- **`DtThresholder()`**
  default constructor
- **`virtual ~DtThresholder()`**
  destructor
- **`DtThresholder(const DtThresholder& orig)`**
  copy constructor
- **`DtThresholder& operator=(const DtThresholder& orig)`**

assignment operator

- **virtual void setRotationThreshold(DtReal thresh)**

  Set/Get rotation threshold in radians. Negative number means use default.

- **virtual double rotationThreshold() const**

- **virtual double rotationTraceThreshold() const**

  Returns the "RotationTrace" threshold

- **virtual void setTranslationThreshold(DtReal thresh)**

  Set/Get translation threshold in meters. Negative number means use default.

- **virtual double translationThreshold() const**

- **virtual void setAggDimensionThreshold(DtReal thresh)**

  Set/Get aggregate dimension threshold. Thresh is a number between 0 and 1, indicating the percentage change that an aggregate dimension can undergo before causing an update to be generated. Negative number means use default. Applies only to aggregate entities.

- **virtual double aggDimensionThreshold() const**

- **virtual DtArtPartThresholder* artPartThresholder()**

  Returns a pointer to an object that maintains thresholds for articulated parts

- **virtual const DtArtPartThresholder& artPartThresholder() const**

- **static void setDfltTranslationThreshold(DtReal thresh)**

  Set/Get default translation threshold in radians.

- **static double dfltTranslationThreshold()**

- **static void setDfltRotationThreshold(DtReal thresh)**

  Set/Get default rotation threshold in radians.

- **static double dfltRotationThreshold()**

- **static double dfltRotationTraceThreshold()**

  Returns the "RotationTrace" threshold

- **static void setDfltAggDimensionThreshold(DtReal thresh)**

  Set/Get aggregate dimension threshold. Thresh is a number between 0 and 1, indicating the percentage change that an aggregate dimension can undergo before causing an update to be generated. Applies only to aggregate entities.

- **static double dfltAggDimensionThreshold()**

- **double myTransThresh**

- **double myRotThresh**

- **double myTraceRotThresh**

- **double myAggDimThresh**

- **DtBoolean myUsingDfltTrans**

- **DtBoolean myUsingDfltRot**

- DtBoolean myUsingDfltAggDim
- DtArtPartThresholder myArtPartThresh
- static double theDfltTransThresh
- static double theDfltRotThresh
- static double theDfltTraceRotThresh
- static double theDfltAggDimThresh
- inline DtArtPartThresholder* artPartThresholder()
- inline const DtArtPartThresholder& artPartThresholder() const
- inline double dfltAggDimensionThreshold()
- inline double dfltRotationThreshold()
- inline double dfltRotationTraceThreshold()
- inline double dfltTranslationThreshold()
- inline void setDfltAggDimensionThreshold(DtReal thresh)
- inline void setDfltRotationThreshold(DtReal thresh)
- inline void setDfltTranslationThreshold(DtReal thresh)

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file entitysr.h:

class DtEntityStateRepository: public DtBaseEntityStateRepository

class DtEntityStateRepository: Instances of DtEntityStateRepository are used to maintain entity state

---

Inheritance:

DtEntityStateRepository - DtBaseEntityStateRepository - DtStateRepository

---

Public Methods

	DtEntityStateRepository () *default constructor*
virtual	**~DtEntityStateRepository** () *destructor*
	DtEntityStateRepository (const DtEntityStateRepository& orig) *copy constructor*
DtEntityStateRepository&	**operator=** (const DtEntityStateRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
virtual void	**setForceId** (DtForceType force) *Set/Get the force type*
virtual DtForceType	**forceId** () const
virtual void	**setGuise** (const DtEntityType& type) *Set/Get the alternate entity type or guise*
virtual const DtEntityType&	**guise** () const
virtual void	**setFrozen** (DtBoolean onOff) *Set/Get whether frozen or not*
virtual DtBoolean	**frozen** () const
virtual void	**setMarkingCharSet** (int charSet) *Set/Get the character set used for the marking*
virtual int	**markingCharSet** () const
virtual void	**setMarkingText** (const char* text) *Set/Get the marking text*
virtual const char*	**markingText** () const

virtual void **setPowerPlantOn** (DtBoolean onOff)
Set/Get whether entity's power plant is on or not

virtual DtBoolean **powerPlantOn** () const

virtual void **setDamageState** (DtDamageState s)
Set/Get the damage state

virtual DtDamageState **damageState** () const

virtual void **setEngineSmokeOn** (DtBoolean onOff)
Set/Get whether engine smoke is on or not

virtual DtBoolean **engineSmokeOn** () const

virtual void **setFlamesPresent** (DtBoolean onOff)
Set/Get whether flames are present or not

virtual DtBoolean **flamesPresent** () const

virtual void **setHasFuelSupplyCap** (DtBoolean onOff)
Set/Get whether entity has fuel supplying capability

virtual DtBoolean **hasFuelSupplyCap** () const

virtual void **setHasRecoveryCap** (DtBoolean onOff)
Set/Get whether entity has recovery capability

virtual DtBoolean **hasRecoveryCap** () const

virtual void **setHasRepairCap** (DtBoolean onOff)
Set/Get whether entity has repair capability

virtual DtBoolean **hasRepairCap** () const

virtual void **setHatchState** (DtHatchState s)
Set/Get the entity's hatch state

virtual DtHatchState **hatchState** () const

virtual void **setImmobilized** (DtBoolean onOff)
Set/Get whether the entity is immobilized or not

virtual DtBoolean **immobilized** () const

virtual void **setLifeformState** (DtLifeformState s)
Set/Get entity life form state

virtual DtLifeformState **lifeformState** () const

virtual void **setLightState** (DtLightState s)
Set/Get light state of entity

virtual DtLightState **lightState** () const

virtual void **setDensity** (DtDensityType dens)
Set/Get light state of entity

virtual DtDensityType **density** () const

virtual void **setRampDeployed** (DtBoolean onOff)
Set/Get whether entity has ramp deployed or not

virtual DtBoolean **rampDeployed** () const

virtual void **setSmokePlumePresent** (DtBoolean onOff)
Set/Get whether a smoke plume is present or not

virtual DtBoolean **smokePlumePresent** () const

virtual void **setTentDeployed** (DtBoolean onOff)
Set/Get whether a tent is deployed or not

virtual DtBoolean **tentDeployed** () const

virtual void **setTrailState** (DtTrailState s)
Set/Get trailing effect state

virtual DtTrailState **trailState** () const

virtual void **setCamouflageType** (DtCamouflageType type)
Set/Get camouflage type

virtual DtCamouflageType **camouflageType** () const

virtual void **setFirePowerDisabled** (DtBoolean onOff)
Set/Get whether entity is fire power disabled or not

virtual DtBoolean **firePowerDisabled** () const

virtual void **setIsConcealed** (DtBoolean onOff)
Set/Get whether entity is concealed or not

virtual DtBoolean **isConcealed** () const

virtual void **setAfterburnerOn** (DtBoolean onOff)
Set/Get whether entity has after burner on or not

virtual DtBoolean **afterburnerOn** () const

virtual void **setHasAmmunitionSupplyCap** (DtBoolean onOff)
Set/Get whether entity has ammunition supply capability or not

virtual DtBoolean **hasAmmunitionSupplyCap** () const

virtual void **setLauncherRaised** (DtBoolean onOff)
Set/Get whether entity has launcher raised or not

virtual DtBoolean **launcherRaised** () const

virtual void **setLauncherFlashPresent** (DtBoolean onOff)
Set/Get whether launcher flash is present or not

virtual DtBoolean **launcherFlashPresent** () const

virtual void **setPrimaryWeaponState** (DtWeaponState s)
Set/Get primary weapon deployment state

virtual DtWeaponState **primaryWeaponState** () const

virtual void **setSecondaryWeaponState** (DtWeaponState s)
Set/Get secondary weapon deployment state

virtual DtWeaponState **secondaryWeaponState** () const

virtual void **initArtParts** (const DtArtPartSpec* specList)
initialize articulated parts

virtual const DtArtPartList* **artPartList** () const
get the articulated parts list

virtual DtArtPartList* **artPartList** ()

virtual const DtAttachedPartList* **attPartList** () const
get the attached parts list

virtual DtAttachedPartList* **attPartList** ()

virtual void **printData** () const
Print the state data

- static DtEntityStateRepository* **create** ()
- inline DtU32 **appearance** () const
- inline const DtArtPartList* **artPartList** () const
- inline DtArtPartList* **artPartList** ()
- inline const DtAttachedPartList* **attPartList** () const
- inline DtAttachedPartList* **attPartList** ()
- inline DtU32 **capabilities** () const
- inline DtForceType **forceId** () const
- inline const DtEntityType& **guise** () const
- inline void **internalSetMarkingText** (const char* text)
- inline int **markingCharSet** () const
- inline const char* **markingText** () const
- inline DtEntityStateRepository* **self** () const
- inline void **setAppearance** (DtU32 app)
- inline void **setCapabilities** (DtU32 cap)
- inline void **setForceId** (DtForceType force)
- inline void **setGuise** (const DtEntityType& type)
- inline void **setMarkingCharSet** (int charSet)
- inline void **setMarkingText** (const char* text)

Protected Fields

- DtArtPartList **myArtPartList**
- DtAttachedPartList **myAttPartList**
- DtForceType **myForceId**
- DtEntityType **myGuise**
- DtU32 **myAppearance**
- int **myMarkingCharSet**
- char* **myMarkingText**
  myMarkingText is an array of theMaxMarkingSize
- DtU32 **myCapabilities**
- static const int **theMaxMarkingSize**

Protected Methods

virtual void	**setAppearance** (DtU32 app) *Set/Get the appearance field*
virtual DtU32	**appearance** () const
virtual void	**setAppearanceAttribute** (const DtAppearAttribute attr, const DtAppearAttrVal val) *Set/Get an appearance attribute*
virtual DtAppearAttrVal	**appearanceAttribute** (const DtAppearAttribute attr) const
virtual void	**setCapabilities** (DtU32 cap) *Set/Get the capabilities field*
virtual DtU32	**capabilities** () const
DtEntityStateRepository*	**self** () const *Cast off const-ness for trivial internal change*
void	**internalSetMarkingText** (const char* text) *Set the marking text*

Inherited from DtBaseEntityStateRepository:

Public Methods

- void **setTimeStampType**(DtTimeStampType type)
- DtTimeStampType **timeStampType**() const
- virtual void **setLocation**(DtConstVector location, DtTime timeNow = DtGetSimTime())
- virtual DtConstVector **location**(DtTime timeNow = DtGetSimTime()) const
- virtual DtConstVector **lastSetLocation**() const
- virtual void **setVelocity**(DtConstVector Velocity, DtTime timeNow = DtGetSimTime())
- virtual DtConstVector **velocity**(DtTime timeNow = DtGetSimTime()) const
- virtual DtConstVector **lastSetVelocity**() const
- virtual void **setAcceleration**(DtConstVector Acceleration, DtTime timeNow = DtGetSimTime())
- virtual DtConstVector **acceleration**(DtTime timeNow = DtGetSimTime()) const
- virtual DtConstVector **lastSetAcceleration**() const
- virtual void **setOrientation**(const DtTaitBryan& orient, DtTime timeNow = DtGetSimTime())
- virtual const DtTaitBryan& **orientation**(DtTime timeNow = DtGetSimTime()) const
- virtual const DtTaitBryan& **lastSetOrientation**() const
- virtual void **setRotationalVelocity**(DtConstVector BodyRates, DtTime timeNow = DtGetSimTime())
- virtual DtConstVector **rotationalVelocity**(DtTime timeNow = DtGetSimTime()) const
- virtual DtConstVector **lastSetRotationalVelocity**() const

- virtual DtConstDcm **bodyToGeoc**(DtTime timeNow = DtGetSimTime()) const
- virtual void **setAlgorithm**(DtDeadReckonTypes alg)
- virtual DtDeadReckonTypes **algorithm**() const
- virtual void **setEntityId**(const DtEntityIdentifier& id)
- virtual const DtEntityIdentifier& **entityId**() const
- virtual void **setEntityType**(const DtEntityType& type)
- virtual const DtEntityType& **entityType**() const
- virtual void **setFederateId**(const DtFederateId& id)
- virtual const DtFederateId& **federateId**() const
- virtual void **setApproximator**(DtApproximator* approx)
- virtual DtApproximator* **approximator**() const
- virtual void **useDeadReckoner**(DtDeadReckoner* dr)
- virtual void **useDeadReckoner**()
- virtual DtDeadReckoner* **usingDeadReckoner**()
- virtual void **useSmoother**(DtSmoother* sm)
- virtual void **useSmoother**()
- virtual DtSmoother* **usingSmoother**()
- virtual DtThresholder* **thresholder**()
- virtual const DtThresholder& **thresholder**() const
- virtual void **setTimeForSets**(DtTime time)
- inline const DtEntityIdentifier& **entityId**() const
- inline const DtEntityType& **entityType**() const
- inline const DtFederateId& **federateId**() const
- inline DtConstVector **lastSetAcceleration**( ) const
- inline DtConstVector **lastSetLocation**() const
- inline const DtTaitBryan& **lastSetOrientation**( ) const
- inline DtConstVector **lastSetRotationalVelocity**( ) const
- inline DtConstVector **lastSetVelocity**() const
- inline void **setEntityId**(const DtEntityIdentifier& id)
- inline void **setEntityType**(const DtEntityType& type)

- inline void **setFederateId**(const DtFederateId& id)
- inline void **setTimeStampType**(DtTimeStampType type)
- inline DtThresholder* **thresholder**()
- inline const DtThresholder& **thresholder**() const
- inline DtTimeStampType **timeStampType**() const

Protected Fields

- DtApproximator* **myApproximator**
- DtDeadReckoner* **myDeadReckoner**
- DtSmoother* **mySmoother**
- DtDeadReckoner* **myDfltDeadReckoner**
- DtSmoother* **myDfltSmoother**
- DtVector **myLocation**
- DtVector **myVelocity**
- DtVector **myAcceleration**
- DtTaitBryan **myOrientation**
- DtVector **myRotationalVelocity**
- DtDeadReckonTypes **myAlgorithm**
- DtDcm **myBodyToGeoc**
- DtBoolean **myBodyToGeocDirty**
- DtTimeStampType **myTimeStampType**
- DtEntityIdentifier **myEntityId**
- DtFederateId **myFederateId**
- DtEntityType **myEntityType**
- DtBoolean **myFrozen**
- DtThresholder **myThresholder**
- DtTime **myTimeForSets**

Protected Methods

- virtual DtTime **timeForSets**() const

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

-

Documentation

class DtEntityStateRepository:

Instances of DtEntityStateRepository are used to maintain entity state. It provides a fat interface which covers various types of entities. As a result, individual fields may not apply depending on the type of entity.

- **`DtEntityStateRepository()`**
  default constructor
- **`virtual ~DtEntityStateRepository()`**
  destructor
- **`DtEntityStateRepository(const DtEntityStateRepository& orig)`**
  copy constructor
- **`DtEntityStateRepository& operator=(const DtEntityStateRepository& orig)`**
  assignment operator
- **`virtual DtStateRepository* clone() const`**
  return an empty object of the same type as this

- **virtual void setForceId(DtForceType force)**

  Set/Get the force type

- **virtual DtForceType forceId() const**

- **virtual void setGuise(const DtEntityType& type)**

  Set/Get the alternate entity type or guise

- **virtual const DtEntityType& guise() const**

- **virtual void setFrozen(DtBoolean onOff)**

  Set/Get whether frozen or not

- **virtual DtBoolean frozen() const**

- **virtual void setMarkingCharSet(int charSet)**

  Set/Get the character set used for the marking. Defaults to 1.

- **virtual int markingCharSet() const**

- **virtual void setMarkingText(const char* text)**

  Set/Get the marking text

- **virtual const char* markingText() const**

- **virtual void setPowerPlantOn(DtBoolean onOff)**

  Set/Get whether entity's power plant is on or not

- **virtual DtBoolean powerPlantOn() const**

- **virtual void setDamageState(DtDamageState s)**

  Set/Get the damage state

- **virtual DtDamageState damageState() const**

- **virtual void setEngineSmokeOn(DtBoolean onOff)**

  Set/Get whether engine smoke is on or not

- **virtual DtBoolean engineSmokeOn() const**

- **virtual void setFlamesPresent(DtBoolean onOff)**

  Set/Get whether flames are present or not

- **virtual DtBoolean flamesPresent() const**

- **virtual void setHasFuelSupplyCap(DtBoolean onOff)**

  Set/Get whether entity has fuel supplying capability

- **virtual DtBoolean hasFuelSupplyCap() const**

- **virtual void setHasRecoveryCap(DtBoolean onOff)**

  Set/Get whether entity has recovery capability

- **virtual DtBoolean hasRecoveryCap() const**

- **virtual void setHasRepairCap(DtBoolean onOff)**

  Set/Get whether entity has repair capability

- **virtual DtBoolean hasRepairCap() const**
- **virtual void setHatchState(DtHatchState s)**

  Set/Get the entity's hatch state

- **virtual DtHatchState hatchState() const**
- **virtual void setImmobilized(DtBoolean onOff)**

  Set/Get whether the entity is immobilized or not

- **virtual DtBoolean immobilized() const**
- **virtual void setLifeformState(DtLifeformState s)**

  Set/Get entity life form state

- **virtual DtLifeformState lifeformState() const**
- **virtual void setLightState(DtLightState s)**

  Set/Get light state of entity

- **virtual DtLightState lightState() const**
- **virtual void setDensity(DtDensityType dens)**

  Set/Get light state of entity. (Environment entities).

- **virtual DtDensityType density() const**
- **virtual void setRampDeployed(DtBoolean onOff)**

  Set/Get whether entity has ramp deployed or not

- **virtual DtBoolean rampDeployed() const**
- **virtual void setSmokePlumePresent(DtBoolean onOff)**

  Set/Get whether a smoke plume is present or not

- **virtual DtBoolean smokePlumePresent() const**
- **virtual void setTentDeployed(DtBoolean onOff)**

  Set/Get whether a tent is deployed or not

- **virtual DtBoolean tentDeployed() const**
- **virtual void setTrailState(DtTrailState s)**

  Set/Get trailing effect state

- **virtual DtTrailState trailState() const**
- **virtual void setCamouflageType(DtCamouflageType type)**

  Set/Get camouflage type

- **virtual DtCamouflageType camouflageType() const**

- **virtual void setFirePowerDisabled(DtBoolean onOff)**
  Set/Get whether entity is fire power disabled or not
- **virtual DtBoolean firePowerDisabled() const**
- **virtual void setIsConcealed(DtBoolean onOff)**
  Set/Get whether entity is concealed or not
- **virtual DtBoolean isConcealed() const**
- **virtual void setAfterburnerOn(DtBoolean onOff)**
  Set/Get whether entity has after burner on or not
- **virtual DtBoolean afterburnerOn() const**
- **virtual void setHasAmmunitionSupplyCap(DtBoolean onOff)**
  Set/Get whether entity has ammunition supply capability or not
- **virtual DtBoolean hasAmmunitionSupplyCap() const**
- **virtual void setLauncherRaised(DtBoolean onOff)**
  Set/Get whether entity has launcher raised or not
- **virtual DtBoolean launcherRaised() const**
- **virtual void setLauncherFlashPresent(DtBoolean onOff)**
  Set/Get whether launcher flash is present or not
- **virtual DtBoolean launcherFlashPresent() const**
- **virtual void setPrimaryWeaponState(DtWeaponState s)**
  Set/Get primary weapon deployment state
- **virtual DtWeaponState primaryWeaponState() const**
- **virtual void setSecondaryWeaponState(DtWeaponState s)**
  Set/Get secondary weapon deployment state
- **virtual DtWeaponState secondaryWeaponState() const**
- **virtual void initArtParts(const DtArtPartSpec* specList)**
  initialize articulated parts
- **virtual const DtArtPartList* artPartList() const**
  get the articulated parts list
- **virtual DtArtPartList* artPartList()**
- **virtual const DtAttachedPartList* attPartList() const**
  get the attached parts list
- **virtual DtAttachedPartList* attPartList()**
- **virtual void setAppearance(DtU32 app)**

Set/Get the appearance field

- **virtual DtU32 appearance() const**
- **virtual void setAppearanceAttribute(const DtAppearAttribute attr, const DtAppearAttrVal val)**

  Set/Get an appearance attribute. attr is the appearance bitfield mask val is the hex number to be or'ed in
- **virtual DtAppearAttrVal appearanceAttribute(const DtAppearAttribute attr) const**
- **virtual void setCapabilities(DtU32 cap)**

  Set/Get the capabilities field
- **virtual DtU32 capabilities() const**
- **virtual void printData() const**

  Print the state data
- **static DtEntityStateRepository* create()**
- **DtEntityStateRepository* self() const**

  Cast off const-ness for trivial internal change. Non-virtual since can't override on return type.
- **void internalSetMarkingText(const char* text)**

  Set the marking text. Non-virtual since meant to be used by constructor.
- **DtArtPartList myArtPartList**
- **DtAttachedPartList myAttPartList**
- **DtForceType myForceId**
- **DtEntityType myGuise**
- **DtU32 myAppearance**
- **int myMarkingCharSet**
- **char* myMarkingText**

  myMarkingText is an array of theMaxMarkingSize. String length is theMaxMarkingSize - 1 (doesn't include NULL).
- **DtU32 myCapabilities**
- **static const int theMaxMarkingSize**
- **inline DtU32 appearance() const**
- **inline const DtArtPartList* artPartList() const**
- **inline DtArtPartList* artPartList()**
- **inline const DtAttachedPartList* attPartList() const**
- **inline DtAttachedPartList* attPartList()**
- **inline DtU32 capabilities() const**
- **inline DtForceType forceId() const**

- `inline const DtEntityType& guise() const`
- `inline void internalSetMarkingText(const char* text)`
- `inline int markingCharSet() const`
- `inline const char* markingText() const`
- `inline DtEntityStateRepository* self() const`
- `inline void setAppearance(DtU32 app)`
- `inline void setCapabilities(DtU32 cap)`
- `inline void setForceId(DtForceType force)`
- `inline void setGuise(const DtEntityType& type)`
- `inline void setMarkingCharSet(int charSet)`
- `inline void setMarkingText(const char* text)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file aggregatesr.h:

class DtAggregateStateRepository: public DtBaseEntityStateRepository

class DtAggregateStateRepository: Instances of DtAggregateStateRepository are used to maintain the state of an aggregate

---

Inheritance:

DtAggregateStateRepository - DtBaseEntityStateRepository - DtStateRepository

---

Public Methods

	DtAggregateStateRepository () *default constructor*
virtual	**~DtAggregateStateRepository** () *destructor*
	DtAggregateStateRepository (const DtAggregateStateRepository& orig) *copy constructor*
DtAggregateStateRepository&	**operator=** (const DtAggregateStateRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
virtual void	**setAggregateID** (const DtEntityIdentifier& id) *Set/Get the aggregate's identifier*
virtual const DtEntityIdentifier&	**aggregateID** () const
virtual void	**setForceId** (DtForceType force) *Set/Get the force type*
virtual DtForceType	**forceId** () const
virtual void	**setAggregateState** (DtAggregateState value) *Set/Get the aggregate state*
virtual DtAggregateState	**aggregateState** () const
virtual void	**setAggregateType** (const DtEntityType& type) *Set/Get the aggregate type*
virtual const DtEntityType&	**aggregateType** () const
virtual void	**setFormation** (DtAggregateFormation value) *Set/Get the formation*

virtual DtAggregateFormation	**formation** () const
virtual void	**setMarkingCharacterSet** (DtCharacterSet value) *Set/Get marking character set*
virtual DtCharacterSet	**markingCharacterSet** () const
virtual void	**setMarkingText** (DtU8* copyFrom) *Set/Get marking text (copies 31 bytes)*
virtual void	**getMarkingText** (DtU8* copyTo) const
virtual const char*	**markingText** () const
virtual void	**setDimensions** (DtConstVector dims) *Set/Get the dimensions*
virtual DtConstVector	**dimensions** () const
const DtGlobalObjectDesignatorList&	**entities** () const *Return the list of published entities in the aggregate*
DtGlobalObjectDesignatorList*	**entities** ()
DtGlobalObjectDesignatorList*	**subAggregates** () *Return the list of published subaggregates in the aggregate*
const DtGlobalObjectDesignatorList&	**subAggregates** () const
virtual DtBoolean	**setNumberOfDIS_Subaggregates** (int newNumberOfAggregates) *Set/Get the number of DIS sub-aggregates*
virtual int	**numberOfDIS_Subaggregates** () const
virtual DtBoolean	**setNumberOfDIS_Entities** (int newNumberOfEntities) *Set/Get the number of DIS entities*
virtual int	**numberOfDIS_Entities** () const
virtual DtBoolean	**setNumberOfSilentSubaggregates** (int newNumberOfAggregates) *Set/Get the number of silent aggregates*
virtual int	**numberOfSilentSubaggregates** () const
virtual DtBoolean	**setNumberOfSilentEntities** (int newNumberOfEntities) *Set/Get the number of silent entities*
virtual int	**numberOfSilentEntities** () const
virtual DtBoolean	**setSubaggregateID** (int index, const DtGlobalObjectDesignator& aggID) *Set/Get a DIS sub-aggregate ID*
virtual DtBoolean	**getSubaggregateID** (int index, DtGlobalObjectDesignator& aggID) const
virtual DtBoolean	**setEntityID** (int index, const DtGlobalObjectDesignator& entID) *Set/Get a DIS entity ID*
virtual DtBoolean	**getEntityID** (int index, DtGlobalObjectDesignator& entID) const
virtual DtBoolean	**setSilentAggregateList** (int index, const DtSilentAggregateList& aggList) *Set/Get a silent aggregate list*
virtual DtBoolean	**getSilentAggregateList** (int index, DtSilentAggregateList& aggList) const
virtual DtBoolean	**setSilentEntityList** (int index, const DtSilentEntityList& entList) *Set/Get a silent entity list*
virtual DtBoolean	**getSilentEntityList** (int index, DtSilentEntityList& entList) const
virtual DtBoolean	**setNumberOfDatumRecords** (int newNumberOfRecords) *Set the number of variable datum records*

virtual int	**numberOfDatumRecords** () const *Get the number of variable datum records*
virtual DtBoolean	**setVariableDatumRecord** (int index, const DtVariableDatumRecord& rec) *Set/Get a variable datum record*
virtual DtBoolean	**getVariableDatumRecord** (int index, DtVariableDatumRecord& rec) const
virtual DtVariableDatumRecord&	**variableDatumRecordRef** (int index) *Get a reference to a variable datum record*
virtual const DtVariableDatumRecord&	**variableDatumRecordConstRef** (int index) const *Get a constant reference to a variable datum record*
virtual void	**printData** () const *Print the state data*
static DtAggregateStateRepository*	**create** ()

Protected Fields

DtForceType	**myForceID**
DtAggregateState	**myAggregateState**
DtAggregateFormation	**myFormation**
DtCharacterSet	**myMarkingCharSet**
char	**myMarkingText** [AGGREGATE_SR_MARKING_TEXT_LENGTH]
DtVector	**myDimensions**
DtGlobalObjectDesignatorList	**myEntities**
DtGlobalObjectDesignatorList	**mySubAggs**
int	**myNumberOfSilentAggregateTypes**
DtSilentAggregateList*	**mySilentAggregates**
int	**myNumberOfSilentEntityTypes**
DtSilentEntityList*	**mySilentEntities**
int	**myNumberOfVariableDatumRecords**
DtVariableDatumRecord*	**myVariableDatumRecords**
DtVariableDatumRecord	**myDumRec**

Protected Methods

DtAggregateStateRepository*	**self** () const *Cast off const-ness for trivial internal change*

Protected

- #define **AGGREGATE_SR_MARKING_TEXT_LENGTH**
  Aggregate state PDU marking is 31 bytes long, RPR marking struct is 11
- #define **AGGREGATE_SR_MARKING_TEXT_LENGTH**

Inherited from DtBaseEntityStateRepository:

Public Methods

- void **setTimeStampType**(DtTimeStampType type)
- DtTimeStampType **timeStampType**() const
- virtual void **setLocation**(DtConstVector location, DtTime timeNow = DtGetSimTime())
- virtual DtConstVector **location**(DtTime timeNow = DtGetSimTime()) const
- virtual DtConstVector **lastSetLocation**() const
- virtual void **setVelocity**(DtConstVector Velocity, DtTime timeNow = DtGetSimTime())
- virtual DtConstVector **velocity**(DtTime timeNow = DtGetSimTime()) const
- virtual DtConstVector **lastSetVelocity**() const
- virtual void **setAcceleration**(DtConstVector Acceleration, DtTime timeNow = DtGetSimTime())
- virtual DtConstVector **acceleration**(DtTime timeNow = DtGetSimTime()) const
- virtual DtConstVector **lastSetAcceleration**() const
- virtual void **setOrientation**(const DtTaitBryan& orient, DtTime timeNow = DtGetSimTime())
- virtual const DtTaitBryan& **orientation**(DtTime timeNow = DtGetSimTime()) const
- virtual const DtTaitBryan& **lastSetOrientation**() const
- virtual void **setRotationalVelocity**(DtConstVector BodyRates, DtTime timeNow = DtGetSimTime())
- virtual DtConstVector **rotationalVelocity**(DtTime timeNow = DtGetSimTime()) const
- virtual DtConstVector **lastSetRotationalVelocity**() const
- virtual DtConstDcm **bodyToGeoc**(DtTime timeNow = DtGetSimTime()) const
- virtual void **setAlgorithm**(DtDeadReckonTypes alg)
- virtual DtDeadReckonTypes **algorithm**() const
- virtual void **setEntityId**(const DtEntityIdentifier& id)
- virtual const DtEntityIdentifier& **entityId**() const
- virtual void **setEntityType**(const DtEntityType& type)

- virtual const DtEntityType& **entityType**() const
- virtual void **setFederateId**(const DtFederateId& id)
- virtual const DtFederateId& **federateId**() const
- virtual void **setFrozen**(DtBoolean onOff)
- virtual DtBoolean **frozen**() const
- virtual void **setApproximator**(DtApproximator* approx)
- virtual DtApproximator* **approximator**() const
- virtual void **useDeadReckoner**(DtDeadReckoner* dr)
- virtual void **useDeadReckoner**()
- virtual DtDeadReckoner* **usingDeadReckoner**()
- virtual void **useSmoother**(DtSmoother* sm)
- virtual void **useSmoother**()
- virtual DtSmoother* **usingSmoother**()
- virtual DtThresholder* **thresholder**()
- virtual const DtThresholder& **thresholder**() const
- virtual void **setTimeForSets**(DtTime time)
- inline const DtEntityIdentifier& **entityId**() const
- inline const DtEntityType& **entityType**() const
- inline const DtFederateId& **federateId**() const
- inline DtConstVector **lastSetAcceleration**( ) const
- inline DtConstVector **lastSetLocation**() const
- inline const DtTaitBryan& **lastSetOrientation**( ) const
- inline DtConstVector **lastSetRotationalVelocity**( ) const
- inline DtConstVector **lastSetVelocity**() const
- inline void **setEntityId**(const DtEntityIdentifier& id)
- inline void **setEntityType**(const DtEntityType& type)
- inline void **setFederateId**(const DtFederateId& id)
- inline void **setTimeStampType**(DtTimeStampType type)
- inline DtThresholder* **thresholder**()
- inline const DtThresholder& **thresholder**() const

- inline DtTimeStampType **timeStampType**() const

Protected Fields

- DtApproximator* **myApproximator**
- DtDeadReckoner* **myDeadReckoner**
- DtSmoother* **mySmoother**
- DtDeadReckoner* **myDfltDeadReckoner**
- DtSmoother* **myDfltSmoother**
- DtVector **myLocation**
- DtVector **myVelocity**
- DtVector **myAcceleration**
- DtTaitBryan **myOrientation**
- DtVector **myRotationalVelocity**
- DtDeadReckonTypes **myAlgorithm**
- DtDcm **myBodyToGeoc**
- DtBoolean **myBodyToGeocDirty**
- DtTimeStampType **myTimeStampType**
- DtEntityIdentifier **myEntityId**
- DtFederateId **myFederateId**
- DtEntityType **myEntityType**
- DtBoolean **myFrozen**
- DtThresholder **myThresholder**
- DtTime **myTimeForSets**

Protected Methods

- virtual DtTime **timeForSets**() const

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

-

Documentation

class DtAggregateStateRepository:

Instances of DtAggregateStateRepository are used to maintain the state of an aggregate. It provides an interface which covers the various features which define the state of an aggregate.

- **`DtAggregateStateRepository()`**
  default constructor
- **`virtual ~DtAggregateStateRepository()`**
  destructor
- **`DtAggregateStateRepository(const DtAggregateStateRepository& orig)`**
  copy constructor
- **`DtAggregateStateRepository& operator=( const DtAggregateStateRepository& orig)`**
  assignment operator
- **`virtual DtStateRepository* clone() const`**
  return an empty object of the same type as this
- **`virtual void setAggregateID(const DtEntityIdentifier& id)`**
  Set/Get the aggregate's identifier. Calls down to base class.
- **`virtual const DtEntityIdentifier& aggregateID() const`**
- **`virtual void setForceId(DtForceType force)`**
  Set/Get the force type

- **virtual DtForceType forceId() const**
- **virtual void setAggregateState(DtAggregateState value)**

  Set/Get the aggregate state
- **virtual DtAggregateState aggregateState() const**
- **virtual void setAggregateType(const DtEntityType& type)**

  Set/Get the aggregate type. Calls down to base class.
- **virtual const DtEntityType& aggregateType() const**
- **virtual void setFormation(DtAggregateFormation value)**

  Set/Get the formation
- **virtual DtAggregateFormation formation() const**
- **virtual void setMarkingCharacterSet(DtCharacterSet value)**

  Set/Get marking character set
- **virtual DtCharacterSet markingCharacterSet() const**
- **virtual void setMarkingText(DtU8* copyFrom)**

  Set/Get marking text (copies 31 bytes)
- **virtual void getMarkingText(DtU8* copyTo) const**
- **virtual const char* markingText() const**
- **virtual void setDimensions(DtConstVector dims)**

  Set/Get the dimensions
- **virtual DtConstVector dimensions() const**
- **const DtGlobalObjectDesignatorList& entities() const**

  Return the list of published entities in the aggregate
- **DtGlobalObjectDesignatorList* entities()**
- **DtGlobalObjectDesignatorList* subAggregates()**

  Return the list of published subaggregates in the aggregate
- **const DtGlobalObjectDesignatorList& subAggregates() const**
- **virtual DtBoolean setNumberOfDIS_Subaggregates(int newNumberOfAggregates)**

  Set/Get the number of DIS sub-aggregates. Set returns DtFalse if the requested number is too large. Changing this number will erase all previous data. Can be done through subAggregates() instead.
- **virtual int numberOfDIS_Subaggregates() const**
- **virtual DtBoolean setNumberOfDIS_Entities(int newNumberOfEntities)**

  Set/Get the number of DIS entities. Set returns DtFalse if the requested number is too large. Changing this number will erase all previous data. Can be done through entities() instead.
- **virtual int numberOfDIS_Entities() const**

- **virtual DtBoolean setNumberOfSilentSubaggregates(int newNumberOfAggregates)**

  Set/Get the number of silent aggregates. Set returns DtFalse if the requested number is too large. Changing this number will erase all previous data.

- **virtual int numberOfSilentSubaggregates() const**

- **virtual DtBoolean setNumberOfSilentEntities(int newNumberOfEntities)**

  Set/Get the number of silent entities. Set returns DtFalse if the requested number is too large. Changing this number will erase all previous data.

- **virtual int numberOfSilentEntities() const**

- **virtual DtBoolean setSubaggregateID(int index, const DtGlobalObjectDesignator& aggID)**

  Set/Get a DIS sub-aggregate ID. Returns DtFalse if the index is invalid. Can be done through subAggregates() instead.

- **virtual DtBoolean getSubaggregateID(int index, DtGlobalObjectDesignator& aggID) const**

- **virtual DtBoolean setEntityID(int index, const DtGlobalObjectDesignator& entID)**

  Set/Get a DIS entity ID. Returns DtFalse if the index is invalid. Can be done through entities() instead.

- **virtual DtBoolean getEntityID(int index, DtGlobalObjectDesignator& entID) const**

- **virtual DtBoolean setSilentAggregateList(int index, const DtSilentAggregateList& aggList)**

  Set/Get a silent aggregate list. Returns DtFalse if the index is invalid.

- **virtual DtBoolean getSilentAggregateList(int index, DtSilentAggregateList& aggList) const**

- **virtual DtBoolean setSilentEntityList(int index, const DtSilentEntityList& entList)**

  Set/Get a silent entity list. Returns DtFalse if the index is invalid.

- **virtual DtBoolean getSilentEntityList(int index, DtSilentEntityList& entList) const**

- **virtual DtBoolean setNumberOfDatumRecords(int newNumberOfRecords)**

  Set the number of variable datum records. Returns DtFalse if the number of records is too large. This will erase the contents of existing datum records.

- **virtual int numberOfDatumRecords() const**

  Get the number of variable datum records

- **virtual DtBoolean setVariableDatumRecord(int index, const DtVariableDatumRecord& rec)**

  Set/Get a variable datum record. Returns DtFalse if the index is invalid.

- **virtual DtBoolean getVariableDatumRecord(int index, DtVariableDatumRecord& rec) const**

- **virtual DtVariableDatumRecord& variableDatumRecordRef(int index)**

  Get a reference to a variable datum record

- **virtual const DtVariableDatumRecord& variableDatumRecordConstRef(int index) const**

  Get a constant reference to a variable datum record

- **virtual void printData() const**

Print the state data

- **static DtAggregateStateRepository* create()**
- **DtAggregateStateRepository* self() const**

  Cast off const-ness for trivial internal change. Non-virtual since can't override on return type.
- **DtForceType myForceID**
- **DtAggregateState myAggregateState**
- **DtAggregateFormation myFormation**
- **DtCharacterSet myMarkingCharSet**
- **#define AGGREGATE_SR_MARKING_TEXT_LENGTH**

  Aggregate state PDU marking is 31 bytes long, RPR marking struct is 11
- **#define AGGREGATE_SR_MARKING_TEXT_LENGTH**
- **char myMarkingText[AGGREGATE_SR_MARKING_TEXT_LENGTH]**
- **DtVector myDimensions**
- **DtGlobalObjectDesignatorList myEntities**
- **DtGlobalObjectDesignatorList mySubAggs**
- **int myNumberOfSilentAggregateTypes**
- **DtSilentAggregateList* mySilentAggregates**
- **int myNumberOfSilentEntityTypes**
- **DtSilentEntityList* mySilentEntities**
- **int myNumberOfVariableDatumRecords**
- **DtVariableDatumRecord* myVariableDatumRecords**
- **DtVariableDatumRecord myDumRec**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file genericsr.h:

class DtGenericStateRepository: public DtStateRepository

class DtGenericStateRepository: Instances of DtGenericStateRepository are repositories for the state of an HLA object of unknown type

---

Inheritance:

DtGenericStateRepository - DtStateRepository

Public Methods

	DtGenericStateRepository () *default constructor*
virtual	**~DtGenericStateRepository** () *destructor*
	DtGenericStateRepository (const DtGenericStateRepository& orig) *copy constructor*
DtGenericStateRepository&	**operator=** (const DtGenericStateRepository& orig) *assignment operator*
virtual void	**setValue** (RTI::AttributeHandle handle, void* buffer, int size) *Sets the value of an attribute by handle*
virtual void*	**value** (RTI::AttributeHandle handle) const *Returns the value of the attribute associated with handle, or NULL if none exists*
virtual int	**valueLength** (RTI::AttributeHandle handle) const *Returns the length of the value of the attribute associated with handle, or - if none exists*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
virtual void	**printData** () const *Print the state data*

Protected Fields

RTI::AttributeHandleValuePairSet*	**myAttrs**

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

-

Documentation

class DtGenericStateRepository:

Instances of DtGenericStateRepository are repositories for the state of an HLA object of unknown type. State is represented exactly the same way as in the FOM.

- **`DtGenericStateRepository()`**
  default constructor
- **`virtual ~DtGenericStateRepository()`**
  destructor
- **`DtGenericStateRepository(const DtGenericStateRepository& orig)`**
  copy constructor
- **`DtGenericStateRepository& operator=( const DtGenericStateRepository& orig)`**
  assignment operator
- **`virtual void setValue(RTI::AttributeHandle handle, void* buffer, int size)`**
  Sets the value of an attribute by handle
- **`virtual void* value(RTI::AttributeHandle handle) const`**
  Returns the value of the attribute associated with handle, or NULL if none exists
- **`virtual int valueLength(RTI::AttributeHandle handle) const`**
  Returns the length of the value of the attribute associated with handle, or - if none exists

- **`virtual DtStateRepository* clone() const`**

  return an empty object of the same type as this

- **`virtual void printData() const`**

  Print the state data

- **`RTI::AttributeHandleValuePairSet* myAttrs`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file emittrbeamsr.h:

class DtEmitterBeamRepository: public DtStateRepository

class DtEmitterBeamRepository: Instances of DtEmitterBeamRepository are

---

Inheritance:

DtEmitterBeamRepository - DtStateRepository

Public Methods

	DtEmitterBeamRepository () *default constructor*
virtual	**~DtEmitterBeamRepository** () *destructor*
	DtEmitterBeamRepository (const DtEmitterBeamRepository& orig) *copy constructor*
DtEmitterBeamRepository&	**operator=** (const DtEmitterBeamRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
static DtEmitterBeamRepository*	**create** ()
virtual void	**setAzimuthCenter** (DtFloat32 center) *Set/Get Azimuth Center*
virtual DtFloat32	**azimuthCenter** () const
virtual void	**setAzimuthSweep** (DtFloat32 sweep) *Set/Get Azimuth Sweep*
virtual DtFloat32	**azimuthSweep** () const
virtual void	**setBeamFunction** (DtBeamFunc func) *Set/Get Beam Function*
virtual DtBeamFunc	**beamFunction** () const
virtual void	**setBeamId** (DtU8 id) *Set/Get Beam Id*
virtual DtU8	**beamId** () const
virtual void	**setBeamParameterIndex** (DtU16 index) *Set/Get Beam Parameter Index*
virtual DtU16	**beamParameterIndex** () const

virtual void **setElevationCenter** (DtFloat32 center)
Set/Get Elevation Center

virtual DtFloat32 **elevationCenter** () const

virtual void **setElevationSweep** (DtFloat32 sweep)
Set/Get Elevation Sweep

virtual DtFloat32 **elevationSweep** () const

virtual void **setEmittingSystemId** (const DtGlobalObjectDesignator& id)
Set/Get Emitting System Id

virtual const DtGlobalObjectDesignator& **emittingSystemId** () const

virtual void **setERP** (DtFloat32 erp)
Set/Get ERP

virtual DtFloat32 **ERP** () const

virtual void **setEntityId** (const DtEntityIdentifier& id)
Set/Get Entity Id - not used after RPR 0

virtual const DtEntityIdentifier& **entityId** () const

virtual void **setEventId** (const DtEventId& id)
Set/Get Event Id - not used until RPR 0

virtual const DtEventId& **eventId** () const

virtual void **setFrequency** (DtFloat32 freq)
Set/Get Frequency

virtual DtFloat32 **frequency** () const

virtual void **setFrequencyRange** (DtFloat32 freq)
Set/Get Frequency Range

virtual DtFloat32 **frequencyRange** () const

virtual void **setPRF** (DtFloat32 prf)
Set/Get PRF

virtual DtFloat32 **PRF** () const

virtual void **setPulseWidth** (DtFloat32 width)
Set/Get Pulse Width

virtual DtFloat32 **pulseWidth** () const

virtual void **setSweepSync** (DtFloat32 sync)
Set/Get Sweep Sync

virtual DtFloat32 **sweepSync** () const

virtual void **setBeamType** (DtBeamType flag)
Set/Get beam type - track, jam or neither

virtual DtBeamType **beamType** () const

virtual void **setJammingModeSequence** (DtU32 sequence)
Set/Get Jamming Mode Sequence

virtual DtU32 **jammingModeSequence** () const

virtual const DtTrackJamDesignatorList& **trackJamTargets** () const
Get set of track jam targets

virtual DtTrackJamDesignatorList* **trackJamTargets** ()

virtual void **setHighDensity** (DtBoolean val)
Set/Get the high density track/jam flag

virtual DtBoolean **highDensity** () const

- virtual void **printData** () const

Public

- *The next two accessors should only be called if the beam type is DtTrackBeam or DtJamBeam*

Protected Fields

- DtFloat32 **myAzimuthCenter**
  Common to emitter beams and track jam beams
- DtFloat32 **myAzimuthSweep**
- DtBeamFunc **myBeamFunction**
- DtU8 **myBeamId**
- DtU16 **myBeamParameterIndex**
- DtFloat32 **myElevationCenter**
- DtFloat32 **myElevationSweep**
- DtGlobalObjectDesignator **myEmittingSystemId**
- DtFloat32 **myERP**
- DtEntityIdentifier **myEntityId**
- DtEventId **myEventId**
- DtFloat32 **myFrequency**
- DtFloat32 **myFrequencyRange**
- DtFloat32 **myPRF**
- DtFloat32 **myPulseWidth**
- DtFloat32 **mySweepSync**
- DtU32 **myJammingModeSequence**
  For Track Jam Beams
- DtTrackJamDesignatorList **myTrackJamTargets**
- DtBeamType **myBeamType**
- DtBoolean **myHighDensity**

Protected Methods

- virtual void **printTrackJam** () const
  Prints track/jam target information

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Documentation

class DtEmitterBeamRepository: Instances of DtEmitterBeamRepository are

- **`DtEmitterBeamRepository()`**
  default constructor
- **`virtual ~DtEmitterBeamRepository()`**
  destructor
- **`DtEmitterBeamRepository(const DtEmitterBeamRepository& orig)`**
  copy constructor
- **`DtEmitterBeamRepository& operator=( const DtEmitterBeamRepository& orig)`**
  assignment operator
- **`virtual DtStateRepository* clone() const`**
  return an empty object of the same type as this
- **`static DtEmitterBeamRepository* create()`**
- **`virtual void setAzimuthCenter(DtFloat32 center)`**
  Set/Get Azimuth Center
- **`virtual DtFloat32 azimuthCenter() const`**
- **`virtual void setAzimuthSweep(DtFloat32 sweep)`**
  Set/Get Azimuth Sweep
- **`virtual DtFloat32 azimuthSweep() const`**
- **`virtual void setBeamFunction(DtBeamFunc func)`**
  Set/Get Beam Function

- **virtual DtBeamFunc beamFunction() const**
- **virtual void setBeamId(DtU8 id)**
  Set/Get Beam Id
- **virtual DtU8 beamId() const**
- **virtual void setBeamParameterIndex(DtU16 index)**
  Set/Get Beam Parameter Index
- **virtual DtU16 beamParameterIndex() const**
- **virtual void setElevationCenter(DtFloat32 center)**
  Set/Get Elevation Center
- **virtual DtFloat32 elevationCenter() const**
- **virtual void setElevationSweep(DtFloat32 sweep)**
  Set/Get Elevation Sweep
- **virtual DtFloat32 elevationSweep() const**
- **virtual void setEmittingSystemId(const DtGlobalObjectDesignator& id)**
  Set/Get Emitting System Id
- **virtual const DtGlobalObjectDesignator& emittingSystemId() const**
- **virtual void setERP(DtFloat32 erp)**
  Set/Get ERP
- **virtual DtFloat32 ERP() const**
- **virtual void setEntityId(const DtEntityIdentifier& id)**
  Set/Get Entity Id - not used after RPR 0
- **virtual const DtEntityIdentifier& entityId() const**
- **virtual void setEventId(const DtEventId& id)**
  Set/Get Event Id - not used until RPR 0
- **virtual const DtEventId& eventId() const**
- **virtual void setFrequency(DtFloat32 freq)**
  Set/Get Frequency
- **virtual DtFloat32 frequency() const**
- **virtual void setFrequencyRange(DtFloat32 freq)**
  Set/Get Frequency Range
- **virtual DtFloat32 frequencyRange() const**
- **virtual void setPRF(DtFloat32 prf)**
  Set/Get PRF
- **virtual DtFloat32 PRF() const**

- **virtual void setPulseWidth(DtFloat32 width)**

  Set/Get Pulse Width

- **virtual DtFloat32 pulseWidth() const**
- **virtual void setSweepSync(DtFloat32 sync)**

  Set/Get Sweep Sync

- **virtual DtFloat32 sweepSync() const**
- **virtual void setBeamType(DtBeamType flag)**

  Set/Get beam type - track, jam or neither

- **virtual DtBeamType beamType() const**
-

  The next two accessors should only be called if the beam type is DtTrackBeam or DtJamBeam

- **virtual void setJammingModeSequence(DtU32 sequence)**

  Set/Get Jamming Mode Sequence

- **virtual DtU32 jammingModeSequence() const**
- **virtual const DtTrackJamDesignatorList& trackJamTargets() const**

  Get set of track jam targets. Returns pointer to internal object. Do not delete. It will be deleted when the SR is deleted.

- **virtual DtTrackJamDesignatorList* trackJamTargets()**
- **virtual void setHighDensity(DtBoolean val)**

  Set/Get the high density track/jam flag

- **virtual DtBoolean highDensity() const**
- **virtual void printData() const**
- **virtual void printTrackJam() const**

  Prints track/jam target information

- **DtFloat32 myAzimuthCenter**

  Common to emitter beams and track jam beams

- **DtFloat32 myAzimuthSweep**
- **DtBeamFunc myBeamFunction**
- **DtU8 myBeamId**
- **DtU16 myBeamParameterIndex**
- **DtFloat32 myElevationCenter**
- **DtFloat32 myElevationSweep**
- **DtGlobalObjectDesignator myEmittingSystemId**
- **DtFloat32 myERP**

- **DtEntityIdentifier myEntityId**
- **DtEventId myEventId**
- **DtFloat32 myFrequency**
- **DtFloat32 myFrequencyRange**
- **DtFloat32 myPRF**
- **DtFloat32 myPulseWidth**
- **DtFloat32 mySweepSync**
- **DtU32 myJammingModeSequence**
  For Track Jam Beams
- **DtTrackJamDesignatorList myTrackJamTargets**
- **DtBeamType myBeamType**
- **DtBoolean myHighDensity**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file embedsystemsr.h:

class DtEmbeddedSystemRepository: public DtStateRepository

class DtEmbeddedSystemRepository: Instances of DtEmbeddedSystemRepository are

---

Inheritance:

DtEmbeddedSystemRepository - DtStateRepository

Public Methods

	DtEmbeddedSystemRepository () *default constructor*
	DtEmbeddedSystemRepository (const DtEmbeddedSystemRepository& orig) *copy constructor*
virtual	**~DtEmbeddedSystemRepository** () *destructor*
DtEmbeddedSystemRepository&	**operator=** (const DtEmbeddedSystemRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
static DtEmbeddedSystemRepository*	**create** ()
virtual void	**setEntityId** (const DtEntityIdentifier& id) *Set/Get the entity identifier*
virtual const DtEntityIdentifier&	**entityId** () const
virtual void	**setHostId** (const DtGlobalObjectDesignator& id) *Set/Get the host identifier*
virtual const DtGlobalObjectDesignator&	**hostId** () const
virtual void	**setRelativePosition** (DtConstVector position) *Set/Get the relative position*
virtual DtConstVector	**relativePosition** () const
virtual void	**printData** () const

Protected Fields

- DtEntityIdentifier **myEntityId**
  embedded system class attributes
- DtGlobalObjectDesignator **myHostId**
- DtVector **myRelativePosition**

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

-

Documentation

class DtEmbeddedSystemRepository: Instances of DtEmbeddedSystemRepository are

- **`DtEmbeddedSystemRepository()`**
  default constructor
- **`DtEmbeddedSystemRepository(const DtEmbeddedSystemRepository& orig)`**
  copy constructor
- **`virtual ~DtEmbeddedSystemRepository()`**
  destructor
- **`DtEmbeddedSystemRepository& operator=( const DtEmbeddedSystemRepository& orig)`**
  assignment operator
- **`virtual DtStateRepository* clone() const`**

return an empty object of the same type as this

- **`static DtEmbeddedSystemRepository* create()`**
- **`virtual void setEntityId(const DtEntityIdentifier& id)`**

  Set/Get the entity identifier
- **`virtual const DtEntityIdentifier& entityId() const`**
- **`virtual void setHostId(const DtGlobalObjectDesignator& id)`**

  Set/Get the host identifier
- **`virtual const DtGlobalObjectDesignator& hostId() const`**
- **`virtual void setRelativePosition(DtConstVector position)`**

  Set/Get the relative position
- **`virtual DtConstVector relativePosition() const`**
- **`virtual void printData() const`**
- **`DtEntityIdentifier myEntityId`**

  embedded system class attributes
- **`DtGlobalObjectDesignator myHostId`**
- **`DtVector myRelativePosition`**

Direct child classes:

DtRadioTransmitterRepository
DtRadioReceiverRepository
DtEmitterSystemRepository
DtDesignatorRepository

Alphabetic index *Hierarchy of classes*

In file netstructs.h:

class DtNetEntityIdentifier

Public Fields

- DtNetSimulationAddress **DtSimulator**
- DtNetU16 **DtEntity**

Public Methods

- DtEntityIdentifier **operator =** (DtEntityIdentifier)
- DtNetEntityIdentifier **operator =** (DtNetEntityIdentifier)

Documentation

- **`DtNetSimulationAddress DtSimulator`**
- **`DtNetU16 DtEntity`**
- **`DtEntityIdentifier operator = (DtEntityIdentifier)`**
- **`DtNetEntityIdentifier operator = (DtNetEntityIdentifier)`**

This class has no child classes.

In file netstructs.h:

class DtNetFederateId

Public Fields

- DtNetU16 **site**
- DtNetU16 **app**

Public Methods

- DtFederateId **operator =** (DtFederateId)
- DtNetFederateId **operator =** (DtNetFederateId)

Documentation

- **`DtNetU16 site`**
- **`DtNetU16 app`**
- **`DtFederateId operator = (DtFederateId)`**
- **`DtNetFederateId operator = (DtNetFederateId)`**

This class has no child classes.

In file disenums.h:

enum DtForceType

Section 4

- **DtForceOther**
- **DtForceFriendly**
- **DtForceOpposing**
- **DtForceNeutral**

Documentation

Section 4

- `DtForceOther`
- `DtForceFriendly`
- `DtForceOpposing`
- `DtForceNeutral`

In file disenums.h:

enum DtDamageState

This is for the tank ditch and concertina wire breach appearance Areal object specific appearance

---

- **DtDamageNone**
- **DtDamageSlight**
- **DtDamageModerate**
- **DtDamageDestroyed**

Documentation

This is for the tank ditch and concertina wire breach appearance Areal object specific appearance

- `DtDamageNone`
- `DtDamageSlight`
- `DtDamageModerate`
- `DtDamageDestroyed`

In file disenums.h:

enum DtHatchState

- **DtHatchNA**
- **DtHatchClosed**
- **DtHatchPopped**
- **DtHatchPoppedAndPerson**
- **DtHatchOpen**
- **DtHatchOpenAndPerson**

Documentation

- `DtHatchNA`
- `DtHatchClosed`
- `DtHatchPopped`
- `DtHatchPoppedAndPerson`
- `DtHatchOpen`
- `DtHatchOpenAndPerson`

In file disenums.h:

enum DtLifeformState

- **DtLifeformNA**
- **DtLifeformStanding**
- **DtLifeformWalking**
- **DtLifeformRunning**
- **DtLifeformKneeling**
- **DtLifeformProne**
- **DtLifeformCrawling**
- **DtLifeformSwimming**
- **DtLifeformParachuting**
- **DtLifeformJumping**
- **DtLifeformSitting**
- **DtLifeformSquatting**
- **DtLifeformCrouching**
- **DtLifeformWading**

Documentation

- `DtLifeformNA`
- `DtLifeformStanding`
- `DtLifeformWalking`
- `DtLifeformRunning`
- `DtLifeformKneeling`
- `DtLifeformProne`
- `DtLifeformCrawling`
- `DtLifeformSwimming`
- `DtLifeformParachuting`

- **DtLifeformJumping**
- **DtLifeformSitting**
- **DtLifeformSquatting**
- **DtLifeformCrouching**
- **DtLifeformWading**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtLightState

- **DtLightsNone**
- **DtLightsRunning**
- **DtLightsNavigation**
- **DtLightsFormation**

Documentation

- `DtLightsNone`
- `DtLightsRunning`
- `DtLightsNavigation`
- `DtLightsFormation`

In file disenums.h:

enum DtDensityType

- **DtDensityClear**
- **DtDensityHazy**
- **DtDensityDense**
- **DtDensityVeryDense**
- **DtDensityOpaque**

Documentation

- `DtDensityClear`
- `DtDensityHazy`
- `DtDensityDense`
- `DtDensityVeryDense`
- `DtDensityOpaque`

In file disenums.h:

enum DtTrailState

- **DtTrailingEffectsNone**
- **DtTrailingEffectsSmall**
- **DtTrailingEffectsMedium**
- **DtTrailingEffectsLarge**

Documentation

- **`DtTrailingEffectsNone`**
- **`DtTrailingEffectsSmall`**
- **`DtTrailingEffectsMedium`**
- **`DtTrailingEffectsLarge`**

In file disenums.h:

enum DtCamouflageType

- **DtDesertCamouflage**
- **DtWinterCamouflage**
- **DtForestCamouflage**

Documentation

- `DtDesertCamouflage`
- `DtWinterCamouflage`
- `DtForestCamouflage`

In file disenums.h:

enum DtWeaponState

- **DtNoWeapon**
- **DtWeaponStowed**
- **DtWeaponDeployed**
- **DtWeaponInFirePosition**

Documentation

- `DtNoWeapon`
- `DtWeaponStowed`
- `DtWeaponDeployed`
- `DtWeaponInFirePosition`

In file disenums.h:

enum DtAppearAttribute

if you put spaces between the "<<" and the surrounding strings, enumstr gets confused and thinks the shift is another value

---

- **DtAppPaintScheme**
- **DtAppMobilityKill**
- **DtAppFirePowerKill**
- **DtAppDamage**
- **DtAppSmoke**
- **DtAppTrailingEffects**
- **DtAppHatch**
- **DtAppLights**
- **DtAppFlaming**
- **DtAppFrozen**
- **DtAppPowerPlantStatus**
- **DtAppDummy**

Documentation

if you put spaces between the "<<" and the surrounding strings, enumstr gets confused and thinks the shift is another value

- `DtAppPaintScheme`
- `DtAppMobilityKill`
- `DtAppFirePowerKill`
- `DtAppDamage`
- `DtAppSmoke`
- `DtAppTrailingEffects`
- `DtAppHatch`
- `DtAppLights`
- `DtAppFlaming`

- **DtAppFrozen**
- **DtAppPowerPlantStatus**
- **DtAppDummy**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtAppearAttrVal

Create an enumeration type, but since many of the names will have the same value, define them all as consts below

---

DtAppearInvalid

Documentation

Create an enumeration type, but since many of the names will have the same value, define them all as consts below.

`DtAppearInvalid`

In file disenums.h:

enum DtAggregateState

Aggregate State PDU enums

- **DtOtherAggregateState**
- **DtAggregated**
- **DtDisaggregated**
- **DtFullyDisaggregated**
- **DtPseudoDisaggregated**
- **DtPartiallyDisaggregated**

Documentation

Aggregate State PDU enums

- `DtOtherAggregateState`
- `DtAggregated`
- `DtDisaggregated`
- `DtFullyDisaggregated`
- `DtPseudoDisaggregated`
- `DtPartiallyDisaggregated`

In file disenums.h:

enum DtAggregateFormation

- **DtOtherAggregateFormation**
- **DtAssemblyFormation**
- **DtVeeFormation**
- **DtWedgeFormation**
- **DtLineFormation**
- **DtColumnFormation**

Documentation

- `DtOtherAggregateFormation`
- `DtAssemblyFormation`
- `DtVeeFormation`
- `DtWedgeFormation`
- `DtLineFormation`
- `DtColumnFormation`

In file disenums.h:

enum DtCharacterSet

- **DtUnusedCharacterSet**
- **DtASCII_CharacterSet**

Documentation

- `DtUnusedCharacterSet`
- `DtASCII_CharacterSet`

In file aggagglist.h:

class DtSilentAggregateList

class DtSilentAggregateList: DtSilentAggregateList is a class which contains the information needed to describe a list of silent aggregates for the aggregate state PDU

Public Methods

	DtSilentAggregateList () *Constructors*
	DtSilentAggregateList (const DtSilentAggregateList& orig)
virtual	**~DtSilentAggregateList** () *Destructor*
DtSilentAggregateList&	**operator=** (const DtSilentAggregateList& orig) *Assignment operator*
DtSilentAggregateList&	**operator=** (const DtNetSilentAggregateList& orig)
int	**operator==** (const DtSilentAggregateList& otherguy) const *Comparison operators*
int	**operator!=** (const DtSilentAggregateList& otherguy) const
virtual void	**setNumberOfAggregates** (const DtU16 value) *Set/Get the number of aggregates*
virtual DtU16	**numberOfAggregates** () const
virtual void	**setAggregateType** (const DtEntityType& type) *Set/Get the aggregate type*
virtual const DtEntityType&	**aggregateType** () const
virtual void	**print** () const *Print the record*
virtual void	**unloadNetRecord** (const DtNetSilentAggregateList& record) *Load/unload data from a packet record*
virtual void	**loadNetRecord** (DtNetSilentAggregateList& record) const
static int	**byteLengthOfNetworkRecord** () *Get the length of the network representation of a record*

Protected Fields

DtU16	**myNumberOfAggregates** *Data members:*
DtEntityType	**myEntityType**

Documentation

class DtSilentAggregateList: DtSilentAggregateList is a class which contains the information needed to describe a list of silent aggregates for the aggregate state PDU

- **`DtSilentAggregateList()`**
  Constructors
- **`DtSilentAggregateList(const DtSilentAggregateList& orig)`**
- **`virtual ~DtSilentAggregateList()`**
  Destructor
- **`DtSilentAggregateList& operator=(const DtSilentAggregateList& orig)`**
  Assignment operator
- **`DtSilentAggregateList& operator=(const DtNetSilentAggregateList& orig)`**
- **`int operator==(const DtSilentAggregateList& otherguy) const`**
  Comparison operators
- **`int operator!=(const DtSilentAggregateList& otherguy) const`**
- **`virtual void setNumberOfAggregates(const DtU16 value)`**
  Set/Get the number of aggregates
- **`virtual DtU16 numberOfAggregates() const`**
- **`virtual void setAggregateType(const DtEntityType& type)`**
  Set/Get the aggregate type
- **`virtual const DtEntityType& aggregateType() const`**
- **`virtual void print() const`**
  Print the record
- **`virtual void unloadNetRecord(const DtNetSilentAggregateList& record)`**
  Load/unload data from a packet record
- **`virtual void loadNetRecord(DtNetSilentAggregateList& record) const`**
- **`static int byteLengthOfNetworkRecord()`**
  Get the length of the network representation of a record
- **`DtU16 myNumberOfAggregates`**
  Data members:
- **`DtEntityType myEntityType`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file aggentlist.h:

class DtSilentEntityList

class DtSilentEntityList: DtSilentEntityList is a class which contains the information needed to describe a list of silent entities for the aggregate state PDU

Public Methods

	DtSilentEntityList () *Constructors*
	DtSilentEntityList (const DtSilentEntityList& orig)
virtual	**~DtSilentEntityList** () *Destructor*
DtSilentEntityList&	**operator=** (const DtSilentEntityList& orig) *Assignment operator*
DtSilentEntityList&	**operator=** (const DtNetSilentEntityList& orig)
int	**operator==** (const DtSilentEntityList& otherguy) const *Comparison operators*
int	**operator!=** (const DtSilentEntityList& otherguy) const
virtual void	**setNumberOfEntities** (const DtU16 value) *Set/Get the number of aggregates*
virtual DtU16	**numberOfEntities** () const
virtual void	**setEntityType** (const DtEntityType& type) *Set/Get the entity type*
virtual const DtEntityType&	**entityType** () const
virtual DtBoolean	**setNumberOfAppearanceRecords** (const DtU16 value) *Set/Get the number of appearance records*
virtual DtU16	**numberOfAppearanceRecords** () const
virtual DtBoolean	**setAppearance** (int index, DtU32 app) *Set/Get the appearance of an entity (as a 32 bit integer)*
virtual DtBoolean	**getAppearance** (int index, DtU32& app) const
virtual DtBoolean	**setAppearanceAttribute** (int index, const DtAppearAttribute attr, const DtAppearAttrVal val) *Set/Get appearance by attribute*
virtual DtBoolean	**getAppearanceAttribute** (int index, const DtAppearAttribute attr, DtAppearAttrVal& val) const
virtual void	**print** () const *Print the record*
virtual void	**unloadNetRecord** (const DtNetSilentEntityList& record) *Load/unload data from a packet record*
virtual void	**loadNetRecord** (DtNetSilentEntityList& record) const

- virtual int — **networkByteLength** () const
  Get the network byte length of this record
- static int — **byteLengthOfNetworkRecord** (DtNetSilentEntityList* recPtr)
  Get the length of the network representation of a record

Protected Fields

- DtU16 — **myNumberOfEntities**
  Data members:
- DtEntityType — **myEntityType**
- DtU16 — **myNumberOfAppearanceRecords**
- DtU32* — **myEntityAppearanceList**

Documentation

class DtSilentEntityList: DtSilentEntityList is a class which contains the information needed to describe a list of silent entities for the aggregate state PDU

- **`DtSilentEntityList()`**
  Constructors
- **`DtSilentEntityList(const DtSilentEntityList& orig)`**
- **`virtual ~DtSilentEntityList()`**
  Destructor
- **`DtSilentEntityList& operator=(const DtSilentEntityList& orig)`**
  Assignment operator
- **`DtSilentEntityList& operator=(const DtNetSilentEntityList& orig)`**
- **`int operator==(const DtSilentEntityList& otherguy) const`**
  Comparison operators
- **`int operator!=(const DtSilentEntityList& otherguy) const`**
- **`virtual void setNumberOfEntities(const DtU16 value)`**
  Set/Get the number of aggregates
- **`virtual DtU16 numberOfEntities() const`**
- **`virtual void setEntityType(const DtEntityType& type)`**
  Set/Get the entity type
- **`virtual const DtEntityType& entityType() const`**
- **`virtual DtBoolean setNumberOfAppearanceRecords(const DtU16 value)`**
  Set/Get the number of appearance records. Set returns DtFalse if the value is too large, old values are not retained when the number changes.

- **virtual DtU16 numberOfAppearanceRecords() const**
- **virtual DtBoolean setAppearance(int index, DtU32 app)**

  Set/Get the appearance of an entity (as a 32 bit integer). Returns DtFalse if the index is out of range. Set will erase all previous values.
- **virtual DtBoolean getAppearance(int index, DtU32& app) const**
- **virtual DtBoolean setAppearanceAttribute(int index, const DtAppearAttribute attr, const DtAppearAttrVal val)**

  Set/Get appearance by attribute. Returns DtFalse if the index is out of range.
- **virtual DtBoolean getAppearanceAttribute(int index, const DtAppearAttribute attr, DtAppearAttrVal& val) const**
- **virtual void print() const**

  Print the record
- **virtual void unloadNetRecord(const DtNetSilentEntityList& record)**

  Load/unload data from a packet record
- **virtual void loadNetRecord(DtNetSilentEntityList& record) const**
- **virtual int networkByteLength() const**

  Get the network byte length of this record
- **static int byteLengthOfNetworkRecord(DtNetSilentEntityList* recPtr)**

  Get the length of the network representation of a record
- **DtU16 myNumberOfEntities**

  Data members:
- **DtEntityType myEntityType**
- **DtU16 myNumberOfAppearanceRecords**
- **DtU32* myEntityAppearanceList**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vardatumrec.h:

class DtVariableDatumRecord

class DtVariableDatumRecord: Instances of DtVariableDatumRecord are used to hold a single variable datum record and to provide access to its network representation

---

Public Methods

	DtVariableDatumRecord () *default constructor*
virtual	**~DtVariableDatumRecord** () *destructor*
	DtVariableDatumRecord (const DtVariableDatumRecord& orig) *copy constructor*
DtVariableDatumRecord&	**operator=** (const DtVariableDatumRecord& orig) *assignment operator*
int	**operator==** (const DtVariableDatumRecord& otherguy) const *comparison operators*
int	**operator!=** (const DtVariableDatumRecord& otherguy) const
virtual void	**setExternalBuffer** (char* bufferPtr) *Set the working buffer to an externally supplied buffer*
virtual void	**setDatumBitLength** (DtU32 newDatumBitLength) *Set/Get the datum length in bits*
virtual DtU32	**datumBitLength** () const
virtual int	**datumByteLength** () const *Get the datum length in bytes (padded to an 8 byte boundary)*
virtual void	**setDatumID** (DtDatumParam value) *Set/Get the variable datum ID*
virtual DtDatumParam	**datumID** () const
virtual void	**setDatumValFloat64** (DtFloat64 value) *Set/Get datum value as a DtFloat64 Set will set the datum bit length to 64*
virtual DtFloat64	**datumValFloat64** () const
virtual void	**setDatumValFloat32** (DtFloat32 value) *Set/Get datum value as a DtFloat32 Set will set the datum bit length to 32*
virtual DtFloat32	**datumValFloat32** () const
virtual void	**setDatumValInt32** (DtInt32 value) *Set/Get datum value as a DtInt32 Set will set the datum bit length to 32*
virtual DtInt32	**datumValInt32** () const
virtual void	**setDatumValU32** (DtU32 value) *Set/Get datum value as a DtU32 Set will set the datum bit length to 32*
virtual DtU32	**datumValU32** () const
virtual void	**setDatumValInt16** (DtInt16 value) *Set/Get datum value as a DtInt16 Set will set the datum bit length to 16*

- virtual DtInt16 **datumValInt16** () const
- virtual void **setDatumValU16** (DtU16 value)
  Set/Get datum value as a DtU16 Set will set the datum bit length to 16
- virtual DtU16 **datumValU16** () const
- virtual void **setDatumValU8** (DtU8 value)
  Set/Get datum value as a DtU8 Set will set the datum bit length to 8
- virtual DtU8 **datumValU8** () const
- virtual void **setDatumValByteArray** (const DtU8* setFrom)
  Set/Get datum value as a byte array
- virtual void **getDatumValByteArray** (DtU8* copyTo) const
- virtual char* **datumValPtr** ()
  Get a pointer to the datum value inside of the record
- virtual const char* **constDatumValPtr** () const
  Get a const pointer to the datum value inside of the record
- virtual int **networkByteLength** () const
  Get the network byte length of the entire datum record
- virtual const char* **networkDataBuffer** () const
  Get a pointer to the buffer which stores the network representation of the datum record
- virtual void **print** () const
  Print the contents of the record

Protected Fields

- char* **myRecordBuffer**
- char* **myExternalBuffer**

Protected Methods

- virtual DtNetVarDatum* **netStruct** ()
  Cast the buffer to a DtNetVarDatum struct
- virtual const DtNetVarDatum* **constNetStruct** () const
  Cast the buffer to a const DtNetVarDatum struct
- virtual void **deleteIfNoExternal** ()
  Delete the internal buffer if there is no external buffer
- virtual int **datumByteSize** (int datumBitLength) const
  Compute the total datum size in bytes from the datum bit length
- virtual int **recordByteSize** (int datumBitLength) const
  Compute the total record size in bytes from the datum bit length

---

Documentation

class DtVariableDatumRecord: Instances of DtVariableDatumRecord are used to hold a single variable datum record and to provide access to its network representation

- **`DtVariableDatumRecord()`**

  default constructor

- **virtual ~DtVariableDatumRecord()**
  destructor
- **DtVariableDatumRecord(const DtVariableDatumRecord& orig)**
  copy constructor
- **DtVariableDatumRecord& operator=(const DtVariableDatumRecord& orig)**
  assignment operator
- **int operator==(const DtVariableDatumRecord& otherguy) const**
  comparison operators
- **int operator!=(const DtVariableDatumRecord& otherguy) const**
- **virtual void setExternalBuffer(char* bufferPtr)**
  Set the working buffer to an externally supplied buffer
- **virtual void setDatumBitLength(DtU32 newDatumBitLength)**
  Set/Get the datum length in bits. This method will re-allocate memory if the buffer is an internal buffer.
- **virtual DtU32 datumBitLength() const**
- **virtual int datumByteLength() const**
  Get the datum length in bytes (padded to an 8 byte boundary)
- **virtual void setDatumID(DtDatumParam value)**
  Set/Get the variable datum ID
- **virtual DtDatumParam datumID() const**
- **virtual void setDatumValFloat64(DtFloat64 value)**
  Set/Get datum value as a DtFloat64 Set will set the datum bit length to 64
- **virtual DtFloat64 datumValFloat64() const**
- **virtual void setDatumValFloat32(DtFloat32 value)**
  Set/Get datum value as a DtFloat32 Set will set the datum bit length to 32
- **virtual DtFloat32 datumValFloat32() const**
- **virtual void setDatumValInt32(DtInt32 value)**
  Set/Get datum value as a DtInt32 Set will set the datum bit length to 32
- **virtual DtInt32 datumValInt32() const**
- **virtual void setDatumValU32(DtU32 value)**
  Set/Get datum value as a DtU32 Set will set the datum bit length to 32
- **virtual DtU32 datumValU32() const**
- **virtual void setDatumValInt16(DtInt16 value)**
  Set/Get datum value as a DtInt16 Set will set the datum bit length to 16
- **virtual DtInt16 datumValInt16() const**
- **virtual void setDatumValU16(DtU16 value)**

Set/Get datum value as a DtU16 Set will set the datum bit length to 16

- **`virtual DtU16 datumValU16() const`**
- **`virtual void setDatumValU8(DtU8 value)`**

  Set/Get datum value as a DtU8 Set will set the datum bit length to 8

- **`virtual DtU8 datumValU8() const`**
- **`virtual void setDatumValByteArray(const DtU8* setFrom)`**

  Set/Get datum value as a byte array. Data is copied to and from the network data buffer according to the specified length of the record.

- **`virtual void getDatumValByteArray(DtU8* copyTo) const`**
- **`virtual char* datumValPtr()`**

  Get a pointer to the datum value inside of the record

- **`virtual const char* constDatumValPtr() const`**

  Get a const pointer to the datum value inside of the record

- **`virtual int networkByteLength() const`**

  Get the network byte length of the entire datum record

- **`virtual const char* networkDataBuffer() const`**

  Get a pointer to the buffer which stores the network representation of the datum record

- **`virtual void print() const`**

  Print the contents of the record

- **`virtual DtNetVarDatum* netStruct()`**

  Cast the buffer to a DtNetVarDatum struct

- **`virtual const DtNetVarDatum* constNetStruct() const`**

  Cast the buffer to a const DtNetVarDatum struct

- **`virtual void deleteIfNoExternal()`**

  Delete the internal buffer if there is no external buffer

- **`virtual int datumByteSize(int datumBitLength) const`**

  Compute the total datum size in bytes from the datum bit length

- **`virtual int recordByteSize(int datumBitLength) const`**

  Compute the total record size in bytes from the datum bit length

- **`char* myRecordBuffer`**
- **`char* myExternalBuffer`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file libmatrix.h:

class DtCoordTransform

Public Fields

- DtVector **offset**
  these were protected, but it is often more efficient to set these by passing them to another function
- DtDcm **rotation**
  "to" origin in base coords

Public Methods

- **DtCoordTransform** ()
- **DtCoordTransform** (DtConstVector toOriginInFromCoord, DtConstDcm from2to)
- void **setOffset** (DtConstVector toOriginInFromCoord)
- void **setRotation** (DtConstDcm from2to)
- void **setByInverse** (const DtCoordTransform &to2from)
- void **getOffset** (DtVectorRef toOriginInFromCoord) const
- void **getOrientation** (DtDcmRef from_to) const
- void **coordTrans** (DtConstVector from, DtVectorRef to) const
- void **vecTrans** (DtConstVector from, DtVectorRef to) const
- void **eulerTrans** (const DtTaitBryan &from, DtTaitBryan *to) const

---

Documentation

- `DtCoordTransform()`
- `DtCoordTransform(DtConstVector toOriginInFromCoord, DtConstDcm from2to)`
- `void setOffset(DtConstVector toOriginInFromCoord)`
- `void setRotation(DtConstDcm from2to)`
- `void setByInverse(const DtCoordTransform &to2from)`
- `void getOffset(DtVectorRef toOriginInFromCoord) const`
- `void getOrientation(DtDcmRef from_to) const`
- `void coordTrans(DtConstVector from, DtVectorRef to) const`

- **`void vecTrans(DtConstVector from, DtVectorRef to) const`**
- **`void eulerTrans(const DtTaitBryan &from, DtTaitBryan *to) const`**
- **`DtVector offset`**

  these were protected, but it is often more efficient to set these by passing them to another function
- **`DtDcm rotation`**

  "to" origin in base coords

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file topoview.h:

class DtTopoView: public DtCartesianView

class DtTopoView: Instances of DtTopoView are used to create a Topographic coordinate view of the geocentric coordinates in an Entity State Repository

---

Inheritance:

DtTopoView - DtCartesianView

Public Methods

	DtTopoView (DtBaseEntityStateRepository* esr, double latitiude, double longitude) *constructor*
virtual	**~DtTopoView** () *destructor*
	DtTopoView (const DtTopoView& orig) *copy constructor*
DtTopoView&	**operator=** (const DtTopoView& orig) *assignment operator*

Inherited from DtCartesianView:

Public Methods

- virtual void **setOffset**(DtConstVector originInGeoc)
- virtual void **setRotation**(DtConstDcm geocToLocalRot)
- virtual void **setLocation**(DtConstVector Location)
- virtual DtConstVector **location**() const
- virtual void **setVelocity**(DtConstVector Velocity)
- virtual DtConstVector **velocity**() const
- virtual void **setOrientation**(const DtTaitBryan& Orient)
- virtual const DtTaitBryan& **orientation**() const
- virtual DtConstDcm **bodyToLocal**() const

- virtual void **setAcceleration**(DtConstVector Acceleration)
- virtual DtConstVector **acceleration**() const
- virtual void **setRotationalVelocity**(DtConstVector BodyRates)
- virtual DtConstVector **rotationalVelocity**() const
- inline DtCartesianView* **self**() const

Protected Fields

- DtBaseEntityStateRepository* **myESR**
- DtCoordTransform **myGeocToLocal**
- DtCoordTransform **myLocalToGeoc**
- DtVector **myLocalLoc**
- DtVector **myLocalVel**
- DtVector **myLocalAccel**
- DtTaitBryan **myLocalOrient**
- DtDcm **myBodyToLocal**

Protected Methods

- DtCartesianView* **self**() const

Documentation

class DtTopoView: Instances of DtTopoView are used to create a Topographic coordinate view of the geocentric coordinates in an Entity State Repository

- `DtTopoView(DtBaseEntityStateRepository* esr, double latitiude, double longitude)`

  constructor
- `virtual ~DtTopoView()`

  destructor
- `DtTopoView(const DtTopoView& orig)`

  copy constructor
- `DtTopoView& operator=(const DtTopoView& orig)`

  assignment operator

This class has no child classes.

In file disenums.h:

enum DtBeamFunc

- **DtBeamFuncOther**
- **DtBeamFuncSearch**
- **DtBeamFuncHeightFinder**
- **DtBeamFuncAcquisition**
- **DtBeamFuncTracking**
- **DtBeamFuncAcquisitionAndTracking**
- **DtBeamFuncCommandGuidance**
- **DtBeamFuncIllumination**
- **DtBeamFuncRangeOnlyRadar**
- **DtBeamFuncMissileBeacon**
- **DtBeamFuncMissileFuze**
- **DtBeamFuncActiveRadarMissileSeeker**
- **BeamFuncJammer**

Documentation

- `DtBeamFuncOther`
- `DtBeamFuncSearch`
- `DtBeamFuncHeightFinder`
- `DtBeamFuncAcquisition`
- `DtBeamFuncTracking`
- `DtBeamFuncAcquisitionAndTracking`
- `DtBeamFuncCommandGuidance`
- `DtBeamFuncIllumination`
- `DtBeamFuncRangeOnlyRadar`
- `DtBeamFuncMissileBeacon`

- **DtBeamFuncMissileFuze**
- **DtBeamFuncActiveRadarMissileSeeker**
- **BeamFuncJammer**

Alphabetic index *Hierarchy of classes*

In file eventid.h:

class DtEventId

Public Methods

	DtEventId (const char* str) *Construct a DtEventId from a character string consisting of the event number followed by a colon, and then an object name*
	DtEventId (const DtGlobalObjectDesignator& des, int eventNum) *Construct a DtEventId from a name and an event number*
	DtEventId (const DtNetEventId& netId) *Construct a DtEventId from network representation*
	DtEventId () *Same as DtEventId("", 0);*
void	**setFromNet** (const DtNetEventId& netId) *Set from network representation/*
DtNetEventId*	**netRep** () const *Convert to network representation*
int	**operator==** (const DtEventId &otherGuy) const *equivalence operators*
int	**operator!=** (const DtEventId &otherGuy) const
void	**setEventNum** (int eventNum) *Set/Get event number component*
int	**eventNum** () const
const DtGlobalObjectDesignator&	**objectId** () const *Set/Get object designator component*
void	**setObjectId** (const DtGlobalObjectDesignator& name)
const DtGlobalObjectDesignator&	**globalId** () const *Set/Get object designator component*
void	**setGlobalId** (const DtGlobalObjectDesignator& name)
char*	**string** () const *Convert to string*
static const DtEventId&	**nullId** () *Return a null DtEventId*

Protected Fields

DtString	**myObjName**
int	**myEventNum**

Protected Methods

- void **init** (const DtGlobalObjectDesignator& des, int eventNum)
  initialize

Documentation

- **`DtEventId(const char* str)`**

  Construct a DtEventId from a character string consisting of the event number followed by a colon, and then an object name. For example: DtEventId event("36:Object1");

- **`DtEventId(const DtGlobalObjectDesignator& des, int eventNum)`**

  Construct a DtEventId from a name and an event number

- **`DtEventId(const DtNetEventId& netId)`**

  Construct a DtEventId from network representation

- **`DtEventId()`**

  Same as DtEventId("", 0);

- **`void setFromNet(const DtNetEventId& netId)`**

  Set from network representation/

- **`DtNetEventId* netRep() const`**

  Convert to network representation. Returns ptr to internal memory.

- **`int operator==(const DtEventId &otherGuy) const`**

  equivalence operators

- **`int operator!=(const DtEventId &otherGuy) const`**

- **`void setEventNum(int eventNum)`**

  Set/Get event number component

- **`int eventNum() const`**

- **`const DtGlobalObjectDesignator& objectId() const`**

  Set/Get object designator component

- **`void setObjectId(const DtGlobalObjectDesignator& name)`**

- **`const DtGlobalObjectDesignator& globalId() const`**

  Set/Get object designator component

- **`void setGlobalId(const DtGlobalObjectDesignator& name)`**

- **`char* string() const`**

  Convert to string

- **`static const DtEventId& nullId()`**

  Return a null DtEventId

- **`void init(const DtGlobalObjectDesignator& des, int eventNum)`**
  initialize
- **`DtString myObjName`**
- **`int myEventNum`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file emittrbeamsr.h:

enum DtBeamType

- **DtRegularBeam**
- **DtRadarBeam**
- **DtTrackBeam**
- **DtJamBeam**

Documentation

- `DtRegularBeam`
- `DtRadarBeam`
- `DtTrackBeam`
- `DtJamBeam`

In file radioxmitsr.h:

class DtRadioTransmitterRepository: public DtEmbeddedSystemRepository

class DtRadioTransmitterRepository: Instances of DtRadioTransmitterRepository are

---

Inheritance:

DtRadioTransmitterRepository - DtEmbeddedSystemRepository - DtStateRepository

---

Public Methods

	DtRadioTransmitterRepository () *default constructor*
virtual	**~DtRadioTransmitterRepository** () *destructor*
	DtRadioTransmitterRepository (const DtRadioTransmitterRepository& orig) *copy constructor*
DtRadioTransmitterRepository&	**operator=** (const DtRadioTransmitterRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
virtual void	**setAntennaPattern** (const DtAntennaPatternRecord& rec) *Set/Get the antenna pattern*
virtual const DtAntennaPatternRecord&	**antennaPattern** () const
virtual void	**setAntennaPatternType** (DtAntennaPatternType pattern) *Set/Get the antenna pattern type*
virtual DtAntennaPatternType	**antennaPatternType** () const
virtual void	**setCryptoKeyId** (DtU16 key) *Set/Get the crypto key id*
virtual DtU16	**cryptoKeyId** () const
virtual void	**setCryptoMode** (DtCryptoMode mode) *Set/Get the crypto key id*
virtual DtCryptoMode	**cryptoMode** () const
virtual void	**setCryptoSystem** (DtRadioCryptoSystem system) *Set/Get the crypto system*
virtual DtRadioCryptoSystem	**cryptoSystem** () const

virtual void	**setDetailedModulationType** (DtDetailedModulationType modulation) *Set/Get the detailed modulation*
virtual DtDetailedModulationType	**detailedModulationType** () const
virtual void	**setFrequency** (unsigned long upper, unsigned long lower) *Set the frequency, either as upper and lower word, or as full 64 bit integer*
virtual void	**setFrequency** (unsigned long freq)
virtual unsigned long	**frequency** (DtLongWordPortion upperLower = DtLowerWord) const *Get either the upper or lower word of the 64 bit frequency*
virtual void	**setFrequencyBandwidth** (DtFloat32 bandwidth) *Set/Get the frequency bandwidth*
virtual DtFloat32	**frequencyBandwidth** () const
virtual void	**setFrequencyHopInUse** (DtBoolean hopInUse) *Set/Get the frequency hop in use*
virtual DtBoolean	**frequencyHopInUse** () const
virtual void	**setInputSource** (DtRadioInputSource source) *Set/Get the input source*
virtual DtRadioInputSource	**inputSource** () const
virtual void	**setModulationParams** (const void *networkRepresentation, DtU8 size) *Set/Get the modulation parameters' network representation*
virtual const void*	**modulationParamsAddr** () const
virtual DtU8	**modulationParamsLength** () const
virtual void	**setMajorModulationType** (DtRadioMajorModulation modulation) *Set/Get the major modulation type*
virtual DtRadioMajorModulation	**majorModulationType** () const
virtual void	**setModulationSystemType** (DtRadioSystem system) *Set/Get the modulation system*
virtual DtRadioSystem	**modulationSystemType** () const
virtual void	**setPower** (DtFloat32 power) *Set/Get the power*
virtual DtFloat32	**power** () const
virtual void	**setPsuedoNoiseSpectrumInUse** (DtBoolean inUse) *Set/Get the Psuedo Noise Spectrum In Use*
virtual DtBoolean	**psuedoNoiseSpectrumInUse** () const
virtual void	**setRadioId** (unsigned int number) *Set/Get the Transmitter radio number*
virtual unsigned int	**radioId** () const
virtual void	**setRadioEntityType** (const DtRadioEntityType& type) *Set/Get the radio entity type*
virtual const DtRadioEntityType&	**radioEntityType** () const
virtual void	**setTimeHopInUse** (DtBoolean inUse) *Set/Get the Time Hop In Use*
virtual DtBoolean	**timeHopInUse** () const
virtual void	**setTransmitState** (DtRadioTransmitState state) *Set/Get the transmit state*

virtual DtRadioTransmitState	**transmitState** () const
virtual void	**setWorldAntennaLoc** (DtConstVector location) *Set/Get the world location*
virtual DtConstVector	**worldAntennaLoc** () const
virtual void	**setRelativeAntennaLoc** (DtConstVector location) *Set/Get the relative location (wrapper around EmbedSysSR member func, for compatibility with DIS transmitter PDU*
virtual DtConstVector	**relativeAntennaLoc** () const
virtual void	**printData** () const
static DtRadioTransmitterRepository*	**create** ()

Protected Fields

DtAntennaPatternType	**myAntennaPatternType**
DtU16	**myCryptoKeyId**
DtCryptoMode	**myCryptoMode**
DtRadioCryptoSystem	**myCryptoSystem**
DtDetailedModulationType	**myDetailedModulation**
unsigned long	**myFreqHi**
unsigned long	**myFreqLo**
DtFloat32	**myFrequencyBandwidth**
DtBoolean	**myFrequencyHopInUse**
DtRadioInputSource	**myInputSource**
char*	**myModulationParamsAddr**
DtU8	**myModulationParamsLength**
DtAntennaPatternRecord*	**myAntPat**
DtRadioMajorModulation	**myMajorModulationType**
DtRadioSystem	**myModulationSystem**
DtFloat32	**myPower**
DtBoolean	**myPsuedoNoiseSpectrumInUse**
DtU16	**myTransmitterRadioNumber**
DtRadioEntityType	**myRadioEntityType**
DtBoolean	**myTimeHopInUse**
DtRadioTransmitState	**myTransmitState**
DtVector	**myWorldLocation**

Inherited from DtEmbeddedSystemRepository:

Public Methods

- virtual void **setEntityId**(const DtEntityIdentifier& id)
- virtual const DtEntityIdentifier& **entityId**() const
- virtual void **setHostId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **hostId**() const
- virtual void **setRelativePosition**(DtConstVector position)
- virtual DtConstVector **relativePosition**() const

Protected Fields

- DtEntityIdentifier **myEntityId**
- DtGlobalObjectDesignator **myHostId**
- DtVector **myRelativePosition**

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

-

Documentation

class DtRadioTransmitterRepository: Instances of DtRadioTransmitterRepository are

- **`DtRadioTransmitterRepository()`**
  default constructor
- **`virtual ~DtRadioTransmitterRepository()`**
  destructor
- **`DtRadioTransmitterRepository(const DtRadioTransmitterRepository& orig)`**
  copy constructor
- **`DtRadioTransmitterRepository& operator=( const DtRadioTransmitterRepository& orig)`**
  assignment operator
- **`virtual DtStateRepository* clone() const`**
  return an empty object of the same type as this
- **`virtual void setAntennaPattern(const DtAntennaPatternRecord& rec)`**
  Set/Get the antenna pattern
- **`virtual const DtAntennaPatternRecord& antennaPattern() const`**
- **`virtual void setAntennaPatternType(DtAntennaPatternType pattern)`**
  Set/Get the antenna pattern type
- **`virtual DtAntennaPatternType antennaPatternType() const`**
- **`virtual void setCryptoKeyId(DtU16 key)`**
  Set/Get the crypto key id
- **`virtual DtU16 cryptoKeyId() const`**
- **`virtual void setCryptoMode(DtCryptoMode mode)`**
  Set/Get the crypto key id
- **`virtual DtCryptoMode cryptoMode() const`**
- **`virtual void setCryptoSystem(DtRadioCryptoSystem system)`**
  Set/Get the crypto system
- **`virtual DtRadioCryptoSystem cryptoSystem() const`**
- **`virtual void setDetailedModulationType(DtDetailedModulationType modulation)`**
  Set/Get the detailed modulation
- **`virtual DtDetailedModulationType detailedModulationType() const`**
- **`virtual void setFrequency(unsigned long upper, unsigned long lower)`**
  Set the frequency, either as upper and lower word, or as full 64 bit integer
- **`virtual void setFrequency(unsigned long freq)`**

- **virtual unsigned long frequency(DtLongWordPortion upperLower = DtLowerWord) const**
  Get either the upper or lower word of the 64 bit frequency
- **virtual void setFrequencyBandwidth(DtFloat32 bandwidth)**
  Set/Get the frequency bandwidth
- **virtual DtFloat32 frequencyBandwidth() const**
- **virtual void setFrequencyHopInUse(DtBoolean hopInUse)**
  Set/Get the frequency hop in use
- **virtual DtBoolean frequencyHopInUse() const**
- **virtual void setInputSource(DtRadioInputSource source)**
  Set/Get the input source
- **virtual DtRadioInputSource inputSource() const**
- **virtual void setModulationParams(const void *networkRepresentation, DtU8 size)**
  Set/Get the modulation parameters' network representation
- **virtual const void* modulationParamsAddr() const**
- **virtual DtU8 modulationParamsLength() const**
- **virtual void setMajorModulationType(DtRadioMajorModulation modulation)**
  Set/Get the major modulation type
- **virtual DtRadioMajorModulation majorModulationType() const**
- **virtual void setModulationSystemType(DtRadioSystem system)**
  Set/Get the modulation system
- **virtual DtRadioSystem modulationSystemType() const**
- **virtual void setPower(DtFloat32 power)**
  Set/Get the power
- **virtual DtFloat32 power() const**
- **virtual void setPsuedoNoiseSpectrumInUse(DtBoolean inUse)**
  Set/Get the Psuedo Noise Spectrum In Use
- **virtual DtBoolean psuedoNoiseSpectrumInUse() const**
- **virtual void setRadioId(unsigned int number)**
  Set/Get the Transmitter radio number
- **virtual unsigned int radioId() const**
- **virtual void setRadioEntityType(const DtRadioEntityType& type)**
  Set/Get the radio entity type
- **virtual const DtRadioEntityType& radioEntityType() const**

- **virtual void setTimeHopInUse(DtBoolean inUse)**

  Set/Get the Time Hop In Use

- **virtual DtBoolean timeHopInUse() const**

- **virtual void setTransmitState(DtRadioTransmitState state)**

  Set/Get the transmit state

- **virtual DtRadioTransmitState transmitState() const**

- **virtual void setWorldAntennaLoc(DtConstVector location)**

  Set/Get the world location

- **virtual DtConstVector worldAntennaLoc() const**

- **virtual void setRelativeAntennaLoc(DtConstVector location)**

  Set/Get the relative location (wrapper around EmbedSysSR member func, for compatibility with DIS transmitter PDU

- **virtual DtConstVector relativeAntennaLoc() const**

- **virtual void printData() const**

- **static DtRadioTransmitterRepository* create()**

- **DtAntennaPatternType myAntennaPatternType**

- **DtU16 myCryptoKeyId**

- **DtCryptoMode myCryptoMode**

- **DtRadioCryptoSystem myCryptoSystem**

- **DtDetailedModulationType myDetailedModulation**

- **unsigned long myFreqHi**

- **unsigned long myFreqLo**

- **DtFloat32 myFrequencyBandwidth**

- **DtBoolean myFrequencyHopInUse**

- **DtRadioInputSource myInputSource**

- **char* myModulationParamsAddr**

- **DtU8 myModulationParamsLength**

- **DtAntennaPatternRecord* myAntPat**

- **DtRadioMajorModulation myMajorModulationType**

- **DtRadioSystem myModulationSystem**

- **DtFloat32 myPower**

- **DtBoolean myPsuedoNoiseSpectrumInUse**

- **DtU16 myTransmitterRadioNumber**
- **DtRadioEntityType myRadioEntityType**
- **DtBoolean myTimeHopInUse**
- **DtRadioTransmitState myTransmitState**
- **DtVector myWorldLocation**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file radiorecvrsr.h:

class DtRadioReceiverRepository: public DtEmbeddedSystemRepository

class DtRadioReceiverRepository: Instances of DtRadioReceiverRepository are

---

Inheritance:

DtRadioReceiverRepository - DtEmbeddedSystemRepository - DtStateRepository

---

Public Methods

	DtRadioReceiverRepository () *default constructor*
virtual	**~DtRadioReceiverRepository** () *destructor*
	DtRadioReceiverRepository (const DtRadioReceiverRepository& orig) *copy constructor*
DtRadioReceiverRepository&	**operator=** (const DtRadioReceiverRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
static DtRadioReceiverRepository*	**create** ()
virtual void	**setRadioId** (unsigned int number) *Set/Get the Receiver radio number*
virtual unsigned int	**radioId** () const
virtual void	**setReceivedPower** (DtFloat32 power) *Set/Get the received power*
virtual DtFloat32	**receivedPower** () const
virtual void	**setTransmitRadioId** (const DtGlobalObjectDesignator& id) *Set/Get the received transmitter*
virtual const DtGlobalObjectDesignator&	**transmitRadioId** () const
virtual void	**setReceiverState** (DtRadioReceiverState state) *Set/Get the receiver state*
virtual DtRadioReceiverState	**receiverState** () const
virtual void	**printData** () const

Protected Fields

- int **myReceiverRadioNumber**
- DtFloat32 **myReceivedPower**
- DtGlobalObjectDesignator **myTransmitId**
- DtRadioReceiverState **myReceiverState**

Inherited from DtEmbeddedSystemRepository:

Public Methods

- virtual void **setEntityId**(const DtEntityIdentifier& id)
- virtual const DtEntityIdentifier& **entityId**() const
- virtual void **setHostId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **hostId**() const
- virtual void **setRelativePosition**(DtConstVector position)
- virtual DtConstVector **relativePosition**() const

Protected Fields

- DtEntityIdentifier **myEntityId**
- DtGlobalObjectDesignator **myHostId**
- DtVector **myRelativePosition**

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

-

Documentation

class DtRadioReceiverRepository: Instances of DtRadioReceiverRepository are

- **`DtRadioReceiverRepository()`**
  default constructor
- **`virtual ~DtRadioReceiverRepository()`**
  destructor
- **`DtRadioReceiverRepository(const DtRadioReceiverRepository& orig)`**
  copy constructor
- **`DtRadioReceiverRepository& operator=(const DtRadioReceiverRepository& orig)`**
  assignment operator
- **`virtual DtStateRepository* clone() const`**
  return an empty object of the same type as this
- **`static DtRadioReceiverRepository* create()`**
- **`virtual void setRadioId(unsigned int number)`**
  Set/Get the Receiver radio number
- **`virtual unsigned int radioId() const`**
- **`virtual void setReceivedPower(DtFloat32 power)`**
  Set/Get the received power
- **`virtual DtFloat32 receivedPower() const`**
- **`virtual void setTransmitRadioId(const DtGlobalObjectDesignator& id)`**
  Set/Get the received transmitter
- **`virtual const DtGlobalObjectDesignator& transmitRadioId() const`**
- **`virtual void setReceiverState(DtRadioReceiverState state)`**
  Set/Get the receiver state
- **`virtual DtRadioReceiverState receiverState() const`**
- **`virtual void printData() const`**

- **int myReceiverRadioNumber**
- **DtFloat32 myReceivedPower**
- **DtGlobalObjectDesignator myTransmitId**
- **DtRadioReceiverState myReceiverState**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file emittersyssr.h:

class DtEmitterSystemRepository: public DtEmbeddedSystemRepository

class DtEmitterSystemRepository: Instances of DtEmitterSystemRepository are

---

Inheritance:

DtEmitterSystemRepository - DtEmbeddedSystemRepository - DtStateRepository

---

Public Methods

	DtEmitterSystemRepository () *default constructor*
virtual	**~DtEmitterSystemRepository** () *destructor*
	DtEmitterSystemRepository (const DtEmitterSystemRepository& orig) *copy constructor*
DtEmitterSystemRepository&	**operator=** (const DtEmitterSystemRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
static DtEmitterSystemRepository*	**create** ()
virtual void	**setEmitterFunction** (DtEmissionFunc function) *Set/Get the emitter function*
virtual DtEmissionFunc	**emitterFunction** () const
virtual void	**setEmitterName** (DtEmitterSysName name) *Set/Get the emitter name*
virtual DtEmitterSysName	**emitterName** () const
virtual void	**setEmitterNumber** (DtU8 number) *Set/Get the emitter number*
virtual DtU8	**emitterNumber** () const
virtual void	**setEventId** (const DtEventId& id) *Set/Get Event Id - not used until RPR 0*
virtual const DtEventId&	**eventId** () const
virtual void	**printData** () const

- virtual DtEmitterBeamRepository* **beamSRById** (DtU8 beamId, DtEmitterSysRetCode* retcode = NULL)
  Return a pointer to the first BeamSR with specified beamID
- virtual DtEmitterBeamRepository* **addBeam** (DtU8 beamId = 0, DtBeamType type = DtRegularBeam, DtEmitterSysRetCode *retcode = NULL)
  add/remove a new emitter beam within this system if add a beam with duplicate ID, retcode will be set to DtDuplicateBeam, and beamSRById will return the first beamSR with the given ID
- virtual DtEmitterSysRetCode **removeBeam** (DtU8 beamId)
- virtual int **numBeams** () const
  Returns number of beams
- virtual DtList* **beamList** ()
  Returns DtList of DtEmitterBeamStateRepositories
- DtBoolean **isMember** (DtEmitterBeamRepository* bsr) const
  Returns DtTrue if the give BSR is in this ESR's list of BSRs

Protected Fields

- DtEventId **myEventId**
- DtEmissionFunc **myEmitterFunction**
- DtEmitterSysName **myEmitterName**
- DtU8 **myEmitterNumber**
- DtList **myBeamList**
  list of beams in this system

Protected Methods

- virtual void **purgeBeamList** ()

Inherited from DtEmbeddedSystemRepository:

Public Methods

- virtual void **setEntityId**(const DtEntityIdentifier& id)
- virtual const DtEntityIdentifier& **entityId**() const
- virtual void **setHostId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **hostId**() const
- virtual void **setRelativePosition**(DtConstVector position)
- virtual DtConstVector **relativePosition**() const

Protected Fields

- DtEntityIdentifier **myEntityId**
- DtGlobalObjectDesignator **myHostId**
- DtVector **myRelativePosition**

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

-

Documentation

class DtEmitterSystemRepository: Instances of DtEmitterSystemRepository are

- **`DtEmitterSystemRepository()`**
  default constructor
- **`virtual ~DtEmitterSystemRepository()`**
  destructor
- **`DtEmitterSystemRepository(const DtEmitterSystemRepository& orig)`**
  copy constructor
- **`DtEmitterSystemRepository& operator=(const DtEmitterSystemRepository& orig)`**
  assignment operator
- **`virtual DtStateRepository* clone() const`**

return an empty object of the same type as this

- **static DtEmitterSystemRepository* create()**
- **virtual void setEmitterFunction(DtEmissionFunc function)**

  Set/Get the emitter function
- **virtual DtEmissionFunc emitterFunction() const**
- **virtual void setEmitterName(DtEmitterSysName name)**

  Set/Get the emitter name
- **virtual DtEmitterSysName emitterName() const**
- **virtual void setEmitterNumber(DtU8 number)**

  Set/Get the emitter number
- **virtual DtU8 emitterNumber() const**
- **virtual void setEventId(const DtEventId& id)**

  Set/Get Event Id - not used until RPR 0
- **virtual const DtEventId& eventId() const**
- **virtual void printData() const**
- **virtual DtEmitterBeamRepository* beamSRById(DtU8 beamId, DtEmitterSysRetCode* retcode = NULL)**

  Return a pointer to the first BeamSR with specified beamID
- **virtual DtEmitterBeamRepository* addBeam(DtU8 beamId = 0, DtBeamType type = DtRegularBeam, DtEmitterSysRetCode *retcode = NULL)**

  add/remove a new emitter beam within this system if add a beam with duplicate ID, retcode will be set to DtDuplicateBeam, and beamSRById will return the first beamSR with the given ID
- **virtual DtEmitterSysRetCode removeBeam(DtU8 beamId)**
- **virtual int numBeams() const**

  Returns number of beams
- **virtual DtList* beamList()**

  Returns DtList of DtEmitterBeamStateRepositories
- **DtBoolean isMember(DtEmitterBeamRepository* bsr) const**

  Returns DtTrue if the give BSR is in this ESR's list of BSRs
- **virtual void purgeBeamList()**
- **DtEventId myEventId**
- **DtEmissionFunc myEmitterFunction**
- **DtEmitterSysName myEmitterName**
- **DtU8 myEmitterNumber**

DtList myBeamList

list of beams in this system

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file designatorsr.h:

class DtDesignatorRepository: public DtEmbeddedSystemRepository

class DtDesignatorRepository: Instances of DtDesignatorRepository are

---

Inheritance:

DtDesignatorRepository - DtEmbeddedSystemRepository - DtStateRepository

---

Public Methods

	DtDesignatorRepository () *default constructor*
virtual	**~DtDesignatorRepository** () *destructor*
	DtDesignatorRepository (const DtDesignatorRepository& orig) *copy constructor*
DtDesignatorRepository&	**operator=** (const DtDesignatorRepository& orig) *assignment operator*
virtual DtStateRepository*	**clone** () const *return an empty object of the same type as this*
static DtDesignatorRepository*	**create** ()
virtual void	**printData** () const
virtual void	**setCodeName** (const DtDesignatorCodeName& name) *Set/Get the code name*
virtual const DtDesignatorCodeName&	**codeName** () const
virtual void	**setDesignatedObject** (const DtGlobalObjectDesignator& name) *Set/Get the designated object name*
virtual const DtGlobalObjectDesignator&	**designatedObject** () const
virtual void	**setDesignatorCode** (const DtDesignatorCode& code) *Set/Get the designator code*
virtual const DtDesignatorCode&	**designatorCode** () const
virtual void	**setDesignatorId** (const DtGlobalObjectDesignator& id) *Set/Get the designator object id*
virtual const DtGlobalObjectDesignator&	**designatorId** () const

- virtual void **setDesignatorPower** (DtFloat32 power)
  Set/Get the designator power
- virtual DtFloat32 **designatorPower** () const
- virtual void **setDesignatorSpotLocation** (DtConstVector location)
  Set/Get the designator spot location
- virtual DtConstVector **designatorSpotLocation** () const
- virtual void **setDesignatorWavelength** (DtFloat32 wavelength)
  Set/Get the designator wavelength
- virtual DtFloat32 **designatorWavelength** () const
- virtual void **setDRAlgorithm** (DtDeadReckonTypes algorithm)
  Set/Get the dead reckoning algorithm
- virtual DtDeadReckonTypes **dRAlgorithm** () const
- virtual void **setRelativeSpotLocation** (DtConstVector location)
  Set/Get the relative spot location
- virtual DtConstVector **relativeSpotLocation** () const
- virtual void **setSpotLinearAcceleration** (DtConstVector acceleration)
  Set/Get the spot linear acceleration
- virtual DtConstVector **spotLinearAcceleration** () const

Protected Fields

- DtDesignatorCodeName **myCodeName**
- DtGlobalObjectDesignator **myDesignatedObject**
- DtGlobalObjectDesignator **myDesignatorId**
- DtDesignatorCode **myDesignatorCode**
- DtFloat32 **myDesignatorPower**
- DtVector **myDesignatorSpotLocation**
- DtFloat32 **myDesignatorWavelength**
- DtDeadReckonTypes **myDRAlgorithm**
- DtVector **myRelativeSpotLocation**
- DtVector **mySpotLinearAcceleration**

Inherited from DtEmbeddedSystemRepository:

Public Methods

- virtual void **setEntityId**(const DtEntityIdentifier& id)
- virtual const DtEntityIdentifier& **entityId**() const
- virtual void **setHostId**(const DtGlobalObjectDesignator& id)

- virtual const DtGlobalObjectDesignator& **hostId**() const
- virtual void **setRelativePosition**(DtConstVector position)
- virtual DtConstVector **relativePosition**() const

Protected Fields

- DtEntityIdentifier **myEntityId**
- DtGlobalObjectDesignator **myHostId**
- DtVector **myRelativePosition**

Inherited from DtStateRepository:

Public Methods

- virtual void **setGlobalId**(const DtGlobalObjectDesignator& id)
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **setLocalId**(const DtObjectId& id)
- virtual const DtObjectId& **localId**() const

Protected Fields

- DtGlobalObjectDesignator **myGlobalId**
- DtObjectId **myLocalId**

Protected

-

Documentation

class DtDesignatorRepository: Instances of DtDesignatorRepository are

- **`DtDesignatorRepository()`**
  default constructor
- **`virtual ~DtDesignatorRepository()`**
  destructor
- **`DtDesignatorRepository(const DtDesignatorRepository& orig)`**
  copy constructor

- **DtDesignatorRepository& operator=(const DtDesignatorRepository& orig)**
  assignment operator
- **virtual DtStateRepository* clone() const**
  return an empty object of the same type as this
- **static DtDesignatorRepository* create()**
- **virtual void printData() const**
- **virtual void setCodeName(const DtDesignatorCodeName& name)**
  Set/Get the code name
- **virtual const DtDesignatorCodeName& codeName() const**
- **virtual void setDesignatedObject(const DtGlobalObjectDesignator& name)**
  Set/Get the designated object name
- **virtual const DtGlobalObjectDesignator& designatedObject() const**
- **virtual void setDesignatorCode(const DtDesignatorCode& code)**
  Set/Get the designator code
- **virtual const DtDesignatorCode& designatorCode() const**
- **virtual void setDesignatorId(const DtGlobalObjectDesignator& id)**
  Set/Get the designator object id. Not communicated in RPR FOM 0.7.
- **virtual const DtGlobalObjectDesignator& designatorId() const**
- **virtual void setDesignatorPower(DtFloat32 power)**
  Set/Get the designator power
- **virtual DtFloat32 designatorPower() const**
- **virtual void setDesignatorSpotLocation(DtConstVector location)**
  Set/Get the designator spot location
- **virtual DtConstVector designatorSpotLocation() const**
- **virtual void setDesignatorWavelength(DtFloat32 wavelength)**
  Set/Get the designator wavelength
- **virtual DtFloat32 designatorWavelength() const**
- **virtual void setDRAlgorithm(DtDeadReckonTypes algorithm)**
  Set/Get the dead reckoning algorithm
- **virtual DtDeadReckonTypes dRAlgorithm() const**
- **virtual void setRelativeSpotLocation(DtConstVector location)**
  Set/Get the relative spot location
- **virtual DtConstVector relativeSpotLocation() const**
- **virtual void setSpotLinearAcceleration(DtConstVector acceleration)**

Set/Get the spot linear acceleration

- **virtual DtConstVector spotLinearAcceleration() const**
- **DtDesignatorCodeName myCodeName**
- **DtGlobalObjectDesignator myDesignatedObject**
- **DtGlobalObjectDesignator myDesignatorId**
- **DtDesignatorCode myDesignatorCode**
- **DtFloat32 myDesignatorPower**
- **DtVector myDesignatorSpotLocation**
- **DtFloat32 myDesignatorWavelength**
- **DtDeadReckonTypes myDRAlgorithm**
- **DtVector myRelativeSpotLocation**
- **DtVector mySpotLinearAcceleration**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file netstructs.h:

class DtNetSimulationAddress

Public Fields

- DtNetU16 **DtSite**
- DtNetU16 **DtHost**

Public Methods

- DtSimulationAddress **operator =** (DtSimulationAddress)
- DtNetSimulationAddress **operator =** (DtNetSimulationAddress)

Documentation

- **`DtNetU16 DtSite`**
- **`DtNetU16 DtHost`**
- **`DtSimulationAddress operator = (DtSimulationAddress)`**
- **`DtNetSimulationAddress operator = (DtNetSimulationAddress)`**

This class has no child classes.

In file libmatrix.h:

class DtCoordTranslation

Public Methods

- **DtCoordTranslation** ()
- **DtCoordTranslation** (DtConstVector toOriginInFromCoord)
- void **setOffset** (DtConstVector toOriginInFromCoord)
- void **setByInverse** (DtCoordTranslation &to2from)
- void **getOffset** (DtVectorRef toOriginInFromCoord)
- void **coordTrans** (DtConstVector from, DtVectorRef to)

Private Fields

- DtVector **offset**
- *"to" origin in base coords*

Documentation

- `DtCoordTranslation()`
- `DtCoordTranslation(DtConstVector toOriginInFromCoord)`
- `void setOffset(DtConstVector toOriginInFromCoord)`
- `void setByInverse(DtCoordTranslation &to2from)`
- `void getOffset(DtVectorRef toOriginInFromCoord)`
- `void coordTrans(DtConstVector from, DtVectorRef to)`
- `DtVector offset`
- "to" origin in base coords

This class has no child classes.

In file dscvobjcbinfo.h:

class DtDiscoverObjectCbInfo

Public Methods

	DtDiscoverObjectCbInfo (RTI::ObjectClassHandle h, DtDiscoverObjectCb cb, void* usr) *constructor*
virtual	**~DtDiscoverObjectCbInfo** () *destructor*
	DtDiscoverObjectCbInfo (const DtDiscoverObjectCbInfo& orig) *copy constructor*
DtDiscoverObjectCbInfo&	**operator=** (const DtDiscoverObjectCbInfo& orig) *assignment operator*
virtual int	**match** (const DtDiscoverObjectCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtDiscoverObjectCbInfo& other) const *Check whether one callback info is equal to another*
virtual const RTI::ObjectClassHandle&	**objectClass** () const *Get the object class handle*
virtual const DtDiscoverObjectCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtDiscoverObjectCb&	**cbFunction** () const
inline const RTI::ObjectClassHandle&	**objectClass** () const
inline void*	**userData** () const

Protected Fields

RTI::ObjectClassHandle	**myClass**
DtDiscoverObjectCb	**myFunc**
void*	**myUsr**

Documentation

- **`DtDiscoverObjectCbInfo(RTI::ObjectClassHandle h, DtDiscoverObjectCb cb, void* usr)`**
  constructor

- **`virtual ~DtDiscoverObjectCbInfo()`**
  destructor
- **`DtDiscoverObjectCbInfo( const DtDiscoverObjectCbInfo& orig)`**
  copy constructor
- **`DtDiscoverObjectCbInfo& operator=( const DtDiscoverObjectCbInfo& orig)`**
  assignment operator
- **`virtual int match(const DtDiscoverObjectCbInfo* const other) const`**
  Check whether one callback info is equal to another
- **`virtual int operator==(const DtDiscoverObjectCbInfo& other) const`**
  Check whether one callback info is equal to another
- **`virtual const RTI::ObjectClassHandle& objectClass() const`**
  Get the object class handle
- **`virtual const DtDiscoverObjectCb& cbFunction() const`**
  Get the callback function
- **`virtual void* userData() const`**
  Get the user data
- **`RTI::ObjectClassHandle myClass`**
- **`DtDiscoverObjectCb myFunc`**
- **`void* myUsr`**
- **`inline const DtDiscoverObjectCb& cbFunction() const`**
- **`inline const RTI::ObjectClassHandle& objectClass() const`**
- **`inline void* userData() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file netstructs.h:

class DtNetEventId

Public Fields

- DtNetU16 **eventNum**
- DtNetGlobalObjectDesignator **objectId**

Public Methods

- **DtNetEventId** ()
- inline **DtNetEventId** ()

Protected Methods

- **DtNetEventId** (const DtNetEventId& orig)
  Copy ctor and assignment operator not implemented, since we don't want to change size when necessary
- DtNetEventId **operator=** (const DtNetEventId& orig)

Documentation

- **`DtNetEventId()`**
- **`DtNetU16 eventNum`**
- **`DtNetGlobalObjectDesignator objectId`**
- **`DtNetEventId(const DtNetEventId& orig)`**

  Copy ctor and assignment operator not implemented, since we don't want to change size when necessary. (NetTypes are sometimes used in structures, where we count on their size.)
- **`DtNetEventId operator= (const DtNetEventId& orig)`**
- **`inline DtNetEventId()`**

This class has no child classes.

In file fedreader.h:

class DtFedReader

class DtFedReader: Instances of DtFedReader are used to read a FED file, and fill in a DtFedTarget with the information read

---

Public Methods

	DtFedReader () *Constructor*
virtual	**~DtFedReader** () *Destructor*
	DtFedReader (const DtFedReader& orig) *copy constructor*
DtFedReader&	**operator=** (const DtFedReader& orig) *assignment operator*
virtual DtFedReadStatus	**readFed** (const char* fedFileName, DtFedTarget* fedTarget) *Read the FED file fedFileName, filling in intMgr and objMgr*

Protected Methods

virtual int	**findString** (FILE* fp, const char* str) *Returns 1 after reading a line containing str Returns 0 if hit EOF without finding such a line*
virtual char*	**readLine** (FILE* fp) *Read a line from the file*
virtual int	**readObjects** (FILE* fp, DtFedTarget* fedTarget) *Starting from the line after "(objects", read through the ")" that denotes the end of the objects definitions, filling in objMgr as we go*
virtual int	**readObjClass** (FILE* fp, DtFedTarget* fedTarget, char* line, const char* withinClass) *Read the description of a single object class, filling in objMgr with data read*
virtual int	**readInteractions** (FILE* fp, DtFedTarget* fedTarget) *Starting from the line after "(interactions", read through the ")" that denotes the end of the interactions definitions, filling in interMgr as we go*
virtual int	**readInterClass** (FILE* fp, DtFedTarget* fedTarget, char* line, const char* withinClass) *Read the description of a single interaction class, filling in interMgr with data read*
virtual FILE*	**openFile** (const char* fedFileName) *Open the file fedFileName*

Documentation

class DtFedReader:

Instances of DtFedReader are used to read a FED file, and fill in a DtFedTarget with the information read.

- **`DtFedReader()`**

  Constructor

- **`virtual ~DtFedReader()`**

  Destructor

- **`DtFedReader(const DtFedReader& orig)`**

  copy constructor

- **`DtFedReader& operator=(const DtFedReader& orig)`**

  assignment operator

- **`virtual DtFedReadStatus readFed(const char* fedFileName, DtFedTarget* fedTarget)`**

  Read the FED file fedFileName, filling in intMgr and objMgr

- **`virtual int findString(FILE* fp, const char* str)`**

  Returns 1 after reading a line containing str Returns 0 if hit EOF without finding such a line

- **`virtual char* readLine(FILE* fp)`**

  Read a line from the file. Returns NULL if hit EOF. Returns a ptr to static memory that is reused in subsequent calls.

- **`virtual int readObjects(FILE* fp, DtFedTarget* fedTarget)`**

  Starting from the line after "(objects", read through the ")" that denotes the end of the objects definitions, filling in objMgr as we go. Returns 0 if hit EOF during read, 1 otherwise.

- **`virtual int readObjClass(FILE* fp, DtFedTarget* fedTarget, char* line, const char* withinClass)`**

  Read the description of a single object class, filling in objMgr with data read. line should be the line that contains the "class" keyword and the class name. withinClass is the name of the class enclosing the class being read, or NULL if we are reading a base class. Returns 0 if hit EOF before finishing reading the class, 1 otherwise. Recursive calls read and process subclasses. Reads through the class-closing "0".

- **`virtual int readInteractions(FILE* fp, DtFedTarget* fedTarget)`**

  Starting from the line after "(interactions", read through the ")" that denotes the end of the interactions definitions, filling in interMgr as we go. Returns 0 if hit EOF during read, 1 otherwise.

- **`virtual int readInterClass(FILE* fp, DtFedTarget* fedTarget, char* line, const char* withinClass)`**

  Read the description of a single interaction class, filling in interMgr with data read. line should be the line that contains the "class" keyword and the class name. withinClass is the name of the class enclosing the class being read, or NULL if we are reading a base class. Returns 0 if hit EOF before finishing reading the class, 1 otherwise. Recursive calls read and process subclasses. Reads through the class-closing "0".

- **`virtual FILE* openFile(const char* fedFileName)`**

  Open the file fedFileName

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file fedreader.h:

enum DtFedReadStatus

- **DtFedReadOK**
- **DtCouldNotOpenFed**
- **DtErrorReadingFed**

Documentation

- `DtFedReadOK`
- `DtCouldNotOpenFed`
- `DtErrorReadingFed`

In file socket.h:

class DtFilterTest

Inheritance:

DtFilterTest

Public Methods

	DtFilterTest ()
virtual	**~DtFilterTest** ()
void	**disableTest** ()
void	**setTest** (DtTestOperator op, DtU32 value)
void	**extendTest** (DtU32 value)
void	**reduceTest** (DtU32 value)
virtual void	**setOperator** (DtTestOperator op)
virtual void	**setTest** (DtTestOperator op, int n, DtU32 *values)
virtual void	**extendTest** (int n, DtU32 *values)
virtual void	**reduceTest** (int n, DtU32 *values)
virtual DtBoolean	**evalTest** (char *buf, int offset, DtFilterType type)

Protected Fields

DtTestOperator	**testOp**
DtList	**values**

Protected Methods

virtual void	**clearList** ()

Documentation

- `DtFilterTest()`
- `virtual ~DtFilterTest()`
- `void disableTest()`
- `void setTest(DtTestOperator op, DtU32 value)`
- `void extendTest(DtU32 value)`
- `void reduceTest(DtU32 value)`
- `virtual void setOperator(DtTestOperator op)`
- `virtual void setTest(DtTestOperator op, int n, DtU32 *values)`
- `virtual void extendTest(int n, DtU32 *values)`
- `virtual void reduceTest(int n, DtU32 *values)`
- `virtual DtBoolean evalTest(char *buf, int offset, DtFilterType type)`
- `virtual void clearList()`
- `DtTestOperator testOp`
- `DtList values`

Direct child classes:

DtCsFilterTest

Alphabetic index *Hierarchy of classes*

In file socket.h:

enum DtTestOperator

- **Dto_EQ**
- **Dto_NEQ**

Documentation

- `Dto_EQ`
- `Dto_NEQ`

Alphabetic index *Hierarchy of classes*

In file socket.h:

enum DtFilterType

- **DtFtInetAddr**
- **DtFtNetU8**

Documentation

- `DtFtInetAddr`
- `DtFtNetU8`

In file clientsocket.h:

class DtCsFilterTest: public DtFilterTest

Inheritance:

DtCsFilterTest - DtFilterTest

Public Methods

	DtCsFilterTest (int fd, DtU8 filterType, DtU32 clientId)
virtual	**~DtCsFilterTest** ()
void	**setOperator** (DtTestOperator op)
void	**extendTest** (int n, DtU32 *values)
void	**reduceTest** (int n, DtU32 *vals)
DtBoolean	**evalTest** (char *, int, DtFilterType) *No-op*

Private Fields

- DtCsFilterMsg **filterMsg**

Inherited from DtFilterTest:

Public Methods

- void **disableTest**()
- void **setTest**(DtTestOperator op, DtU32 value)
- virtual void **setTest**(DtTestOperator op, int n, DtU32 *values)

Protected Fields

- DtTestOperator **testOp**
- DtList **values**

Protected Methods

- virtual void **clearList**()

Documentation

- **`DtCsFilterTest(int fd, DtU8 filterType, DtU32 clientId)`**
- **`virtual ~DtCsFilterTest()`**
- **`void setOperator(DtTestOperator op)`**
- **`void extendTest(int n, DtU32 *values)`**
- **`void reduceTest(int n, DtU32 *vals)`**
- **`DtBoolean evalTest(char *, int, DtFilterType)`**

  No-op. Real test done on server
- **`DtCsFilterMsg filterMsg`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file physentdec.h:

class DtPhysicalEntityDecoder: public DtBaseEntityDecoder

class DtPhysicalEntityDecoder: Instances of DtPhysicalEntityDecoder are used to decode a physical entity state message into a state repository in an HLA (as opposed to DIS) environment

---

Inheritance:

DtPhysicalEntityDecoder - DtBaseEntityDecoder - DtHlaStateDecoder

---

Public Methods

	DtPhysicalEntityDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtPhysicalEntityDecoder** () *destructor*
	DtPhysicalEntityDecoder (const DtPhysicalEntityDecoder& orig) *copy constructor*
DtPhysicalEntityDecoder&	**operator=** (const DtPhysicalEntityDecoder& orig) *assignment operator*
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtPhysicalEntityDecoder*

Protected Methods

void	**addMyDecoders** () *Add decoding functions for the attributes*
	DtDECLARE_ENTITY_ATTR_DECODER (ArticulatedParametersArray) *static decode<AttributeName> member functions*
	DtDECLARE_ENTITY_ATTR_DECODER (ArticulatedParametersCount)
	DtDECLARE_ENTITY_ATTR_DECODER (DamageState)
	DtDECLARE_ENTITY_ATTR_DECODER (EngineSmokeOn)
	DtDECLARE_ENTITY_ATTR_DECODER (FlamesPresent)
	DtDECLARE_ENTITY_ATTR_DECODER (HasFuelSupplyCap)
	DtDECLARE_ENTITY_ATTR_DECODER (HasRecoveryCap)
	DtDECLARE_ENTITY_ATTR_DECODER (HasRepairCap)

- **DtDECLARE_ENTITY_ATTR_DECODER** (HatchState)
- **DtDECLARE_ENTITY_ATTR_DECODER** (Immobilized)
- **DtDECLARE_ENTITY_ATTR_DECODER** (LifeformState)
- **DtDECLARE_ENTITY_ATTR_DECODER** (LightsState)
- **DtDECLARE_ENTITY_ATTR_DECODER** (Marking)
- **DtDECLARE_ENTITY_ATTR_DECODER** (PowerPlantOn)
- **DtDECLARE_ENTITY_ATTR_DECODER** (RampDeployed)
- **DtDECLARE_ENTITY_ATTR_DECODER** (SmokePlumePresent)
- **DtDECLARE_ENTITY_ATTR_DECODER** (TentDeployed)
- **DtDECLARE_ENTITY_ATTR_DECODER** (TrailState)
- static void **decodeArtParts** (int numParams, DtNetArtParamRecord* netParams, DtArtPartList* apList)
  called by the decoder for the ArticulatedParametersArray attribute
- static void **decodeAttParts** (int numParams, DtNetArtParamRecord* netParams, DtAttachedPartList* attList)
  called by the decoder for the ArticulatedParametersArray attribute

---

Inherited from DtBaseEntityDecoder:

Public Methods

- virtual DtHlaStateDecoder* **clone**() const

Protected Methods

- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(VelocityVector)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void(***DtAttributeDecoder**)(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

class DtPhysicalEntityDecoder: Instances of DtPhysicalEntityDecoder are used to decode a physical entity state message into a state repository in an HLA (as opposed to DIS) environment

- **DtPhysicalEntityDecoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**

  constructor

- **virtual ~DtPhysicalEntityDecoder()**

  destructor

- **DtPhysicalEntityDecoder(const DtPhysicalEntityDecoder& orig)**

  copy constructor

- **DtPhysicalEntityDecoder& operator=(const DtPhysicalEntityDecoder& orig)**

  assignment operator

- **static DtHlaStateDecoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**

  Create a DtPhysicalEntityDecoder. User is responsible for deletion.

- **void addMyDecoders()**

  Add decoding functions for the attributes. Note: Non-virtual since meant to be called from constructors to add decoders.

- **DtDECLARE_ENTITY_ATTR_DECODER(ArticulatedParametersArray)**

  static decode<AttributeName> member functions

- **DtDECLARE_ENTITY_ATTR_DECODER(ArticulatedParametersCount)**
- **DtDECLARE_ENTITY_ATTR_DECODER(DamageState)**
- **DtDECLARE_ENTITY_ATTR_DECODER(EngineSmokeOn)**
- **DtDECLARE_ENTITY_ATTR_DECODER(FlamesPresent)**
- **DtDECLARE_ENTITY_ATTR_DECODER(HasFuelSupplyCap)**
- **DtDECLARE_ENTITY_ATTR_DECODER(HasRecoveryCap)**
- **DtDECLARE_ENTITY_ATTR_DECODER(HasRepairCap)**
- **DtDECLARE_ENTITY_ATTR_DECODER(HatchState)**
- **DtDECLARE_ENTITY_ATTR_DECODER(Immobilized)**
- **DtDECLARE_ENTITY_ATTR_DECODER(LifeformState)**
- **DtDECLARE_ENTITY_ATTR_DECODER(LightsState)**
- **DtDECLARE_ENTITY_ATTR_DECODER(Marking)**
- **DtDECLARE_ENTITY_ATTR_DECODER(PowerPlantOn)**
- **DtDECLARE_ENTITY_ATTR_DECODER(RampDeployed)**
- **DtDECLARE_ENTITY_ATTR_DECODER(SmokePlumePresent)**
- **DtDECLARE_ENTITY_ATTR_DECODER(TentDeployed)**

- **`DtDECLARE_ENTITY_ATTR_DECODER(TrailState)`**

- **`static void decodeArtParts(int numParams, DtNetArtParamRecord* netParams, DtArtPartList* apList)`**

  called by the decoder for the ArticulatedParametersArray attribute

- **`static void decodeAttParts(int numParams, DtNetArtParamRecord* netParams, DtAttachedPartList* attList)`**

  called by the decoder for the ArticulatedParametersArray attribute

---

Direct child classes:

DtMilitaryEntityDecoder

Alphabetic index *Hierarchy of classes*

In file envdec.h:

class DtEnvironmentEntityDecoder: public DtBaseEntityDecoder

class DtEnvironmentEntityDecoder: Instances of DtEnvironmentEntityDecoder are used to decode an environmental entity state message into a state repository in an HLA (as opposed to DIS) environment

---

Inheritance:

DtEnvironmentEntityDecoder - DtBaseEntityDecoder - DtHlaStateDecoder

---

Public Methods

	DtEnvironmentEntityDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtEnvironmentEntityDecoder** () *destructor*
	DtEnvironmentEntityDecoder (const DtEnvironmentEntityDecoder& orig) *copy constructor*
DtEnvironmentEntityDecoder&	**operator=** (const DtEnvironmentEntityDecoder& orig) *assignment operator*
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtEnvironmentEntityDecoder*

Protected Methods

void	**addMyDecoders** () *Add decoding functions for the attributes*
	DtDECLARE_ENTITY_ATTR_DECODER (Density) *static decode<AttributeName> member functions*

Inherited from DtBaseEntityDecoder:

Public Methods

- virtual DtHlaStateDecoder* **clone**() const

Protected Methods

- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(VelocityVector)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const

- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

class DtEnvironmentEntityDecoder: Instances of DtEnvironmentEntityDecoder are used to decode an environmental entity state message into a state repository in an HLA (as opposed to DIS) environment

- **`DtEnvironmentEntityDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtEnvironmentEntityDecoder()`**

  destructor

- **`DtEnvironmentEntityDecoder(const DtEnvironmentEntityDecoder& orig)`**

  copy constructor

- **`DtEnvironmentEntityDecoder& operator=( const DtEnvironmentEntityDecoder& orig)`**

  assignment operator

- **`static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtEnvironmentEntityDecoder. User is responsible for deletion.

- **`void addMyDecoders()`**

  Add decoding functions for the attributes. Note: Non-virtual since meant to be called from constructors to add decoders.

- **`DtDECLARE_ENTITY_ATTR_DECODER(Density)`**

  static decode<AttributeName> member functions

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file aggentdec.h:

class DtAggregateEntityDecoder: public DtBaseEntityDecoder

class DtAggregateEntityDecoder: Instances of DtAggregateEntityDecoder are used to decode an aggregate state message into a state repository in an HLA (as opposed to DIS) environment

---

Inheritance:

DtAggregateEntityDecoder - DtBaseEntityDecoder - DtHlaStateDecoder

---

Public Methods

	DtAggregateEntityDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtAggregateEntityDecoder** () *destructor*
	DtAggregateEntityDecoder (const DtAggregateEntityDecoder& orig) *copy constructor*
DtAggregateEntityDecoder&	**operator=** (const DtAggregateEntityDecoder& orig) *assignment operator*
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtAggregateEntityDecoder*

Protected Methods

void	**addMyDecoders** () *Add decoding functions for the attributes*
	DtDECLARE_AGGREGATE_ATTR_DECODER (AggregateMarking) *static decode<AttributeName> member functions*
	DtDECLARE_AGGREGATE_ATTR_DECODER (AggregateState)
	DtDECLARE_AGGREGATE_ATTR_DECODER (Dimensions)
	DtDECLARE_AGGREGATE_ATTR_DECODER (EntityIDs)
	DtDECLARE_AGGREGATE_ATTR_DECODER (ForceID)
	DtDECLARE_AGGREGATE_ATTR_DECODER (Formation)
	DtDECLARE_AGGREGATE_ATTR_DECODER (NumberOfEntities)
	DtDECLARE_AGGREGATE_ATTR_DECODER (SilentAggregates)

- **DtDECLARE_AGGREGATE_ATTR_DECODER** (SilentEntities)
- **DtDECLARE_AGGREGATE_ATTR_DECODER** (SubAggregateIDs)
- **DtDECLARE_AGGREGATE_ATTR_DECODER** (VariableDatums)

Inherited from DtBaseEntityDecoder:

Public Methods

- virtual DtHlaStateDecoder* **clone**() const

Protected Methods

- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(VelocityVector)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const

- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

class DtAggregateEntityDecoder: Instances of DtAggregateEntityDecoder are used to decode an aggregate state message into a state repository in an HLA (as opposed to DIS) environment

- **`DtAggregateEntityDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtAggregateEntityDecoder()`**

  destructor

- **`DtAggregateEntityDecoder(const DtAggregateEntityDecoder& orig)`**

  copy constructor

- **`DtAggregateEntityDecoder& operator=(const DtAggregateEntityDecoder& orig)`**

  assignment operator

- **`static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtAggregateEntityDecoder. User is responsible for deletion.

- **`void addMyDecoders()`**

  Add decoding functions for the attributes. Note: Non-virtual since meant to be called from constructors to add decoders.

- **`DtDECLARE_AGGREGATE_ATTR_DECODER(AggregateMarking)`**

  static decode<AttributeName> member functions

- **`DtDECLARE_AGGREGATE_ATTR_DECODER(AggregateState)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(Dimensions)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(EntityIDs)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(ForceID)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(Formation)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(NumberOfEntities)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(SilentAggregates)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(SilentEntities)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(SubAggregateIDs)`**
- **`DtDECLARE_AGGREGATE_ATTR_DECODER(VariableDatums)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file trkjambeamdec.h:

class DtTrackJamBeamDecoder: public DtBaseEmitterBeamDecoder

Inheritance:

DtTrackJamBeamDecoder - DtBaseEmitterBeamDecoder - DtHlaStateDecoder

---

Public Methods

	DtTrackJamBeamDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtTrackJamBeamDecoder** ()
	DtTrackJamBeamDecoder (const DtTrackJamBeamDecoder& orig)
DtTrackJamBeamDecoder&	**operator=** (const DtTrackJamBeamDecoder& orig)
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

void	**addMyDecoders** ()
	DtDECLARE_TRKJAMBEAM_ATTR_DECODER (JammingModeSequence)
	DtDECLARE_TRKJAMBEAM_ATTR_DECODER (TrackJamObjectIDs)
	DtDECLARE_TRKJAMBEAM_ATTR_DECODER (TrackOrJammer)

Inherited from DtBaseEmitterBeamDecoder:

Public Methods

DtBaseEmitterBeamDecoder&	**operator=** (const DtBaseEmitterBeamDecoder& orig)

Protected Methods

- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(AzimuthCenter)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(AzimuthSweep)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamFunction)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamParameterIndex)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ElevationCenter)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ElevationSweep)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EmittingSystemID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ERP)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EntityID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EventID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(Frequency)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(FrequencyRange)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(PRF)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(PulseWidth)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(SweepSynch)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const

- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- `DtTrackJamBeamDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `virtual ~DtTrackJamBeamDecoder()`
- `DtTrackJamBeamDecoder(const DtTrackJamBeamDecoder& orig)`
- `DtTrackJamBeamDecoder& operator=(const DtTrackJamBeamDecoder& orig)`
- `static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `void addMyDecoders()`
- `DtDECLARE_TRKJAMBEAM_ATTR_DECODER(JammingModeSequence)`
- `DtDECLARE_TRKJAMBEAM_ATTR_DECODER(TrackJamObjectIDs)`
- `DtDECLARE_TRKJAMBEAM_ATTR_DECODER(TrackOrJammer)`

This class has no child classes.

In file radarbeamdec.h:

class DtRadarBeamDecoder: public DtBaseEmitterBeamDecoder

Inheritance:

DtRadarBeamDecoder - DtBaseEmitterBeamDecoder - DtHlaStateDecoder

---

Public Methods

	DtRadarBeamDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtRadarBeamDecoder** ()
	DtRadarBeamDecoder (const DtRadarBeamDecoder& orig)
DtRadarBeamDecoder&	**operator=** (const DtRadarBeamDecoder& orig)
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

void	**addMyDecoders** ()
	DtDECLARE_RADARBEAM_ATTR_DECODER (TrackObjectIDs)
	DtDECLARE_RADARBEAM_ATTR_DECODER (HighDensityTrack)

Inherited from DtBaseEmitterBeamDecoder:

Public Methods

DtBaseEmitterBeamDecoder&	**operator=** (const DtBaseEmitterBeamDecoder& orig)

Protected Methods

- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(AzimuthCenter)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(AzimuthSweep)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamFunction)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamParameterIndex)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ElevationCenter)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ElevationSweep)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EmittingSystemID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ERP)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EntityID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EventID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(Frequency)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(FrequencyRange)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(PRF)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(PulseWidth)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(SweepSynch)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const

- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- **`DtRadarBeamDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
- **`virtual ~DtRadarBeamDecoder()`**
- **`DtRadarBeamDecoder(const DtRadarBeamDecoder& orig)`**
- **`DtRadarBeamDecoder& operator=(const DtRadarBeamDecoder& orig)`**
- **`static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
- **`void addMyDecoders()`**
- **`DtDECLARE_RADARBEAM_ATTR_DECODER(TrackObjectIDs)`**
- **`DtDECLARE_RADARBEAM_ATTR_DECODER(HighDensityTrack)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file jammerbeamdec.h:

class DtJammerBeamDecoder: public DtBaseEmitterBeamDecoder

Inheritance:

DtJammerBeamDecoder - DtBaseEmitterBeamDecoder - DtHlaStateDecoder

---

Public Methods

- **DtJammerBeamDecoder** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
- virtual **~DtJammerBeamDecoder** ()
- **DtJammerBeamDecoder** (const DtJammerBeamDecoder& orig)
- DtJammerBeamDecoder& **operator=** (const DtJammerBeamDecoder& orig)
- static DtHlaStateDecoder* **create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- void **addMyDecoders** ()
- **DtDECLARE_JAMMERBEAM_ATTR_DECODER** (JammingModeSequence)
- **DtDECLARE_JAMMERBEAM_ATTR_DECODER** (JammedObjectIDs)
- **DtDECLARE_JAMMERBEAM_ATTR_DECODER** (HighDensityJam)

Inherited from DtBaseEmitterBeamDecoder:

Public Methods

- DtBaseEmitterBeamDecoder& **operator=** (const DtBaseEmitterBeamDecoder& orig)

Protected Methods

- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(AzimuthCenter)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(AzimuthSweep)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamFunction)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(BeamParameterIndex)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ElevationCenter)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ElevationSweep)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EmittingSystemID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(ERP)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EntityID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(EventID)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(Frequency)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(FrequencyRange)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(PRF)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(PulseWidth)
- **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(SweepSynch)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const

- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- **`DtJammerBeamDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
- **`virtual ~DtJammerBeamDecoder()`**
- **`DtJammerBeamDecoder(const DtJammerBeamDecoder& orig)`**
- **`DtJammerBeamDecoder& operator=(const DtJammerBeamDecoder& orig)`**
- **`static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
- **`void addMyDecoders()`**
- **`DtDECLARE_JAMMERBEAM_ATTR_DECODER(JammingModeSequence)`**
- **`DtDECLARE_JAMMERBEAM_ATTR_DECODER(JammedObjectIDs)`**
- **`DtDECLARE_JAMMERBEAM_ATTR_DECODER(HighDensityJam)`**

This class has no child classes.

In file radioxmitdec.h:

class DtRadioTransmitterDecoder: public DtBaseEmbeddedSystemDecoder

---

Inheritance:

DtRadioTransmitterDecoder - DtBaseEmbeddedSystemDecoder - DtHlaStateDecoder

---

Public Methods

	DtRadioTransmitterDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtRadioTransmitterDecoder** ()
	DtRadioTransmitterDecoder (const DtRadioTransmitterDecoder& orig)
DtRadioTransmitterDecoder&	**operator=** (const DtRadioTransmitterDecoder& orig)
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- void **addMyDecoders** ()
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (AntennaPattern)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (AntennaPatternType)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (CryptoKeyID)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (CryptoMode)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (CryptoSystem)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (DetailedModulationType)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (Frequency)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (FrequencyBandwidth)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (FrequencyHopInUse)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (InputSource)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (ModulationParameters)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (MajorModulationType)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (ModulationSystemType)

- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (Power)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (PsuedoNoiseSpectrumInUse)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (RadioNumber)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (RadioType)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (TimeHopInUse)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (TransmitState)
- **DtDECLARE_RADIOXMIT_ATTR_DECODER** (WorldLocation)

Inherited from DtBaseEmbeddedSystemDecoder:

Protected Methods

- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(EntityID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(HostObjectID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(RelativePosition)

Inherited from DtHlaStateDecoder:

Public Methods

- DtHlaStateDecoder& **operator=**(const DtHlaStateDecoder& orig)
- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const

- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- **DtRadioTransmitterDecoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtRadioTransmitterDecoder()**
- **DtRadioTransmitterDecoder(const DtRadioTransmitterDecoder& orig)**
- **DtRadioTransmitterDecoder& operator= (const DtRadioTransmitterDecoder& orig)**
- **static DtHlaStateDecoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **void addMyDecoders()**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(AntennaPattern)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(AntennaPatternType)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(CryptoKeyID)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(CryptoMode)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(CryptoSystem)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(DetailedModulationType)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(Frequency)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(FrequencyBandwidth)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(FrequencyHopInUse)**

- **DtDECLARE_RADIOXMIT_ATTR_DECODER(InputSource)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(ModulationParameters)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(MajorModulationType)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(ModulationSystemType)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(Power)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(PsuedoNoiseSpectrumInUse)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(RadioNumber)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(RadioType)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(TimeHopInUse)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(TransmitState)**
- **DtDECLARE_RADIOXMIT_ATTR_DECODER(WorldLocation)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file radiorecvrdec.h:

class DtRadioReceiverDecoder: public DtBaseEmbeddedSystemDecoder

---

Inheritance:

DtRadioReceiverDecoder - DtBaseEmbeddedSystemDecoder - DtHlaStateDecoder

---

Public Methods

	DtRadioReceiverDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtRadioReceiverDecoder** ()
	DtRadioReceiverDecoder (const DtRadioReceiverDecoder& orig)
DtRadioReceiverDecoder&	**operator=** (const DtRadioReceiverDecoder& orig)
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

void	**addMyDecoders** ()
	DtDECLARE_RADIORECVR_ATTR_DECODER (RadioNumber)
	DtDECLARE_RADIORECVR_ATTR_DECODER (ReceivedPower)
	DtDECLARE_RADIORECVR_ATTR_DECODER (ReceivedTransmitter)
	DtDECLARE_RADIORECVR_ATTR_DECODER (ReceiverState)

Inherited from DtBaseEmbeddedSystemDecoder:

Public Methods

DtBaseEmbeddedSystemDecoder&	**operator=** (const DtBaseEmbeddedSystemDecoder& orig)

Protected Methods

- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(EntityID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(HostObjectID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(RelativePosition)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**

- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- **DtRadioReceiverDecoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtRadioReceiverDecoder()**
- **DtRadioReceiverDecoder(const DtRadioReceiverDecoder& orig)**
- **DtRadioReceiverDecoder& operator=(const DtRadioReceiverDecoder& orig)**
- **static DtHlaStateDecoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **void addMyDecoders()**
- **DtDECLARE_RADIORECVR_ATTR_DECODER(RadioNumber)**
- **DtDECLARE_RADIORECVR_ATTR_DECODER(ReceivedPower)**
- **DtDECLARE_RADIORECVR_ATTR_DECODER(ReceivedTransmitter)**
- **DtDECLARE_RADIORECVR_ATTR_DECODER(ReceiverState)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file emittersysdec.h:

class DtEmitterSystemDecoder: public DtBaseEmbeddedSystemDecoder

---

Inheritance:

DtEmitterSystemDecoder - DtBaseEmbeddedSystemDecoder - DtHlaStateDecoder

---

Public Methods

	DtEmitterSystemDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtEmitterSystemDecoder** ()
	DtEmitterSystemDecoder (const DtEmitterSystemDecoder& orig)
DtEmitterSystemDecoder&	**operator=** (const DtEmitterSystemDecoder& orig)
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

void	**addMyDecoders** ()
	DtDECLARE_EMITTERSYS_ATTR_DECODER (EmitterFunction)
	DtDECLARE_EMITTERSYS_ATTR_DECODER (EmitterName)
	DtDECLARE_EMITTERSYS_ATTR_DECODER (EmitterNumber)
	DtDECLARE_EMITTERSYS_ATTR_DECODER (EventID)

Inherited from DtBaseEmbeddedSystemDecoder:

Public Methods

DtBaseEmbeddedSystemDecoder&	**operator=** (const DtBaseEmbeddedSystemDecoder& orig)

Protected Methods

- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(EntityID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(HostObjectID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(RelativePosition)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**

- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- **DtEmitterSystemDecoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtEmitterSystemDecoder()**
- **DtEmitterSystemDecoder(const DtEmitterSystemDecoder& orig)**
- **DtEmitterSystemDecoder& operator=(const DtEmitterSystemDecoder& orig)**
- **static DtHlaStateDecoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **void addMyDecoders()**
- **DtDECLARE_EMITTERSYS_ATTR_DECODER(EmitterFunction)**
- **DtDECLARE_EMITTERSYS_ATTR_DECODER(EmitterName)**
- **DtDECLARE_EMITTERSYS_ATTR_DECODER(EmitterNumber)**
- **DtDECLARE_EMITTERSYS_ATTR_DECODER(EventID)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file designatordec.h:

class DtDesignatorDecoder: public DtBaseEmbeddedSystemDecoder

---

Inheritance:

DtDesignatorDecoder - DtBaseEmbeddedSystemDecoder - DtHlaStateDecoder

---

Public Methods

	DtDesignatorDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtDesignatorDecoder** ()
	DtDesignatorDecoder (const DtDesignatorDecoder& orig)
DtDesignatorDecoder&	**operator=** (const DtDesignatorDecoder& orig)
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- void **addMyDecoders** ()
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (CodeName)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (DesignatedObject)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (DesignatorCode)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (DesignatorID)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (DesignatorPower)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (DesignatorSpotLocation)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (DesignatorWavelength)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (DRAlgorithm)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (RelativeSpotLocation)
- **DtDECLARE_DESIGNATOR_ATTR_DECODER** (SpotLinearAcceleration)

Inherited from DtBaseEmbeddedSystemDecoder:

Public Methods

- DtBaseEmbeddedSystemDecoder& **operator=** (const DtBaseEmbeddedSystemDecoder& orig)

Protected Methods

- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(EntityID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(HostObjectID)
- **DtDECLARE_EMBEDSYS_ATTR_DECODER**(RelativePosition)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void**(*DtAttributeDecoder)**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual DtHlaStateDecoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**

- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

- **DtDesignatorDecoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtDesignatorDecoder()**
- **DtDesignatorDecoder(const DtDesignatorDecoder& orig)**
- **DtDesignatorDecoder& operator=(const DtDesignatorDecoder& orig)**
- **static DtHlaStateDecoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **void addMyDecoders()**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(CodeName)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(DesignatedObject)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(DesignatorCode)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(DesignatorID)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(DesignatorPower)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(DesignatorSpotLocation)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(DesignatorWavelength)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(DRAlgorithm)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(RelativeSpotLocation)**
- **DtDECLARE_DESIGNATOR_ATTR_DECODER(SpotLinearAcceleration)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file physentenc.h:

class DtPhysicalEntityEncoder: public DtBaseEntityEncoder

class DtPhysicalEntityEncoder: Instances of DtPhysicalEntityEncoder are used to encode a physical entity's state repository into a state message in an HLA environment

---

Inheritance:

DtPhysicalEntityEncoder - DtBaseEntityEncoder - DtHlaStateEncoder

---

Public Methods

	DtPhysicalEntityEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtPhysicalEntityEncoder** () *destructor*
	DtPhysicalEntityEncoder (const DtPhysicalEntityEncoder& orig) *copy constructor*
DtPhysicalEntityEncoder&	**operator=** (const DtPhysicalEntityEncoder& orig) *assignment operator*
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtPhysicalEntityEncoder*

Protected Methods

- **DtDECLARE_ENTITY_ATTR_CHECKER** (ArticulatedParametersArray)
  static need<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_CHECKER** (ArticulatedParametersCount)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (DamageState)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (EngineSmokeOn)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (FlamesPresent)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (HasFuelSupplyCap)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (HasRecoveryCap)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (HasRepairCap)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (HatchState)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (Immobilized)

- **DtDECLARE_ENTITY_ATTR_CHECKER** (LifeformState)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (LightsState)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (Marking)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (PowerPlantOn)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (RampDeployed)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (SmokePlumePresent)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (TentDeployed)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (TrailState)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (ArticulatedParametersArray)
  static decode<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_ENCODER** (ArticulatedParametersCount)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (DamageState)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (EngineSmokeOn)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (FlamesPresent)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (HasFuelSupplyCap)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (HasRecoveryCap)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (HasRepairCap)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (HatchState)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (Immobilized)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (LifeformState)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (LightsState)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (Marking)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (PowerPlantOn)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (RampDeployed)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (SmokePlumePresent)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (TentDeployed)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (TrailState)

Inherited from DtBaseEntityEncoder:

Public Methods

- static DtBoolean **tpsiChanged**(const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)

Protected Methods

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityType)

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(VelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(VelocityVector)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)

- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

class DtPhysicalEntityEncoder: Instances of DtPhysicalEntityEncoder are used to encode a physical entity's state repository into a state message in an HLA environment

- **`DtPhysicalEntityEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtPhysicalEntityEncoder()`**

  destructor

- **`DtPhysicalEntityEncoder(const DtPhysicalEntityEncoder& orig)`**

  copy constructor

- **DtPhysicalEntityEncoder& operator=(const DtPhysicalEntityEncoder& orig)**
  assignment operator
- **static DtHlaStateEncoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
  Create a DtPhysicalEntityEncoder. Caller is responsible for deletion.
- **DtDECLARE_ENTITY_ATTR_CHECKER(ArticulatedParametersArray)**
  static need<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_CHECKER(ArticulatedParametersCount)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(DamageState)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(EngineSmokeOn)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(FlamesPresent)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(HasFuelSupplyCap)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(HasRecoveryCap)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(HasRepairCap)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(HatchState)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(Immobilized)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(LifeformState)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(LightsState)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(Marking)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(PowerPlantOn)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(RampDeployed)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(SmokePlumePresent)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(TentDeployed)**
- **DtDECLARE_ENTITY_ATTR_CHECKER(TrailState)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(ArticulatedParametersArray)**
  static decode<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_ENCODER(ArticulatedParametersCount)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(DamageState)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(EngineSmokeOn)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(FlamesPresent)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(HasFuelSupplyCap)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(HasRecoveryCap)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(HasRepairCap)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(HatchState)**

- **DtDECLARE_ENTITY_ATTR_ENCODER(Immobilized)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(LifeformState)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(LightsState)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(Marking)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(PowerPlantOn)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(RampDeployed)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(SmokePlumePresent)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(TentDeployed)**
- **DtDECLARE_ENTITY_ATTR_ENCODER(TrailState)**

Direct child classes:

DtMilitaryEntityEncoder

Alphabetic index *Hierarchy of classes*

In file envenc.h:

class DtEnvironmentEntityEncoder: public DtBaseEntityEncoder

class DtEnvironmentEntityEncoder: Instances of DtEnvironmentEntityEncoder are used to encode an environment entity's entity's state repository into a state message in an HLA environment

---

Inheritance:

DtEnvironmentEntityEncoder - DtBaseEntityEncoder - DtHlaStateEncoder

---

Public Methods

	DtEnvironmentEntityEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtEnvironmentEntityEncoder** () *destructor*
	DtEnvironmentEntityEncoder (const DtEnvironmentEntityEncoder& orig) *copy constructor*
DtEnvironmentEntityEncoder&	**operator=** (const DtEnvironmentEntityEncoder& orig) *assignment operator*
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtEnvironmentEntityEncoder*

Protected Methods

- **DtDECLARE_ENTITY_ATTR_CHECKER** (Density)
  static need<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_ENCODER** (Density)
  static decode<Attribute> member functions

Inherited from DtBaseEntityEncoder:

Public Methods

- static DtBoolean **tpsiChanged**(const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)

Protected Methods

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(VelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(VelocityVector)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)

- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

class DtEnvironmentEntityEncoder: Instances of DtEnvironmentEntityEncoder are used to encode an environment entity's entity's state repository into a state message in an HLA environment

- **`DtEnvironmentEntityEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtEnvironmentEntityEncoder()`**

  destructor

- **`DtEnvironmentEntityEncoder(const DtEnvironmentEntityEncoder& orig)`**

  copy constructor

- **`DtEnvironmentEntityEncoder& operator=( const DtEnvironmentEntityEncoder& orig)`**

  assignment operator

- **`static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtEnvironmentEntityEncoder. Caller is responsible for deletion.

- **`DtDECLARE_ENTITY_ATTR_CHECKER(Density)`**

  static need<Attribute> member functions

- **`DtDECLARE_ENTITY_ATTR_ENCODER(Density)`**

  static decode<Attribute> member functions

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file aggentenc.h:

class DtAggregateEntityEncoder: public DtBaseEntityEncoder

class DtAggregateEntityEncoder: Instances of DtAggregateEntityEncoder are used to encode an aggregate entity's state repository into a state message in an HLA environment

---

Inheritance:

DtAggregateEntityEncoder - DtBaseEntityEncoder - DtHlaStateEncoder

---

Public Methods

	DtAggregateEntityEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtAggregateEntityEncoder** () *destructor*
	DtAggregateEntityEncoder (const DtAggregateEntityEncoder& orig) *copy constructor*
DtAggregateEntityEncoder&	**operator=** (const DtAggregateEntityEncoder& orig) *assignment operator*
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtAggregateEntityEncoder*

Protected Methods

- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (AggregateMarking)
  static need<Attribute> member functions
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (AggregateState)
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (Dimensions)
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (EntityIDs)
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (ForceID)
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (Formation)
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (NumberOfEntities)
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (SilentAggregates)
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (SilentEntities)
- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (SubAggregateIDs)

- **DtDECLARE_AGGREGATE_ATTR_CHECKER** (VariableDatums)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (AggregateMarking)
  static decode<Attribute> member functions
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (AggregateState)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (Dimensions)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (EntityIDs)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (ForceID)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (Formation)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (NumberOfEntities)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (SilentAggregates)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (SilentEntities)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (SubAggregateIDs)
- **DtDECLARE_AGGREGATE_ATTR_ENCODER** (VariableDatums)

Inherited from DtBaseEntityEncoder:

Public Methods

- static DtBoolean **tpsiChanged**(const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)

Protected Methods

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(VelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityID)

- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(VelocityVector)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const

- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

class DtAggregateEntityEncoder: Instances of DtAggregateEntityEncoder are used to encode an aggregate entity's state repository into a state message in an HLA environment

- **`DtAggregateEntityEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor
- **`virtual ~DtAggregateEntityEncoder()`**

  destructor
- **`DtAggregateEntityEncoder(const DtAggregateEntityEncoder& orig)`**

  copy constructor
- **`DtAggregateEntityEncoder& operator=(const DtAggregateEntityEncoder& orig)`**

  assignment operator
- **`static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtAggregateEntityEncoder. Caller is responsible for deletion.
- **`DtDECLARE_AGGREGATE_ATTR_CHECKER(AggregateMarking)`**

  static need<Attribute> member functions
- **`DtDECLARE_AGGREGATE_ATTR_CHECKER(AggregateState)`**
- **`DtDECLARE_AGGREGATE_ATTR_CHECKER(Dimensions)`**
- **`DtDECLARE_AGGREGATE_ATTR_CHECKER(EntityIDs)`**
- **`DtDECLARE_AGGREGATE_ATTR_CHECKER(ForceID)`**
- **`DtDECLARE_AGGREGATE_ATTR_CHECKER(Formation)`**
- **`DtDECLARE_AGGREGATE_ATTR_CHECKER(NumberOfEntities)`**

- **DtDECLARE_AGGREGATE_ATTR_CHECKER(SilentAggregates)**
- **DtDECLARE_AGGREGATE_ATTR_CHECKER(SilentEntities)**
- **DtDECLARE_AGGREGATE_ATTR_CHECKER(SubAggregateIDs)**
- **DtDECLARE_AGGREGATE_ATTR_CHECKER(VariableDatums)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(AggregateMarking)**
  static decode<Attribute> member functions
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(AggregateState)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(Dimensions)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(EntityIDs)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(ForceID)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(Formation)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(NumberOfEntities)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(SilentAggregates)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(SilentEntities)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(SubAggregateIDs)**
- **DtDECLARE_AGGREGATE_ATTR_ENCODER(VariableDatums)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file trkjambeamenc.h:

class DtTrackJamBeamEncoder: public DtBaseEmitterBeamEncoder

Inheritance:

DtTrackJamBeamEncoder - DtBaseEmitterBeamEncoder - DtHlaStateEncoder

---

Public Methods

	DtTrackJamBeamEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtTrackJamBeamEncoder** ()
	DtTrackJamBeamEncoder (const DtTrackJamBeamEncoder& orig)
DtTrackJamBeamEncoder&	**operator=** (const DtTrackJamBeamEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER** (JammingModeSequence)
- **DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER** (TrackJamObjectIDs)
- **DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER** (TrackOrJammer)
- **DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER** (JammingModeSequence)
- **DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER** (TrackJamObjectIDs)
- **DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER** (TrackOrJammer)

Inherited from DtBaseEmitterBeamEncoder:

Protected Methods

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(AzimuthCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(AzimuthSweep)

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamFunction)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamParameterIndex)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ElevationCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ElevationSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EmittingSystemID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ERP)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EventID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(Frequency)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(FrequencyRange)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(PRF)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(PulseWidth)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(SweepSynch)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(AzimuthCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(AzimuthSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamFunction)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamParameterIndex)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ElevationCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ElevationSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EmittingSystemID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ERP)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EventID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(Frequency)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(FrequencyRange)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(PRF)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(PulseWidth)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(SweepSynch)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void(***DtAttributeEncoder**)(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean(***DtAttributeChecker**)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**

- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- `DtTrackJamBeamEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `virtual ~DtTrackJamBeamEncoder()`
- `DtTrackJamBeamEncoder(const DtTrackJamBeamEncoder& orig)`
- `DtTrackJamBeamEncoder& operator=(const DtTrackJamBeamEncoder& orig)`
- `static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER(JammingModeSequence)`
- `DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER(TrackJamObjectIDs)`
- `DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER(TrackOrJammer)`
- `DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER(JammingModeSequence)`
- `DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER(TrackJamObjectIDs)`
- `DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER(TrackOrJammer)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file radarbeamenc.h:

class DtRadarBeamEncoder: public DtBaseEmitterBeamEncoder

Inheritance:

DtRadarBeamEncoder - DtBaseEmitterBeamEncoder - DtHlaStateEncoder

---

Public Methods

	DtRadarBeamEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtRadarBeamEncoder** ()
	DtRadarBeamEncoder (const DtRadarBeamEncoder& orig)
DtRadarBeamEncoder&	**operator=** (const DtRadarBeamEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_RADARBEAM_ATTR_ENCODER** (TrackObjectIDs)
- **DtDECLARE_RADARBEAM_ATTR_ENCODER** (HighDensityTrack)
- **DtDECLARE_RADARBEAM_ATTR_CHECKER** (TrackObjectIDs)
- **DtDECLARE_RADARBEAM_ATTR_CHECKER** (HighDensityTrack)

Inherited from DtBaseEmitterBeamEncoder:

Protected Methods

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(AzimuthCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(AzimuthSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamFunction)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamID)

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamParameterIndex)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ElevationCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ElevationSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EmittingSystemID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ERP)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EventID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(Frequency)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(FrequencyRange)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(PRF)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(PulseWidth)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(SweepSynch)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(AzimuthCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(AzimuthSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamFunction)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamParameterIndex)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ElevationCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ElevationSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EmittingSystemID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ERP)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EventID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(Frequency)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(FrequencyRange)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(PRF)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(PulseWidth)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(SweepSynch)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void(***DtAttributeEncoder**)(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean(***DtAttributeChecker**)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**

- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- `DtRadarBeamEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `virtual ~DtRadarBeamEncoder()`
- `DtRadarBeamEncoder(const DtRadarBeamEncoder& orig)`
- `DtRadarBeamEncoder& operator=(const DtRadarBeamEncoder& orig)`
- `static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `DtDECLARE_RADARBEAM_ATTR_ENCODER(TrackObjectIDs)`
- `DtDECLARE_RADARBEAM_ATTR_ENCODER(HighDensityTrack)`
- `DtDECLARE_RADARBEAM_ATTR_CHECKER(TrackObjectIDs)`
- `DtDECLARE_RADARBEAM_ATTR_CHECKER(HighDensityTrack)`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file jammerbeamenc.h:

class DtJammerBeamEncoder: public DtBaseEmitterBeamEncoder

Inheritance:

DtJammerBeamEncoder - DtBaseEmitterBeamEncoder - DtHlaStateEncoder

---

Public Methods

	DtJammerBeamEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtJammerBeamEncoder** ()
	DtJammerBeamEncoder (const DtJammerBeamEncoder& orig)
DtJammerBeamEncoder&	**operator=** (const DtJammerBeamEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_JAMMERBEAM_ATTR_ENCODER** (JammingModeSequence)
- **DtDECLARE_JAMMERBEAM_ATTR_ENCODER** (JammedObjectIDs)
- **DtDECLARE_JAMMERBEAM_ATTR_ENCODER** (HighDensityJam)
- **DtDECLARE_JAMMERBEAM_ATTR_CHECKER** (JammingModeSequence)
- **DtDECLARE_JAMMERBEAM_ATTR_CHECKER** (JammedObjectIDs)
- **DtDECLARE_JAMMERBEAM_ATTR_CHECKER** (HighDensityJam)

Inherited from DtBaseEmitterBeamEncoder:

Protected Methods

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(AzimuthCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(AzimuthSweep)

- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamFunction)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(BeamParameterIndex)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ElevationCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ElevationSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EmittingSystemID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(ERP)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(EventID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(Frequency)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(FrequencyRange)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(PRF)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(PulseWidth)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(SweepSynch)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(AzimuthCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(AzimuthSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamFunction)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(BeamParameterIndex)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ElevationCenter)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ElevationSweep)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EmittingSystemID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(ERP)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(EventID)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(Frequency)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(FrequencyRange)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(PRF)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(PulseWidth)
- **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(SweepSynch)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void(***DtAttributeEncoder**)(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean(***DtAttributeChecker**)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**

- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- **DtJammerBeamEncoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtJammerBeamEncoder()**
- **DtJammerBeamEncoder(const DtJammerBeamEncoder& orig)**
- **DtJammerBeamEncoder& operator=(const DtJammerBeamEncoder& orig)**
- **static DtHlaStateEncoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **DtDECLARE_JAMMERBEAM_ATTR_ENCODER(JammingModeSequence)**
- **DtDECLARE_JAMMERBEAM_ATTR_ENCODER(JammedObjectIDs)**
- **DtDECLARE_JAMMERBEAM_ATTR_ENCODER(HighDensityJam)**
- **DtDECLARE_JAMMERBEAM_ATTR_CHECKER(JammingModeSequence)**
- **DtDECLARE_JAMMERBEAM_ATTR_CHECKER(JammedObjectIDs)**
- **DtDECLARE_JAMMERBEAM_ATTR_CHECKER(HighDensityJam)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file radioxmitenc.h:

class DtRadioTransmitterEncoder: public DtBaseEmbeddedSystemEncoder

---

Inheritance:

DtRadioTransmitterEncoder - DtBaseEmbeddedSystemEncoder - DtHlaStateEncoder

---

Public Methods

	DtRadioTransmitterEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtRadioTransmitterEncoder** ()
	DtRadioTransmitterEncoder (const DtRadioTransmitterEncoder& orig)
DtRadioTransmitterEncoder&	**operator=** (const DtRadioTransmitterEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (AntennaPattern)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (AntennaPatternType)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (CryptoKeyID)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (CryptoMode)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (CryptoSystem)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (DetailedModulationType)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (Frequency)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (FrequencyBandwidth)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (FrequencyHopInUse)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (InputSource)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (ModulationParameters)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (MajorModulationType)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (ModulationSystemType)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (Power)

- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (PsuedoNoiseSpectrumInUse)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (RadioNumber)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (RadioType)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (TimeHopInUse)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (TransmitState)
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER** (WorldLocation)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (AntennaPattern)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (AntennaPatternType)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (CryptoKeyID)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (CryptoMode)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (CryptoSystem)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (DetailedModulationType)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (Frequency)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (FrequencyBandwidth)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (FrequencyHopInUse)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (InputSource)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (ModulationParameters)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (MajorModulationType)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (ModulationSystemType)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (Power)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (PsuedoNoiseSpectrumInUse)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (RadioNumber)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (RadioType)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (TimeHopInUse)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (TransmitState)
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER** (WorldLocation)

Inherited from DtBaseEmbeddedSystemEncoder:

Protected Methods

- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(RelativePosition)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(RelativePosition)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void(***DtAttributeEncoder**)(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean(***DtAttributeChecker**)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**

- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- `DtRadioTransmitterEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `virtual ~DtRadioTransmitterEncoder()`
- `DtRadioTransmitterEncoder(const DtRadioTransmitterEncoder& orig)`
- `DtRadioTransmitterEncoder& operator=(const DtRadioTransmitterEncoder& orig)`
- `static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(AntennaPattern)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(AntennaPatternType)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(CryptoKeyID)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(CryptoMode)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(CryptoSystem)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(DetailedModulationType)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(Frequency)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(FrequencyBandwidth)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(FrequencyHopInUse)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(InputSource)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(ModulationParameters)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(MajorModulationType)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(ModulationSystemType)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(Power)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(PsuedoNoiseSpectrumInUse)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(RadioNumber)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(RadioType)`
- `DtDECLARE_RADIOXMIT_ATTR_CHECKER(TimeHopInUse)`

- **DtDECLARE_RADIOXMIT_ATTR_CHECKER(TransmitState)**
- **DtDECLARE_RADIOXMIT_ATTR_CHECKER(WorldLocation)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(AntennaPattern)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(AntennaPatternType)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(CryptoKeyID)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(CryptoMode)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(CryptoSystem)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(DetailedModulationType)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(Frequency)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(FrequencyBandwidth)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(FrequencyHopInUse)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(InputSource)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(ModulationParameters)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(MajorModulationType)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(ModulationSystemType)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(Power)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(PsuedoNoiseSpectrumInUse)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(RadioNumber)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(RadioType)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(TimeHopInUse)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(TransmitState)**
- **DtDECLARE_RADIOXMIT_ATTR_ENCODER(WorldLocation)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file radiorecvrenc.h:

class DtRadioReceiverEncoder: public DtBaseEmbeddedSystemEncoder

---

Inheritance:

DtRadioReceiverEncoder - DtBaseEmbeddedSystemEncoder - DtHlaStateEncoder

---

Public Methods

	DtRadioReceiverEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtRadioReceiverEncoder** ()
	DtRadioReceiverEncoder (const DtRadioReceiverEncoder& orig)
DtRadioReceiverEncoder&	**operator=** (const DtRadioReceiverEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_RADIORECVR_ATTR_CHECKER** (RadioNumber)
- **DtDECLARE_RADIORECVR_ATTR_CHECKER** (ReceivedPower)
- **DtDECLARE_RADIORECVR_ATTR_CHECKER** (ReceivedTransmitter)
- **DtDECLARE_RADIORECVR_ATTR_CHECKER** (ReceiverState)
- **DtDECLARE_RADIORECVR_ATTR_ENCODER** (RadioNumber)
- **DtDECLARE_RADIORECVR_ATTR_ENCODER** (ReceivedPower)
- **DtDECLARE_RADIORECVR_ATTR_ENCODER** (ReceivedTransmitter)
- **DtDECLARE_RADIORECVR_ATTR_ENCODER** (ReceiverState)

Inherited from DtBaseEmbeddedSystemEncoder:

Protected Methods

- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(RelativePosition)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(RelativePosition)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)

- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- **DtRadioReceiverEncoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtRadioReceiverEncoder()**
- **DtRadioReceiverEncoder(const DtRadioReceiverEncoder& orig)**
- **DtRadioReceiverEncoder& operator=(const DtRadioReceiverEncoder& orig)**
- **static DtHlaStateEncoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **DtDECLARE_RADIORECVR_ATTR_CHECKER(RadioNumber)**
- **DtDECLARE_RADIORECVR_ATTR_CHECKER(ReceivedPower)**
- **DtDECLARE_RADIORECVR_ATTR_CHECKER(ReceivedTransmitter)**
- **DtDECLARE_RADIORECVR_ATTR_CHECKER(ReceiverState)**
- **DtDECLARE_RADIORECVR_ATTR_ENCODER(RadioNumber)**
- **DtDECLARE_RADIORECVR_ATTR_ENCODER(ReceivedPower)**
- **DtDECLARE_RADIORECVR_ATTR_ENCODER(ReceivedTransmitter)**
- **DtDECLARE_RADIORECVR_ATTR_ENCODER(ReceiverState)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file emittersysenc.h:

class DtEmitterSystemEncoder: public DtBaseEmbeddedSystemEncoder

---

Inheritance:

DtEmitterSystemEncoder - DtBaseEmbeddedSystemEncoder - DtHlaStateEncoder

---

Public Methods

	DtEmitterSystemEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtEmitterSystemEncoder** ()
	DtEmitterSystemEncoder (const DtEmitterSystemEncoder& orig)
DtEmitterSystemEncoder&	**operator=** (const DtEmitterSystemEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_EMITTERSYS_ATTR_CHECKER** (EmitterFunction)
- **DtDECLARE_EMITTERSYS_ATTR_CHECKER** (EmitterName)
- **DtDECLARE_EMITTERSYS_ATTR_CHECKER** (EmitterNumber)
- **DtDECLARE_EMITTERSYS_ATTR_CHECKER** (EventID)
- **DtDECLARE_EMITTERSYS_ATTR_ENCODER** (EmitterFunction)
- **DtDECLARE_EMITTERSYS_ATTR_ENCODER** (EmitterName)
- **DtDECLARE_EMITTERSYS_ATTR_ENCODER** (EmitterNumber)
- **DtDECLARE_EMITTERSYS_ATTR_ENCODER** (EventID)

Inherited from DtBaseEmbeddedSystemEncoder:

Protected Methods

- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(RelativePosition)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(RelativePosition)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)

- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- **DtEmitterSystemEncoder(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **virtual ~DtEmitterSystemEncoder()**
- **DtEmitterSystemEncoder(const DtEmitterSystemEncoder& orig)**
- **DtEmitterSystemEncoder& operator=(const DtEmitterSystemEncoder& orig)**
- **static DtHlaStateEncoder* create(DtExerciseConn* exConn, DtObjClassDesc* classDesc)**
- **DtDECLARE_EMITTERSYS_ATTR_CHECKER(EmitterFunction)**
- **DtDECLARE_EMITTERSYS_ATTR_CHECKER(EmitterName)**
- **DtDECLARE_EMITTERSYS_ATTR_CHECKER(EmitterNumber)**
- **DtDECLARE_EMITTERSYS_ATTR_CHECKER(EventID)**
- **DtDECLARE_EMITTERSYS_ATTR_ENCODER(EmitterFunction)**
- **DtDECLARE_EMITTERSYS_ATTR_ENCODER(EmitterName)**
- **DtDECLARE_EMITTERSYS_ATTR_ENCODER(EmitterNumber)**
- **DtDECLARE_EMITTERSYS_ATTR_ENCODER(EventID)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file designatorenc.h:

class DtDesignatorEncoder: public DtBaseEmbeddedSystemEncoder

---

Inheritance:

DtDesignatorEncoder - DtBaseEmbeddedSystemEncoder - DtHlaStateEncoder

---

Public Methods

	DtDesignatorEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc)
virtual	**~DtDesignatorEncoder** ()
	DtDesignatorEncoder (const DtDesignatorEncoder& orig)
DtDesignatorEncoder&	**operator=** (const DtDesignatorEncoder& orig)
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc)

Protected Methods

- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (CodeName)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (DesignatedObject)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (DesignatorCode)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (DesignatorID)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (DesignatorPower)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (DesignatorSpotLocation)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (DesignatorWavelength)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (DRAlgorithm)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (RelativeSpotLocation)
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER** (SpotLinearAcceleration)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (CodeName)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (DesignatedObject)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (DesignatorCode)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (DesignatorID)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (DesignatorPower)

- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (DesignatorSpotLocation)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (DesignatorWavelength)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (DRAlgorithm)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (RelativeSpotLocation)
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER** (SpotLinearAcceleration)

Inherited from DtBaseEmbeddedSystemEncoder:

Protected Methods

- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(RelativePosition)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(HostObjectID)
- **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(RelativePosition)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void**(*DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean**(*DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const

- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

- **`DtDesignatorEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
- **`virtual ~DtDesignatorEncoder()`**
- **`DtDesignatorEncoder(const DtDesignatorEncoder& orig)`**
- **`DtDesignatorEncoder& operator=(const DtDesignatorEncoder& orig)`**
- **`static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
- **`DtDECLARE_DESIGNATOR_ATTR_CHECKER(CodeName)`**
- **`DtDECLARE_DESIGNATOR_ATTR_CHECKER(DesignatedObject)`**
- **`DtDECLARE_DESIGNATOR_ATTR_CHECKER(DesignatorCode)`**

- **DtDECLARE_DESIGNATOR_ATTR_CHECKER(DesignatorID)**
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER(DesignatorPower)**
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER(DesignatorSpotLocation)**
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER(DesignatorWavelength)**
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER(DRAlgorithm)**
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER(RelativeSpotLocation)**
- **DtDECLARE_DESIGNATOR_ATTR_CHECKER(SpotLinearAcceleration)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(CodeName)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(DesignatedObject)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(DesignatorCode)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(DesignatorID)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(DesignatorPower)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(DesignatorSpotLocation)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(DesignatorWavelength)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(DRAlgorithm)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(RelativeSpotLocation)**
- **DtDECLARE_DESIGNATOR_ATTR_ENCODER(SpotLinearAcceleration)**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file globidhlst.h:

class DtGlobalIdHashlist

class DtGlobalIdHashlist: Instances of DtGlobalIdHashlist are hashlists, where you can store associations between DtGlobalObjectDesignators and arbitrary pieces of data

---

Public Methods

	DtGlobalIdHashlist (DtBoolean intrusive = DtFalse) *default constructor*
virtual	**~DtGlobalIdHashlist** () *destructor*
virtual DtListItem*	**add** (const DtGlobalObjectDesignator& key, void* data) *Add an association between key and data to the list*
virtual void*	**remove** (const DtGlobalObjectDesignator& key) *Remove the current association involving key from the list*
virtual void*	**lookup** (const DtGlobalObjectDesignator& key) const *Lookup the data associated with key*
virtual void*	**operator[]** (const DtGlobalObjectDesignator& key) const
virtual const DtList*	**list** () const *Get a DtList containing all of the void* data that you have listed, regardless of keys*
inline void*	**operator[]** (const DtGlobalObjectDesignator& key) const

Public

In the event that an item with key already exists, then add will replace the data and move it to the location in list

Protected Fields

DtList*	**myList**
DtHashlist*	**myHashlist**

Protected Methods

	DtGlobalIdHashlist (const DtGlobalIdHashlist& orig) *copy constructor*
DtGlobalIdHashlist&	**operator=** (const DtGlobalIdHashlist& orig) *assignment operator*

Documentation

class DtGlobalIdHashlist: Instances of DtGlobalIdHashlist are hashlists, where you can store associations between DtGlobalObjectDesignators and arbitrary pieces of data

- **DtGlobalIdHashlist(DtBoolean intrusive = DtFalse)**

  default constructor. Pass DtTrue only if the data that you are listing are instances of a class derived from DtListedObject.

- **virtual ~DtGlobalIdHashlist()**

  destructor

-

  In the event that an item with key already exists, then add will replace the data and move it to the location in list. If your data was alloced, do not add unless you know key is unique, or a memory leak will result.

- **virtual DtListItem* add(const DtGlobalObjectDesignator& key, void* data)**

  Add an association between key and data to the list

- **virtual void* remove(const DtGlobalObjectDesignator& key)**

  Remove the current association involving key from the list. Returns the data that was associated with key.

- **virtual void* lookup(const DtGlobalObjectDesignator& key) const**

  Lookup the data associated with key

- **virtual void* operator[](const DtGlobalObjectDesignator& key) const**

- **virtual const DtList* list() const**

  Get a DtList containing all of the void* data that you have listed, regardless of keys

- **DtGlobalIdHashlist(const DtGlobalIdHashlist& orig)**

  copy constructor

- **DtGlobalIdHashlist& operator=(const DtGlobalIdHashlist& orig)**

  assignment operator

- **DtList* myList**

- **DtHashlist* myHashlist**

- **inline void* operator[](const DtGlobalObjectDesignator& key) const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hashitem.h:

class DtHashItem

class DtHashItem: Instances of DtHashItem are used internally to a hashlist to associate a pointer to an item in a list with a key

---

Public Methods

	DtHashItem (const DtBaseHashKey& key, DtListItem* item) *default constructor*
virtual	**~DtHashItem** () *destructor*
	DtHashItem (const DtHashItem& orig) *copy constructor*
DtHashItem&	**operator=** (const DtHashItem& orig) *assignment operator*
virtual const DtBaseHashKey&	**hashKey** () const *get the key for the hashed item*
virtual DtListItem*	**listItem** () const *get the list item*

Protected Fields

DtBaseHashKey*	**myKey**
DtListItem*	**myItem**

Documentation

class DtHashItem: Instances of DtHashItem are used internally to a hashlist to associate a pointer to an item in a list with a key

`DtHashItem(const DtBaseHashKey& key, DtListItem* item)`

default constructor

`virtual ~DtHashItem()`

destructor

`DtHashItem(const DtHashItem& orig)`

copy constructor

`DtHashItem& operator=(const DtHashItem& orig)`

assignment operator

- **`virtual const DtBaseHashKey& hashKey() const`**

  get the key for the hashed item

- **`virtual DtListItem* listItem() const`**

  get the list item

- **`DtBaseHashKey* myKey`**

- **`DtListItem* myItem`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtAntennaPatternType

- **DtOmniDirectional**
- **DtBeam**
- **DtSphericalHarmonic**

Documentation

- `DtOmniDirectional`
- `DtBeam`
- `DtSphericalHarmonic`

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtCryptoMode

- **DtBasebandEncryption**
- **DtDiphaseEncryption**

Documentation

- `DtBasebandEncryption`
- `DtDiphaseEncryption`

In file disenums.h:

enum DtRadioCryptoSystem

- **DtRadioCryptoSysOther**
- **DtRadioCryptoSysKY_28**
- **DtRadioCryptoSysKY_8**
- **DtRadioCryptoSysNarrowSpectrumSecureVoice_NSVE**
- **DtRadioCryptoSysWideSpectrumSecureVoice_WSVE**
- **DtRadioCryptoSysSINCGARS_ICOM**

Documentation

- `DtRadioCryptoSysOther`
- `DtRadioCryptoSysKY_28`
- `DtRadioCryptoSysKY_8`
- `DtRadioCryptoSysNarrowSpectrumSecureVoice_NSVE`
- `DtRadioCryptoSysWideSpectrumSecureVoice_WSVE`
- `DtRadioCryptoSysSINCGARS_ICOM`

In file disenums.h:

enum DtDetailedModulationType

DtDetailedModulationDummy

Documentation

`DtDetailedModulationDummy`

In file radioxmitsr.h:

enum DtLongWordPortion

- **DtUpperWord**
- **DtLowerWord**

Documentation

- `DtUpperWord`
- `DtLowerWord`

In file disenums.h:

enum DtRadioInputSource

- **DtRadioInputOther**
- **DtPilot**
- **DtCoPilot**
- **DtFirstOfficer**
- **DtDriver**
- **DtLoader**
- **DtGunner**
- **DtCommander**
- **DtDigitalDataDevice**
- **DtIntercom**

Documentation

- `DtRadioInputOther`
- `DtPilot`
- `DtCoPilot`
- `DtFirstOfficer`
- `DtDriver`
- `DtLoader`
- `DtGunner`
- `DtCommander`
- `DtDigitalDataDevice`
- `DtIntercom`

In file disenums.h:

enum DtRadioMajorModulation

- **DtRadioMajorModOther**
- **DtAmplitude**
- **DtAmplitudeAndAngle**
- **DtAngle**
- **DtCombination**
- **DtPulse**
- **DtUnmodulated**

Documentation

- `DtRadioMajorModOther`
- `DtAmplitude`
- `DtAmplitudeAndAngle`
- `DtAngle`
- `DtCombination`
- `DtPulse`
- `DtUnmodulated`

In file disenums.h:

enum DtRadioSystem

- **DtRadioSystemOther**
- **DtGeneric**
- **DtHq**
- **DtHqII**
- **DtHqIIA**
- **DtSincgards**

Documentation

- `DtRadioSystemOther`
- `DtGeneric`
- `DtHq`
- `DtHqII`
- `DtHqIIA`
- `DtSincgards`

In file disenums.h:

enum DtRadioTransmitState

- **DtOff**
- **DtRadioOnNotTrasmitting**
- **DtOn**

Documentation

- `DtOff`
- `DtRadioOnNotTrasmitting`
- `DtOn`

In file disenums.h:

enum DtRadioReceiverState

- **DtRadioOff**
- **DtRadioOnNotReceiving**
- **DtRadioOn**

Documentation

- `DtRadioOff`
- `DtRadioOnNotReceiving`
- `DtRadioOn`

In file disenums.h:

enum DtEmissionFunc

- **DtEmissionFuncOther**
- **DtEmissionFuncMultiFunction**
- **DtEmissionFuncEarlyWarningSurveillance**
- **DtEmissionFuncHeightFinding**
- **DtEmissionFuncFireControl**
- **DtEmissionFuncAcquisitionDetection**
- **DtEmissionFuncTracking**
- **DtEmissionFuncGuidanceIllumination**
- **DtEmissionFuncFiringPointLaunchPointLocation**
- **DtEmissionFuncRanging**
- **DtEmissionFuncRadarAltimeter**
- **DtEmissionFuncImaging**
- **DtEmissionFuncMotionDetection**
- **DtEmissionFuncNavigation**
- **DtEmissionFuncWeather**
- **DtEmissionFuncInstrumentation**
- **DtEmissionFuncIdentificationClassification**
- **DtEmissionFuncJammingNoise**
- **DtEmissionFuncJammingDeception**
- **DtEmissionFuncDecoy**
- **DtEmissionFuncWeaponNonLethal**
- **DtEmissionFuncWeaponLethal**

Documentation

- **DtEmissionFuncOther**
- **DtEmissionFuncMultiFunction**
- **DtEmissionFuncEarlyWarningSurveillance**
- **DtEmissionFuncHeightFinding**
- **DtEmissionFuncFireControl**
- **DtEmissionFuncAcquisitionDetection**
- **DtEmissionFuncTracking**
- **DtEmissionFuncGuidanceIllumination**
- **DtEmissionFuncFiringPointLaunchPointLocation**
- **DtEmissionFuncRanging**
- **DtEmissionFuncRadarAltimeter**
- **DtEmissionFuncImaging**
- **DtEmissionFuncMotionDetection**
- **DtEmissionFuncNavigation**
- **DtEmissionFuncWeather**
- **DtEmissionFuncInstrumentation**
- **DtEmissionFuncIdentificationClassification**
- **DtEmissionFuncJammingNoise**
- **DtEmissionFuncJammingDeception**
- **DtEmissionFuncDecoy**
- **DtEmissionFuncWeaponNonLethal**
- **DtEmissionFuncWeaponLethal**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtEmitterSysName

DtEmitterSysOther

Documentation

`DtEmitterSysOther`

In file emittersyssr.h:

enum DtEmitterSysRetCode

error codes for looking up individual beams

- **DtEmitterSuccess**
- **DtBadBeamId**
- **DtDuplicateBeamId**

Documentation

error codes for looking up individual beams

- **`DtEmitterSuccess`**
- **`DtBadBeamId`**
- **`DtDuplicateBeamId`**

In file disenums.h:

enum DtDesignatorCodeName

- **DtDesignatorNameOther**
- **DtDesignatorNameTBD**

Documentation

- `DtDesignatorNameOther`
- `DtDesignatorNameTBD`

In file disenums.h:

enum DtDesignatorCode

DtDesignatorCodeOther

Documentation

`DtDesignatorCodeOther`

In file milentenc.h:

class DtMilitaryEntityEncoder: public DtPhysicalEntityEncoder

class DtMilitaryEntityEncoder: Instances of DtMilitaryEntityEncoder are used to encode a military entity's state repository into a state message in an HLA environment

---

Inheritance:

DtMilitaryEntityEncoder - DtPhysicalEntityEncoder - DtBaseEntityEncoder - DtHlaStateEncoder

---

Public Methods

	DtMilitaryEntityEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtMilitaryEntityEncoder** () *destructor*
	DtMilitaryEntityEncoder (const DtMilitaryEntityEncoder& orig) *copy constructor*
DtMilitaryEntityEncoder&	**operator=** (const DtMilitaryEntityEncoder& orig) *assignment operator*
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtMilitaryEntityEncoder*

Protected Methods

- **DtDECLARE_ENTITY_ATTR_CHECKER** (AlternateEntityType)
  static need<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_CHECKER** (CamouflageType)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (FirePowerDisabled)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (ForceID)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (IsConcealed)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (AlternateEntityType)
  static decode<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_ENCODER** (CamouflageType)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (FirePowerDisabled)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (ForceID)

- **DtDECLARE_ENTITY_ATTR_ENCODER** (IsConcealed)

Inherited from DtPhysicalEntityEncoder:

Inherited from DtBaseEntityEncoder:

Public Methods

- static DtBoolean **tpsiChanged**(const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)

Protected Methods

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(VelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(VelocityVector)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void(***DtAttributeEncoder)**(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean(***DtAttributeChecker)**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**

- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

class DtMilitaryEntityEncoder: Instances of DtMilitaryEntityEncoder are used to encode a military entity's state repository into a state message in an HLA environment

- **`DtMilitaryEntityEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
  constructor
- **`virtual ~DtMilitaryEntityEncoder()`**
  destructor
- **`DtMilitaryEntityEncoder(const DtMilitaryEntityEncoder& orig)`**
  copy constructor
- **`DtMilitaryEntityEncoder& operator=(const DtMilitaryEntityEncoder& orig)`**
  assignment operator
- **`static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**
  Create a DtMilitaryEntityEncoder. Caller is responsible for deletion.
- **`DtDECLARE_ENTITY_ATTR_CHECKER(AlternateEntityType)`**
  static need<Attribute> member functions
- **`DtDECLARE_ENTITY_ATTR_CHECKER(CamouflageType)`**
- **`DtDECLARE_ENTITY_ATTR_CHECKER(FirePowerDisabled)`**
- **`DtDECLARE_ENTITY_ATTR_CHECKER(ForceID)`**
- **`DtDECLARE_ENTITY_ATTR_CHECKER(IsConcealed)`**
- **`DtDECLARE_ENTITY_ATTR_ENCODER(AlternateEntityType)`**
  static decode<Attribute> member functions
- **`DtDECLARE_ENTITY_ATTR_ENCODER(CamouflageType)`**
- **`DtDECLARE_ENTITY_ATTR_ENCODER(FirePowerDisabled)`**
- **`DtDECLARE_ENTITY_ATTR_ENCODER(ForceID)`**
- **`DtDECLARE_ENTITY_ATTR_ENCODER(IsConcealed)`**

Direct child classes:

DtSoldierEncoder

DtMunitionEntityEncoder
DtMilitaryPlatformEntityEncoder

Alphabetic index *Hierarchy of classes*

In file milentdec.h:

class DtMilitaryEntityDecoder: public DtPhysicalEntityDecoder

class DtMilitaryEntityDecoder: Instances of DtMilitaryEntityDecoder are used to decode a military entity state message into a state repository in an HLA (as opposed to DIS) environment

---

Inheritance:

DtMilitaryEntityDecoder - DtPhysicalEntityDecoder - DtBaseEntityDecoder - DtHlaStateDecoder

---

Public Methods

	DtMilitaryEntityDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtMilitaryEntityDecoder** () *destructor*
	DtMilitaryEntityDecoder (const DtMilitaryEntityDecoder& orig) *copy constructor*
DtMilitaryEntityDecoder&	**operator=** (const DtMilitaryEntityDecoder& orig) *assignment operator*
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtMilitaryEntityDecoder*

Protected Methods

void	**addMyDecoders** () *Add decoding functions for the attributes*
	DtDECLARE_ENTITY_ATTR_DECODER (AlternateEntityType) *static decode<AttributeName> member functions*
	DtDECLARE_ENTITY_ATTR_DECODER (CamouflageType)
	DtDECLARE_ENTITY_ATTR_DECODER (FirePowerDisabled)
	DtDECLARE_ENTITY_ATTR_DECODER (ForceID)
	DtDECLARE_ENTITY_ATTR_DECODER (IsConcealed)

Inherited from DtPhysicalEntityDecoder:

Protected Methods

- static void **decodeArtParts**(int numParams, DtNetArtParamRecord* netParams, DtArtPartList* apList)
- static void **decodeAttParts**(int numParams, DtNetArtParamRecord* netParams, DtAttachedPartList* attList)

Inherited from DtBaseEntityDecoder:

Public Methods

- virtual DtHlaStateDecoder* **clone**() const

Protected Methods

- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(VelocityVector)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void(***DtAttributeDecoder**)(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)

- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

class DtMilitaryEntityDecoder: Instances of DtMilitaryEntityDecoder are used to decode a military entity state message into a state repository in an HLA (as opposed to DIS) environment

- **`DtMilitaryEntityDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtMilitaryEntityDecoder()`**

  destructor

- **`DtMilitaryEntityDecoder(const DtMilitaryEntityDecoder& orig)`**

  copy constructor

- **`DtMilitaryEntityDecoder& operator=(const DtMilitaryEntityDecoder& orig)`**

assignment operator

- **`static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtMilitaryEntityDecoder. User is responsible for deletion.

- **`void addMyDecoders()`**

  Add decoding functions for the attributes. Note: Non-virtual since meant to be called from constructors to add decoders.

- **`DtDECLARE_ENTITY_ATTR_DECODER(AlternateEntityType)`**

  static decode<AttributeName> member functions

- **`DtDECLARE_ENTITY_ATTR_DECODER(CamouflageType)`**
- **`DtDECLARE_ENTITY_ATTR_DECODER(FirePowerDisabled)`**
- **`DtDECLARE_ENTITY_ATTR_DECODER(ForceID)`**
- **`DtDECLARE_ENTITY_ATTR_DECODER(IsConcealed)`**

Direct child classes:

DtSoldierDecoder
DtMunitionEntityDecoder
DtMilitaryPlatformEntityDecoder

Alphabetic index *Hierarchy of classes*

In file soldierdec.h:

class DtSoldierDecoder: public DtMilitaryEntityDecoder

class DtSoldierDecoder: Instances of DtSoldierDecoder are used to decode a soldier entity state message into a state repository in an HLA (as opposed to DIS) environment

---

Inheritance:

DtSoldierDecoder - DtMilitaryEntityDecoder - DtPhysicalEntityDecoder - DtBaseEntityDecoder - DtHlaStateDecoder

---

Public Methods

	DtSoldierDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtSoldierDecoder** () *destructor*
	DtSoldierDecoder (const DtSoldierDecoder& orig) *copy constructor*
DtSoldierDecoder&	**operator=** (const DtSoldierDecoder& orig) *assignment operator*
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtSoldierDecoder*

Protected Methods

void	**addMyDecoders** () *Add decoding functions for the attributes*
	DtDECLARE_ENTITY_ATTR_DECODER (PrimaryWeaponState) *static decode<AttributeName> member functions*
	DtDECLARE_ENTITY_ATTR_DECODER (SecondaryWeaponState)

Inherited from DtMilitaryEntityDecoder:

Inherited from DtPhysicalEntityDecoder:

Protected Methods

- static void **decodeArtParts**(int numParams, DtNetArtParamRecord* netParams, DtArtPartList* apList)
- static void **decodeAttParts**(int numParams, DtNetArtParamRecord* netParams, DtAttachedPartList* attList)

Inherited from DtBaseEntityDecoder:

Public Methods

- virtual DtHlaStateDecoder* **clone**() const

Protected Methods

- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(VelocityVector)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void(***DtAttributeDecoder**)(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)

- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

class DtSoldierDecoder: Instances of DtSoldierDecoder are used to decode a soldier entity state message into a state repository in an HLA (as opposed to DIS) environment

- **`DtSoldierDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtSoldierDecoder()`**

  destructor

- **`DtSoldierDecoder(const DtSoldierDecoder& orig)`**

  copy constructor

- **`DtSoldierDecoder& operator=(const DtSoldierDecoder& orig)`**

  assignment operator

- **`static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtSoldierDecoder. User is responsible for deletion.

- **`void addMyDecoders()`**

  Add decoding functions for the attributes. Note: Non-virtual since meant to be called from constructors to add decoders.

- **`DtDECLARE_ENTITY_ATTR_DECODER(PrimaryWeaponState)`**

  static decode<AttributeName> member functions

- **`DtDECLARE_ENTITY_ATTR_DECODER(SecondaryWeaponState)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file munentdec.h:

class DtMunitionEntityDecoder: public DtMilitaryEntityDecoder

class DtMunitionEntityDecoder: Instances of DtMunitionEntityDecoder are used to decode a munition entity state message into a state repository in an HLA (as opposed to DIS) environment

---

Inheritance:

DtMunitionEntityDecoder - DtMilitaryEntityDecoder - DtPhysicalEntityDecoder - DtBaseEntityDecoder - DtHlaStateDecoder

---

Public Methods

	DtMunitionEntityDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtMunitionEntityDecoder** () *destructor*
	DtMunitionEntityDecoder (const DtMunitionEntityDecoder& orig) *copy constructor*
DtMunitionEntityDecoder&	**operator=** (const DtMunitionEntityDecoder& orig) *assignment operator*
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtMunitionEntityDecoder*

Protected Methods

void	**addMyDecoders** () *Add decoding functions for the attributes*
	DtDECLARE_ENTITY_ATTR_DECODER (LauncherFlashPresent) *static decode<AttributeName> member functions*

Inherited from DtMilitaryEntityDecoder:

Inherited from DtPhysicalEntityDecoder:

Protected Methods

- static void **decodeArtParts**(int numParams, DtNetArtParamRecord* netParams, DtArtPartList* apList)
- static void **decodeAttParts**(int numParams, DtNetArtParamRecord* netParams, DtAttachedPartList* attList)

Inherited from DtBaseEntityDecoder:

Public Methods

- virtual DtHlaStateDecoder* **clone**() const

Protected Methods

- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(VelocityVector)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void(***DtAttributeDecoder**)(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)
- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)

- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

class DtMunitionEntityDecoder: Instances of DtMunitionEntityDecoder are used to decode a munition entity state message into a state repository in an HLA (as opposed to DIS) environment

- **`DtMunitionEntityDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtMunitionEntityDecoder()`**

  destructor

- **`DtMunitionEntityDecoder(const DtMunitionEntityDecoder& orig)`**

  copy constructor

- **`DtMunitionEntityDecoder& operator=(const DtMunitionEntityDecoder& orig)`**

assignment operator

- **`static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtMunitionEntityDecoder. User is responsible for deletion.

- **`void addMyDecoders()`**

  Add decoding functions for the attributes. Note: Non-virtual since meant to be called from constructors to add decoders.

- **`DtDECLARE_ENTITY_ATTR_DECODER(LauncherFlashPresent)`**

  static decode<AttributeName> member functions

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file milplatdec.h:

class DtMilitaryPlatformEntityDecoder: public DtMilitaryEntityDecoder

class DtMilitaryPlatformEntityDecoder: Instances of DtMilitaryPlatformEntityDecoder are used to decode a military platform entity state message into a state repository in an HLA (as opposed to DIS) environment

---

Inheritance:

DtMilitaryPlatformEntityDecoder - DtMilitaryEntityDecoder - DtPhysicalEntityDecoder - DtBaseEntityDecoder - DtHlaStateDecoder

---

Public Methods

	DtMilitaryPlatformEntityDecoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtMilitaryPlatformEntityDecoder** () *destructor*
	DtMilitaryPlatformEntityDecoder (const DtMilitaryPlatformEntityDecoder& orig) *copy constructor*
DtMilitaryPlatformEntityDecoder&	**operator=** (const DtMilitaryPlatformEntityDecoder& orig) *assignment operator*
static DtHlaStateDecoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtMilitaryPlatformEntityDecoder*

Protected Methods

void	**addMyDecoders** () *Add decoding functions for the attributes*
	DtDECLARE_ENTITY_ATTR_DECODER (AfterburnerOn) *static decode<AttributeName> member functions*
	DtDECLARE_ENTITY_ATTR_DECODER (HasAmmunitionSupplyCap)
	DtDECLARE_ENTITY_ATTR_DECODER (LauncherRaised)

Inherited from DtMilitaryEntityDecoder:

Inherited from DtPhysicalEntityDecoder:

Protected Methods

- static void **decodeArtParts**(int numParams, DtNetArtParamRecord* netParams, DtArtPartList* apList)
- static void **decodeAttParts**(int numParams, DtNetArtParamRecord* netParams, DtAttachedPartList* attList)

Inherited from DtBaseEntityDecoder:

Public Methods

- virtual DtHlaStateDecoder* **clone**() const

Protected Methods

- virtual void **decode**(const DtStateMsg& msg, DtStateRepository* stateRep) const
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(VelocityVector)

Inherited from DtHlaStateDecoder:

Public Methods

- typedef void(***DtAttributeDecoder**)(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& avlist, int pairSetIndex)

- DtAttributeDecoder **addDecoder**(RTI::AttributeHandle h, DtAttributeDecoder func)
- DtAttributeDecoder **removeDecoder**(RTI::AttributeHandle h)
- DtAttributeDecoder **addDecoder**(char* name, DtAttributeDecoder func, DtBoolean complain = DtTrue)
- DtAttributeDecoder **removeDecoder**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeDecoder **decoder**(int handle) const
- virtual RTI::ObjectClassHandle **objectClassHandle**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static char* **decodingBuffer**()
- inline char* **decodingBuffer**()
- inline RTI::ObjectClassHandle **objectClassHandle**() const
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::ObjectClassHandle **myHandle**
- DtAttributeDecoder* **myDecoderList**
- static const unsigned int **theMaxAttributesPerObject**
- static char **theDecBuffer**[2000]

Protected Methods

- virtual void **decodeAttribute**(DtStateRepository* stateRep, const RTI::AttributeHandleValuePairSet& attrs, int pairSetIndex) const

---

Documentation

class DtMilitaryPlatformEntityDecoder: Instances of DtMilitaryPlatformEntityDecoder are used to decode a military platform entity state message into a state repository in an HLA (as opposed to DIS) environment

- **`DtMilitaryPlatformEntityDecoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtMilitaryPlatformEntityDecoder()`**

  destructor

- **`DtMilitaryPlatformEntityDecoder( const DtMilitaryPlatformEntityDecoder& orig)`**

  copy constructor

- **`DtMilitaryPlatformEntityDecoder& operator=( const DtMilitaryPlatformEntityDecoder& orig)`**

  assignment operator

- **`static DtHlaStateDecoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtMilitaryPlatformEntityDecoder. User is responsible for deletion.

- **`void addMyDecoders()`**

  Add decoding functions for the attributes. Note: Non-virtual since meant to be called from constructors to add decoders.

- **`DtDECLARE_ENTITY_ATTR_DECODER(AfterburnerOn)`**

  static decode<AttributeName> member functions

- **`DtDECLARE_ENTITY_ATTR_DECODER(HasAmmunitionSupplyCap)`**

- **`DtDECLARE_ENTITY_ATTR_DECODER(LauncherRaised)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file soldierenc.h:

class DtSoldierEncoder: public DtMilitaryEntityEncoder

class DtSoldierEncoder: Instances of DtSoldierEncoder are used to encode a soldier entity's state repository into a state message in an HLA environment

---

Inheritance:

DtSoldierEncoder - DtMilitaryEntityEncoder - DtPhysicalEntityEncoder - DtBaseEntityEncoder - DtHlaStateEncoder

---

Public Methods

	DtSoldierEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtSoldierEncoder** () *destructor*
	DtSoldierEncoder (const DtSoldierEncoder& orig) *copy constructor*
DtSoldierEncoder&	**operator=** (const DtSoldierEncoder& orig) *assignment operator*
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtSoldierEncoder*

Protected Methods

- **DtDECLARE_ENTITY_ATTR_CHECKER** (PrimaryWeaponState)
  static need<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_CHECKER** (SecondaryWeaponState)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (PrimaryWeaponState)
  static decode<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_ENCODER** (SecondaryWeaponState)

Inherited from DtMilitaryEntityEncoder:

Inherited from DtPhysicalEntityEncoder:

Inherited from DtBaseEntityEncoder:

Public Methods

- static DtBoolean **tpsiChanged**(const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)

Protected Methods

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(VelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(VelocityVector)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void(***DtAttributeEncoder**)(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean(***DtAttributeChecker**)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

class DtSoldierEncoder: Instances of DtSoldierEncoder are used to encode a soldier entity's state repository into a state message in an HLA environment

- **`DtSoldierEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtSoldierEncoder()`**

  destructor

- **`DtSoldierEncoder(const DtSoldierEncoder& orig)`**

  copy constructor

- **`DtSoldierEncoder& operator=(const DtSoldierEncoder& orig)`**

  assignment operator

- **`static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtSoldierEncoder. Caller is responsible for deletion.

- **`DtDECLARE_ENTITY_ATTR_CHECKER(PrimaryWeaponState)`**

  static need<Attribute> member functions

- **`DtDECLARE_ENTITY_ATTR_CHECKER(SecondaryWeaponState)`**
- **`DtDECLARE_ENTITY_ATTR_ENCODER(PrimaryWeaponState)`**

  static decode<Attribute> member functions

- **`DtDECLARE_ENTITY_ATTR_ENCODER(SecondaryWeaponState)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file munentenc.h:

class DtMunitionEntityEncoder: public DtMilitaryEntityEncoder

class DtMunitionEntityEncoder: Instances of DtMunitionEntityEncoder are used to encode a munition entity's state repository into a state message in an HLA environment

---

Inheritance:

DtMunitionEntityEncoder - DtMilitaryEntityEncoder - DtPhysicalEntityEncoder - DtBaseEntityEncoder - DtHlaStateEncoder

---

Public Methods

	DtMunitionEntityEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtMunitionEntityEncoder** () *destructor*
	DtMunitionEntityEncoder (const DtMunitionEntityEncoder& orig) *copy constructor*
DtMunitionEntityEncoder&	**operator=** (const DtMunitionEntityEncoder& orig) *assignment operator*
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtMunitionEntityEncoder*

Protected Methods

- **DtDECLARE_ENTITY_ATTR_CHECKER** (LauncherFlashPresent)
  static need<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_ENCODER** (LauncherFlashPresent)
  static decode<Attribute> member functions

Inherited from DtMilitaryEntityEncoder:

Inherited from DtPhysicalEntityEncoder:

Inherited from DtBaseEntityEncoder:

Public Methods

- static DtBoolean **tpsiChanged**(const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)

Protected Methods

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(VelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(VelocityVector)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void(***DtAttributeEncoder**)(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean(***DtAttributeChecker**)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**

- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

class DtMunitionEntityEncoder: Instances of DtMunitionEntityEncoder are used to encode a munition entity's state repository into a state message in an HLA environment

- **`DtMunitionEntityEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtMunitionEntityEncoder()`**

  destructor

- **`DtMunitionEntityEncoder(const DtMunitionEntityEncoder& orig)`**

  copy constructor

- **`DtMunitionEntityEncoder& operator=(const DtMunitionEntityEncoder& orig)`**

  assignment operator

- **`static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtMunitionEntityEncoder. Caller is responsible for deletion.

- **`DtDECLARE_ENTITY_ATTR_CHECKER(LauncherFlashPresent)`**

  static need<Attribute> member functions

- **`DtDECLARE_ENTITY_ATTR_ENCODER(LauncherFlashPresent)`**

  static decode<Attribute> member functions

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file milplatenc.h:

class DtMilitaryPlatformEntityEncoder: public DtMilitaryEntityEncoder

class DtMilitaryPlatformEntityEncoder: Instances of DtMilitaryPlatformEntityEncoder are used to encode a military platform entity's state repository into a state message in an HLA environment

---

Inheritance:

DtMilitaryPlatformEntityEncoder - DtMilitaryEntityEncoder - DtPhysicalEntityEncoder - DtBaseEntityEncoder - DtHlaStateEncoder

---

Public Methods

	DtMilitaryPlatformEntityEncoder (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *constructor*
virtual	**~DtMilitaryPlatformEntityEncoder** () *destructor*
	DtMilitaryPlatformEntityEncoder (const DtMilitaryPlatformEntityEncoder& orig) *copy constructor*
DtMilitaryPlatformEntityEncoder&	**operator=** (const DtMilitaryPlatformEntityEncoder& orig) *assignment operator*
static DtHlaStateEncoder*	**create** (DtExerciseConn* exConn, DtObjClassDesc* classDesc) *Create a DtMilitaryPlatformEntityEncoder*

Protected Methods

- **DtDECLARE_ENTITY_ATTR_CHECKER** (AfterburnerOn)
  static need<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_CHECKER** (HasAmmunitionSupplyCap)
- **DtDECLARE_ENTITY_ATTR_CHECKER** (LauncherRaised)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (AfterburnerOn)
  static decode<Attribute> member functions
- **DtDECLARE_ENTITY_ATTR_ENCODER** (HasAmmunitionSupplyCap)
- **DtDECLARE_ENTITY_ATTR_ENCODER** (LauncherRaised)

Inherited from DtMilitaryEntityEncoder:

Inherited from DtPhysicalEntityEncoder:

Inherited from DtBaseEntityEncoder:

Public Methods

- static DtBoolean **tpsiChanged**(const DtBaseEntityStateRepository& stateRep, const DtBaseEntityStateRepository& asSeenByRemote)

Protected Methods

- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(VelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AccelerationVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(AngularVelocityVector)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(DRAlgorithm)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityType)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(EntityID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(FederateID)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(IsFrozen)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Orientation)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(Position)
- **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(VelocityVector)

Inherited from DtHlaStateEncoder:

Public Methods

- typedef void(***DtAttributeEncoder**)(const DtStateRepository& stateRep, RTI::AttributeHandleValuePairSet* avList, RTI::AttributeHandle handle)
- typedef DtBoolean(***DtAttributeChecker**)(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- DtAttributeEncoder **addEncoder**(RTI::AttributeHandle hand, DtAttributeEncoder func)
- DtAttributeEncoder **removeEncoder**(RTI::AttributeHandle hand)
- DtAttributeEncoder **addEncoder**(char* name, DtAttributeEncoder func, DtBoolean complain = DtTrue)
- DtAttributeEncoder **removeEncoder**(char* name, DtBoolean complain = DtTrue)
- DtAttributeChecker **addChecker**(RTI::AttributeHandle hand, DtAttributeChecker func)
- DtAttributeChecker **removeChecker**(RTI::AttributeHandle hand)
- DtAttributeChecker **addChecker**(char* name, DtAttributeChecker func, DtBoolean complain = DtTrue)
- DtAttributeChecker **removeChecker**(char* name, DtBoolean complain = DtTrue)
- virtual DtAttributeEncoder **encoder**(int handle) const
- virtual DtAttributeChecker **checker**(int handle) const
- virtual void **encode**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote, const RTI::AttributeHandleSet& neededAttrs, const RTI::AttributeHandleSet& includeThese, DtStateMsg* msg)
- virtual RTI::ObjectClassHandle **objectClassHandle**()
- virtual DtHlaStateEncoder* **clone**() const
- virtual void **setClassDesc**(DtObjClassDesc* desc)
- static DtBoolean **trueChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- static DtBoolean **falseChecker**(const DtStateRepository& stateRep, const DtStateRepository& asSeenByRemote)
- inline DtHlaStateEncoder::DtAttributeChecker **checker**(int hand) const
- inline DtHlaStateEncoder::DtAttributeEncoder **encoder**(int hand) const
- inline RTI::ObjectClassHandle **objectClassHandle**()
- inline void **setClassDesc**(DtObjClassDesc* desc)

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjClassDesc* **myClassDesc**
- DtVrlRtiAmbassador* **myRtiAmb**
- RTI::InteractionClassHandle **myHandle**
- DtAttributeEncoder* **myEncoderList**
- DtAttributeChecker* **myCheckerList**
- static const int **theMaxAttributesPerObject**

Documentation

class DtMilitaryPlatformEntityEncoder: Instances of DtMilitaryPlatformEntityEncoder are used to encode a military platform entity's state repository into a state message in an HLA environment

- **`DtMilitaryPlatformEntityEncoder( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  constructor

- **`virtual ~DtMilitaryPlatformEntityEncoder()`**

  destructor

- **`DtMilitaryPlatformEntityEncoder( const DtMilitaryPlatformEntityEncoder& orig)`**

  copy constructor

- **`DtMilitaryPlatformEntityEncoder& operator=( const DtMilitaryPlatformEntityEncoder& orig)`**

  assignment operator

- **`static DtHlaStateEncoder* create( DtExerciseConn* exConn, DtObjClassDesc* classDesc)`**

  Create a DtMilitaryPlatformEntityEncoder. Caller is responsible for deletion.

- **`DtDECLARE_ENTITY_ATTR_CHECKER(AfterburnerOn)`**

  static need<Attribute> member functions

- **`DtDECLARE_ENTITY_ATTR_CHECKER(HasAmmunitionSupplyCap)`**
- **`DtDECLARE_ENTITY_ATTR_CHECKER(LauncherRaised)`**
- **`DtDECLARE_ENTITY_ATTR_ENCODER(AfterburnerOn)`**

  static decode<Attribute> member functions

- **`DtDECLARE_ENTITY_ATTR_ENCODER(HasAmmunitionSupplyCap)`**
- **`DtDECLARE_ENTITY_ATTR_ENCODER(LauncherRaised)`**

This class has no child classes.

In file disenums.h:

enum DtEventType

- **DtEventOther**
- **DtEventOutOfFuel**
- **DtEventOutOfAmmo**
- **DtEventKilledInAction**
- **DtEventDamage**

Documentation

- `DtEventOther`
- `DtEventOutOfFuel`
- `DtEventOutOfAmmo`
- `DtEventKilledInAction`
- `DtEventDamage`

In file disenums.h:

enum DtRequestStatus

- **DtRequestStatusOther**
- **DtRequestStatusPending**
- **DtRequestStatusExecuting**
- **DtRequestStatusPartial**
- **DtRequestStatusComplete**
- **DtRequestStatusRejected**
- **DtRequestStatusRetransmitNow**
- **DtRequestStatusRetransmitLater**
- **DtRequestStatusInvalidTimeParameters**
- **DtRequestStatusSimulationTimeExceeded**
- **DtRequestStatusRequestDone**

Documentation

- `DtRequestStatusOther`
- `DtRequestStatusPending`
- `DtRequestStatusExecuting`
- `DtRequestStatusPartial`
- `DtRequestStatusComplete`
- `DtRequestStatusRejected`
- `DtRequestStatusRetransmitNow`
- `DtRequestStatusRetransmitLater`
- `DtRequestStatusInvalidTimeParameters`
- `DtRequestStatusSimulationTimeExceeded`
- `DtRequestStatusRequestDone`

In file vllist.h:

class DtListedObj

This is a base class from which elements of a DtIntrusiveList must be derived

---

Inheritance:

DtListedObj

Public Methods

	DtListedObj () *Constructor*
virtual	**~DtListedObj** () *Destructor*
void*	**next** () const *Return next (previous) elements on the list, or NULL if hit the end (beginning) of the list*
void*	**prev** () const
void*	**wrapNext** () const *Return next (previous) elements on the list, but wrap to the beginning (end) of the list when you hit the end (beginning)*
void*	**wrapPrev** () const
inline void*	**next** () const
inline void*	**prev** () const
inline void*	**wrapNext** () const
inline void*	**wrapPrev** () const

Protected Fields

DtList*	**myList**
DtListItem*	**myItem**

Documentation

This is a base class from which elements of a DtIntrusiveList must be derived. Previous and next pointers can be obtained directly from the DtListedObj, without going through the DtList to which it has been added.

- **`DtListedObj()`**

  Constructor

- **`virtual ~DtListedObj()`**

  Destructor

- **`void* next() const`**

  Return next (previous) elements on the list, or NULL if hit the end (beginning) of the list

- **`void* prev() const`**

- **`void* wrapNext() const`**

  Return next (previous) elements on the list, but wrap to the beginning (end) of the list when you hit the end (beginning). Only returns NULL when the list is totally empty.

- **`void* wrapPrev() const`**

- **`DtList* myList`**

- **`DtListItem* myItem`**

- **`inline void* next() const`**

- **`inline void* prev() const`**

- **`inline void* wrapNext() const`**

- **`inline void* wrapPrev() const`**

Direct child classes:

DtReflectedObjectCallbackInfo
DtReflectedObject
DtBaseReflectedEntity
DtBaseReflectedAggregate

Alphabetic index *Hierarchy of classes*

In file refobjcbinfo.h:

class DtReflectedObjectCallbackInfo: public DtListedObj

class DtReflectedObjectCallbackInfo: Instances of DtReflectedObjectCallbackInfo are used to contain the function and user data associated with a DtReflectedObject callback,

---

Inheritance:

DtReflectedObjectCallbackInfo - DtListedObj

Public Methods

	DtReflectedObjectCallbackInfo () *default constructor*
	DtReflectedObjectCallbackInfo (DtReflectedObjectCallbackFunction cb, void* userData) *constructor*
virtual	**~DtReflectedObjectCallbackInfo** () *destructor*
	DtReflectedObjectCallbackInfo (const DtReflectedObjectCallbackInfo& orig) *copy constructor*
DtReflectedObjectCallbackInfo&	**operator=** (const DtReflectedObjectCallbackInfo& orig) *assignment operator*
int	**operator==** (const DtReflectedObjectCallbackInfo& orig) const *equality operators*
int	**operator!=** (const DtReflectedObjectCallbackInfo& orig) const
virtual void	**setCallbackFunction** (DtReflectedObjectCallbackFunction funct) *set/get the callback function*
virtual DtReflectedObjectCallbackFunction	**callbackFunction** () const
virtual void	**setUserData** (void* data) *set/get the user data*
virtual void*	**userData** () const

Protected Fields

- DtReflectedObjectCallbackFunction **myCallbackFunction**
- void* **myUserData**

Inherited from DtListedObj:

Public Methods

- void* **next**() const
- void* **prev**() const
- void* **wrapNext**() const
- void* **wrapPrev**() const
- inline void* **next**() const
- inline void* **prev**() const
- inline void* **wrapNext**() const
- inline void* **wrapPrev**() const

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedObjectCallbackInfo: Instances of DtReflectedObjectCallbackInfo are used to contain the function and user data associated with a DtReflectedObject callback,

- **`DtReflectedObjectCallbackInfo()`**
  default constructor
- **`DtReflectedObjectCallbackInfo( DtReflectedObjectCallbackFunction cb, void* userData)`**
  constructor
- **`virtual ~DtReflectedObjectCallbackInfo()`**
  destructor
- **`DtReflectedObjectCallbackInfo(const DtReflectedObjectCallbackInfo& orig)`**
  copy constructor
- **`DtReflectedObjectCallbackInfo& operator=( const DtReflectedObjectCallbackInfo& orig)`**

assignment operator

- **int operator==(const DtReflectedObjectCallbackInfo& orig) const**

  equality operators

- **int operator!=(const DtReflectedObjectCallbackInfo& orig) const**

- **virtual void setCallbackFunction(DtReflectedObjectCallbackFunction funct)**

  set/get the callback function

- **virtual DtReflectedObjectCallbackFunction callbackFunction() const**

- **virtual void setUserData(void* data)**

  set/get the user data

- **virtual void* userData() const**

- **DtReflectedObjectCallbackFunction myCallbackFunction**

- **void* myUserData**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hreflectedobj.h:

class DtReflectedObject: public DtListedObj

class DtReflectedObject: Instances of DtReflectedObject are used to represent a simulation entity reflected across a network (i

---

Inheritance:

DtReflectedObject - DtListedObj

Public Methods

typedef DtStateRepository*	**(*DtBaseStateRepCreator)** () *Function prototype for functions that create DtStateRepositories*
	DtReflectedObject (DtExerciseConn* conn, DtStateRepository* stateRep) *Primary constructor*
	DtReflectedObject (DtHlaObject* obj, DtExerciseConn* conn, DtBaseStateRepCreator creator) *Older constructor*
	DtReflectedObject (const DtObjectId& id, const DtGlobalObjectDesignator& des, DtExerciseConn* conn, DtBaseStateRepCreator creator) *Older constructor*
virtual	**~DtReflectedObject** () *destructor*
virtual DtStateRepository*	**stateRep** () const *Returns a pointer to the StateRepository - through which you can inspect current state information*
virtual DtStateRepository*	**sr** () const *Synonym for stateRep()*
virtual DtHlaStateDecoder*	**decoder** () *Returns a pointer to the Decoder being used to decode state messages into our SR*
virtual DtHlaObject*	**hlaObject** () const *Get the entity's hlaObject*
virtual const DtObjectId&	**objectId** () const *Get the HLA object's id*
virtual const DtObjectId&	**id** () const *Get the HLA object's id (local ID)*
virtual const DtObjectId&	**localId** () const *Get the HLA object's id (local ID)*
virtual const DtString&	**name** () const *Get the HLA object name*

- virtual const DtGlobalObjectDesignator& **globalId** () const
  Get the HLA object's global object designator - the object name in RTI1
- virtual DtTime **lastSimTimeUpdated** () const
  Returns the value of simulation time at the time an update was last received for the entity
- virtual void **baseAddPostUpdateCallback** (DtReflectedObjectCallbackFunction cb, void* userData)
  Object state callback registration/deregistration
- virtual void **baseRemovePostUpdateCallback** (DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj** () const
  Get the next remote object
- virtual DtReflectedObject* **prevObj** () const
  Get the previous remote object
- virtual DtReflectedObject* **wrapNextObj** () const
  Get the next remote object
- virtual DtReflectedObject* **wrapPrevObj** () const
  Get the previous remote object
- virtual void **setHlaObject** (DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)
  Start reflecting this object, using this decoder
- virtual void **setNoHlaObject** (const DtObjectId& id, const DtGlobalObjectDesignator& des)
  Used to set up a proxy DtReflectedEntity - one that is not actually managing a real reflected HLA object
- virtual void **detachHlaObject** ()
  Ends managing of current HLA object
- inline DtHlaObject* **hlaObject** () const
- inline const DtObjectId& **localId** () const
- inline DtReflectedObject* **nextObj** () const
- inline DtReflectedObject* **prevObj** () const
- inline DtStateRepository* **sr** () const
- inline DtStateRepository* **stateRep** () const
- inline DtReflectedObject* **wrapNextObj** () const
- inline DtReflectedObject* **wrapPrevObj** () const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**

DtObjectId	**myId**
DtString	**myName**

Protected Methods

	DtReflectedObject (const DtReflectedObject& orig) *copy constructor*
DtReflectedObject&	**operator=** (const DtReflectedObject& orig) *assignment operator*
virtual void	**requestUpdate** () *Request that the owner of this entity send an attribute update*
virtual void	**processPostUpdateCallbacks** () *process the post update callbacks*
static void	**requestUpdatePostDrainCb** (void* usr) *calls requestUpdate*
static void	**processPostUpdateCallbacksCb** (const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr) *callback from HlaObject to process post update callbacks*

Inherited from DtListedObj:

Public Methods

- void* **next**() const
- void* **prev**() const
- void* **wrapNext**() const
- void* **wrapPrev**() const
- inline void* **next**() const
- inline void* **prev**() const
- inline void* **wrapNext**() const
- inline void* **wrapPrev**() const

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedObject:

Instances of DtReflectedObject are used to represent a simulation entity reflected across a network (i.e. this is not an full simulation running on this machine).

- **typedef DtStateRepository* (*DtBaseStateRepCreator)()**

  Function prototype for functions that create DtStateRepositories

- **DtReflectedObject(DtExerciseConn* conn, DtStateRepository* stateRep)**

  Primary constructor

- **DtReflectedObject(DtHlaObject* obj, DtExerciseConn* conn, DtBaseStateRepCreator creator)**

  Older constructor. Use the primary constructor followed by setHlaObject instead.

- **DtReflectedObject(const DtObjectId& id, const DtGlobalObjectDesignator& des, DtExerciseConn* conn, DtBaseStateRepCreator creator)**

  Older constructor. Use the primary constructor followed by setNoHlaObject instead.

- **virtual ~DtReflectedObject()**

  destructor

- **virtual DtStateRepository* stateRep() const**

  Returns a pointer to the StateRepository - through which you can inspect current state information

- **virtual DtStateRepository* sr() const**

  Synonym for stateRep()

- **virtual DtHlaStateDecoder* decoder()**

  Returns a pointer to the Decoder being used to decode state messages into our SR

- **virtual DtHlaObject* hlaObject() const**

  Get the entity's hlaObject

- **virtual const DtObjectId& objectId() const**

  Get the HLA object's id

- **virtual const DtObjectId& id() const**

  Get the HLA object's id (local ID)

- **virtual const DtObjectId& localId() const**

  Get the HLA object's id (local ID)

- **virtual const DtString& name() const**

  Get the HLA object name

- **virtual const DtGlobalObjectDesignator& globalId() const**

  Get the HLA object's global object designator - the object name in RTI1.3.3, but the objectId in RTI1.0.3.

- **virtual DtTime lastSimTimeUpdated() const**

Returns the value of simulation time at the time an update was last received for the entity

- **virtual void baseAddPostUpdateCallback(DtReflectedObjectCallbackFunction cb, void* userData)**

  Object state callback registration/deregistration

- **virtual void baseRemovePostUpdateCallback(DtReflectedObjectCallbackFunction cb, void* userData)**

- **virtual DtReflectedObject* nextObj() const**

  Get the next remote object

- **virtual DtReflectedObject* prevObj() const**

  Get the previous remote object

- **virtual DtReflectedObject* wrapNextObj() const**

  Get the next remote object. Will cycle back to the first remote object.

- **virtual DtReflectedObject* wrapPrevObj() const**

  Get the previous remote object. Will cycle back to the last remote object.

- **virtual void setHlaObject(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)**

  Start reflecting this object, using this decoder. If decoder is NULL, we get one from the FOM Mapper. The decoder is deleted by the DtReflectedObject destructor. Usually called by derived class constructor.

- **virtual void setNoHlaObject(const DtObjectId& id, const DtGlobalObjectDesignator& des)**

  Used to set up a proxy DtReflectedEntity - one that is not actually managing a real reflected HLA object

- **virtual void detachHlaObject()**

  Ends managing of current HLA object

- **DtReflectedObject(const DtReflectedObject& orig)**

  copy constructor

- **DtReflectedObject& operator=(const DtReflectedObject& orig)**

  assignment operator

- **virtual void requestUpdate()**

  Request that the owner of this entity send an attribute update

- **virtual void processPostUpdateCallbacks()**

  process the post update callbacks

- **static void requestUpdatePostDrainCb(void* usr)**

  calls requestUpdate

- **static void processPostUpdateCallbacksCb(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)**

  callback from HlaObject to process post update callbacks

- **DtExerciseConn* myConn**

- **DtHlaObjectWithStateRep* myHlaObj**

- **DtStateRepository* myStateRep**

- DtHlaStateDecoder* myDecoder
- DtBoolean myRequestCbRegistered
- DtIntrusiveList myStateUpdateCallbacks
- DtObjectId myId
- DtString myName
- inline DtHlaObject* hlaObject() const
- inline const DtObjectId& localId() const
- inline DtReflectedObject* nextObj() const
- inline DtReflectedObject* prevObj() const
- inline DtStateRepository* sr() const
- inline DtStateRepository* stateRep() const
- inline DtReflectedObject* wrapNextObj() const
- inline DtReflectedObject* wrapPrevObj() const

Direct child classes:

DtReflectedRadioTransmitter
DtReflectedRadioReceiver
DtReflectedGenericObject
DtReflectedEntity
DtReflectedEmitterSystem
DtReflectedEmitterBeam
DtReflectedDesignator
DtReflectedAggregate

Alphabetic index *Hierarchy of classes*

In file hreflradxmitr.h:

class DtReflectedRadioTransmitter: public DtReflectedObject

class DtReflectedRadioTransmitter: Instances of DtReflectedRadioTransmitter represent RadioTransmitter object simulated by remote applications

---

Inheritance:

DtReflectedRadioTransmitter - DtReflectedObject - DtListedObj

---

Public Methods

typedef DtRadioTransmitterRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtRadioTransmitterRepositories*
	DtReflectedRadioTransmitter (DtHlaObject* obj, DtExerciseConn* conn) *constructor*
virtual	**~DtReflectedRadioTransmitter** () *destructor*
virtual DtRadioTransmitterRepository*	**transmitterRep** () const *Get the transmitter's state repository*
virtual DtRadioTransmitterRepository*	**tsr** () const *Synonym for transmitterRep()*
virtual DtReflectedRadioTransmitter*	**next** () const *Get the next remote transmitter*
virtual DtReflectedRadioTransmitter*	**prev** () const *Get the previous remote transmitter*
virtual DtReflectedRadioTransmitter*	**wrapNext** () const *Get the next remote transmitter*
virtual DtReflectedRadioTransmitter*	**wrapPrev** () const *Get the previous remote transmitter*
virtual void	**addPostUpdateCallback** (DtRadioTransmitterCallbackFunction cb, void* userData) *transmitter state callback registration/deregistration*
virtual void	**removePostUpdateCallback** (DtRadioTransmitterCallbackFunction cb, void* userData)
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtRadioTransmitterRepository*	**transmitterRep** () const
inline DtRadioTransmitterRepository*	**tsr** () const

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtReflectedRadioTransmitter** (const DtReflectedRadioTransmitter& orig)
  copy constructor
- DtReflectedRadioTransmitter& **operator=** (const DtReflectedRadioTransmitter& orig)
  assignment operator

Inherited from DtReflectedObject:

Public Methods

- typedef DtStateRepository***(*DtBaseStateRepCreator)**()
- virtual DtStateRepository* **stateRep**() const
- virtual DtStateRepository* **sr**() const
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaObject* **hlaObject**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual DtTime **lastSimTimeUpdated**() const
- virtual void **baseAddPostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **baseRemovePostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj**() const
- virtual DtReflectedObject* **prevObj**() const
- virtual DtReflectedObject* **wrapNextObj**() const
- virtual DtReflectedObject* **wrapPrevObj**() const
- virtual void **setHlaObject**(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)

- virtual void **setNoHlaObject**(const DtObjectId& id, const DtGlobalObjectDesignator& des)
- virtual void **detachHlaObject**()
- inline DtHlaObject* **hlaObject**() const
- inline const DtObjectId& **localId**() const
- inline DtReflectedObject* **nextObj**() const
- inline DtReflectedObject* **prevObj**() const
- inline DtStateRepository* **sr**() const
- inline DtStateRepository* **stateRep**() const
- inline DtReflectedObject* **wrapNextObj**() const
- inline DtReflectedObject* **wrapPrevObj**() const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**
- DtObjectId **myId**
- DtString **myName**

Protected Methods

- virtual void **requestUpdate**()
- virtual void **processPostUpdateCallbacks**()
- static void **requestUpdatePostDrainCb**(void* usr)
- static void **processPostUpdateCallbacksCb**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)

Inherited from DtListedObj:

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedRadioTransmitter:

Instances of DtReflectedRadioTransmitter represent RadioTransmitter object simulated by remote applications. State data can be viewed through it's state repository.

- **typedef DtRadioTransmitterRepository* (*DtStateRepCreator)()**

  Function prototype for functions that create DtRadioTransmitterRepositories

- **DtReflectedRadioTransmitter(DtHlaObject* obj, DtExerciseConn* conn)**

  constructor

- **virtual ~DtReflectedRadioTransmitter()**

  destructor

- **virtual DtRadioTransmitterRepository* transmitterRep() const**

  Get the transmitter's state repository

- **virtual DtRadioTransmitterRepository* tsr() const**

  Synonym for transmitterRep()

- **virtual DtReflectedRadioTransmitter* next() const**

  Get the next remote transmitter

- **virtual DtReflectedRadioTransmitter* prev() const**

  Get the previous remote transmitter

- **virtual DtReflectedRadioTransmitter* wrapNext() const**

  Get the next remote transmitter. Will cycle back to the first remote transmitter.

- **virtual DtReflectedRadioTransmitter* wrapPrev() const**

  Get the previous remote transmitter. Will cycle back to the last remote transmitter.

- **virtual void addPostUpdateCallback(DtRadioTransmitterCallbackFunction cb, void* userData)**

  transmitter state callback registration/deregistration

- **virtual void removePostUpdateCallback(DtRadioTransmitterCallbackFunction cb, void* userData)**

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**

  Give us a function to be used for creating SRs

- **DtReflectedRadioTransmitter(const DtReflectedRadioTransmitter& orig)**

copy constructor

- **DtReflectedRadioTransmitter& operator=(const DtReflectedRadioTransmitter& orig)**

  assignment operator

- **static DtStateRepCreator theStateRepCreator**

- **inline DtRadioTransmitterRepository* transmitterRep() const**

- **inline DtRadioTransmitterRepository* tsr() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hreflradrecvr.h:

class DtReflectedRadioReceiver: public DtReflectedObject

class DtReflectedRadioReceiver: Instances of DtReflectedRadioReceiver represent RadioReceiver object simulated by remote applications

---

Inheritance:

DtReflectedRadioReceiver - DtReflectedObject - DtListedObj

Public Methods

typedef DtRadioReceiverRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtRadioReceiverRepositories*
	DtReflectedRadioReceiver (DtHlaObject* obj, DtExerciseConn* conn) *constructor*
virtual	**~DtReflectedRadioReceiver** () *destructor*
virtual DtRadioReceiverRepository*	**receiverRep** () const *Get the receiver's state repository*
virtual DtRadioReceiverRepository*	**rsr** () const *Synonym for receiverRep()*
virtual DtReflectedRadioReceiver*	**next** () const *Get the next remote receiver*
virtual DtReflectedRadioReceiver*	**prev** () const *Get the previous remote receiver*
virtual DtReflectedRadioReceiver*	**wrapNext** () const *Get the next remote receiver*
virtual DtReflectedRadioReceiver*	**wrapPrev** () const *Get the previous remote receiver*
virtual void	**addPostUpdateCallback** (DtRadioReceiverCallbackFunction cb, void* userData) *receiver state callback registration/deregistration*
virtual void	**removePostUpdateCallback** (DtRadioReceiverCallbackFunction cb, void* userData)
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtRadioReceiverRepository*	**receiverRep** () const
inline DtRadioReceiverRepository*	**rsr** () const

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtReflectedRadioReceiver** (const DtReflectedRadioReceiver& orig)
  copy constructor
- DtReflectedRadioReceiver& **operator=** (const DtReflectedRadioReceiver& orig)
  assignment operator

Inherited from DtReflectedObject:

Public Methods

- typedef DtStateRepository***(*DtBaseStateRepCreator)**()
- virtual DtStateRepository* **stateRep**() const
- virtual DtStateRepository* **sr**() const
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaObject* **hlaObject**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual DtTime **lastSimTimeUpdated**() const
- virtual void **baseAddPostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **baseRemovePostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj**() const
- virtual DtReflectedObject* **prevObj**() const
- virtual DtReflectedObject* **wrapNextObj**() const
- virtual DtReflectedObject* **wrapPrevObj**() const
- virtual void **setHlaObject**(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)

- virtual void **setNoHlaObject**(const DtObjectId& id, const DtGlobalObjectDesignator& des)
- virtual void **detachHlaObject**()
- inline DtHlaObject* **hlaObject**() const
- inline const DtObjectId& **localId**() const
- inline DtReflectedObject* **nextObj**() const
- inline DtReflectedObject* **prevObj**() const
- inline DtStateRepository* **sr**() const
- inline DtStateRepository* **stateRep**() const
- inline DtReflectedObject* **wrapNextObj**() const
- inline DtReflectedObject* **wrapPrevObj**() const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**
- DtObjectId **myId**
- DtString **myName**

Protected Methods

- virtual void **requestUpdate**()
- virtual void **processPostUpdateCallbacks**()
- static void **requestUpdatePostDrainCb**(void* usr)
- static void **processPostUpdateCallbacksCb**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)

Inherited from DtListedObj:

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedRadioReceiver:

Instances of DtReflectedRadioReceiver represent RadioReceiver object simulated by remote applications. State data can be viewed through it's state repository.

- **`typedef DtRadioReceiverRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtRadioReceiverRepositories

- **`DtReflectedRadioReceiver(DtHlaObject* obj, DtExerciseConn* conn)`**

  constructor

- **`virtual ~DtReflectedRadioReceiver()`**

  destructor

- **`virtual DtRadioReceiverRepository* receiverRep() const`**

  Get the receiver's state repository

- **`virtual DtRadioReceiverRepository* rsr() const`**

  Synonym for receiverRep()

- **`virtual DtReflectedRadioReceiver* next() const`**

  Get the next remote receiver

- **`virtual DtReflectedRadioReceiver* prev() const`**

  Get the previous remote receiver

- **`virtual DtReflectedRadioReceiver* wrapNext() const`**

  Get the next remote receiver. Will cycle back to the first remote receiver.

- **`virtual DtReflectedRadioReceiver* wrapPrev() const`**

  Get the previous remote receiver. Will cycle back to the last remote receiver.

- **`virtual void addPostUpdateCallback( DtRadioReceiverCallbackFunction cb, void* userData)`**

  receiver state callback registration/deregistration

- **`virtual void removePostUpdateCallback( DtRadioReceiverCallbackFunction cb, void* userData)`**

- **`static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)`**

  Give us a function to be used for creating SRs

- **`DtReflectedRadioReceiver(const DtReflectedRadioReceiver& orig)`**

  copy constructor

- **DtReflectedRadioReceiver& operator=(const DtReflectedRadioReceiver& orig)**

  assignment operator

- **static DtStateRepCreator theStateRepCreator**
- **inline DtRadioReceiverRepository* receiverRep() const**
- **inline DtRadioReceiverRepository* rsr() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file refgenericobj.h:

class DtReflectedGenericObject: public DtReflectedObject

class DtReflectedGenericObject: Instances of DtReflectedGenericObject are used to represent "GenericObject" objects simulated remotely, that we hear about through the RTI

Inheritance:

DtReflectedGenericObject - DtReflectedObject - DtListedObj

Public Methods

	DtReflectedGenericObject (DtHlaObject* obj, DtExerciseConn* conn) *constructor*
virtual	**~DtReflectedGenericObject** () *destructor*
virtual DtGenericStateRepository*	**genericStateRep** () const *Returns a pointer to the DtGenericStateRepository, through which you can inspect current state information*
virtual DtGenericStateRepository*	**gsr** () const *Synonym for genericStateRep()*
virtual DtReflectedGenericObject*	**next** () const *Functions to get the next and previous objects in the list*
virtual DtReflectedGenericObject*	**prev** () const
virtual DtReflectedGenericObject*	**wrapNext** () const
virtual DtReflectedGenericObject*	**wrapPrev** () const
virtual void	**addPostUpdateCallback** (DtGenericObjectCallbackFunction cb, void* userData) *Registration/deregistration of postUpdate callbacks*
virtual void	**removePostUpdateCallback** (DtGenericObjectCallbackFunction cb, void* userData)

Inherited from DtReflectedObject:

Public Methods

- typedef DtStateRepository*(***DtBaseStateRepCreator**)()
- virtual DtStateRepository* **stateRep**() const
- virtual DtStateRepository* **sr**() const
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaObject* **hlaObject**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual DtTime **lastSimTimeUpdated**() const
- virtual void **baseAddPostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **baseRemovePostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj**() const
- virtual DtReflectedObject* **prevObj**() const
- virtual DtReflectedObject* **wrapNextObj**() const
- virtual DtReflectedObject* **wrapPrevObj**() const
- virtual void **setHlaObject**(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)
- virtual void **setNoHlaObject**(const DtObjectId& id, const DtGlobalObjectDesignator& des)
- virtual void **detachHlaObject**()
- inline DtHlaObject* **hlaObject**() const
- inline const DtObjectId& **localId**() const
- inline DtReflectedObject* **nextObj**() const
- inline DtReflectedObject* **prevObj**() const
- inline DtStateRepository* **sr**() const
- inline DtStateRepository* **stateRep**() const
- inline DtReflectedObject* **wrapNextObj**() const
- inline DtReflectedObject* **wrapPrevObj**() const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**
- DtObjectId **myId**
- DtString **myName**

Protected Methods

- DtReflectedObject& **operator=**(const DtReflectedObject& orig)
- virtual void **requestUpdate**()
- virtual void **processPostUpdateCallbacks**()
- static void **requestUpdatePostDrainCb**(void* usr)
- static void **processPostUpdateCallbacksCb**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)

Inherited from DtListedObj:

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedGenericObject:

Instances of DtReflectedGenericObject are used to represent "GenericObject" objects simulated remotely, that we hear about through the RTI. Instances are created and deleted automatically by DtReflectedGenericObjectList in response to discover and remove RTI service invocations. This class is derived from DtDtReflectedObject, which does most of the work. The derived class mostly exists to provide typesafety, and to configure the base class object.

- **`DtReflectedGenericObject(DtHlaObject* obj, DtExerciseConn* conn)`**

  constructor

- **`virtual ~DtReflectedGenericObject()`**

  destructor

- **`virtual DtGenericStateRepository* genericStateRep() const`**

  Returns a pointer to the DtGenericStateRepository, through which you can inspect current state information

- **`virtual DtGenericStateRepository* gsr() const`**

  Synonym for genericStateRep()

- **`virtual DtReflectedGenericObject* next() const`**

  Functions to get the next and previous objects in the list. The "wrap" functions cycle to the beginning/end of the list.

- **`virtual DtReflectedGenericObject* prev() const`**

- **`virtual DtReflectedGenericObject* wrapNext() const`**

- **`virtual DtReflectedGenericObject* wrapPrev() const`**

- **`virtual void addPostUpdateCallback( DtGenericObjectCallbackFunction cb, void* userData)`**

  Registration/deregistration of postUpdate callbacks

- **`virtual void removePostUpdateCallback( DtGenericObjectCallbackFunction cb, void* userData)`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hreflectedent.h:

class DtReflectedEntity: public DtReflectedObject

class DtReflectedEntity: Instances of DtReflectedEntity are used to represent a simulation entity reflected across a network (i

---

Inheritance:

DtReflectedEntity - DtReflectedObject - DtListedObj

Public Fields

- typedef DtStateRepCreator **DtEsrCreator**
  For backwards compatibility

Public Methods

- typedef DtEntityStateRepository* **(*DtStateRepCreator)** ()
  Function prototype for functions that create DtEntityStateRepositories
- **DtReflectedEntity** (DtHlaObject* obj, DtExerciseConn* conn)
  constructor
- virtual **~DtReflectedEntity** ()
  destructor
- virtual DtEntityStateRepository* **entityStateRep** () const
  Returns a pointer to the EntityStateRepository - through which you can inspect current state information
- virtual DtEntityStateRepository* **esr** () const
  Synonym for entityStateRep()
- virtual DtReflectedEntity* **next** () const
  Get the next remote entity
- virtual DtReflectedEntity* **prev** () const
  Get the previous remote entity
- virtual DtReflectedEntity* **wrapNext** () const
  Get the next remote entity
- virtual DtReflectedEntity* **wrapPrev** () const
  Get the previous remote entity
- virtual void **addPostUpdateCallback** (DtEntityCallbackFunction cb, void* userData)
  Entity state callback registration/deregistration
- virtual void **removePostUpdateCallback** (DtEntityCallbackFunction cb, void* userData)
- static DtStateRepCreator **setStateRepCreator** (DtStateRepCreator creator)
  Give us a function to be used for creating SRs

- virtual RTI::ObjectHandle **entityObjectId** () const
  For backwards compatibility
- static DtEsrCreator **setEsrCreator** (DtEsrCreator creator)
  For backwards compatibility
- inline DtEntityStateRepository* **entityStateRep** () const
- inline DtEntityStateRepository* **esr** () const

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtReflectedEntity** (const DtReflectedEntity& orig)
  copy constructor
- DtReflectedEntity& **operator=** (const DtReflectedEntity& orig)
  assignment operator

Inherited from DtReflectedObject:

Public Methods

- typedef DtStateRepository***(*DtBaseStateRepCreator)**()
- virtual DtStateRepository* **stateRep**() const
- virtual DtStateRepository* **sr**() const
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaObject* **hlaObject**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual DtTime **lastSimTimeUpdated**() const
- virtual void **baseAddPostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **baseRemovePostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj**() const

- virtual DtReflectedObject* **prevObj**() const
- virtual DtReflectedObject* **wrapNextObj**() const
- virtual DtReflectedObject* **wrapPrevObj**() const
- virtual void **setHlaObject**(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)
- virtual void **setNoHlaObject**(const DtObjectId& id, const DtGlobalObjectDesignator& des)
- virtual void **detachHlaObject**()
- inline DtHlaObject* **hlaObject**() const
- inline const DtObjectId& **localId**() const
- inline DtReflectedObject* **nextObj**() const
- inline DtReflectedObject* **prevObj**() const
- inline DtStateRepository* **sr**() const
- inline DtStateRepository* **stateRep**() const
- inline DtReflectedObject* **wrapNextObj**() const
- inline DtReflectedObject* **wrapPrevObj**() const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**
- DtObjectId **myId**
- DtString **myName**

Protected Methods

- virtual void **requestUpdate**()
- virtual void **processPostUpdateCallbacks**()
- static void **requestUpdatePostDrainCb**(void* usr)
- static void **processPostUpdateCallbacksCb**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)

Inherited from DtListedObj:

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedEntity:

Instances of DtReflectedEntity are used to represent a simulation entity reflected across a network (i.e. this is not an full simulation running on this machine).

- **`typedef DtEntityStateRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtEntityStateRepositories

- **`DtReflectedEntity(DtHlaObject* obj, DtExerciseConn* conn)`**

  constructor

- **`virtual ~DtReflectedEntity()`**

  destructor

- **`virtual DtEntityStateRepository* entityStateRep() const`**

  Returns a pointer to the EntityStateRepository - through which you can inspect current state information

- **`virtual DtEntityStateRepository* esr() const`**

  Synonym for entityStateRep()

- **`virtual DtReflectedEntity* next() const`**

  Get the next remote entity

- **`virtual DtReflectedEntity* prev() const`**

  Get the previous remote entity

- **`virtual DtReflectedEntity* wrapNext() const`**

  Get the next remote entity. Will cycle back to the first remote entity.

- **`virtual DtReflectedEntity* wrapPrev() const`**

  Get the previous remote entity. Will cycle back to the last remote entity.

- **`virtual void addPostUpdateCallback( DtEntityCallbackFunction cb, void* userData)`**

  Entity state callback registration/deregistration

- **`virtual void removePostUpdateCallback( DtEntityCallbackFunction cb, void* userData)`**

- **`static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)`**

  Give us a function to be used for creating SRs

- **`typedef DtStateRepCreator DtEsrCreator`**
  For backwards compatibility. Use DtStateRepCreator instead.
- **`virtual RTI::ObjectHandle entityObjectId() const`**
  For backwards compatibility. Use id() instead.
- **`static DtEsrCreator setEsrCreator(DtEsrCreator creator)`**
  For backwards compatibility. Use setStateRepCreator instead.
- **`DtReflectedEntity(const DtReflectedEntity& orig)`**
  copy constructor
- **`DtReflectedEntity& operator=(const DtReflectedEntity& orig)`**
  assignment operator
- **`static DtStateRepCreator theStateRepCreator`**
- **`inline DtEntityStateRepository* entityStateRep() const`**
- **`inline DtEntityStateRepository* esr() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hreflemitrsys.h:

class DtReflectedEmitterSystem: public DtReflectedObject

class DtReflectedEmitterSystem: Instances of DtReflectedEmitterSystem represent EmitterSystem object simulated by remote applications

---

Inheritance:

DtReflectedEmitterSystem - DtReflectedObject - DtListedObj

Public Methods

typedef DtEmitterSystemRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtEmitterSystemRepositories*
	DtReflectedEmitterSystem (DtHlaObject* obj, DtExerciseConn* conn, DtBoolean beamsUpdateSystem) *constructor If beamsUpdateSystem == DtTrue, then a system's post-update callbacks are called whenever one of the beams in the system is updated*
	DtReflectedEmitterSystem (const DtObjectId& id, const DtGlobalObjectDesignator& des, DtExerciseConn* conn, DtBoolean beamsUpdateSystem) *Constructor used to construct a placeholder system when we discover a beam whose system hasn't been discovered yet*
virtual	**~DtReflectedEmitterSystem** () *destructor*
virtual DtEmitterSystemRepository*	**emitterSysRep** () const *Get the emitter system's state repository*
virtual DtEmitterSystemRepository*	**esr** () const *Synonym for emitterSysRep()*
virtual DtReflectedEmitterSystem*	**next** () const *Get the next remote emitterSystem*
virtual DtReflectedEmitterSystem*	**prev** () const *Get the previous remote emitterSystem*
virtual DtReflectedEmitterSystem*	**wrapNext** () const *Get the next remote emitterSystem*
virtual DtReflectedEmitterSystem*	**wrapPrev** () const *Get the previous remote emitterSystem*
virtual void	**addPostUpdateCallback** (DtEmitterSystemCallbackFunction cb, void* userData) *emitterSystem state callback registration/deregistration*

- virtual void **removePostUpdateCallback** (DtEmitterSystemCallbackFunction cb, void* userData)
- virtual void **beamUpdated** ()
  Tell the emitter system that one of its beams has been updated
- static DtStateRepCreator **setStateRepCreator** (DtStateRepCreator creator)
  Give us a function to be used for creating SRs
- inline DtEmitterSystemRepository* **emitterSysRep** () const
- inline DtEmitterSystemRepository* **esr** () const

Protected Fields

- static DtStateRepCreator **theStateRepCreator**
- DtBoolean **myBeamsUpdateSystem**

Protected Methods

- **DtReflectedEmitterSystem** (const DtReflectedEmitterSystem& orig)
  copy constructor
- DtReflectedEmitterSystem& **operator=** (const DtReflectedEmitterSystem& orig)
  assignment operator

Inherited from DtReflectedObject:

Public Methods

- typedef DtStateRepository***(*DtBaseStateRepCreator)**()
- virtual DtStateRepository* **stateRep**() const
- virtual DtStateRepository* **sr**() const
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaObject* **hlaObject**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual DtTime **lastSimTimeUpdated**() const
- virtual void **baseAddPostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)

- virtual void **baseRemovePostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj**() const
- virtual DtReflectedObject* **prevObj**() const
- virtual DtReflectedObject* **wrapNextObj**() const
- virtual DtReflectedObject* **wrapPrevObj**() const
- virtual void **setHlaObject**(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)
- virtual void **setNoHlaObject**(const DtObjectId& id, const DtGlobalObjectDesignator& des)
- virtual void **detachHlaObject**()
- inline DtHlaObject* **hlaObject**() const
- inline const DtObjectId& **localId**() const
- inline DtReflectedObject* **nextObj**() const
- inline DtReflectedObject* **prevObj**() const
- inline DtStateRepository* **sr**() const
- inline DtStateRepository* **stateRep**() const
- inline DtReflectedObject* **wrapNextObj**() const
- inline DtReflectedObject* **wrapPrevObj**() const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**
- DtObjectId **myId**
- DtString **myName**

Protected Methods

- virtual void **requestUpdate**()
- virtual void **processPostUpdateCallbacks**()

- static void **requestUpdatePostDrainCb**(void* usr)
- static void **processPostUpdateCallbacksCb**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)

Inherited from DtListedObj:

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedEmitterSystem:

Instances of DtReflectedEmitterSystem represent EmitterSystem object simulated by remote applications. State data can be viewed through it's state repository.

- **`typedef DtEmitterSystemRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtEmitterSystemRepositories

- **`DtReflectedEmitterSystem(DtHlaObject* obj, DtExerciseConn* conn, DtBoolean beamsUpdateSystem)`**

  constructor If beamsUpdateSystem == DtTrue, then a system's post-update callbacks are called whenever one of the beams in the system is updated

- **`DtReflectedEmitterSystem(const DtObjectId& id, const DtGlobalObjectDesignator& des, DtExerciseConn* conn, DtBoolean beamsUpdateSystem)`**

  Constructor used to construct a placeholder system when we discover a beam whose system hasn't been discovered yet

- **`virtual ~DtReflectedEmitterSystem()`**

  destructor

- **`virtual DtEmitterSystemRepository* emitterSysRep() const`**

  Get the emitter system's state repository

- **`virtual DtEmitterSystemRepository* esr() const`**

  Synonym for emitterSysRep()

- **`virtual DtReflectedEmitterSystem* next() const`**

  Get the next remote emitterSystem

- **`virtual DtReflectedEmitterSystem* prev() const`**

  Get the previous remote emitterSystem

- **`virtual DtReflectedEmitterSystem* wrapNext() const`**

  Get the next remote emitterSystem. Will cycle back to the first remote emitterSystem.

- **virtual DtReflectedEmitterSystem* wrapPrev() const**

  Get the previous remote emitterSystem. Will cycle back to the last remote emitterSystem.

- **virtual void addPostUpdateCallback(DtEmitterSystemCallbackFunction cb, void* userData)**

  emitterSystem state callback registration/deregistration

- **virtual void removePostUpdateCallback(DtEmitterSystemCallbackFunction cb, void* userData)**

- **virtual void beamUpdated()**

  Tell the emitter system that one of its beams has been updated

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**

  Give us a function to be used for creating SRs

- **DtReflectedEmitterSystem(const DtReflectedEmitterSystem& orig)**

  copy constructor

- **DtReflectedEmitterSystem& operator=(const DtReflectedEmitterSystem& orig)**

  assignment operator

- **static DtStateRepCreator theStateRepCreator**

- **DtBoolean myBeamsUpdateSystem**

- **inline DtEmitterSystemRepository* emitterSysRep() const**

- **inline DtEmitterSystemRepository* esr() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hreflbeam.h:

class DtReflectedEmitterBeam: public DtReflectedObject

class DtReflectedEmitterBeam: Instances of DtReflectedEmitterBeam represent EmitterBeam object simulated by remote applications

---

Inheritance:

DtReflectedEmitterBeam - DtReflectedObject - DtListedObj

Public Methods

typedef DtEmitterBeamRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtEmitterBeamRepositories*
	DtReflectedEmitterBeam (DtHlaObject* obj, DtExerciseConn* conn) *constructor*
virtual	**~DtReflectedEmitterBeam** () *destructor*
virtual DtEmitterBeamRepository*	**beamRep** () const *Get the beam's state repository*
virtual DtEmitterBeamRepository*	**bsr** () const *Synonym for beamRep()*
virtual void	**setBsr** (DtEmitterBeamRepository* bsr) *Set beam state repository*
virtual DtReflectedEmitterBeam*	**next** () const *Get the next remote beam*
virtual DtReflectedEmitterBeam*	**prev** () const *Get the previous remote beam*
virtual DtReflectedEmitterBeam*	**wrapNext** () const *Get the next remote beam*
virtual DtReflectedEmitterBeam*	**wrapPrev** () const *Get the previous remote beam*
virtual void	**addPostUpdateCallback** (DtEmitterBeamCallbackFunction cb, void* userData) *beam state callback registration/deregistration*
virtual void	**removePostUpdateCallback** (DtEmitterBeamCallbackFunction cb, void* userData)
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtEmitterBeamRepository*	**beamRep** () const

- inline DtEmitterBeamRepository* **bsr** () const

Protected Fields

- DtEmitterBeamRepository* **myConstructedBsr**
- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtReflectedEmitterBeam** (const DtReflectedEmitterBeam& orig)
  copy constructor
- DtReflectedEmitterBeam& **operator=** (const DtReflectedEmitterBeam& orig)
  assignment operator

Inherited from DtReflectedObject:

Public Methods

- typedef DtStateRepository***(*DtBaseStateRepCreator)**()
- virtual DtStateRepository* **stateRep**() const
- virtual DtStateRepository* **sr**() const
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaObject* **hlaObject**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual DtTime **lastSimTimeUpdated**() const
- virtual void **baseAddPostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **baseRemovePostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj**() const
- virtual DtReflectedObject* **prevObj**() const
- virtual DtReflectedObject* **wrapNextObj**() const
- virtual DtReflectedObject* **wrapPrevObj**() const

- virtual void **setHlaObject**(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)
- virtual void **setNoHlaObject**(const DtObjectId& id, const DtGlobalObjectDesignator& des)
- virtual void **detachHlaObject**()
- inline DtHlaObject* **hlaObject**() const
- inline const DtObjectId& **localId**() const
- inline DtReflectedObject* **nextObj**() const
- inline DtReflectedObject* **prevObj**() const
- inline DtStateRepository* **sr**() const
- inline DtStateRepository* **stateRep**() const
- inline DtReflectedObject* **wrapNextObj**() const
- inline DtReflectedObject* **wrapPrevObj**() const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**
- DtObjectId **myId**
- DtString **myName**

Protected Methods

- virtual void **requestUpdate**()
- virtual void **processPostUpdateCallbacks**()
- static void **requestUpdatePostDrainCb**(void* usr)
- static void **processPostUpdateCallbacksCb**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)

Inherited from DtListedObj:

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedEmitterBeam:

Instances of DtReflectedEmitterBeam represent EmitterBeam object simulated by remote applications. State data can be viewed through it's state repository.

- **typedef DtEmitterBeamRepository* (*DtStateRepCreator)()**

  Function prototype for functions that create DtEmitterBeamRepositories

- **DtReflectedEmitterBeam(DtHlaObject* obj, DtExerciseConn* conn)**

  constructor

- **virtual ~DtReflectedEmitterBeam()**

  destructor

- **virtual DtEmitterBeamRepository* beamRep() const**

  Get the beam's state repository

- **virtual DtEmitterBeamRepository* bsr() const**

  Synonym for beamRep()

- **virtual void setBsr(DtEmitterBeamRepository* bsr)**

  Set beam state repository. Only necessary when you want to use something other than the default, which is created internally.

- **virtual DtReflectedEmitterBeam* next() const**

  Get the next remote beam

- **virtual DtReflectedEmitterBeam* prev() const**

  Get the previous remote beam

- **virtual DtReflectedEmitterBeam* wrapNext() const**

  Get the next remote beam. Will cycle back to the first remote beam.

- **virtual DtReflectedEmitterBeam* wrapPrev() const**

  Get the previous remote beam. Will cycle back to the last remote beam.

- **virtual void addPostUpdateCallback(DtEmitterBeamCallbackFunction cb, void* userData)**

  beam state callback registration/deregistration

- **virtual void removePostUpdateCallback(DtEmitterBeamCallbackFunction cb, void***

userData)

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**
  Give us a function to be used for creating SRs
- **DtReflectedEmitterBeam(const DtReflectedEmitterBeam& orig)**
  copy constructor
- **DtReflectedEmitterBeam& operator=(const DtReflectedEmitterBeam& orig)**
  assignment operator
- **DtEmitterBeamRepository* myConstructedBsr**
- **static DtStateRepCreator theStateRepCreator**
- **inline DtEmitterBeamRepository* beamRep() const**
- **inline DtEmitterBeamRepository* bsr() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrefldesigntr.h:

class DtReflectedDesignator: public DtReflectedObject

class DtReflectedDesignator: Instances of DtReflectedDesignator are

---

Inheritance:

DtReflectedDesignator - DtReflectedObject - DtListedObj

Public Methods

typedef DtDesignatorRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtDesignatorRepositories*
	DtReflectedDesignator (DtHlaObject* obj, DtExerciseConn* conn) *constructor*
virtual	**~DtReflectedDesignator** () *destructor*
virtual DtDesignatorRepository*	**designatorRep** () const *Get the designator's state repository*
virtual DtDesignatorRepository*	**dsr** () const *Synonym for designatorRep()*
virtual DtReflectedDesignator*	**next** () const *Get the next remote designator*
virtual DtReflectedDesignator*	**prev** () const *Get the previous remote designator*
virtual DtReflectedDesignator*	**wrapNext** () const *Get the next remote designator*
virtual DtReflectedDesignator*	**wrapPrev** () const *Get the previous remote designator*
virtual void	**addPostUpdateCallback** (DtDesignatorCallbackFunction cb, void* userData) *designator state callback registration/deregistration*
virtual void	**removePostUpdateCallback** (DtDesignatorCallbackFunction cb, void* userData)
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtDesignatorRepository*	**designatorRep** () const
inline DtDesignatorRepository*	**dsr** () const

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtReflectedDesignator** (const DtReflectedDesignator& orig)
  copy constructor
- DtReflectedDesignator& **operator=** (const DtReflectedDesignator& orig)
  assignment operator

Inherited from DtReflectedObject:

Public Methods

- typedef DtStateRepository***(*DtBaseStateRepCreator)**()
- virtual DtStateRepository* **stateRep**() const
- virtual DtStateRepository* **sr**() const
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaObject* **hlaObject**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual DtTime **lastSimTimeUpdated**() const
- virtual void **baseAddPostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **baseRemovePostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj**() const
- virtual DtReflectedObject* **prevObj**() const
- virtual DtReflectedObject* **wrapNextObj**() const
- virtual DtReflectedObject* **wrapPrevObj**() const
- virtual void **setHlaObject**(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)
- virtual void **setNoHlaObject**(const DtObjectId& id, const DtGlobalObjectDesignator& des)

- virtual void **detachHlaObject**()
- inline DtHlaObject* **hlaObject**() const
- inline const DtObjectId& **localId**() const
- inline DtReflectedObject* **nextObj**() const
- inline DtReflectedObject* **prevObj**() const
- inline DtStateRepository* **sr**() const
- inline DtStateRepository* **stateRep**() const
- inline DtReflectedObject* **wrapNextObj**() const
- inline DtReflectedObject* **wrapPrevObj**() const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**
- DtObjectId **myId**
- DtString **myName**

Protected Methods

- virtual void **requestUpdate**()
- virtual void **processPostUpdateCallbacks**()
- static void **requestUpdatePostDrainCb**(void* usr)
- static void **processPostUpdateCallbacksCb**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)

Inherited from DtListedObj:

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedDesignator: Instances of DtReflectedDesignator are

- **typedef DtDesignatorRepository* (*DtStateRepCreator)()**
  Function prototype for functions that create DtDesignatorRepositories
- **DtReflectedDesignator(DtHlaObject* obj, DtExerciseConn* conn)**
  constructor
- **virtual ~DtReflectedDesignator()**
  destructor
- **virtual DtDesignatorRepository* designatorRep() const**
  Get the designator's state repository
- **virtual DtDesignatorRepository* dsr() const**
  Synonym for designatorRep()
- **virtual DtReflectedDesignator* next() const**
  Get the next remote designator
- **virtual DtReflectedDesignator* prev() const**
  Get the previous remote designator
- **virtual DtReflectedDesignator* wrapNext() const**
  Get the next remote designator. Will cycle back to the first remote designator.
- **virtual DtReflectedDesignator* wrapPrev() const**
  Get the previous remote designator. Will cycle back to the last remote designator.
- **virtual void addPostUpdateCallback(DtDesignatorCallbackFunction cb, void* userData)**
  designator state callback registration/deregistration
- **virtual void removePostUpdateCallback(DtDesignatorCallbackFunction cb, void* userData)**
- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**
  Give us a function to be used for creating SRs
- **DtReflectedDesignator(const DtReflectedDesignator& orig)**
  copy constructor
- **DtReflectedDesignator& operator=(const DtReflectedDesignator& orig)**
  assignment operator

- `static DtStateRepCreator theStateRepCreator`
- `inline DtDesignatorRepository* designatorRep() const`
- `inline DtDesignatorRepository* dsr() const`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hreflectedagg.h:

class DtReflectedAggregate: public DtReflectedObject

class DtReflectedAggregate: Instances of DtReflectedAggregate are used to represent a simulation aggregate reflected across a network (i

---

Inheritance:

DtReflectedAggregate - DtReflectedObject - DtListedObj

Public Methods

typedef DtAggregateStateRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtAggregateStateRepositories*
	DtReflectedAggregate (DtHlaObject* obj, DtExerciseConn* conn) *constructor*
virtual	**~DtReflectedAggregate** () *destructor*
virtual DtAggregateStateRepository*	**aggregateStateRep** () const *Returns a pointer to the AggregateStateRepository - through which you can inspect current state information*
virtual DtAggregateStateRepository*	**asr** () const *Synonym for aggregateStateRep()*
virtual DtReflectedAggregate*	**next** () const *Get the next remote aggregate*
virtual DtReflectedAggregate*	**prev** () const *Get the previous remote aggregate*
virtual DtReflectedAggregate*	**wrapNext** () const *Get the next remote aggregate*
virtual DtReflectedAggregate*	**wrapPrev** () const *Get the previous remote aggregate*
virtual void	**addPostUpdateCallback** (DtAggregateCallbackFunction cb, void* userData) *Aggregate state callback registration/deregistration*
virtual void	**removePostUpdateCallback** (DtAggregateCallbackFunction cb, void* userData)
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
virtual RTI::ObjectHandle	**aggregateObjectId** () const *Get the aggregate HLA object's id For backwards compatibility*
inline DtAggregateStateRepository*	**aggregateStateRep** () const

- inline DtAggregateStateRepository* **asr** () const

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtReflectedAggregate** (const DtReflectedAggregate& orig)
  copy constructor
- DtReflectedAggregate& **operator=** (const DtReflectedAggregate& orig)
  assignment operator

Inherited from DtReflectedObject:

Public Methods

- typedef DtStateRepository***(*DtBaseStateRepCreator)**()
- virtual DtStateRepository* **stateRep**() const
- virtual DtStateRepository* **sr**() const
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaObject* **hlaObject**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual DtTime **lastSimTimeUpdated**() const
- virtual void **baseAddPostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **baseRemovePostUpdateCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual DtReflectedObject* **nextObj**() const
- virtual DtReflectedObject* **prevObj**() const
- virtual DtReflectedObject* **wrapNextObj**() const
- virtual DtReflectedObject* **wrapPrevObj**() const

- virtual void **setHlaObject**(DtHlaObject* obj, DtHlaStateDecoder* decoder = NULL)
- virtual void **setNoHlaObject**(const DtObjectId& id, const DtGlobalObjectDesignator& des)
- virtual void **detachHlaObject**()
- inline DtHlaObject* **hlaObject**() const
- inline const DtObjectId& **localId**() const
- inline DtReflectedObject* **nextObj**() const
- inline DtReflectedObject* **prevObj**() const
- inline DtStateRepository* **sr**() const
- inline DtStateRepository* **stateRep**() const
- inline DtReflectedObject* **wrapNextObj**() const
- inline DtReflectedObject* **wrapPrevObj**() const

Protected Fields

- DtExerciseConn* **myConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtHlaStateDecoder* **myDecoder**
- DtBoolean **myRequestCbRegistered**
- DtIntrusiveList **myStateUpdateCallbacks**
- DtObjectId **myId**
- DtString **myName**

Protected Methods

- virtual void **requestUpdate**()
- virtual void **processPostUpdateCallbacks**()
- static void **requestUpdatePostDrainCb**(void* usr)
- static void **processPostUpdateCallbacksCb**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr)

Inherited from DtListedObj:

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

class DtReflectedAggregate:

Instances of DtReflectedAggregate are used to represent a simulation aggregate reflected across a network (i.e. this is not an full simulation running on this machine).

- **`typedef DtAggregateStateRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtAggregateStateRepositories

- **`DtReflectedAggregate(DtHlaObject* obj, DtExerciseConn* conn)`**

  constructor

- **`virtual ~DtReflectedAggregate()`**

  destructor

- **`virtual DtAggregateStateRepository* aggregateStateRep() const`**

  Returns a pointer to the AggregateStateRepository - through which you can inspect current state information

- **`virtual DtAggregateStateRepository* asr() const`**

  Synonym for aggregateStateRep()

- **`virtual DtReflectedAggregate* next() const`**

  Get the next remote aggregate

- **`virtual DtReflectedAggregate* prev() const`**

  Get the previous remote aggregate

- **`virtual DtReflectedAggregate* wrapNext() const`**

  Get the next remote aggregate. Will cycle back to the first remote aggregate.

- **`virtual DtReflectedAggregate* wrapPrev() const`**

  Get the previous remote aggregate. Will cycle back to the last remote aggregate.

- **`virtual void addPostUpdateCallback( DtAggregateCallbackFunction cb, void* userData)`**

  Aggregate state callback registration/deregistration

- **`virtual void removePostUpdateCallback( DtAggregateCallbackFunction cb, void* userData)`**

- **`static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)`**

  Give us a function to be used for creating SRs

- **`virtual RTI::ObjectHandle aggregateObjectId() const`**

Get the aggregate HLA object's id For backwards compatibility. Use id() instead.

- **`DtReflectedAggregate(const DtReflectedAggregate& orig)`**

  copy constructor

- **`DtReflectedAggregate& operator=(const DtReflectedAggregate& orig)`**

  assignment operator

- **`static DtStateRepCreator theStateRepCreator`**

- **`inline DtAggregateStateRepository* aggregateStateRep() const`**

- **`inline DtAggregateStateRepository* asr() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file basereflent.h:

class DtBaseReflectedEntity: public DtListedObj

Inheritance:

DtBaseReflectedEntity - DtListedObj

Public Methods

- virtual **~DtBaseReflectedEntity** ()
  destructor
- DtBaseReflectedEntity& **operator=** (const DtBaseReflectedEntity& orig)
  assignment operator
- virtual DtEntityStateRepository* **entityStateRep** () const
  Get the entity's state repository
- virtual const DtObjectId& **id** () const
  Get the entity's identifier
- **DtBaseReflectedEntity** (const DtObjectId& id)
  constructor
- inline const DtObjectId& **id** () const

Protected Fields

- DtObjectId **myObjectId**

Protected Methods

- **DtBaseReflectedEntity** (const DtBaseReflectedEntity& orig)
  copy constructor

Protected

- *Constructors are protected since this is an abstract base class*

Inherited from DtListedObj:

Public Methods

- void* **next**() const
- void* **prev**() const
- void* **wrapNext**() const
- void* **wrapPrev**() const
- inline void* **next**() const
- inline void* **prev**() const
- inline void* **wrapNext**() const
- inline void* **wrapPrev**() const

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

- **`virtual ~DtBaseReflectedEntity()`**
  destructor
- **`DtBaseReflectedEntity& operator=(const DtBaseReflectedEntity& orig)`**
  assignment operator
- **`virtual DtEntityStateRepository* entityStateRep() const`**
  Get the entity's state repository
- **`virtual const DtObjectId& id() const`**
  Get the entity's identifier
-
  Constructors are protected since this is an abstract base class
- **`DtBaseReflectedEntity(const DtObjectId& id)`**
  constructor
- **`DtBaseReflectedEntity(const DtBaseReflectedEntity& orig)`**
  copy constructor
- **`DtObjectId myObjectId`**

`inline const DtObjectId& id() const`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file basereflagg.h:

class DtBaseReflectedAggregate: public DtListedObj

Inheritance:

DtBaseReflectedAggregate - DtListedObj

Public Methods

virtual	**~DtBaseReflectedAggregate** () *destructor*
DtBaseReflectedAggregate&	**operator=** (const DtBaseReflectedAggregate& orig) *assignment operator*
virtual DtAggregateStateRepository*	**aggregateStateRep** () const *Get the aggregate's state repository*
virtual const DtObjectId&	**id** () const *Get the aggregate's identifier*
	DtBaseReflectedAggregate (const DtObjectId& id) *constructor*

Protected Fields

- DtObjectId **myObjectId**

Protected Methods

- **DtBaseReflectedAggregate** (const DtBaseReflectedAggregate& orig)
  copy constructor

Protected

- *Constructors are protected since this is an abstract base class*

Inherited from DtListedObj:

Public Methods

- void* **next**() const
- void* **prev**() const
- void* **wrapNext**() const
- void* **wrapPrev**() const
- inline void* **next**() const
- inline void* **prev**() const
- inline void* **wrapNext**() const
- inline void* **wrapPrev**() const

Protected Fields

- DtList* **myList**
- DtListItem* **myItem**

Documentation

- **`virtual ~DtBaseReflectedAggregate()`**
  destructor
- **`DtBaseReflectedAggregate& operator=(const DtBaseReflectedAggregate& orig)`**
  assignment operator
- **`virtual DtAggregateStateRepository* aggregateStateRep() const`**
  Get the aggregate's state repository
- **`virtual const DtObjectId& id() const`**
  Get the aggregate's identifier
-
  Constructors are protected since this is an abstract base class
- **`DtBaseReflectedAggregate(const DtObjectId& id)`**
  constructor
- **`DtBaseReflectedAggregate(const DtBaseReflectedAggregate& orig)`**
  copy constructor
- **`DtObjectId myObjectId`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file netstructs.h:

class DtNet32Vector

Public Methods

	DtNet32Vector ()
	DtNet32Vector (const DtVector &vec)
DtVector	**operator =** (const DtVector &vec)
	operator DtVector () const
DtNetFloat32&	**operator[]** (int i)
DtNetFloat32	**operator[]** (int i) const

Private Fields

DtNetFloat32	**rep** [3]

Documentation

- `DtNet32Vector()`
- `DtNet32Vector(const DtVector &vec)`
- `DtVector operator = (const DtVector &vec)`
- `operator DtVector() const`
- `DtNetFloat32& operator[](int i)`
- `DtNetFloat32 operator[](int i) const`
- `DtNetFloat32 rep[3]`

This class has no child classes.

In file netstructs.h:

if DT_LITTLE_ENDIAN class DtNet64Vector

Public Methods

	DtNet64Vector ()
	DtNet64Vector (const DtVector &vec)
DtVector	**operator =** (const DtVector &vec)
	operator DtVector () const
DtNetFloat64&	**operator[]** (int i)
DtNetFloat64	**operator[]** (int i) const

Private Fields

DtNetFloat64	**rep** [3]

Documentation

- `DtNet64Vector()`
- `DtNet64Vector(const DtVector &vec)`
- `DtVector operator = (const DtVector &vec)`
- `operator DtVector() const`
- `DtNetFloat64& operator[](int i)`
- `DtNetFloat64 operator[](int i) const`
- `DtNetFloat64 rep[3]`

This class has no child classes.

In file netboolean.h:

class DtNetBoolean

Public Methods

- **DtNetBoolean** ()
  default constructor
- **~DtNetBoolean** ()
  destructor
- **DtNetBoolean** (const DtNetBoolean& orig)
  copy constructor
- DtNetBoolean& **operator=** (const DtNetBoolean& orig)
  assignment operator
- **DtNetBoolean** (DtBoolean val)
  from host constructor
- DtNetBoolean& **operator=** (DtBoolean val)
  from host assignment operator
- DtNetBoolean& **operator DtBoolean** () const
  to host cast operator

Public

- *If a copy constructor is implemented, the assignment operator should also be implemented*

Protected Fields

- DtNetU32 **myRep**

Documentation

- **`DtNetBoolean()`**
  default constructor
- **`~DtNetBoolean()`**
  destructor
- If a copy constructor is implemented, the assignment operator should also be implemented. Should generally implement both of them anyway to avoid memory problems due to inappropriate compiler provided defaults. Access varies so need to put proper access keyword above these declarations.

- **`DtNetBoolean(const DtNetBoolean& orig)`**
  copy constructor
- **`DtNetBoolean& operator=(const DtNetBoolean& orig)`**
  assignment operator
- **`DtNetBoolean(DtBoolean val)`**
  from host constructor
- **`DtNetBoolean& operator=(DtBoolean val)`**
  from host assignment operator
- **`DtNetBoolean& operator DtBoolean() const`**
  to host cast operator
- **`DtNetU32 myRep`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file netstructs.h:

class DtNetEntityMarking

Public Fields

- DtNetU8 **DtCharacterSet**
- DtNetU8 **DtText** [DtMaxEntityMarkingLength]

Public

- #define **DtMaxEntityMarkingLength**

Documentation

- **`DtNetU8 DtCharacterSet`**
- **`#define DtMaxEntityMarkingLength`**
- **`DtNetU8 DtText[DtMaxEntityMarkingLength]`**

This class has no child classes.

In file netstructs.h:

class DtNetEulerAngles

Public Fields

- DtNetFloat32 **psi**
- DtNetFloat32 **theta**
  Counter clockwise about z
- DtNetFloat32 **phi**
  Counter clockwise about y
- *Counter clockwise about x*

Public Methods

- **DtNetEulerAngles** ()
- **DtNetEulerAngles** (const DtTaitBryan &tait)
- DtTaitBryan **operator =** (const DtTaitBryan &tait)
- **operator DtTaitBryan** () const

Documentation

- **`DtNetEulerAngles()`**
- **`DtNetEulerAngles(const DtTaitBryan &tait)`**
- **`DtTaitBryan operator = (const DtTaitBryan &tait)`**
- **`operator DtTaitBryan() const`**
- **`DtNetFloat32 psi`**
- **`DtNetFloat32 theta`**
  Counter clockwise about z
- **`DtNetFloat32 phi`**
  Counter clockwise about y
- Counter clockwise about x

This class has no child classes.

In file nettypes.h:

class DtNetFloat32

Network-order 32-bit IEEE-754 floating-point quantity

Public Fields

- DtNetU32 **bits**

Public Methods

- **DtNetFloat32** ()
- **DtNetFloat32** (DtFloat32)
- **DtNetFloat32** (const DtNetFloat32&)
- DtFloat32 **operator =** (DtFloat32)
- **operator DtFloat32** () const

Documentation

Network-order 32-bit IEEE-754 floating-point quantity

- `DtNetU32 bits`
- `DtNetFloat32()`
- `DtNetFloat32(DtFloat32)`
- `DtNetFloat32(const DtNetFloat32&)`
- `DtFloat32 operator = (DtFloat32)`
- `operator DtFloat32() const`

This class has no child classes.

In file nettypes.h:

class DtNetFloat64

Network-order 64-bit IEEE-754 floating-point quantity

Public Fields

- DtNetU32 **mslong**
- DtNetU32 **lslong**
  Most Significant Long

Public Methods

- **DtNetFloat64** ()
  Least Significant Least
- **DtNetFloat64** (DtFloat64)
- **DtNetFloat64** (const DtNetFloat64&)
- DtFloat64 **operator =** (DtFloat64)
- **operator DtFloat64** () const

Documentation

Network-order 64-bit IEEE-754 floating-point quantity

- **`DtNetU32 mslong`**
- **`DtNetU32 lslong`**
  Most Significant Long
- **`DtNetFloat64()`**
  Least Significant Least
- **`DtNetFloat64(DtFloat64)`**
- **`DtNetFloat64(const DtNetFloat64&)`**
- **`DtFloat64 operator = (DtFloat64)`**
- **`operator DtFloat64() const`**

This class has no child classes.

In file netstructs.h:

class DtNetGlobalObjectDesignator

Public Methods

- DtNetGlobalObjectDesignator& **operator=** (const DtGlobalObjectDesignator& des)
- **operator DtGlobalObjectDesignator** () const

Protected Fields

- char **myRep** [DtMAX_OBJECT_NAME_LENGTH]

Documentation

- **`DtNetGlobalObjectDesignator& operator=( const DtGlobalObjectDesignator& des)`**
- **`operator DtGlobalObjectDesignator() const`**
- **`char myRep[DtMAX_OBJECT_NAME_LENGTH]`**

This class has no child classes.

In file nettypes.h:

class DtNetInt16

Network-order signed 16-bit quantity

Public Fields

- DtNetInt8 **msbyte**
- DtNetInt8 **lsbyte**

Public Methods

- **DtNetInt16** ()
- **DtNetInt16** (DtInt16)
- **DtNetInt16** (const DtNetInt16&)
- DtInt16 **operator =** (DtInt16)
- **operator DtInt16** () const

Documentation

Network-order signed 16-bit quantity

- `DtNetInt8 msbyte`
- `DtNetInt8 lsbyte`
- `DtNetInt16()`
- `DtNetInt16(DtInt16)`
- `DtNetInt16(const DtNetInt16&)`
- `DtInt16 operator = (DtInt16)`
- `operator DtInt16() const`

This class has no child classes.

In file nettypes.h:

class DtNetInt32

Network-order signed 32-bit quantity

Public Fields

- DtNetInt16 **msword**
- DtNetInt16 **lsword**
  Most Significant Word

Public Methods

- **DtNetInt32** ()
  Least Significant Word
- **DtNetInt32** (DtInt32)
- **DtNetInt32** (const DtNetInt32&)
- DtInt32 **operator =** (DtInt32)
- **operator DtInt32** () const

Documentation

Network-order signed 32-bit quantity

- **`DtNetInt16 msword`**
- **`DtNetInt16 lsword`**

  Most Significant Word
- **`DtNetInt32()`**

  Least Significant Word
- **`DtNetInt32(DtInt32)`**
- **`DtNetInt32(const DtNetInt32&)`**
- **`DtInt32 operator = (DtInt32)`**
- **`operator DtInt32() const`**

This class has no child classes.

In file nettypes.h:

class DtNetU16

Network-order unsigned 16-bit quantity

Public Fields

- DtNetU8 **msbyte**
- DtNetU8 **lsbyte**
  Most Significant Byte

Public Methods

- **DtNetU16** ()
  Least Significant Byte
- **DtNetU16** (DtU16)
- **DtNetU16** (const DtNetU16&)
- DtU16 **operator =** (DtU16)
- **operator DtU16** () const

Documentation

Network-order unsigned 16-bit quantity

- **`DtNetU8 msbyte`**
- **`DtNetU8 lsbyte`**
  Most Significant Byte
- **`DtNetU16()`**
  Least Significant Byte
- **`DtNetU16(DtU16)`**
- **`DtNetU16(const DtNetU16&)`**
- **`DtU16 operator = (DtU16)`**
- **`operator DtU16() const`**

This class has no child classes.

In file nettypes.h:

class DtNetU32

Network-order unsigned 32-bit quantity

Public Fields

- DtNetU16 **msword**
- DtNetU16 **lsword**
  Most Significant Word

Public Methods

- **DtNetU32** ()
  Least Significant Word
- **DtNetU32** (DtU32)
- **DtNetU32** (const DtNetU32&)
- DtU32 **operator =** (DtU32)
- **operator DtU32** () const

Documentation

Network-order unsigned 32-bit quantity

- **`DtNetU16 msword`**
- **`DtNetU16 lsword`**
  Most Significant Word
- **`DtNetU32()`**
  Least Significant Word
- **`DtNetU32(DtU32)`**
- **`DtNetU32(const DtNetU32&)`**
- **`DtU32 operator = (DtU32)`**
- **`operator DtU32() const`**

This class has no child classes.

In file hobjpub.h:

class DtObjectPublisher

class DtObjectPublisher: Instances of DtObjectPublisher are used for local or source simulations of objects, (i

---

Inheritance:

DtObjectPublisher

Public Methods

	DtObjectPublisher (DtExerciseConn* conn, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote) *Constructor*
virtual	**~DtObjectPublisher** () *Destructor*
virtual void	**setHlaObject** (const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL) *A DtObjectPublisher is not capable of communicating data through the RTI until setHlaObject is called*
virtual void	**setHlaObject** (RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL) *Same as above, but you pass object class name rather than handle*
virtual void	**detachHlaObject** () *Detach HLA object - bringing the DtObjectPublisher back to the state it was in before setHlaObject was called*
virtual DtStateRepository*	**stateRep** () *Returns a pointer to the DtStateRepository - through which you can set or inspect current state information*
virtual DtStateRepository*	**sr** () *Synonym for stateRep()*
virtual DtHlaStateDecoder*	**decoder** () *Returns a pointer to the Decoder being used to decode state messages that we send back into the asSeenByRemote SR*
virtual DtHlaStateEncoder*	**encoder** () *Returns a pointer to the Encoder being used to encode current state into an outgoing message*
virtual DtHlaObject*	**hlaObject** () *Get the object's hlaObject*
virtual DtExerciseConn*	**exerciseConn** () *Get our exercise connection*

virtual const DtObjectId&	**id** () const *Get the object's local identifier*
virtual const DtObjectId&	**objectId** () const *Get the object's local identifier*
virtual const DtObjectId&	**localId** () const *Get the object's local identifier*
virtual const DtString&	**name** () const *Get the object's name*
virtual const DtGlobalObjectDesignator&	**globalId** () const *Get the HLA object's global object designator - the object name in RTI1*
virtual void	**forceUpdate** () *Specify a set of attributes that should be included in the ahvps the next time tick is called, even if they would not otherwise need to be sent*
virtual void	**forceUpdate** (const RTI::AttributeHandleSet& set)
virtual void	**tick** () *Tick (update) this object*
	DtObjectPublisher (DtExerciseConn* conn, const char* name = NULL) *These constructors and init functions are obsolete, but are here for backwards compatibility with pre-3*
void	**init** (RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
void	**init** (const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
inline DtExerciseConn*	**exerciseConn** ()
inline DtHlaObject*	**hlaObject** ()
inline const DtObjectId&	**localId** () const
inline DtStateRepository*	**sr** ()
inline DtStateRepository*	**stateRep** ()

Protected Fields

DtExerciseConn*	**myExConn**
DtHlaObjectWithStateRep*	**myHlaObj**
DtStateRepository*	**myStateRep**
DtStateRepository*	**myAsSeenByRemote**
DtHlaStateEncoder*	**myEncoder**
DtHlaStateDecoder*	**myDecoder**
DtObjectId	**myObjectId**
DtString	**myName**

Protected Methods

	DtObjectPublisher (const DtObjectPublisher& orig) *copy constructor*
DtObjectPublisher&	**operator=** (const DtObjectPublisher& orig) *assignment operator*

Documentation

class DtObjectPublisher:

Instances of DtObjectPublisher are used for local or source simulations of objects, (i.e. these objects are not dependent on data being produced elsewhere on the network for their existence). These object send data to the network.

`DtObjectPublisher(DtExerciseConn* conn, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote)`

Constructor. Object is not fully constructed until setHlaObject is called - typically by subclass constructor.

`virtual ~DtObjectPublisher()`

Destructor

`virtual void setHlaObject( const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)`

A DtObjectPublisher is not capable of communicating data through the RTI until setHlaObject is called. It is typically called by subclass constructors. If encoder or decoder are NULL, the FOM Mapper is consulted. This class assumes ownership of the encoder/decoder objects passed to setHlaObject, and will delete them on destruction. If objectName is NULL, a name is obtained from the RTI.

`virtual void setHlaObject( RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)`

Same as above, but you pass object class name rather than handle

`virtual void detachHlaObject()`

Detach HLA object - bringing the DtObjectPublisher back to the state it was in before setHlaObject was called. Typically called only by destructor.

`virtual DtStateRepository* stateRep()`

Returns a pointer to the DtStateRepository - through which you can set or inspect current state information

`virtual DtStateRepository* sr()`

Synonym for stateRep()

`virtual DtHlaStateDecoder* decoder()`

Returns a pointer to the Decoder being used to decode state messages that we send back into the asSeenByRemote SR.

`virtual DtHlaStateEncoder* encoder()`

Returns a pointer to the Encoder being used to encode current state into an outgoing message

`virtual DtHlaObject* hlaObject()`

Get the object's hlaObject

- **virtual DtExerciseConn* exerciseConn()**

  Get our exercise connection

- **virtual const DtObjectId& id() const**

  Get the object's local identifier

- **virtual const DtObjectId& objectId() const**

  Get the object's local identifier

- **virtual const DtObjectId& localId() const**

  Get the object's local identifier

- **virtual const DtString& name() const**

  Get the object's name

- **virtual const DtGlobalObjectDesignator& globalId() const**

  Get the HLA object's global object designator - the object name in RTI1.3.3, but the objectId in RTI1.0.3.

- **virtual void forceUpdate()**

  Specify a set of attributes that should be included in the ahvps the next time tick is called, even if they would not otherwise need to be sent. The call with no arguments, indicates the set of all of the object's attributes. Multiple calls are cumulative, until the set is cleared with clearForcedAttributes, which should happen within tick.

- **virtual void forceUpdate(const RTI::AttributeHandleSet& set)**

- **virtual void tick()**

  Tick (update) this object

- **DtObjectPublisher(const DtObjectPublisher& orig)**

  copy constructor

- **DtObjectPublisher& operator= (const DtObjectPublisher& orig)**

  assignment operator

- **DtExerciseConn* myExConn**

- **DtHlaObjectWithStateRep* myHlaObj**

- **DtStateRepository* myStateRep**

- **DtStateRepository* myAsSeenByRemote**

- **DtHlaStateEncoder* myEncoder**

- **DtHlaStateDecoder* myDecoder**

- **DtObjectId myObjectId**

- **DtString myName**

- **DtObjectPublisher(DtExerciseConn* conn, const char* name = NULL)**

  These constructors and init functions are obsolete, but are here for backwards compatibility with pre-3.4 releases. Please use the constructor and setHlaObject functions described at the top of this class definition instead.

- **void init(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)**
- **void init(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)**
- **inline DtExerciseConn* exerciseConn()**
- **inline DtHlaObject* hlaObject()**
- **inline const DtObjectId& localId() const**
- **inline DtStateRepository* sr()**
- **inline DtStateRepository* stateRep()**

Direct child classes:

DtRadioTransmitterPublisher
DtRadioReceiverPublisher
DtGenericObjectPublisher
DtEntityPublisher
DtEmitterSystemPublisher
DtEmitterBeamPublisher
DtDesignatorPublisher
DtAggregatePublisher

Alphabetic index *Hierarchy of classes*

In file hradioxmitpub.h:

class DtRadioTransmitterPublisher: public DtObjectPublisher

class DtRadioTransmitterPublisher: Instances of DtRadioTransmitterPublisher are used for local or source simulations of radio transmitters, (i

---

Inheritance:

DtRadioTransmitterPublisher - DtObjectPublisher

Public Methods

typedef DtRadioTransmitterRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtRadioTransmitterRepositories*
	DtRadioTransmitterPublisher (RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL) *constructor*
	DtRadioTransmitterPublisher (DtExerciseConn* conn, const char* name = NULL) *constructor*
virtual	**~DtRadioTransmitterPublisher** () *destructor*
virtual DtRadioTransmitterRepository*	**radioXmitRep** () *Returns a pointer to the DtRadioTransmitterRepository - through which you can set or inspect current state information*
virtual DtRadioTransmitterRepository*	**tsr** () *Synonym for radioXmitRep()*
virtual const char*	**classToUse** () const *Returns the name of the FOM object class to use*
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtRadioTransmitterRepository*	**radioXmitRep** ()
inline DtRadioTransmitterRepository*	**tsr** ()

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtRadioTransmitterPublisher** (const DtRadioTransmitterPublisher& orig)
  copy constructor
- DtRadioTransmitterPublisher& **operator=** (const DtRadioTransmitterPublisher& orig)
  copy constructor

Inherited from DtObjectPublisher:

Public Methods

- virtual void **setHlaObject**(const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **setHlaObject**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **detachHlaObject**()
- virtual DtStateRepository* **stateRep**()
- virtual DtStateRepository* **sr**()
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaStateEncoder* **encoder**()
- virtual DtHlaObject* **hlaObject**()
- virtual DtExerciseConn* **exerciseConn**()
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **tick**()
- void **init**(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote,

DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)

- void **init**(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- inline DtExerciseConn* **exerciseConn**()
- inline DtHlaObject* **hlaObject**()
- inline const DtObjectId& **localId**() const
- inline DtStateRepository* **sr**()
- inline DtStateRepository* **stateRep**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**
- DtHlaStateEncoder* **myEncoder**
- DtHlaStateDecoder* **myDecoder**
- DtObjectId **myObjectId**
- DtString **myName**

Protected Methods

- DtObjectPublisher& **operator=** (const DtObjectPublisher& orig)

Documentation

class DtRadioTransmitterPublisher:

Instances of DtRadioTransmitterPublisher are used for local or source simulations of radio transmitters, (i.e. these entities are not dependent on data being produced elsewhere on the network for their existence). These radio transmitters send data to the network.

- **`typedef DtRadioTransmitterRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtRadioTransmitterRepositories

- **`DtRadioTransmitterPublisher(RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL)`**

  constructor

- **`DtRadioTransmitterPublisher(DtExerciseConn* conn, const char* name = NULL)`**

  constructor

- **`virtual ~DtRadioTransmitterPublisher()`**

  destructor

- **`virtual DtRadioTransmitterRepository* radioXmitRep()`**

  Returns a pointer to the DtRadioTransmitterRepository - through which you can set or inspect current state information

- **`virtual DtRadioTransmitterRepository* tsr()`**

  Synonym for radioXmitRep()

- **`virtual const char* classToUse() const`**

  Returns the name of the FOM object class to use. Default implementation is to ask the FOM Mapper.

- **`static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)`**

  Give us a function to be used for creating SRs

- **`DtRadioTransmitterPublisher(const DtRadioTransmitterPublisher& orig)`**

  copy constructor

- **`DtRadioTransmitterPublisher& operator= (const DtRadioTransmitterPublisher& orig)`**

  copy constructor

- **`static DtStateRepCreator theStateRepCreator`**

- **`inline DtRadioTransmitterRepository* radioXmitRep()`**

- **`inline DtRadioTransmitterRepository* tsr()`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hradiorcvrpub.h:

class DtRadioReceiverPublisher: public DtObjectPublisher

class DtRadioReceiverPublisher: Instances of DtRadioReceiverPublisher are used for local or source simulations of radio receivers, (i

---

Inheritance:

DtRadioReceiverPublisher - DtObjectPublisher

Public Methods

typedef DtRadioReceiverRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtRadioReceiverRepositories*
	DtRadioReceiverPublisher (RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL) *constructor*
	DtRadioReceiverPublisher (DtExerciseConn* conn, const char* name = NULL) *constructor*
virtual	**~DtRadioReceiverPublisher** () *destructor*
virtual DtRadioReceiverRepository*	**receiverRep** () *Returns a pointer to the DtRadioReceiverRepository - through which you can set or inspect current state information*
virtual DtRadioReceiverRepository*	**rsr** () *Synonym for receiverRep()*
virtual const char*	**classToUse** () const *Returns the name of the FOM object class to use*
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtRadioReceiverRepository*	**receiverRep** ()
inline DtRadioReceiverRepository*	**rsr** ()

Protected Fields

static DtStateRepCreator	**theStateRepCreator**

Protected Methods

	DtRadioReceiverPublisher (const DtRadioReceiverPublisher& orig) *copy constructor*
DtRadioReceiverPublisher&	**operator=** (const DtRadioReceiverPublisher& orig) *copy constructor*

Inherited from DtObjectPublisher:

Public Methods

- virtual void **setHlaObject**(const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **setHlaObject**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **detachHlaObject**()
- virtual DtStateRepository* **stateRep**()
- virtual DtStateRepository* **sr**()
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaStateEncoder* **encoder**()
- virtual DtHlaObject* **hlaObject**()
- virtual DtExerciseConn* **exerciseConn**()
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **tick**()
- void **init**(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- void **init**(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- inline DtExerciseConn* **exerciseConn**()

- inline DtHlaObject* **hlaObject**()
- inline const DtObjectId& **localId**() const
- inline DtStateRepository* **sr**()
- inline DtStateRepository* **stateRep**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**
- DtHlaStateEncoder* **myEncoder**
- DtHlaStateDecoder* **myDecoder**
- DtObjectId **myObjectId**
- DtString **myName**

Protected Methods

- DtObjectPublisher& **operator=** (const DtObjectPublisher& orig)

Documentation

class DtRadioReceiverPublisher:

Instances of DtRadioReceiverPublisher are used for local or source simulations of radio receivers, (i.e. these entities are not dependent on data being produced elsewhere on the network for their existence). These radio receivers send data to the network.

- **`typedef DtRadioReceiverRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtRadioReceiverRepositories

- **`DtRadioReceiverPublisher(RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL)`**

  constructor

- **`DtRadioReceiverPublisher(DtExerciseConn* conn, const char* name = NULL)`**

  constructor

- **`virtual ~DtRadioReceiverPublisher()`**

  destructor

- **virtual DtRadioReceiverRepository* receiverRep()**

  Returns a pointer to the DtRadioReceiverRepository - through which you can set or inspect current state information

- **virtual DtRadioReceiverRepository* rsr()**

  Synonym for receiverRep()

- **virtual const char* classToUse() const**

  Returns the name of the FOM object class to use. Default implementation is to ask the FOM Mapper.

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**

  Give us a function to be used for creating SRs

- **DtRadioReceiverPublisher(const DtRadioReceiverPublisher& orig)**

  copy constructor

- **DtRadioReceiverPublisher& operator= (const DtRadioReceiverPublisher& orig)**

  copy constructor

- **static DtStateRepCreator theStateRepCreator**

- **inline DtRadioReceiverRepository* receiverRep()**

- **inline DtRadioReceiverRepository* rsr()**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file genericpub.h:

class DtGenericObjectPublisher: public DtObjectPublisher

class DtGenericObjectPublisher: Instances of DtGenericObjectPublisher are used for local or source simulations of objects for which a more specific publisher does not exist

---

Inheritance:

DtGenericObjectPublisher - DtObjectPublisher

Public Methods

	DtGenericObjectPublisher (DtExerciseConn* conn, const char* className, const char* objName = NULL) *Constructor - configure base class with proper FOM class, state repositories, encoder and decoder*
virtual	**~DtGenericObjectPublisher** () *destructor*
virtual DtGenericStateRepository*	**genericStateRep** () *Returns a pointer to the DtGenericStateRepository - through which you can set current state information*
virtual DtGenericStateRepository*	**gsr** () *Synonym for genericStateRep()*

Inherited from DtObjectPublisher:

Public Methods

virtual void **setHlaObject**(const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)

virtual void **setHlaObject**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)

virtual void **detachHlaObject**()

virtual DtStateRepository* **stateRep**()

virtual DtStateRepository* **sr**()

- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaStateEncoder* **encoder**()
- virtual DtHlaObject* **hlaObject**()
- virtual DtExerciseConn* **exerciseConn**()
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **tick**()
- void **init**(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- void **init**(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- inline DtExerciseConn* **exerciseConn**()
- inline DtHlaObject* **hlaObject**()
- inline const DtObjectId& **localId**() const
- inline DtStateRepository* **sr**()
- inline DtStateRepository* **stateRep**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**
- DtHlaStateEncoder* **myEncoder**
- DtHlaStateDecoder* **myDecoder**
- DtObjectId **myObjectId**
- DtString **myName**

Protected Methods

DtObjectPublisher& **operator=** (const DtObjectPublisher& orig)

Documentation

class DtGenericObjectPublisher:

Instances of DtGenericObjectPublisher are used for local or source simulations of objects for which a more specific publisher does not exist. The publisher sends data to the network when necessary. This class is derived from DtObjectPublisher, which does most of the work. The derived class mostly exists to provide typesafety, and to configure the base object.

`DtGenericObjectPublisher(DtExerciseConn* conn, const char* className, const char* objName = NULL)`

Constructor - configure base class with proper FOM class, state repositories, encoder and decoder

`virtual ~DtGenericObjectPublisher()`

destructor

`virtual DtGenericStateRepository* genericStateRep()`

Returns a pointer to the DtGenericStateRepository - through which you can set current state information

`virtual DtGenericStateRepository* gsr()`

Synonym for genericStateRep()

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hentitypub.h:

class DtEntityPublisher: public DtObjectPublisher

class DtEntityPublisher: Instances of DtEntityPublisher are used for local or source simulations of entities, (i

---

Inheritance:

DtEntityPublisher - DtObjectPublisher

Public Methods

typedef DtEntityStateRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtEntityStateRepositories*
	DtEntityPublisher (RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL) *constructor*
	DtEntityPublisher (const DtEntityType& type, DtExerciseConn* conn, const char* name = NULL) *constructor*
	DtEntityPublisher (const DtEntityType& type, DtExerciseConn* conn, DtDeadReckonTypes algorithm, DtForceType forceId, const DtEntityType& guise = gGuiseSameAsType(), const char* name = NULL) *constructor*
virtual	**~DtEntityPublisher** () *destructor*
virtual DtEntityStateRepository*	**entityStateRep** () *Returns a pointer to the EntityStateRepository - through which you can set or inspect current state information*
virtual DtEntityStateRepository*	**esr** () *Synonym for entityStateRep()*
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
static void	**setDfltRotationThreshold** (DtReal r) *These four functions are obsolete, but remain for backwards compatibility*
static double	**dfltRotationThreshold** ()
static void	**setDfltTranslationThreshold** (DtReal t)
static double	**dfltTranslationThreshold** ()
static const DtEntityType&	**guiseSameAsType** () *Returns a constant that can be passed to the constructor indicating that we should use the entity type as our guise*

- inline DtEntityStateRepository* **entityStateRep** ()
- inline DtEntityStateRepository* **esr** ()

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtEntityPublisher** (const DtEntityPublisher& orig)
  copy constructor
- DtEntityPublisher& **operator=** (const DtEntityPublisher& orig)
  assignment operator
- const char* **objectClassNameFromType** (const DtEntityType& type)
  Non-virtual since called from constructor

Inherited from DtObjectPublisher:

Public Methods

- virtual void **setHlaObject**(const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **setHlaObject**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **detachHlaObject**()
- virtual DtStateRepository* **stateRep**()
- virtual DtStateRepository* **sr**()
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaStateEncoder* **encoder**()
- virtual DtHlaObject* **hlaObject**()
- virtual DtExerciseConn* **exerciseConn**()
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const

- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **tick**()
- void **init**(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- void **init**(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- inline DtExerciseConn* **exerciseConn**()
- inline DtHlaObject* **hlaObject**()
- inline const DtObjectId& **localId**() const
- inline DtStateRepository* **sr**()
- inline DtStateRepository* **stateRep**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**
- DtHlaStateEncoder* **myEncoder**
- DtHlaStateDecoder* **myDecoder**
- DtObjectId **myObjectId**
- DtString **myName**

Protected Methods

- DtObjectPublisher& **operator=** (const DtObjectPublisher& orig)

Documentation

class DtEntityPublisher:

Instances of DtEntityPublisher are used for local or source simulations of entities, (i.e. these entities are not dependent on data being produced elsewhere on the network for their existence). These entities send data to the network.

- **`typedef DtEntityStateRepository* (*DtStateRepCreator)()`**

Function prototype for functions that create DtEntityStateRepositories

- **DtEntityPublisher(RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL)**

  constructor

- **DtEntityPublisher(const DtEntityType& type, DtExerciseConn* conn, const char* name = NULL)**

  constructor

- **DtEntityPublisher(const DtEntityType& type, DtExerciseConn* conn, DtDeadReckonTypes algorithm, DtForceType forceId, const DtEntityType& guise = gGuiseSameAsType(), const char* name = NULL)**

  constructor

- **virtual ~DtEntityPublisher()**

  destructor

- **virtual DtEntityStateRepository* entityStateRep()**

  Returns a pointer to the EntityStateRepository - through which you can set or inspect current state information

- **virtual DtEntityStateRepository* esr()**

  Synonym for entityStateRep()

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**

  Give us a function to be used for creating SRs

- **static void setDfltRotationThreshold(DtReal r)**

  These four functions are obsolete, but remain for backwards compatibility. Pleas use the DtThresholder static member functions instead.

- **static double dfltRotationThreshold()**

- **static void setDfltTranslationThreshold(DtReal t)**

- **static double dfltTranslationThreshold()**

- **static const DtEntityType& guiseSameAsType()**

  Returns a constant that can be passed to the constructor indicating that we should use the entity type as our guise

- **DtEntityPublisher(const DtEntityPublisher& orig)**

  copy constructor

- **DtEntityPublisher& operator=(const DtEntityPublisher& orig)**

  assignment operator

- **const char* objectClassNameFromType(const DtEntityType& type)**

  Non-virtual since called from constructor. If you don't like the default mapping between entity type and object class, just pass the object class handle you want to the constructor.

- **static DtStateRepCreator theStateRepCreator**

- **inline DtEntityStateRepository* entityStateRep()**

- **inline DtEntityStateRepository* esr()**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hemittrsyspub.h:

class DtEmitterSystemPublisher: public DtObjectPublisher

class DtEmitterSystemPublisher: Instances of DtEmitterSystemPublisher are used for local or source simulations of emitter systems, (i

---

Inheritance:

DtEmitterSystemPublisher - DtObjectPublisher

Public Methods

typedef DtEmitterSystemRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtEmitterSystemRepositories*
	DtEmitterSystemPublisher (RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL) *constructor*
	DtEmitterSystemPublisher (DtExerciseConn* conn, const char* name = NULL) *constructor*
virtual	**~DtEmitterSystemPublisher** () *destructor*
virtual DtEmitterSystemRepository*	**emitterSystemRep** () *Returns a pointer to the DtEmitterSystemRepository - through which you can set or inspect current state information*
virtual DtEmitterSystemRepository*	**esr** () *Synonym for emitterSystemRep()*
virtual const char*	**classToUse** () const *Returns the name of the FOM object class to use*
virtual void	**tick** () *syncBeamList and update beam objects as well as main system object*
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtEmitterSystemRepository*	**emitterSystemRep** ()
inline DtEmitterSystemRepository*	**esr** ()

Protected Fields

DtList	**myBeamPubs** *List of DtEmitterBeamPubs*
static DtStateRepCreator	**theStateRepCreator**

Protected Methods

	DtEmitterSystemPublisher (const DtEmitterSystemPublisher& orig) *copy constructor*
DtEmitterSystemPublisher&	**operator=** (const DtEmitterSystemPublisher& orig) *assignment operator*
virtual void	**syncBeamList** () *Update myBeamPubs so that it has a publisher for all beamSRs in my systemSR, but no others*
virtual DtEmitterBeamPublisher*	**publisherForBsr** (DtEmitterBeamRepository* bsr) *Returns the publisher responsible for publishing the indicated BSR, NULL if there is none*
virtual int	**numBeams** () const *Return number of beam publishers*

Inherited from DtObjectPublisher:

Public Methods

virtual void **setHlaObject**(const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)

virtual void **setHlaObject**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)

virtual void **detachHlaObject**()

virtual DtStateRepository* **stateRep**()

virtual DtStateRepository* **sr**()

virtual DtHlaStateDecoder* **decoder**()

virtual DtHlaStateEncoder* **encoder**()

virtual DtHlaObject* **hlaObject**()

virtual DtExerciseConn* **exerciseConn**()

virtual const DtObjectId& **id**() const

virtual const DtObjectId& **objectId**() const

virtual const DtObjectId& **localId**() const

- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- void **init**(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- void **init**(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- inline DtExerciseConn* **exerciseConn**()
- inline DtHlaObject* **hlaObject**()
- inline const DtObjectId& **localId**() const
- inline DtStateRepository* **sr**()
- inline DtStateRepository* **stateRep**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**
- DtHlaStateEncoder* **myEncoder**
- DtHlaStateDecoder* **myDecoder**
- DtObjectId **myObjectId**
- DtString **myName**

Protected Methods

- DtObjectPublisher& **operator=** (const DtObjectPublisher& orig)

Documentation

class DtEmitterSystemPublisher:

Instances of DtEmitterSystemPublisher are used for local or source simulations of emitter systems, (i.e. these entities are not dependent on data being produced elsewhere on the network for their existence). These emitter systems send data to the network.

- **typedef DtEmitterSystemRepository* (*DtStateRepCreator)()**

  Function prototype for functions that create DtEmitterSystemRepositories

- **DtEmitterSystemPublisher(RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL)**

  constructor

- **DtEmitterSystemPublisher(DtExerciseConn* conn, const char* name = NULL)**

  constructor

- **virtual ~DtEmitterSystemPublisher()**

  destructor

- **virtual DtEmitterSystemRepository* emitterSystemRep()**

  Returns a pointer to the DtEmitterSystemRepository - through which you can set or inspect current state information

- **virtual DtEmitterSystemRepository* esr()**

  Synonym for emitterSystemRep()

- **virtual const char* classToUse() const**

  Returns the name of the FOM object class to use. Default implementation is to ask the FOM Mapper.

- **virtual void tick()**

  syncBeamList and update beam objects as well as main system object.

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**

  Give us a function to be used for creating SRs

- **DtEmitterSystemPublisher(const DtEmitterSystemPublisher& orig)**

  copy constructor

- **DtEmitterSystemPublisher& operator= (const DtEmitterSystemPublisher& orig)**

  assignment operator

- **virtual void syncBeamList()**

  Update myBeamPubs so that it has a publisher for all beamSRs in my systemSR, but no others. Called by tick() so that all beamPubs can be ticked after systemSR has been updated.

- **virtual DtEmitterBeamPublisher* publisherForBsr(DtEmitterBeamRepository* bsr)**

  Returns the publisher responsible for publishing the indicated BSR, NULL if there is none

- **virtual int numBeams() const**

  Return number of beam publishers

- **DtList myBeamPubs**

  List of DtEmitterBeamPubs

- **static DtStateRepCreator theStateRepCreator**

- **inline DtEmitterSystemRepository* emitterSystemRep()**

- **inline DtEmitterSystemRepository* esr()**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hemitrbeampub.h:

class DtEmitterBeamPublisher: public DtObjectPublisher

class DtEmitterBeamPublisher: Instances of DtEmitterBeamPublisher are used for local or source simulations of emitter beams, (i

---

Inheritance:

DtEmitterBeamPublisher - DtObjectPublisher

Public Methods

typedef DtEmitterBeamRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtBeamRepositories*
	DtEmitterBeamPublisher (RTI::ObjectClassHandle h, DtExerciseConn* conn, DtEmitterBeamRepository* bsr = NULL, const char* name = NULL) *constructor*
	DtEmitterBeamPublisher (DtExerciseConn* conn, DtEmitterBeamRepository* bsr = NULL, const char* name = NULL) *constructor*
virtual	**~DtEmitterBeamPublisher** () *destructor*
virtual DtEmitterBeamRepository*	**emitrBeamRep** () *Returns a pointer to the DtEmitterBeamRepository - through which you can set or inspect current state information*
virtual DtEmitterBeamRepository*	**bsr** () *Synonym for emitrBeamRep()*
virtual const char*	**classToUse** (DtEmitterBeamRepository* bsr) const *Returns the name of the FOM object class handle to use*
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtEmitterBeamRepository*	**bsr** ()
inline DtEmitterBeamRepository*	**emitrBeamRep** ()

Protected Fields

- DtBoolean **myOwnBsr**
- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtEmitterBeamPublisher** (const DtEmitterBeamPublisher& orig)
  copy constructor
- DtEmitterBeamPublisher& **operator=** (const DtEmitterBeamPublisher& orig)
  assignment operator

Inherited from DtObjectPublisher:

Public Methods

- virtual void **setHlaObject**(const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **setHlaObject**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **detachHlaObject**()
- virtual DtStateRepository* **stateRep**()
- virtual DtStateRepository* **sr**()
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaStateEncoder* **encoder**()
- virtual DtHlaObject* **hlaObject**()
- virtual DtExerciseConn* **exerciseConn**()
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **tick**()

- void **init**(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- void **init**(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- inline DtExerciseConn* **exerciseConn**()
- inline DtHlaObject* **hlaObject**()
- inline const DtObjectId& **localId**() const
- inline DtStateRepository* **sr**()
- inline DtStateRepository* **stateRep**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**
- DtHlaStateEncoder* **myEncoder**
- DtHlaStateDecoder* **myDecoder**
- DtObjectId **myObjectId**
- DtString **myName**

Protected Methods

- DtObjectPublisher& **operator=** (const DtObjectPublisher& orig)

Documentation

class DtEmitterBeamPublisher:

Instances of DtEmitterBeamPublisher are used for local or source simulations of emitter beams, (i.e. these entities are not dependent on data being produced elsewhere on the network for their existence). These emitter beams send data to the network.

- **`typedef DtEmitterBeamRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtBeamRepositories

- **`DtEmitterBeamPublisher( RTI::ObjectClassHandle h, DtExerciseConn* conn, DtEmitterBeamRepository* bsr = NULL, const char* name = NULL)`**

  constructor. NULL bsr means create your own.

- **DtEmitterBeamPublisher(DtExerciseConn* conn, DtEmitterBeamRepository* bsr = NULL, const char* name = NULL)**

  constructor. NULL bsr means create your own.

- **virtual ~DtEmitterBeamPublisher()**

  destructor

- **virtual DtEmitterBeamRepository* emitrBeamRep()**

  Returns a pointer to the DtEmitterBeamRepository - through which you can set or inspect current state information

- **virtual DtEmitterBeamRepository* bsr()**

  Synonym for emitrBeamRep()

- **virtual const char* classToUse(DtEmitterBeamRepository* bsr) const**

  Returns the name of the FOM object class handle to use

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**

  Give us a function to be used for creating SRs

- **DtEmitterBeamPublisher(const DtEmitterBeamPublisher& orig)**

  copy constructor

- **DtEmitterBeamPublisher& operator= (const DtEmitterBeamPublisher& orig)**

  assignment operator

- **DtBoolean myOwnBsr**

- **static DtStateRepCreator theStateRepCreator**

- **inline DtEmitterBeamRepository* bsr()**

- **inline DtEmitterBeamRepository* emitrBeamRep()**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hdesignatrpub.h:

class DtDesignatorPublisher: public DtObjectPublisher

class DtDesignatorPublisher: Instances of DtDesignatorPublisher are used for local or source simulations of designators, (i

---

Inheritance:

DtDesignatorPublisher - DtObjectPublisher

Public Methods

- typedef DtDesignatorRepository* **(*DtStateRepCreator)** ()
  Function prototype for functions that create DtDesignatorRepositories
- **DtDesignatorPublisher** (RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL)
  constructor
- **DtDesignatorPublisher** (DtExerciseConn* conn, const char* name = NULL)
  constructor
- virtual **~DtDesignatorPublisher** ()
  destructor
- virtual DtDesignatorRepository* **designatorStateRep** ()
  Returns a pointer to the DtDesignatorRepository - through which you can set or inspect current state information
- virtual DtDesignatorRepository* **dsr** ()
  Synonym for designatorStateRep()
- virtual const char* **classToUse** () const
  Returns the name of the FOM object class to use
- static DtStateRepCreator **setStateRepCreator** (DtStateRepCreator creator)
  Give us a function to be used for creating SRs
- inline DtDesignatorRepository* **designatorStateRep** ()
- inline DtDesignatorRepository* **dsr** ()

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

	DtDesignatorPublisher (const DtDesignatorPublisher& orig) *copy constructor*
DtDesignatorPublisher&	**operator=** (const DtDesignatorPublisher& orig) *copy constructor*

Inherited from DtObjectPublisher:

Public Methods

- virtual void **setHlaObject**(const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **setHlaObject**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **detachHlaObject**()
- virtual DtStateRepository* **stateRep**()
- virtual DtStateRepository* **sr**()
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaStateEncoder* **encoder**()
- virtual DtHlaObject* **hlaObject**()
- virtual DtExerciseConn* **exerciseConn**()
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **tick**()
- void **init**(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- void **init**(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- inline DtExerciseConn* **exerciseConn**()

- inline DtHlaObject* **hlaObject**()
- inline const DtObjectId& **localId**() const
- inline DtStateRepository* **sr**()
- inline DtStateRepository* **stateRep**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**
- DtHlaStateEncoder* **myEncoder**
- DtHlaStateDecoder* **myDecoder**
- DtObjectId **myObjectId**
- DtString **myName**

Protected Methods

- DtObjectPublisher& **operator=** (const DtObjectPublisher& orig)

Documentation

class DtDesignatorPublisher:

Instances of DtDesignatorPublisher are used for local or source simulations of designators, (i.e. these entities are not dependent on data being produced elsewhere on the network for their existence). These designators send data to the network.

- **`typedef DtDesignatorRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtDesignatorRepositories

- **`DtDesignatorPublisher(RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL)`**

  constructor

- **`DtDesignatorPublisher(DtExerciseConn* conn, const char* name = NULL)`**

  constructor

- **`virtual ~DtDesignatorPublisher()`**

  destructor

- **virtual DtDesignatorRepository* designatorStateRep()**

  Returns a pointer to the DtDesignatorRepository - through which you can set or inspect current state information

- **virtual DtDesignatorRepository* dsr()**

  Synonym for designatorStateRep()

- **virtual const char* classToUse() const**

  Returns the name of the FOM object class to use. Default implementation is to ask the FOM Mapper.

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**

  Give us a function to be used for creating SRs

- **DtDesignatorPublisher(const DtDesignatorPublisher& orig)**

  copy constructor

- **DtDesignatorPublisher& operator= (const DtDesignatorPublisher& orig)**

  copy constructor

- **static DtStateRepCreator theStateRepCreator**

- **inline DtDesignatorRepository* designatorStateRep()**

- **inline DtDesignatorRepository* dsr()**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file haggpub.h:

class DtAggregatePublisher: public DtObjectPublisher

class DtAggregatePublisher: Instances of DtAggregatePublisher are used for local or source simulations of aggregate entities, (i

---

Inheritance:

DtAggregatePublisher - DtObjectPublisher

Public Methods

typedef DtAggregateStateRepository*	**(*DtStateRepCreator)** () *Function prototype for functions that create DtAggregateStateRepositories*
	DtAggregatePublisher (RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL) *constructors*
	DtAggregatePublisher (const DtEntityType& type, DtExerciseConn* conn, const char* name = NULL)
	DtAggregatePublisher (const DtEntityType& type, DtExerciseConn *conn, DtDeadReckonTypes algorithm, DtForceType forceId, const char* name = NULL)
virtual	**~DtAggregatePublisher** () *destructor*
virtual DtAggregateStateRepository*	**aggregateStateRep** () *Returns a pointer to the AggregateStateRepository - through which you can set or inspect current state information*
virtual DtAggregateStateRepository*	**asr** () *Synonym for aggregateStateRep()*
static DtStateRepCreator	**setStateRepCreator** (DtStateRepCreator creator) *Give us a function to be used for creating SRs*
inline DtAggregateStateRepository*	**aggregateStateRep** ()
inline DtAggregateStateRepository*	**asr** ()

Protected Fields

- static DtStateRepCreator **theStateRepCreator**

Protected Methods

- **DtAggregatePublisher** (const DtAggregatePublisher& orig)
  copy constructor - not implemented
- DtAggregatePublisher& **operator=** (const DtAggregatePublisher& orig)
  assignment operator - not implemented
- const char* **objectClassNameFromType** (const DtEntityType &type)
  Non-virtual since called from constructor

Inherited from DtObjectPublisher:

Public Methods

- virtual void **setHlaObject**(const char* className, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **setHlaObject**(RTI::ObjectClassHandle hand, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL, const char* objectName = NULL)
- virtual void **detachHlaObject**()
- virtual DtStateRepository* **stateRep**()
- virtual DtStateRepository* **sr**()
- virtual DtHlaStateDecoder* **decoder**()
- virtual DtHlaStateEncoder* **encoder**()
- virtual DtHlaObject* **hlaObject**()
- virtual DtExerciseConn* **exerciseConn**()
- virtual const DtObjectId& **id**() const
- virtual const DtObjectId& **objectId**() const
- virtual const DtObjectId& **localId**() const
- virtual const DtString& **name**() const
- virtual const DtGlobalObjectDesignator& **globalId**() const
- virtual void **forceUpdate**()
- virtual void **forceUpdate**(const RTI::AttributeHandleSet& set)
- virtual void **tick**()

- void **init**(RTI::ObjectClassHandle hand, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- void **init**(const char* className, DtStateRepository* stateRep, DtStateRepository* asSeenByRemote, DtHlaStateEncoder* encoder = NULL, DtHlaStateDecoder* decoder = NULL)
- inline DtExerciseConn* **exerciseConn**()
- inline DtHlaObject* **hlaObject**()
- inline const DtObjectId& **localId**() const
- inline DtStateRepository* **sr**()
- inline DtStateRepository* **stateRep**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtHlaObjectWithStateRep* **myHlaObj**
- DtStateRepository* **myStateRep**
- DtStateRepository* **myAsSeenByRemote**
- DtHlaStateEncoder* **myEncoder**
- DtHlaStateDecoder* **myDecoder**
- DtObjectId **myObjectId**
- DtString **myName**

Protected Methods

- DtObjectPublisher& **operator=** (const DtObjectPublisher& orig)

Documentation

class DtAggregatePublisher:

Instances of DtAggregatePublisher are used for local or source simulations of aggregate entities, (i.e. these entities are not dependent on data being produced elsewhere on the network for their existence). These entities send data to the network.

- **`typedef DtAggregateStateRepository* (*DtStateRepCreator)()`**

  Function prototype for functions that create DtAggregateStateRepositories

- **`DtAggregatePublisher(RTI::ObjectClassHandle h, DtExerciseConn* conn, const char* name = NULL)`**

constructors

- **DtAggregatePublisher(const DtEntityType& type, DtExerciseConn* conn, const char* name = NULL)**

- **DtAggregatePublisher(const DtEntityType& type, DtExerciseConn *conn, DtDeadReckonTypes algorithm, DtForceType forceId, const char* name = NULL)**

- **virtual ~DtAggregatePublisher()**

  destructor

- **virtual DtAggregateStateRepository* aggregateStateRep()**

  Returns a pointer to the AggregateStateRepository - through which you can set or inspect current state information

- **virtual DtAggregateStateRepository* asr()**

  Synonym for aggregateStateRep()

- **static DtStateRepCreator setStateRepCreator(DtStateRepCreator creator)**

  Give us a function to be used for creating SRs

- **DtAggregatePublisher(const DtAggregatePublisher& orig)**

  copy constructor - not implemented

- **DtAggregatePublisher& operator=(const DtAggregatePublisher& orig)**

  assignment operator - not implemented

- **const char* objectClassNameFromType(const DtEntityType &type)**

  Non-virtual since called from constructor. If you don't like the default mapping between entity type and object class, just pass the object class you want to the constructor.

- **static DtStateRepCreator theStateRepCreator**

- **inline DtAggregateStateRepository* aggregateStateRep()**

- **inline DtAggregateStateRepository* asr()**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file objrefcbinfo.h:

class DtObjectReflectAttributeValuesCbInfo

class DtObjectReflectAttributeValuesCbInfo: Instances of DtReflectAttributeValuesCbInfo are used for providing a callback when attributes change for an object in the federation

---

Public Methods

	DtObjectReflectAttributeValuesCbInfo (DtObjectReflectAttributeValuesCb cb, void* usr) *constructor*
virtual	**~DtObjectReflectAttributeValuesCbInfo** () *destructor*
	DtObjectReflectAttributeValuesCbInfo (const DtObjectReflectAttributeValuesCbInfo& orig) *copy constructor*
DtObjectReflectAttributeValuesCbInfo&	**operator=** (const DtObjectReflectAttributeValuesCbInfo& orig) *assignment operator*
virtual int	**match** (const DtObjectReflectAttributeValuesCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtObjectReflectAttributeValuesCbInfo& other) const *Check whether one callback info is equal to another*
virtual const DtObjectReflectAttributeValuesCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtObjectReflectAttributeValuesCb&	**cbFunction** () const
inline void*	**userData** () const

Protected Fields

DtObjectReflectAttributeValuesCb	**myFunc**
void*	**myUsr**

Documentation

class DtObjectReflectAttributeValuesCbInfo: Instances of DtReflectAttributeValuesCbInfo are used for providing a callback when attributes change for an object in the federation

- **`DtObjectReflectAttributeValuesCbInfo( DtObjectReflectAttributeValuesCb cb, void* usr)`**

constructor

- **virtual ~DtObjectReflectAttributeValuesCbInfo()**

  destructor

- **DtObjectReflectAttributeValuesCbInfo(const DtObjectReflectAttributeValuesCbInfo& orig)**

  copy constructor

- **DtObjectReflectAttributeValuesCbInfo& operator=(const DtObjectReflectAttributeValuesCbInfo& orig)**

  assignment operator

- **virtual int match(const DtObjectReflectAttributeValuesCbInfo* const other) const**

  Check whether one callback info is equal to another

- **virtual int operator==(const DtObjectReflectAttributeValuesCbInfo& other) const**

  Check whether one callback info is equal to another

- **virtual const DtObjectReflectAttributeValuesCb& cbFunction() const**

  Get the callback function

- **virtual void* userData() const**

  Get the user data

- **DtObjectReflectAttributeValuesCb myFunc**
- **void* myUsr**
- **inline const DtObjectReflectAttributeValuesCb& cbFunction() const**
- **inline void* userData() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file recintrcbinfo.h:

class DtReceiveInteractionCbInfo

Public Methods

	DtReceiveInteractionCbInfo (RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr) *constructor*
virtual	**~DtReceiveInteractionCbInfo** () *destructor*
	DtReceiveInteractionCbInfo (const DtReceiveInteractionCbInfo& orig) *copy constructor*
DtReceiveInteractionCbInfo&	**operator=** (const DtReceiveInteractionCbInfo& orig) *assignment operator*
virtual int	**match** (const DtReceiveInteractionCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtReceiveInteractionCbInfo& other) const *Check whether one callback info is equal to another*
virtual const RTI::InteractionClassHandle&	**interactionClass** () const *Get the interaction class handle*
virtual const DtReceiveInteractionCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtReceiveInteractionCb&	**cbFunction** () const
inline const RTI::InteractionClassHandle&	**interactionClass** () const
inline void*	**userData** () const

Protected Fields

RTI::InteractionClassHandle	**myClass**
DtReceiveInteractionCb	**myFunc**
void*	**myUsr**

Documentation

`DtReceiveInteractionCbInfo(RTI::InteractionClassHandle h, DtReceiveInteractionCb cb, void* usr)`

constructor

- **`virtual ~DtReceiveInteractionCbInfo()`**
  destructor
- **`DtReceiveInteractionCbInfo( const DtReceiveInteractionCbInfo& orig)`**
  copy constructor
- **`DtReceiveInteractionCbInfo& operator=( const DtReceiveInteractionCbInfo& orig)`**
  assignment operator
- **`virtual int match(const DtReceiveInteractionCbInfo* const other) const`**
  Check whether one callback info is equal to another
- **`virtual int operator==(const DtReceiveInteractionCbInfo& other) const`**
  Check whether one callback info is equal to another
- **`virtual const RTI::InteractionClassHandle& interactionClass() const`**
  Get the interaction class handle
- **`virtual const DtReceiveInteractionCb& cbFunction() const`**
  Get the callback function
- **`virtual void* userData() const`**
  Get the user data
- **`RTI::InteractionClassHandle myClass`**
- **`DtReceiveInteractionCb myFunc`**
- **`void* myUsr`**
- **`inline const DtReceiveInteractionCb& cbFunction() const`**
- **`inline const RTI::InteractionClassHandle& interactionClass() const`**
- **`inline void* userData() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file refattrcbinfo.h:

class DtReflectAttributeValuesCbInfo

class DtReflectAttributeValuesCbInfo: Instances of DtReflectAttributeValuesCbInfo are used for providing a callback when attributes change for an object in federation

---

Public Methods

	DtReflectAttributeValuesCbInfo (RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr) *constructor*
virtual	**~DtReflectAttributeValuesCbInfo** () *destructor*
	DtReflectAttributeValuesCbInfo (const DtReflectAttributeValuesCbInfo& orig) *copy constructor*
DtReflectAttributeValuesCbInfo&	**operator=** (const DtReflectAttributeValuesCbInfo& orig) *assignment operator*
virtual int	**match** (const DtReflectAttributeValuesCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtReflectAttributeValuesCbInfo& other) const *Check whether one callback info is equal to another*
virtual const RTI::ObjectHandle&	**objectId** () const *Get the object id*
virtual const DtReflectAttributeValuesCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtReflectAttributeValuesCb&	**cbFunction** () const
inline const RTI::ObjectHandle&	**objectId** () const
inline void*	**userData** () const

Protected Fields

RTI::ObjectHandle	**myId**
DtReflectAttributeValuesCb	**myFunc**
void*	**myUsr**

Documentation

class DtReflectAttributeValuesCbInfo:

Instances of DtReflectAttributeValuesCbInfo are used for providing a callback when attributes change for an object in federation. This class is a helper class to DtReflectAttributeValuesCbList.

- **`DtReflectAttributeValuesCbInfo(RTI::ObjectHandle id, DtReflectAttributeValuesCb cb, void* usr)`**

  constructor

- **`virtual ~DtReflectAttributeValuesCbInfo()`**

  destructor

- **`DtReflectAttributeValuesCbInfo( const DtReflectAttributeValuesCbInfo& orig)`**

  copy constructor

- **`DtReflectAttributeValuesCbInfo& operator=( const DtReflectAttributeValuesCbInfo& orig)`**

  assignment operator

- **`virtual int match(const DtReflectAttributeValuesCbInfo* const other) const`**

  Check whether one callback info is equal to another

- **`virtual int operator==(const DtReflectAttributeValuesCbInfo& other) const`**

  Check whether one callback info is equal to another

- **`virtual const RTI::ObjectHandle& objectId() const`**

  Get the object id

- **`virtual const DtReflectAttributeValuesCb& cbFunction() const`**

  Get the callback function

- **`virtual void* userData() const`**

  Get the user data

- **`RTI::ObjectHandle myId`**
- **`DtReflectAttributeValuesCb myFunc`**
- **`void* myUsr`**
- **`inline const DtReflectAttributeValuesCb& cbFunction() const`**
- **`inline const RTI::ObjectHandle& objectId() const`**
- **`inline void* userData() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrefobjlist.h:

class DtReflectedObjectList

class DtReflectedObjectList: Instances of DtReflectedObjectList are list of instances of classes derived from DtReflectedObject

---

Inheritance:

DtReflectedObjectList

Public Methods

	DtReflectedObjectList (DtExerciseConn* exConn) *Constructor*
virtual	**~DtReflectedObjectList** () *destructor*
virtual DtReflectedObject*	**lookupObj** (RTI::ObjectHandle id) *Look up object by identifier*
virtual DtReflectedObject*	**lookupObj** (const DtGlobalObjectDesignator& objName) *Look up object by global ID (which is the object name in RTI1*
virtual DtReflectedObject*	**firstObj** () const *Get the first/last object in the list*
virtual DtReflectedObject*	**lastObj** () const
virtual int	**count** () const *Get number of objects*
virtual DtExerciseConn*	**exerciseConn** () const *Get my exerciseConn*
virtual void	**manageObjectClass** (RTI::ObjectClassHandle handle) *Register callbacks with exercise connection so that we get notified when an object of this class is discovered*
virtual void	**removeObject** (const DtObjectId& id) *Have the application act as though we had never discovered the object*
virtual void	**addObjectAdditionCallback** (DtReflectedObjectCallbackFunction cb, void* userData) *Object added callback registration/deregistration*
virtual void	**removeObjectAdditionCallback** (DtReflectedObjectCallbackFunction cb, void* userData)
virtual void	**addObjectRemovalCallback** (DtReflectedObjectCallbackFunction cb, void* userData) *Object removed callback registration/deregistration*
virtual void	**removeObjectRemovalCallback** (DtReflectedObjectCallbackFunction cb, void* userData)
virtual void	**print** () *Print the SRs of all objects*
virtual void	**setTimeoutProcessing** (DtBoolean onOff) *Set whether or not timeout processing occurs*

virtual void	**setTimeoutInterval** (DtTime time) *Set/Get the remote object timeout period*
virtual DtTime	**timeoutInterval** ()
static void	**setRequestClassUpdateFlag** (DtBoolean onOff) *Set/Get flag that determines whether we request updates from all objects of the object classes we subscribe to*
static DtBoolean	**requestClassUpdateFlag** ()
static void	**setRequestObjectUpdateFlag** (DtBoolean onOff) *Set/Get flag that determines whether we request updates from an object as we discover it*
static DtBoolean	**requestObjectUpdateFlag** ()
static void	**findDiscoveredObjectsCb** (void* usr) *Post-drain callback that calls findAlreadyDiscovedObjects()*
inline DtExerciseConn*	**exerciseConn** () const
inline DtReflectedObject*	**firstObj** () const
inline DtReflectedObject*	**lastObj** () const
inline void	**setTimeoutInterval** (DtTime time)
inline DtTime	**timeoutInterval** ()

Protected Fields

DtExerciseConn*	**myExConn**
DtObjectIdHashList	**myObjectList**
DtList	**myTimedOutObjectList**
DtList	**myHandleList**
DtIntrusiveList	**myObjectAddedCallbacks**
DtIntrusiveList	**myObjectRemovedCallbacks**
DtTime	**myTimeoutInterval**
DtBoolean	**myTimeoutProcessingFlag**
static DtBoolean	**theRequestClassUpdateFlag**
static DtBoolean	**theRequestObjectUpdateFlag**

Protected Methods

	DtReflectedObjectList (const DtReflectedObjectList& orig) *copy constructor*
DtReflectedObjectList&	**operator=** (const DtReflectedObjectList& orig) *assignment operator*
virtual void	**processDiscoverObject** (DtHlaObject* obj) *Process the discovery of a remote object on the network*

virtual void	**processRemoveObject** (DtHlaObject* obj) *Process the removal of a remote object from the network*
virtual void	**processTimedOutObjectReflect** (const DtStateMsg& msg, DtHlaObject* obj) *Handle the receipt of attributes for a remote object which was timed out*
virtual void	**doTimeouts** () *Remove any remote objects which have not been updated during the timeout period*
virtual DtReflectedObject*	**newReflectedObject** (DtHlaObject* obj) const *Create a remote object object for the HLA object*
virtual void	**removeAndDelete** (DtReflectedObject* obj, char* msg = NULL) *Remove the remote object from the list if found and delete it*
virtual void	**objectAdded** (DtReflectedObject* object) *Called from processDiscoverObject() after a remote object has been added to the list*
virtual DtBoolean	**shouldBeTimedOut** (DtReflectedObject* object) *Returns whether or not the object should be timed out*
virtual void	**timeOutObject** (DtReflectedObject* object) *Times out object by removing it from the remote object list and deleting it*
virtual void	**unTimeOutObject** (DtHlaObject* obj) *Manages callbacks and data used for reinstating a remote object which was timed out*
virtual void	**addObjectClass** (RTI::ObjectClassHandle handle) *Register interest in this FOM object class*
virtual void	**addObjectClassByName** (const char* className) *Register interest in this FOM object class by className*
virtual void	**init** (const DtList* handleList) *Register interest in a whole list of RTI::ObjectClassHandles all at once*
void	**emptyLists** () *Empty the object list and the timeout list*
virtual void	**processObjectAdditionCallbacks** (DtReflectedObject* object) *Process object addition/removal callbacks*
virtual void	**processObjectRemovalCallbacks** (DtReflectedObject* object)
virtual void	**findAlreadyDiscoveredObjects** () *When list is created, we call this function to go through the list of all discovered HLA objects to see if any should be managed by this list*
static void	**discoverObjectCb** (DtHlaObject* hlaObj, void* rel) *Cast the void* to a DtReflectedObjectList* and calls processDiscoverObject()*
static void	**removeObjectCb** (DtHlaObject* hlaObj, void* rel) *Cast the void* to a DtReflectedObjectList* and calls processRemoveObject()*
static void	**timedOutObjectReflectCb** (const DtStateMsg& msg, DtHlaObject* obj, void* usr) *Cast the void* to a DtReflectedObjectList* and calls processTimedOutObjectReflect()*
static void	**doTimeoutsPostDrainCb** (void* usr) *calls doTimeouts*

Documentation

class DtReflectedObjectList:

Instances of DtReflectedObjectList are list of instances of classes derived from DtReflectedObject. A DtReflectedObjectList manages the comings and goings of its reflected objects based on discoveries and removes received from the RTI.

- **`DtReflectedObjectList(DtExerciseConn* exConn)`**

Constructor

virtual ~DtReflectedObjectList()

destructor

virtual DtReflectedObject* lookupObj(RTI::ObjectHandle id)

Look up object by identifier

virtual DtReflectedObject* lookupObj(const DtGlobalObjectDesignator& objName)

Look up object by global ID (which is the object name in RTI1.3.3).

virtual DtReflectedObject* firstObj() const

Get the first/last object in the list

virtual DtReflectedObject* lastObj() const

virtual int count() const

Get number of objects

virtual DtExerciseConn* exerciseConn() const

Get my exerciseConn

virtual void manageObjectClass(RTI::ObjectClassHandle handle)

Register callbacks with exercise connection so that we get notified when an object of this class is discovered. Automatically called for all object classes passed to the constructor, or for the default set of objects classes if the ctor without a handleList is used. But if you want to handle your own subscription and update requests, pass an empty handleSet to the constructor, use DtObjectClassDesc's registerInterest and requestUpdates, then use this function to tell the ROL to manage the objects that get discovered.

virtual void removeObject(const DtObjectId& id)

Have the application act as though we had never discovered the object. We call the RTI's localDelete service, and delete the DtHlaObject, which causes any DtReflectedObjects associated with id (in this list and in any others) to be removed. The object might be re-discovered later.

virtual void addObjectAdditionCallback(DtReflectedObjectCallbackFunction cb, void* userData)

Object added callback registration/deregistration

virtual void removeObjectAdditionCallback(DtReflectedObjectCallbackFunction cb, void* userData)

virtual void addObjectRemovalCallback(DtReflectedObjectCallbackFunction cb, void* userData)

Object removed callback registration/deregistration

virtual void removeObjectRemovalCallback(DtReflectedObjectCallbackFunction cb, void* userData)

virtual void print()

Print the SRs of all objects

virtual void setTimeoutProcessing(DtBoolean onOff)

Set whether or not timeout processing occurs. Defaults to off.

virtual void setTimeoutInterval(DtTime time)

Set/Get the remote object timeout period

- **virtual DtTime timeoutInterval()**

- **static void setRequestClassUpdateFlag(DtBoolean onOff)**

  Set/Get flag that determines whether we request updates from all objects of the object classes we subscribe to. Default is true.

- **static DtBoolean requestClassUpdateFlag()**

- **static void setRequestObjectUpdateFlag(DtBoolean onOff)**

  Set/Get flag that determines whether we request updates from an object as we discover it. Default is true.

- **static DtBoolean requestObjectUpdateFlag()**

- **static void findDiscoveredObjectsCb(void* usr)**

  Post-drain callback that calls findAlreadyDiscovedObjects()

- **DtReflectedObjectList(const DtReflectedObjectList& orig)**

  copy constructor

- **DtReflectedObjectList& operator=(const DtReflectedObjectList& orig)**

  assignment operator

- **virtual void processDiscoverObject(DtHlaObject* obj)**

  Process the discovery of a remote object on the network. Adds the remote object to this list.

- **virtual void processRemoveObject(DtHlaObject* obj)**

  Process the removal of a remote object from the network. Removes the remote object from this list.

- **virtual void processTimedOutObjectReflect(const DtStateMsg& msg, DtHlaObject* obj)**

  Handle the receipt of attributes for a remote object which was timed out. Reinstates the remote object in this list.

- **virtual void doTimeouts()**

  Remove any remote objects which have not been updated during the timeout period. Gets called automatically by a postDrainCallback registered by the constructor. Users should not need to call this.

- **virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const**

  Create a remote object object for the HLA object. Caller is responsible for deletion.

- **virtual void removeAndDelete(DtReflectedObject* obj, char* msg = NULL)**

  Remove the remote object from the list if found and delete it. Prints a message on deletion if specified.

- **virtual void objectAdded(DtReflectedObject* object)**

  Called from processDiscoverObject() after a remote object has been added to the list. Defaults to do nothing. Subclasses can override this function to add functionality on addition.

- **virtual DtBoolean shouldBeTimedOut(DtReflectedObject* object)**

  Returns whether or not the object should be timed out. Assumes that timeout processing is enabled.

- **virtual void timeOutObject(DtReflectedObject* object)**

  Times out object by removing it from the remote object list and deleting it. Manages callbacks to reinstate remote object upon receiving attribute updates for the timed out object.

- **virtual void unTimeOutObject(DtHlaObject* obj)**

  Manages callbacks and data used for reinstating a remote object which was timed out

- **virtual void addObjectClass(RTI::ObjectClassHandle handle)**

  Register interest in this FOM object class. Typically called by derived class constructor.

- **virtual void addObjectClassByName(const char* className)**

  Register interest in this FOM object class by className. Typically called by derived class constructor.

- **virtual void init(const DtList* handleList)**

  Register interest in a whole list of RTI::ObjectClassHandles all at once. Uses addObjectClass above.

- **void emptyLists()**

  Empty the object list and the timeout list. Non-virtual since meant to be called from destructor.

- **virtual void processObjectAdditionCallbacks(DtReflectedObject* object)**

  Process object addition/removal callbacks

- **virtual void processObjectRemovalCallbacks(DtReflectedObject* object)**

- **virtual void findAlreadyDiscoveredObjects()**

  When list is created, we call this function to go through the list of all discovered HLA objects to see if any should be managed by this list. Otherwise, this list only hears about objects that are discovered by the app after the creation of this list.

- **static void discoverObjectCb(DtHlaObject* hlaObj, void* rel)**

  Cast the void* to a DtReflectedObjectList* and calls processDiscoverObject()

- **static void removeObjectCb(DtHlaObject* hlaObj, void* rel)**

  Cast the void* to a DtReflectedObjectList* and calls processRemoveObject()

- **static void timedOutObjectReflectCb(const DtStateMsg& msg, DtHlaObject* obj, void* usr)**

  Cast the void* to a DtReflectedObjectList* and calls processTimedOutObjectReflect()

- **static void doTimeoutsPostDrainCb(void* usr)**

  calls doTimeouts

- **DtExerciseConn* myExConn**

- **DtObjectIdHashList myObjectList**

- **DtList myTimedOutObjectList**

- **DtList myHandleList**

- **DtIntrusiveList myObjectAddedCallbacks**

- **DtIntrusiveList myObjectRemovedCallbacks**

- **DtTime myTimeoutInterval**

- **DtBoolean myTimeoutProcessingFlag**

- **static DtBoolean theRequestClassUpdateFlag**

- `static DtBoolean theRequestObjectUpdateFlag`
- `inline DtExerciseConn* exerciseConn() const`
- `inline DtReflectedObject* firstObj() const`
- `inline DtReflectedObject* lastObj() const`
- `inline void setTimeoutInterval(DtTime time)`
- `inline DtTime timeoutInterval()`

Direct child classes:

DtReflectedRadioTransmitterList
DtReflectedRadioReceiverList
DtReflectedGenericObjectList
DtReflectedEntityList
DtReflectedEmitterSystemList
DtReflectedEmitterBeamList
DtReflectedDesignatorList
DtReflectedAggregateList

Alphabetic index *Hierarchy of classes*

In file hrefradxmlist.h:

class DtReflectedRadioTransmitterList: public DtReflectedObjectList

class DtReflectedRadioTransmitterList: Instances of DtReflectedRadioTransmitterList manage the list of of remote RadioTransmitters in the exercise

---

Inheritance:

DtReflectedRadioTransmitterList - DtReflectedObjectList

Public Methods

	DtReflectedRadioTransmitterList (DtExerciseConn* exConn) *Constructor*
	DtReflectedRadioTransmitterList (DtExerciseConn* exConn, const DtList *handleList) *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*
virtual	**~DtReflectedRadioTransmitterList** () *destructor*
virtual DtReflectedRadioTransmitter*	**lookup** (RTI::ObjectHandle id) *Look up transmitter by identifier*
virtual DtReflectedRadioTransmitter*	**lookup** (const DtGlobalObjectDesignator& objName) *Look up transmitter by name*
virtual DtReflectedRadioTransmitter*	**first** () const *Get the first/last transmitter in the list*
virtual DtReflectedRadioTransmitter*	**last** () const
virtual void	**addRadioTransmitterAdditionCallback** (DtRadioTransmitterCallbackFunction cb, void* userData) *RadioTransmitter added callback registration/deregistration*
virtual void	**removeRadioTransmitterAdditionCallback** (DtRadioTransmitterCallbackFunction cb, void* userData)
virtual void	**addRadioTransmitterRemovalCallback** (DtRadioTransmitterCallbackFunction cb, void* userData) *RadioTransmitter removed callback registration/deregistration*
virtual void	**removeRadioTransmitterRemovalCallback** (DtRadioTransmitterCallbackFunction cb, void* userData)

Protected Methods

	DtReflectedRadioTransmitterList (const DtReflectedRadioTransmitterList& orig) *copy constructor*
DtReflectedRadioTransmitterList&	**operator=** (const DtReflectedRadioTransmitterList& orig) *assignment operator*
virtual DtReflectedObject*	**newReflectedObject** (DtHlaObject* obj) const *Override of virtual function*
virtual void	**objectAdded** (DtReflectedObject* obj) *Called from processDiscoverObject() after a remote transmitter has been added to the list*
virtual const DtList*	**dfltHandleList** () *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*

Inherited from DtReflectedObjectList:

Public Methods

- virtual DtReflectedObject* **lookupObj**(RTI::ObjectHandle id)
- virtual DtReflectedObject* **lookupObj**(const DtGlobalObjectDesignator& objName)
- virtual DtReflectedObject* **firstObj**() const
- virtual DtReflectedObject* **lastObj**() const
- virtual int **count**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual void **manageObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **removeObject**(const DtObjectId& id)
- virtual void **addObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **addObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **print**()
- virtual void **setTimeoutProcessing**(DtBoolean onOff)
- virtual void **setTimeoutInterval**(DtTime time)
- virtual DtTime **timeoutInterval**()
- static void **setRequestClassUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestClassUpdateFlag**()

- static void **setRequestObjectUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestObjectUpdateFlag**()
- static void **findDiscoveredObjectsCb**(void* usr)
- inline DtExerciseConn* **exerciseConn**() const
- inline DtReflectedObject* **firstObj**() const
- inline DtReflectedObject* **lastObj**() const
- inline void **setTimeoutInterval**(DtTime time)
- inline DtTime **timeoutInterval**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjectIdHashList **myObjectList**
- DtList **myTimedOutObjectList**
- DtList **myHandleList**
- DtIntrusiveList **myObjectAddedCallbacks**
- DtIntrusiveList **myObjectRemovedCallbacks**
- DtTime **myTimeoutInterval**
- DtBoolean **myTimeoutProcessingFlag**
- static DtBoolean **theRequestClassUpdateFlag**
- static DtBoolean **theRequestObjectUpdateFlag**

Protected Methods

- virtual void **processDiscoverObject**(DtHlaObject* obj)
- virtual void **processRemoveObject**(DtHlaObject* obj)
- virtual void **processTimedOutObjectReflect**(const DtStateMsg& msg, DtHlaObject* obj)
- virtual void **doTimeouts**()
- virtual void **removeAndDelete**(DtReflectedObject* obj, char* msg = NULL)
- virtual DtBoolean **shouldBeTimedOut**(DtReflectedObject* object)
- virtual void **timeOutObject**(DtReflectedObject* object)
- virtual void **unTimeOutObject**(DtHlaObject* obj)

- virtual void **addObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **addObjectClassByName**(const char* className)
- virtual void **init**(const DtList* handleList)
- void **emptyLists**()
- virtual void **processObjectAdditionCallbacks**(DtReflectedObject* object)
- virtual void **processObjectRemovalCallbacks**(DtReflectedObject* object)
- virtual void **findAlreadyDiscoveredObjects**()
- static void **discoverObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **removeObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **timedOutObjectReflectCb**(const DtStateMsg& msg, DtHlaObject* obj, void* usr)
- static void **doTimeoutsPostDrainCb**(void* usr)

Documentation

class DtReflectedRadioTransmitterList: Instances of DtReflectedRadioTransmitterList manage the list of of remote RadioTransmitters in the exercise

- **`DtReflectedRadioTransmitterList(DtExerciseConn* exConn)`**

  Constructor

- **`DtReflectedRadioTransmitterList( DtExerciseConn* exConn, const DtList *handleList)`**

  Constructor - only subscribe to the RTI::ObjectClassHandles in handleList

- **`virtual ~DtReflectedRadioTransmitterList()`**

  destructor

- **`virtual DtReflectedRadioTransmitter* lookup(RTI::ObjectHandle id)`**

  Look up transmitter by identifier

- **`virtual DtReflectedRadioTransmitter* lookup( const DtGlobalObjectDesignator& objName)`**

  Look up transmitter by name

- **`virtual DtReflectedRadioTransmitter* first() const`**

  Get the first/last transmitter in the list

- **`virtual DtReflectedRadioTransmitter* last() const`**

- **`virtual void addRadioTransmitterAdditionCallback( DtRadioTransmitterCallbackFunction cb, void* userData)`**

  RadioTransmitter added callback registration/deregistration

- **`virtual void removeRadioTransmitterAdditionCallback( DtRadioTransmitterCallbackFunction cb, void* userData)`**

virtual void addRadioTransmitterRemovalCallback(DtRadioTransmitterCallbackFunction cb, void* userData)

RadioTransmitter removed callback registration/deregistration

virtual void removeRadioTransmitterRemovalCallback(DtRadioTransmitterCallbackFunction cb, void* userData)

DtReflectedRadioTransmitterList(const DtReflectedRadioTransmitterList& orig)

copy constructor

DtReflectedRadioTransmitterList& operator=(const DtReflectedRadioTransmitterList& orig)

assignment operator

virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const

Override of virtual function. Here, we create a DtReflectedRadioTransmitter.

virtual void objectAdded(DtReflectedObject* obj)

Called from processDiscoverObject() after a remote transmitter has been added to the list. Default implementation prints a message. Subclasses can override this function to add functionality on addition. obj will be an instance of a class derived from DtReflectedObj.

virtual const DtList* dfltHandleList()

Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe

---

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrefradrvlist.h:

class DtReflectedRadioReceiverList: public DtReflectedObjectList

class DtReflectedRadioReceiverList: Instances of DtReflectedRadioReceiverList manage the list of of remote RadioReceivers in the exercise

---

Inheritance:

DtReflectedRadioReceiverList - DtReflectedObjectList

Public Methods

	DtReflectedRadioReceiverList (DtExerciseConn* exConn) *Constructor*
	DtReflectedRadioReceiverList (DtExerciseConn* exConn, const DtList *handleList) *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*
virtual	**~DtReflectedRadioReceiverList** () *destructor*
virtual DtReflectedRadioReceiver*	**lookup** (RTI::ObjectHandle id) *Look up receiver by identifier*
virtual DtReflectedRadioReceiver*	**lookup** (const DtGlobalObjectDesignator& objName) *Look up receiver by name*
virtual DtReflectedRadioReceiver*	**first** () const *Get the first/last receiver in the list*
virtual DtReflectedRadioReceiver*	**last** () const
virtual void	**addRadioReceiverAdditionCallback** (DtRadioReceiverCallbackFunction cb, void* userData) *RadioReceiver added callback registration/deregistration*
virtual void	**removeRadioReceiverAdditionCallback** (DtRadioReceiverCallbackFunction cb, void* userData)
virtual void	**addRadioReceiverRemovalCallback** (DtRadioReceiverCallbackFunction cb, void* userData) *RadioReceiver removed callback registration/deregistration*
virtual void	**removeRadioReceiverRemovalCallback** (DtRadioReceiverCallbackFunction cb, void* userData)

Protected Methods

	DtReflectedRadioReceiverList (const DtReflectedRadioReceiverList& orig) *copy constructor*
DtReflectedRadioReceiverList&	**operator=** (const DtReflectedRadioReceiverList& orig) *assignment operator*
virtual DtReflectedObject*	**newReflectedObject** (DtHlaObject* obj) const *Override of virtual function*
virtual void	**objectAdded** (DtReflectedObject* obj) *Called from processDiscoverObject() after a remote receiver has been added to the list*
virtual const DtList*	**dfltHandleList** () *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*

Inherited from DtReflectedObjectList:

Public Methods

- virtual DtReflectedObject* **lookupObj**(RTI::ObjectHandle id)
- virtual DtReflectedObject* **lookupObj**(const DtGlobalObjectDesignator& objName)
- virtual DtReflectedObject* **firstObj**() const
- virtual DtReflectedObject* **lastObj**() const
- virtual int **count**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual void **manageObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **removeObject**(const DtObjectId& id)
- virtual void **addObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **addObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **print**()
- virtual void **setTimeoutProcessing**(DtBoolean onOff)
- virtual void **setTimeoutInterval**(DtTime time)
- virtual DtTime **timeoutInterval**()
- static void **setRequestClassUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestClassUpdateFlag**()

- static void **setRequestObjectUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestObjectUpdateFlag**()
- static void **findDiscoveredObjectsCb**(void* usr)
- inline DtExerciseConn* **exerciseConn**() const
- inline DtReflectedObject* **firstObj**() const
- inline DtReflectedObject* **lastObj**() const
- inline void **setTimeoutInterval**(DtTime time)
- inline DtTime **timeoutInterval**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjectIdHashList **myObjectList**
- DtList **myTimedOutObjectList**
- DtList **myHandleList**
- DtIntrusiveList **myObjectAddedCallbacks**
- DtIntrusiveList **myObjectRemovedCallbacks**
- DtTime **myTimeoutInterval**
- DtBoolean **myTimeoutProcessingFlag**
- static DtBoolean **theRequestClassUpdateFlag**
- static DtBoolean **theRequestObjectUpdateFlag**

Protected Methods

- virtual void **processDiscoverObject**(DtHlaObject* obj)
- virtual void **processRemoveObject**(DtHlaObject* obj)
- virtual void **processTimedOutObjectReflect**(const DtStateMsg& msg, DtHlaObject* obj)
- virtual void **doTimeouts**()
- virtual void **removeAndDelete**(DtReflectedObject* obj, char* msg = NULL)
- virtual DtBoolean **shouldBeTimedOut**(DtReflectedObject* object)
- virtual void **timeOutObject**(DtReflectedObject* object)
- virtual void **unTimeOutObject**(DtHlaObject* obj)

- virtual void **addObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **addObjectClassByName**(const char* className)
- virtual void **init**(const DtList* handleList)
- void **emptyLists**()
- virtual void **processObjectAdditionCallbacks**(DtReflectedObject* object)
- virtual void **processObjectRemovalCallbacks**(DtReflectedObject* object)
- virtual void **findAlreadyDiscoveredObjects**()
- static void **discoverObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **removeObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **timedOutObjectReflectCb**(const DtStateMsg& msg, DtHlaObject* obj, void* usr)
- static void **doTimeoutsPostDrainCb**(void* usr)

Documentation

class DtReflectedRadioReceiverList: Instances of DtReflectedRadioReceiverList manage the list of of remote RadioReceivers in the exercise

- **`DtReflectedRadioReceiverList(DtExerciseConn* exConn)`**

  Constructor

- **`DtReflectedRadioReceiverList( DtExerciseConn* exConn, const DtList *handleList)`**

  Constructor - only subscribe to the RTI::ObjectClassHandles in handleList

- **`virtual ~DtReflectedRadioReceiverList()`**

  destructor

- **`virtual DtReflectedRadioReceiver* lookup(RTI::ObjectHandle id)`**

  Look up receiver by identifier

- **`virtual DtReflectedRadioReceiver* lookup( const DtGlobalObjectDesignator& objName)`**

  Look up receiver by name

- **`virtual DtReflectedRadioReceiver* first() const`**

  Get the first/last receiver in the list

- **`virtual DtReflectedRadioReceiver* last() const`**

- **`virtual void addRadioReceiverAdditionCallback( DtRadioReceiverCallbackFunction cb, void* userData)`**

  RadioReceiver added callback registration/deregistration

- **`virtual void removeRadioReceiverAdditionCallback( DtRadioReceiverCallbackFunction cb, void* userData)`**

- **virtual void addRadioReceiverRemovalCallback(DtRadioReceiverCallbackFunction cb, void* userData)**

  RadioReceiver removed callback registration/deregistration

- **virtual void removeRadioReceiverRemovalCallback(DtRadioReceiverCallbackFunction cb, void* userData)**

- **DtReflectedRadioReceiverList(const DtReflectedRadioReceiverList& orig)**

  copy constructor

- **DtReflectedRadioReceiverList& operator=(const DtReflectedRadioReceiverList& orig)**

  assignment operator

- **virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const**

  Override of virtual function. Here, we create a DtReflectedRadioReceiver.

- **virtual void objectAdded(DtReflectedObject* obj)**

  Called from processDiscoverObject() after a remote receiver has been added to the list. Default implementation prints a message. Subclasses can override this function to add functionality on addition. obj will be an instance of a class derived from DtReflectedObj.

- **virtual const DtList* dfltHandleList()**

  Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe

---

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file refgenlist.h:

class DtReflectedGenericObjectList: public DtReflectedObjectList

class DtReflectedGenericObjectList: An instance of this class manages the list of Generic objects that are simulated remotely, and reflected to us by the RTI

Inheritance:

DtReflectedGenericObjectList - DtReflectedObjectList

Public Methods

	DtReflectedGenericObjectList (DtExerciseConn* exConn, const char* className) *Constructor*
virtual	**~DtReflectedGenericObjectList** () *destructor*
virtual DtReflectedGenericObject*	**lookup** (RTI::ObjectHandle id) *Look up object by identifier*
virtual DtReflectedGenericObject*	**lookup** (const DtGlobalObjectDesignator& objName) *Look up object by name (global object designator)*
virtual DtReflectedGenericObject*	**first** () const *Get the first/last object in the list*
virtual DtReflectedGenericObject*	**last** () const
virtual void	**addGenericAdditionCallback** (DtGenericObjectCallbackFunction cb, void* userData) *Registration/deregistration of object addition callbacks*
virtual void	**removeGenericAdditionCallback** (DtGenericObjectCallbackFunction cb, void* userData)
virtual void	**addGenericRemovalCallback** (DtGenericObjectCallbackFunction cb, void* userData) *Registration/deregistration of object removal callbacks*
virtual void	**removeGenericRemovalCallback** (DtGenericObjectCallbackFunction cb, void* userData)

Public

The following public functions are not required by VR-Link, but they are strongly recommended

Functions that allow a user to register callbacks that will be called when "Generic" objects are added and removed from the list

Protected Methods

virtual DtReflectedObject* **newReflectedObject** (DtHlaObject* obj) const
Override of virtual function that the base class uses to create the appropriate kind of DtReflectedObject

---

Inherited from DtReflectedObjectList:

Public Methods

- virtual DtReflectedObject* **lookupObj**(RTI::ObjectHandle id)
- virtual DtReflectedObject* **lookupObj**(const DtGlobalObjectDesignator& objName)
- virtual DtReflectedObject* **firstObj**() const
- virtual DtReflectedObject* **lastObj**() const
- virtual int **count**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual void **manageObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **removeObject**(const DtObjectId& id)
- virtual void **addObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **addObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **print**()
- virtual void **setTimeoutProcessing**(DtBoolean onOff)
- virtual void **setTimeoutInterval**(DtTime time)
- virtual DtTime **timeoutInterval**()
- static void **setRequestClassUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestClassUpdateFlag**()
- static void **setRequestObjectUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestObjectUpdateFlag**()
- static void **findDiscoveredObjectsCb**(void* usr)

- inline DtExerciseConn* **exerciseConn**() const
- inline DtReflectedObject* **firstObj**() const
- inline DtReflectedObject* **lastObj**() const
- inline void **setTimeoutInterval**(DtTime time)
- inline DtTime **timeoutInterval**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjectIdHashList **myObjectList**
- DtList **myTimedOutObjectList**
- DtList **myHandleList**
- DtIntrusiveList **myObjectAddedCallbacks**
- DtIntrusiveList **myObjectRemovedCallbacks**
- DtTime **myTimeoutInterval**
- DtBoolean **myTimeoutProcessingFlag**
- static DtBoolean **theRequestClassUpdateFlag**
- static DtBoolean **theRequestObjectUpdateFlag**

Protected Methods

- DtReflectedObjectList& **operator=**(const DtReflectedObjectList& orig)
- virtual void **processDiscoverObject**(DtHlaObject* obj)
- virtual void **processRemoveObject**(DtHlaObject* obj)
- virtual void **processTimedOutObjectReflect**(const DtStateMsg& msg, DtHlaObject* obj)
- virtual void **doTimeouts**()
- virtual void **removeAndDelete**(DtReflectedObject* obj, char* msg = NULL)
- virtual void **objectAdded**(DtReflectedObject* object)
- virtual DtBoolean **shouldBeTimedOut**(DtReflectedObject* object)
- virtual void **timeOutObject**(DtReflectedObject* object)
- virtual void **unTimeOutObject**(DtHlaObject* obj)
- virtual void **addObjectClass**(RTI::ObjectClassHandle handle)

- virtual void **addObjectClassByName**(const char* className)
- virtual void **init**(const DtList* handleList)
- void **emptyLists**()
- virtual void **processObjectAdditionCallbacks**(DtReflectedObject* object)
- virtual void **processObjectRemovalCallbacks**(DtReflectedObject* object)
- virtual void **findAlreadyDiscoveredObjects**()
- static void **discoverObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **removeObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **timedOutObjectReflectCb**(const DtStateMsg& msg, DtHlaObject* obj, void* usr)
- static void **doTimeoutsPostDrainCb**(void* usr)

Documentation

class DtReflectedGenericObjectList:

An instance of this class manages the list of Generic objects that are simulated remotely, and reflected to us by the RTI. This class is derived from DtReflectedObjectList, which does most of the work. The derived class mostly exists to provide typesafety, and to let the base class know which FOM classes should be managed by this list.

- **`DtReflectedGenericObjectList(DtExerciseConn* exConn, const char* className)`**

  Constructor

- **`virtual ~DtReflectedGenericObjectList()`**

  destructor

- The following public functions are not required by VR-Link, but they are strongly recommended. The functions call down to base class versions, and cast to and from more generic types.

- **`virtual DtReflectedGenericObject* lookup(RTI::ObjectHandle id)`**

  Look up object by identifier

- **`virtual DtReflectedGenericObject* lookup(const DtGlobalObjectDesignator& objName)`**

  Look up object by name (global object designator)

- **`virtual DtReflectedGenericObject* first() const`**

  Get the first/last object in the list

- **`virtual DtReflectedGenericObject* last() const`**

- Functions that allow a user to register callbacks that will be called when "Generic" objects are added and removed from the list

- **`virtual void addGenericAdditionCallback( DtGenericObjectCallbackFunction cb, void*`**

userData)

Registration/deregistration of object addition callbacks

virtual void removeGenericAdditionCallback(DtGenericObjectCallbackFunction cb, void* userData)

virtual void addGenericRemovalCallback(DtGenericObjectCallbackFunction cb, void* userData)

Registration/deregistration of object removal callbacks

virtual void removeGenericRemovalCallback(DtGenericObjectCallbackFunction cb, void* userData)

virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const

Override of virtual function that the base class uses to create the appropriate kind of DtReflectedObject. Here, we create a DtReflectedGenericObject.

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hreflentlist.h:

class DtReflectedEntityList: public DtReflectedObjectList

class DtReflectedEntityList: Instances of DtReflectedEntityList are used to manage remote entities from the network

---

Inheritance:

DtReflectedEntityList - DtReflectedObjectList

Public Methods

	DtReflectedEntityList (DtExerciseConn* exConn, DtBoolean smoothCapable = DtFalse) *Constructor - obtains the set of object classes to subscribe to from the exConn's fomMapper*
	DtReflectedEntityList (DtExerciseConn* exConn, const DtList *handleList, DtBoolean smoothCapable = DtFalse) *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*
virtual	**~DtReflectedEntityList** () *destructor*
virtual DtReflectedEntity*	**lookup** (RTI::ObjectHandle id) *Look up entity by identifier*
virtual DtReflectedEntity*	**lookup** (const DtGlobalObjectDesignator& objName) *Look up entity by name*
virtual DtReflectedEntity*	**first** () const *Get the first/last entity in the list*
virtual DtReflectedEntity*	**last** () const
virtual void	**addEntityAdditionCallback** (DtEntityCallbackFunction cb, void* userData) *Entity added callback registration/deregistration*
virtual void	**removeEntityAdditionCallback** (DtEntityCallbackFunction cb, void* userData)
virtual void	**addEntityRemovalCallback** (DtEntityCallbackFunction cb, void* userData) *Entity removed callback registration/deregistration*
virtual void	**removeEntityRemovalCallback** (DtEntityCallbackFunction cb, void* userData)

Protected Fields

DtBoolean	**mySmoothCapable**

Protected Methods

	DtReflectedEntityList (const DtReflectedEntityList& orig) *copy constructor*
DtReflectedEntityList&	**operator=** (const DtReflectedEntityList& orig) *assignment operator*
virtual const DtList*	**dfltHandleList** () *Asks the FOM Mapper for the DtList of RTI::ObjectClassHandles to use*
virtual DtReflectedObject*	**newReflectedObject** (DtHlaObject* obj) const *Override of virtual function*
virtual DtReflectedEntity*	**newReflectedEntity** (DtHlaObject* obj) const *Create a remote entity object for the HLA object*
virtual void	**removeAndDelete** (DtReflectedObject* object, char* msg) *Virtual function override*
virtual void	**removeAndDelete** (DtReflectedEntity* entity, char* msg = NULL) *Default calls down to base class version, but can be overridden*
virtual void	**objectAdded** (DtReflectedObject* obj) *Called from processDiscoverObject() after a remote transmitter has been added to the list*
virtual void	**entityAdded** (DtReflectedEntity* ent) *Called by objectAdded (which is called from processDiscoverObject()) after a remote entity has been added to the list*

Inherited from DtReflectedObjectList:

Public Methods

- virtual DtReflectedObject* **lookupObj**(RTI::ObjectHandle id)
- virtual DtReflectedObject* **lookupObj**(const DtGlobalObjectDesignator& objName)
- virtual DtReflectedObject* **firstObj**() const
- virtual DtReflectedObject* **lastObj**() const
- virtual int **count**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual void **manageObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **removeObject**(const DtObjectId& id)
- virtual void **addObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **addObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **print**()
- virtual void **setTimeoutProcessing**(DtBoolean onOff)

- virtual void **setTimeoutInterval**(DtTime time)
- virtual DtTime **timeoutInterval**()
- static void **setRequestClassUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestClassUpdateFlag**()
- static void **setRequestObjectUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestObjectUpdateFlag**()
- static void **findDiscoveredObjectsCb**(void* usr)
- inline DtExerciseConn* **exerciseConn**() const
- inline DtReflectedObject* **firstObj**() const
- inline DtReflectedObject* **lastObj**() const
- inline void **setTimeoutInterval**(DtTime time)
- inline DtTime **timeoutInterval**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjectIdHashList **myObjectList**
- DtList **myTimedOutObjectList**
- DtList **myHandleList**
- DtIntrusiveList **myObjectAddedCallbacks**
- DtIntrusiveList **myObjectRemovedCallbacks**
- DtTime **myTimeoutInterval**
- DtBoolean **myTimeoutProcessingFlag**
- static DtBoolean **theRequestClassUpdateFlag**
- static DtBoolean **theRequestObjectUpdateFlag**

Protected Methods

- virtual void **processDiscoverObject**(DtHlaObject* obj)
- virtual void **processRemoveObject**(DtHlaObject* obj)
- virtual void **processTimedOutObjectReflect**(const DtStateMsg& msg, DtHlaObject* obj)
- virtual void **doTimeouts**()

- virtual DtBoolean **shouldBeTimedOut**(DtReflectedObject* object)
- virtual void **timeOutObject**(DtReflectedObject* object)
- virtual void **unTimeOutObject**(DtHlaObject* obj)
- virtual void **addObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **addObjectClassByName**(const char* className)
- virtual void **init**(const DtList* handleList)
- void **emptyLists**()
- virtual void **processObjectAdditionCallbacks**(DtReflectedObject* object)
- virtual void **processObjectRemovalCallbacks**(DtReflectedObject* object)
- virtual void **findAlreadyDiscoveredObjects**()
- static void **discoverObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **removeObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **timedOutObjectReflectCb**(const DtStateMsg& msg, DtHlaObject* obj, void* usr)
- static void **doTimeoutsPostDrainCb**(void* usr)

Documentation

class DtReflectedEntityList: Instances of DtReflectedEntityList are used to manage remote entities from the network

- **`DtReflectedEntityList(DtExerciseConn* exConn, DtBoolean smoothCapable = DtFalse)`**

  Constructor - obtains the set of object classes to subscribe to from the exConn's fomMapper

- **`DtReflectedEntityList(DtExerciseConn* exConn, const DtList *handleList, DtBoolean smoothCapable = DtFalse)`**

  Constructor - only subscribe to the RTI::ObjectClassHandles in handleList

- **`virtual ~DtReflectedEntityList()`**

  destructor

- **`virtual DtReflectedEntity* lookup(RTI::ObjectHandle id)`**

  Look up entity by identifier

- **`virtual DtReflectedEntity* lookup(const DtGlobalObjectDesignator& objName)`**

  Look up entity by name

- **`virtual DtReflectedEntity* first() const`**

  Get the first/last entity in the list

- **`virtual DtReflectedEntity* last() const`**

- **virtual void addEntityAdditionCallback(DtEntityCallbackFunction cb, void* userData)**
  Entity added callback registration/deregistration
- **virtual void removeEntityAdditionCallback(DtEntityCallbackFunction cb, void* userData)**
- **virtual void addEntityRemovalCallback(DtEntityCallbackFunction cb, void* userData)**
  Entity removed callback registration/deregistration
- **virtual void removeEntityRemovalCallback(DtEntityCallbackFunction cb, void* userData)**
- **DtReflectedEntityList(const DtReflectedEntityList& orig)**
  copy constructor
- **DtReflectedEntityList& operator=(const DtReflectedEntityList& orig)**
  assignment operator
- **virtual const DtList* dfltHandleList()**
  Asks the FOM Mapper for the DtList of RTI::ObjectClassHandles to use
- **virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const**
  Override of virtual function. Calls newReflectedEntity.
- **virtual DtReflectedEntity* newReflectedEntity(DtHlaObject* obj) const**
  Create a remote entity object for the HLA object. Caller is responsible for deletion.
- **virtual void removeAndDelete(DtReflectedObject* object, char* msg)**
  Virtual function override. Calls removeAndDelete(DtReflectedEntity*....)
- **virtual void removeAndDelete(DtReflectedEntity* entity, char* msg = NULL)**
  Default calls down to base class version, but can be overridden
- **virtual void objectAdded(DtReflectedObject* obj)**
  Called from processDiscoverObject() after a remote transmitter has been added to the list. Calls entityAdded.
- **virtual void entityAdded(DtReflectedEntity* ent)**
  Called by objectAdded (which is called from processDiscoverObject()) after a remote entity has been added to the list. Default implementation prints a message. Subclasses can override this function to add functionality on addition.
- **DtBoolean mySmoothCapable**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrefemsyslist.h:

class DtReflectedEmitterSystemList: public DtReflectedObjectList

class DtReflectedEmitterSystemList: Instances of DtReflectedEmitterSystemList manage the list of of remote EmitterSystems in the exercise

---

Inheritance:

DtReflectedEmitterSystemList - DtReflectedObjectList

Public Methods

	DtReflectedEmitterSystemList (DtExerciseConn* exConn, DtBoolean beamsUpdateSystems = DtTrue) *Constructor If beamsUpdateSystems == DtTrue, then a system's post-update callbacks are called whenever one of the beams in the system is updated*
	DtReflectedEmitterSystemList (DtExerciseConn* exConn, const DtList *handleList, DtBoolean beamsUpdateSystems = DtTrue) *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*
virtual	**~DtReflectedEmitterSystemList** () *destructor*
virtual DtReflectedEmitterSystem*	**lookup** (RTI::ObjectHandle id) *Look up emitter system by identifier*
virtual DtReflectedEmitterSystem*	**lookup** (const DtGlobalObjectDesignator& emittingEntity, int systemId) const *lookup an emitter from its object ID and system ID*
virtual DtReflectedEmitterSystem*	**lookup** (const DtGlobalObjectDesignator& objName) *Look up emitter system by name*
virtual DtReflectedEmitterSystem*	**first** () const *Get the first/last emitter system in the list*
virtual DtReflectedEmitterSystem*	**last** () const
virtual void	**addEmitterSystemAdditionCallback** (DtEmitterSystemCallbackFunction cb, void* userData) *EmitterSystem added callback registration/deregistration Called when an emitter system is added due to discovery of a system OR beam*
virtual void	**removeEmitterSystemAdditionCallback** (DtEmitterSystemCallbackFunction cb, void* userData)
virtual void	**addEmitterSystemRemovalCallback** (DtEmitterSystemCallbackFunction cb, void* userData) *EmitterSystem removed callback registration/deregistration Called when an emitter system is removed due to discovery of a system OR beam*

virtual void	**removeEmitterSystemRemovalCallback** (DtEmitterSystemCallbackFunction cb, void* userData)

Protected Fields

DtList	**myOrphanBeams**
DtReflectedEmitterBeamList	**myBeamList**
DtBoolean	**myBeamsUpdateSystems**

Protected Methods

	DtReflectedEmitterSystemList (const DtReflectedEmitterSystemList& orig) *copy constructor*
DtReflectedEmitterSystemList&	**operator=** (const DtReflectedEmitterSystemList& orig) *assignment operator*
virtual void	**processDiscoverObject** (DtHlaObject* obj) *Process the discovery of a remote emitter system on the network*
virtual void	**processRemoveObject** (DtHlaObject* obj) *Process the removal of a remote emitter system from the network*
virtual void	**processEmitterBeamAddition** (DtReflectedEmitterBeam* beam) *Called by similar callback functions registered with myBeamList on beam additions/removals*
virtual void	**processEmitterBeamUpdate** (DtReflectedEmitterBeam* beam)
virtual void	**processEmitterBeamRemoval** (DtReflectedEmitterBeam* beam)
virtual void	**resolveOrphans** () *Check if any orphanBeams have non-NULL emittingSystemId*
virtual DtReflectedObject*	**newReflectedObject** (DtHlaObject* obj) const *Override of virtual function*
virtual DtReflectedEmitterSystem*	**newReflectedEmitterSystem** (const DtGlobalObjectDesignator& des) const *Create a remote emitter system object for the HLA object*
virtual void	**objectAdded** (DtReflectedObject* obj) *Called from processDiscoverObject() after a remote transmitter has been added to the list*
virtual DtBoolean	**shouldBeTimedOut** (DtReflectedObject* object) *Returns whether or not the object should be timed out*
virtual const DtList*	**dfltHandleList** () *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*
static void	**emitterBeamAdditionCb** (DtReflectedEmitterBeam* beam, void* usr) *Callback functions registered with myBeamList on beam additions, updates, and removals*
static void	**emitterBeamUpdateCb** (DtReflectedEmitterBeam* beam, void* usr)
static void	**emitterBeamRemovalCb** (DtReflectedEmitterBeam* beam, void* usr)
static void	**resolveOrphansPostDrainCb** (void *usr) *Callback that calls resolveOrphans*

Inherited from DtReflectedObjectList:

Public Methods

- virtual DtReflectedObject* **lookupObj**(RTI::ObjectHandle id)
- virtual DtReflectedObject* **lookupObj**(const DtGlobalObjectDesignator& objName)
- virtual DtReflectedObject* **firstObj**() const
- virtual DtReflectedObject* **lastObj**() const
- virtual int **count**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual void **manageObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **removeObject**(const DtObjectId& id)
- virtual void **addObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **addObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **print**()
- virtual void **setTimeoutProcessing**(DtBoolean onOff)
- virtual void **setTimeoutInterval**(DtTime time)
- virtual DtTime **timeoutInterval**()
- static void **setRequestClassUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestClassUpdateFlag**()
- static void **setRequestObjectUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestObjectUpdateFlag**()
- static void **findDiscoveredObjectsCb**(void* usr)
- inline DtExerciseConn* **exerciseConn**() const
- inline DtReflectedObject* **firstObj**() const
- inline DtReflectedObject* **lastObj**() const
- inline void **setTimeoutInterval**(DtTime time)
- inline DtTime **timeoutInterval**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjectIdHashList **myObjectList**
- DtList **myTimedOutObjectList**
- DtList **myHandleList**
- DtIntrusiveList **myObjectAddedCallbacks**
- DtIntrusiveList **myObjectRemovedCallbacks**
- DtTime **myTimeoutInterval**
- DtBoolean **myTimeoutProcessingFlag**
- static DtBoolean **theRequestClassUpdateFlag**
- static DtBoolean **theRequestObjectUpdateFlag**

Protected Methods

- DtReflectedObjectList& **operator=**(const DtReflectedObjectList& orig)
- virtual void **processTimedOutObjectReflect**(const DtStateMsg& msg, DtHlaObject* obj)
- virtual void **doTimeouts**()
- virtual void **removeAndDelete**(DtReflectedObject* obj, char* msg = NULL)
- virtual void **timeOutObject**(DtReflectedObject* object)
- virtual void **unTimeOutObject**(DtHlaObject* obj)
- virtual void **addObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **addObjectClassByName**(const char* className)
- virtual void **init**(const DtList* handleList)
- void **emptyLists**()
- virtual void **processObjectAdditionCallbacks**(DtReflectedObject* object)
- virtual void **processObjectRemovalCallbacks**(DtReflectedObject* object)
- virtual void **findAlreadyDiscoveredObjects**()
- static void **discoverObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **removeObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **timedOutObjectReflectCb**(const DtStateMsg& msg, DtHlaObject* obj, void* usr)
- static void **doTimeoutsPostDrainCb**(void* usr)

Documentation

class DtReflectedEmitterSystemList: Instances of DtReflectedEmitterSystemList manage the list of of remote EmitterSystems in the exercise

- **DtReflectedEmitterSystemList(DtExerciseConn* exConn, DtBoolean beamsUpdateSystems = DtTrue)**

  Constructor If beamsUpdateSystems == DtTrue, then a system's post-update callbacks are called whenever one of the beams in the system is updated

- **DtReflectedEmitterSystemList(DtExerciseConn* exConn, const DtList *handleList, DtBoolean beamsUpdateSystems = DtTrue)**

  Constructor - only subscribe to the RTI::ObjectClassHandles in handleList

- **virtual ~DtReflectedEmitterSystemList()**

  destructor

- **virtual DtReflectedEmitterSystem* lookup(RTI::ObjectHandle id)**

  Look up emitter system by identifier

- **virtual DtReflectedEmitterSystem* lookup(const DtGlobalObjectDesignator& emittingEntity, int systemId) const**

  lookup an emitter from its object ID and system ID

- **virtual DtReflectedEmitterSystem* lookup(const DtGlobalObjectDesignator& objName)**

  Look up emitter system by name

- **virtual DtReflectedEmitterSystem* first() const**

  Get the first/last emitter system in the list

- **virtual DtReflectedEmitterSystem* last() const**

- **virtual void addEmitterSystemAdditionCallback(DtEmitterSystemCallbackFunction cb, void* userData)**

  EmitterSystem added callback registration/deregistration Called when an emitter system is added due to discovery of a system OR beam

- **virtual void removeEmitterSystemAdditionCallback(DtEmitterSystemCallbackFunction cb, void* userData)**

- **virtual void addEmitterSystemRemovalCallback(DtEmitterSystemCallbackFunction cb, void* userData)**

  EmitterSystem removed callback registration/deregistration Called when an emitter system is removed due to discovery of a system OR beam

- **virtual void removeEmitterSystemRemovalCallback(DtEmitterSystemCallbackFunction cb, void* userData)**

- **DtReflectedEmitterSystemList(const DtReflectedEmitterSystemList& orig)**

  copy constructor

- **DtReflectedEmitterSystemList& operator= (const DtReflectedEmitterSystemList& orig)**

  assignment operator

- **virtual void processDiscoverObject(DtHlaObject* obj)**

  Process the discovery of a remote emitter system on the network. Can't use base version, since we need to account for systems that are already in the list by virtue of discovered beams.

- **virtual void processRemoveObject(DtHlaObject* obj)**

  Process the removal of a remote emitter system from the network. Removes the remote emitter system from this list. Can't use base version, since we need to keep systems that still have live beams.

- **virtual void processEmitterBeamAddition(DtReflectedEmitterBeam* beam)**

  Called by similar callback functions registered with myBeamList on beam additions/removals. Addition function adds the beam to the set of orphan beams (since we don't know what system he's part of), the removal function removes the beam's beamSR from his system's systemSR. Update function calls the beam's system update callbacks if beamsUpdateSystems == DtTrue in the reflected system list constructor.

- **virtual void processEmitterBeamUpdate(DtReflectedEmitterBeam* beam)**

- **virtual void processEmitterBeamRemoval(DtReflectedEmitterBeam* beam)**

- **virtual void resolveOrphans()**

  Check if any orphanBeams have non-NULL emittingSystemId. If so, add them to their appropriate system (creating one if it doesn't exist), and delete them from orphan list.

- **virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const**

  Override of virtual function. Here, we create a DtReflectedEmitterSystem.

- **virtual DtReflectedEmitterSystem* newReflectedEmitterSystem(const DtGlobalObjectDesignator& des) const**

  Create a remote emitter system object for the HLA object. Caller is responsible for deletion.

- **virtual void objectAdded(DtReflectedObject* obj)**

  Called from processDiscoverObject() after a remote transmitter has been added to the list. Default implementation prints a message. Subclasses can override this function to add functionality on addition. obj will be an instance of a class derived from DtReflectedObj.

- **virtual DtBoolean shouldBeTimedOut(DtReflectedObject* object)**

  Returns whether or not the object should be timed out. Assumes that timeout processing is enabled. Can't use base version since we don't want to time out systems with live beams.

- **virtual const DtList* dfltHandleList()**

  Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe

- **static void emitterBeamAdditionCb(DtReflectedEmitterBeam* beam, void* usr)**

  Callback functions registered with myBeamList on beam additions, updates, and removals. Calls processEmitterBeamAddition/Update/Removal.

- **static void emitterBeamUpdateCb(DtReflectedEmitterBeam* beam, void* usr)**

- **static void emitterBeamRemovalCb(DtReflectedEmitterBeam* beam, void* usr)**

- **static void resolveOrphansPostDrainCb(void *usr)**

  Callback that calls resolveOrphans

- **DtList myOrphanBeams**

- **DtReflectedEmitterBeamList myBeamList**
- **DtBoolean myBeamsUpdateSystems**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrefbeamlist.h:

class DtReflectedEmitterBeamList: public DtReflectedObjectList

class DtReflectedEmitterBeamList: Instances of DtReflectedEmitterBeamList manage the list of of remote EmitterBeams in the exercise

---

Inheritance:

DtReflectedEmitterBeamList - DtReflectedObjectList

Public Methods

	DtReflectedEmitterBeamList (DtExerciseConn* exConn) *Constructor*
	DtReflectedEmitterBeamList (DtExerciseConn* exConn, const DtList *handleList) *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*
virtual	**~DtReflectedEmitterBeamList** () *destructor*
virtual DtReflectedEmitterBeam*	**lookup** (RTI::ObjectHandle id) *Look up beam by identifier*
virtual DtReflectedEmitterBeam*	**lookup** (const DtGlobalObjectDesignator& objName) *Look up beam by name*
virtual DtReflectedEmitterBeam*	**first** () const *Get the first/last beam in the list*
virtual DtReflectedEmitterBeam*	**last** () const
virtual void	**addEmitterBeamAdditionCallback** (DtEmitterBeamCallbackFunction cb, void* userData) *EmitterBeam added callback registration/deregistration*
virtual void	**removeEmitterBeamAdditionCallback** (DtEmitterBeamCallbackFunction cb, void* userData)
virtual void	**addEmitterBeamRemovalCallback** (DtEmitterBeamCallbackFunction cb, void* userData) *EmitterBeam removed callback registration/deregistration*
virtual void	**removeEmitterBeamRemovalCallback** (DtEmitterBeamCallbackFunction cb, void* userData)

Protected Methods

- **DtReflectedEmitterBeamList** (const DtReflectedEmitterBeamList& orig)
  copy constructor
- DtReflectedEmitterBeamList& **operator=** (const DtReflectedEmitterBeamList& orig)
  assignment operator
- virtual DtReflectedObject* **newReflectedObject** (DtHlaObject* obj) const
  Override of virtual function
- virtual void **objectAdded** (DtReflectedObject* obj)
  Called from processDiscoverObject() after a remote beam has been added to the list
- virtual const DtList* **dfltHandleList** ()
  Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe

---

Inherited from DtReflectedObjectList:

Public Methods

- virtual DtReflectedObject* **lookupObj**(RTI::ObjectHandle id)
- virtual DtReflectedObject* **lookupObj**(const DtGlobalObjectDesignator& objName)
- virtual DtReflectedObject* **firstObj**() const
- virtual DtReflectedObject* **lastObj**() const
- virtual int **count**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual void **manageObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **removeObject**(const DtObjectId& id)
- virtual void **addObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **addObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **print**()
- virtual void **setTimeoutProcessing**(DtBoolean onOff)
- virtual void **setTimeoutInterval**(DtTime time)
- virtual DtTime **timeoutInterval**()
- static void **setRequestClassUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestClassUpdateFlag**()

- static void **setRequestObjectUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestObjectUpdateFlag**()
- static void **findDiscoveredObjectsCb**(void* usr)
- inline DtExerciseConn* **exerciseConn**() const
- inline DtReflectedObject* **firstObj**() const
- inline DtReflectedObject* **lastObj**() const
- inline void **setTimeoutInterval**(DtTime time)
- inline DtTime **timeoutInterval**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjectIdHashList **myObjectList**
- DtList **myTimedOutObjectList**
- DtList **myHandleList**
- DtIntrusiveList **myObjectAddedCallbacks**
- DtIntrusiveList **myObjectRemovedCallbacks**
- DtTime **myTimeoutInterval**
- DtBoolean **myTimeoutProcessingFlag**
- static DtBoolean **theRequestClassUpdateFlag**
- static DtBoolean **theRequestObjectUpdateFlag**

Protected Methods

- virtual void **processDiscoverObject**(DtHlaObject* obj)
- virtual void **processRemoveObject**(DtHlaObject* obj)
- virtual void **processTimedOutObjectReflect**(const DtStateMsg& msg, DtHlaObject* obj)
- virtual void **doTimeouts**()
- virtual void **removeAndDelete**(DtReflectedObject* obj, char* msg = NULL)
- virtual DtBoolean **shouldBeTimedOut**(DtReflectedObject* object)
- virtual void **timeOutObject**(DtReflectedObject* object)
- virtual void **unTimeOutObject**(DtHlaObject* obj)

- virtual void **addObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **addObjectClassByName**(const char* className)
- virtual void **init**(const DtList* handleList)
- void **emptyLists**()
- virtual void **processObjectAdditionCallbacks**(DtReflectedObject* object)
- virtual void **processObjectRemovalCallbacks**(DtReflectedObject* object)
- virtual void **findAlreadyDiscoveredObjects**()
- static void **discoverObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **removeObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **timedOutObjectReflectCb**(const DtStateMsg& msg, DtHlaObject* obj, void* usr)
- static void **doTimeoutsPostDrainCb**(void* usr)

Documentation

class DtReflectedEmitterBeamList: Instances of DtReflectedEmitterBeamList manage the list of of remote EmitterBeams in the exercise

- **`DtReflectedEmitterBeamList(DtExerciseConn* exConn)`**

  Constructor

- **`DtReflectedEmitterBeamList( DtExerciseConn* exConn, const DtList *handleList)`**

  Constructor - only subscribe to the RTI::ObjectClassHandles in handleList

- **`virtual ~DtReflectedEmitterBeamList()`**

  destructor

- **`virtual DtReflectedEmitterBeam* lookup(RTI::ObjectHandle id)`**

  Look up beam by identifier

- **`virtual DtReflectedEmitterBeam* lookup( const DtGlobalObjectDesignator& objName)`**

  Look up beam by name

- **`virtual DtReflectedEmitterBeam* first() const`**

  Get the first/last beam in the list

- **`virtual DtReflectedEmitterBeam* last() const`**

- **`virtual void addEmitterBeamAdditionCallback( DtEmitterBeamCallbackFunction cb, void* userData)`**

  EmitterBeam added callback registration/deregistration

- **`virtual void removeEmitterBeamAdditionCallback( DtEmitterBeamCallbackFunction cb, void* userData)`**

virtual void addEmitterBeamRemovalCallback(DtEmitterBeamCallbackFunction cb, void* userData)

EmitterBeam removed callback registration/deregistration

virtual void removeEmitterBeamRemovalCallback(DtEmitterBeamCallbackFunction cb, void* userData)

DtReflectedEmitterBeamList(const DtReflectedEmitterBeamList& orig)

copy constructor

DtReflectedEmitterBeamList& operator=(const DtReflectedEmitterBeamList& orig)

assignment operator

virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const

Override of virtual function. Here, we create a DtReflectedEmitterBeam.

virtual void objectAdded(DtReflectedObject* obj)

Called from processDiscoverObject() after a remote beam has been added to the list. Default implementation prints a message. Subclasses can override this function to add functionality on addition. obj will be an instance of a class derived from DtReflectedObj.

virtual const DtList* dfltHandleList()

Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe

---

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hrefdesiglist.h:

class DtReflectedDesignatorList: public DtReflectedObjectList

class DtReflectedDesignatorList: Instances of DtReflectedDesignatorList manage the list of of remote Designators in the exercise

---

Inheritance:

DtReflectedDesignatorList - DtReflectedObjectList

Public Methods

	DtReflectedDesignatorList (DtExerciseConn* exConn) *Constructor - obtains the set of object classes to subscribe to from the exConn's fomMapper*
	DtReflectedDesignatorList (DtExerciseConn* exConn, const DtList *handleList) *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*
virtual	**~DtReflectedDesignatorList** () *destructor*
virtual DtReflectedDesignator*	**lookup** (RTI::ObjectHandle id) *Look up designator by identifier*
virtual DtReflectedDesignator*	**lookup** (const DtGlobalObjectDesignator& objName) *Look up designator by name*
virtual DtReflectedDesignator*	**first** () const *Get the first/last designator in the list*
virtual DtReflectedDesignator*	**last** () const
virtual void	**addDesignatorAdditionCallback** (DtDesignatorCallbackFunction cb, void* userData) *Designator added callback registration/deregistration*
virtual void	**removeDesignatorAdditionCallback** (DtDesignatorCallbackFunction cb, void* userData)
virtual void	**addDesignatorRemovalCallback** (DtDesignatorCallbackFunction cb, void* userData) *Designator removed callback registration/deregistration*
virtual void	**removeDesignatorRemovalCallback** (DtDesignatorCallbackFunction cb, void* userData)

Protected Methods

	DtReflectedDesignatorList (const DtReflectedDesignatorList& orig) *copy constructor*
DtReflectedDesignatorList&	**operator=** (const DtReflectedDesignatorList& orig) *assignment operator*
virtual DtReflectedObject*	**newReflectedObject** (DtHlaObject* obj) const *Override of virtual function*
virtual void	**objectAdded** (DtReflectedObject* obj) *Called from processDiscoverObject() after a remote designator has been added to the list*
virtual const DtList*	**dfltHandleList** () *Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe*

Inherited from DtReflectedObjectList:

Public Methods

- virtual DtReflectedObject* **lookupObj**(RTI::ObjectHandle id)
- virtual DtReflectedObject* **lookupObj**(const DtGlobalObjectDesignator& objName)
- virtual DtReflectedObject* **firstObj**() const
- virtual DtReflectedObject* **lastObj**() const
- virtual int **count**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual void **manageObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **removeObject**(const DtObjectId& id)
- virtual void **addObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **addObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **print**()
- virtual void **setTimeoutProcessing**(DtBoolean onOff)
- virtual void **setTimeoutInterval**(DtTime time)
- virtual DtTime **timeoutInterval**()
- static void **setRequestClassUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestClassUpdateFlag**()

- static void **setRequestObjectUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestObjectUpdateFlag**()
- static void **findDiscoveredObjectsCb**(void* usr)
- inline DtExerciseConn* **exerciseConn**() const
- inline DtReflectedObject* **firstObj**() const
- inline DtReflectedObject* **lastObj**() const
- inline void **setTimeoutInterval**(DtTime time)
- inline DtTime **timeoutInterval**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjectIdHashList **myObjectList**
- DtList **myTimedOutObjectList**
- DtList **myHandleList**
- DtIntrusiveList **myObjectAddedCallbacks**
- DtIntrusiveList **myObjectRemovedCallbacks**
- DtTime **myTimeoutInterval**
- DtBoolean **myTimeoutProcessingFlag**
- static DtBoolean **theRequestClassUpdateFlag**
- static DtBoolean **theRequestObjectUpdateFlag**

Protected Methods

- virtual void **processDiscoverObject**(DtHlaObject* obj)
- virtual void **processRemoveObject**(DtHlaObject* obj)
- virtual void **processTimedOutObjectReflect**(const DtStateMsg& msg, DtHlaObject* obj)
- virtual void **doTimeouts**()
- virtual void **removeAndDelete**(DtReflectedObject* obj, char* msg = NULL)
- virtual DtBoolean **shouldBeTimedOut**(DtReflectedObject* object)
- virtual void **timeOutObject**(DtReflectedObject* object)
- virtual void **unTimeOutObject**(DtHlaObject* obj)

- virtual void **addObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **addObjectClassByName**(const char* className)
- virtual void **init**(const DtList* handleList)
- void **emptyLists**()
- virtual void **processObjectAdditionCallbacks**(DtReflectedObject* object)
- virtual void **processObjectRemovalCallbacks**(DtReflectedObject* object)
- virtual void **findAlreadyDiscoveredObjects**()
- static void **discoverObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **removeObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **timedOutObjectReflectCb**(const DtStateMsg& msg, DtHlaObject* obj, void* usr)
- static void **doTimeoutsPostDrainCb**(void* usr)

Documentation

class DtReflectedDesignatorList: Instances of DtReflectedDesignatorList manage the list of of remote Designators in the exercise

- **`DtReflectedDesignatorList(DtExerciseConn* exConn)`**

  Constructor - obtains the set of object classes to subscribe to from the exConn's fomMapper

- **`DtReflectedDesignatorList( DtExerciseConn* exConn, const DtList *handleList)`**

  Constructor - only subscribe to the RTI::ObjectClassHandles in handleList

- **`virtual ~DtReflectedDesignatorList()`**

  destructor

- **`virtual DtReflectedDesignator* lookup(RTI::ObjectHandle id)`**

  Look up designator by identifier

- **`virtual DtReflectedDesignator* lookup( const DtGlobalObjectDesignator& objName)`**

  Look up designator by name

- **`virtual DtReflectedDesignator* first() const`**

  Get the first/last designator in the list

- **`virtual DtReflectedDesignator* last() const`**

- **`virtual void addDesignatorAdditionCallback( DtDesignatorCallbackFunction cb, void* userData)`**

  Designator added callback registration/deregistration

- **`virtual void removeDesignatorAdditionCallback( DtDesignatorCallbackFunction cb, void* userData)`**

- **virtual void addDesignatorRemovalCallback(DtDesignatorCallbackFunction cb, void* userData)**

  Designator removed callback registration/deregistration

- **virtual void removeDesignatorRemovalCallback(DtDesignatorCallbackFunction cb, void* userData)**

- **DtReflectedDesignatorList(const DtReflectedDesignatorList& orig)**

  copy constructor

- **DtReflectedDesignatorList& operator=(const DtReflectedDesignatorList& orig)**

  assignment operator

- **virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const**

  Override of virtual function. Here, we create a DtReflectedDesignator.

- **virtual void objectAdded(DtReflectedObject* obj)**

  Called from processDiscoverObject() after a remote designator has been added to the list. Default implementation prints a message. Subclasses can override this function to add functionality on addition. obj will be an instance of a class derived from DtReflectedObj.

- **virtual const DtList* dfltHandleList()**

  Returns a DtList of RTI::ObjectClassHandles that represent the list of object classes to which this list should subscribe

---

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file hreflagglist.h:

class DtReflectedAggregateList: public DtReflectedObjectList

class DtReflectedAggregateList: Instances of DtReflectedAggregateList are used to manage remote aggregates from the network

---

Inheritance:

DtReflectedAggregateList - DtReflectedObjectList

Public Methods

	DtReflectedAggregateList (DtExerciseConn* exConn, DtBoolean smoothCapable = DtFalse) *Constructor - obtains the set of object classes to subscribe to from the exConn's fomMapper*
	DtReflectedAggregateList (DtExerciseConn* exConn, const DtList *handleList, DtBoolean smoothCapable = DtFalse) *Constructor - only subscribe to the RTI::ObjectClassHandles in handleList*
virtual	**~DtReflectedAggregateList** () *destructor*
virtual DtReflectedAggregate*	**lookup** (RTI::ObjectHandle id) *Look up aggregate by identifier*
virtual DtReflectedAggregate*	**lookup** (const DtGlobalObjectDesignator& objName) *Look up aggregate by name*
virtual DtReflectedAggregate*	**first** () const *Get the first/last entity in the list*
virtual DtReflectedAggregate*	**last** () const
virtual void	**addAggregateAdditionCallback** (DtAggregateCallbackFunction cb, void* userData) *Aggregate added callback registration/deregistration*
virtual void	**removeAggregateAdditionCallback** (DtAggregateCallbackFunction cb, void* userData)
virtual void	**addAggregateRemovalCallback** (DtAggregateCallbackFunction cb, void* userData) *Aggregate removed callback registration/deregistration*
virtual void	**removeAggregateRemovalCallback** (DtAggregateCallbackFunction cb, void* userData)

Protected Fields

- DtBoolean **mySmoothCapable**

Protected Methods

	DtReflectedAggregateList (const DtReflectedAggregateList& orig) *copy constructor*
DtReflectedAggregateList&	**operator=** (const DtReflectedAggregateList& orig) *assignment operator*
virtual const DtList*	**dfltHandleList** () *Asks the FOM Mapper for the DtList of RTI::ObjectClassHandles to use*
virtual DtReflectedObject*	**newReflectedObject** (DtHlaObject* obj) const *Override of virtual function*
virtual DtReflectedAggregate*	**newReflectedAggregate** (DtHlaObject* obj) const *Create a remote aggregate object for the HLA object*
virtual void	**removeAndDelete** (DtReflectedObject* object, char* msg) *Virtual function override*
virtual void	**removeAndDelete** (DtReflectedAggregate* aggregate, char* msg = NULL) *Default calls down to base class version, but can be overridden*
virtual void	**objectAdded** (DtReflectedObject* obj) *Called from processDiscoverObject() after a remote transmitter has been added to the list*
virtual void	**aggregateAdded** (DtReflectedAggregate* ent) *Called by objectAdded (which is called from processDiscoverObject()) after a remote aggregate has been added to the list*

Inherited from DtReflectedObjectList:

Public Methods

- virtual DtReflectedObject* **lookupObj**(RTI::ObjectHandle id)
- virtual DtReflectedObject* **lookupObj**(const DtGlobalObjectDesignator& objName)
- virtual DtReflectedObject* **firstObj**() const
- virtual DtReflectedObject* **lastObj**() const
- virtual int **count**() const
- virtual DtExerciseConn* **exerciseConn**() const
- virtual void **manageObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **removeObject**(const DtObjectId& id)
- virtual void **addObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectAdditionCallback**(DtReflectedObjectCallbackFunction cb, void* userData)

- virtual void **addObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **removeObjectRemovalCallback**(DtReflectedObjectCallbackFunction cb, void* userData)
- virtual void **print**()
- virtual void **setTimeoutProcessing**(DtBoolean onOff)
- virtual void **setTimeoutInterval**(DtTime time)
- virtual DtTime **timeoutInterval**()
- static void **setRequestClassUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestClassUpdateFlag**()
- static void **setRequestObjectUpdateFlag**(DtBoolean onOff)
- static DtBoolean **requestObjectUpdateFlag**()
- static void **findDiscoveredObjectsCb**(void* usr)
- inline DtExerciseConn* **exerciseConn**() const
- inline DtReflectedObject* **firstObj**() const
- inline DtReflectedObject* **lastObj**() const
- inline void **setTimeoutInterval**(DtTime time)
- inline DtTime **timeoutInterval**()

Protected Fields

- DtExerciseConn* **myExConn**
- DtObjectIdHashList **myObjectList**
- DtList **myTimedOutObjectList**
- DtList **myHandleList**
- DtIntrusiveList **myObjectAddedCallbacks**
- DtIntrusiveList **myObjectRemovedCallbacks**
- DtTime **myTimeoutInterval**
- DtBoolean **myTimeoutProcessingFlag**
- static DtBoolean **theRequestClassUpdateFlag**
- static DtBoolean **theRequestObjectUpdateFlag**

Protected Methods

- virtual void **processDiscoverObject**(DtHlaObject* obj)
- virtual void **processRemoveObject**(DtHlaObject* obj)
- virtual void **processTimedOutObjectReflect**(const DtStateMsg& msg, DtHlaObject* obj)
- virtual void **doTimeouts**()
- virtual DtBoolean **shouldBeTimedOut**(DtReflectedObject* object)
- virtual void **timeOutObject**(DtReflectedObject* object)
- virtual void **unTimeOutObject**(DtHlaObject* obj)
- virtual void **addObjectClass**(RTI::ObjectClassHandle handle)
- virtual void **addObjectClassByName**(const char* className)
- virtual void **init**(const DtList* handleList)
- void **emptyLists**()
- virtual void **processObjectAdditionCallbacks**(DtReflectedObject* object)
- virtual void **processObjectRemovalCallbacks**(DtReflectedObject* object)
- virtual void **findAlreadyDiscoveredObjects**()
- static void **discoverObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **removeObjectCb**(DtHlaObject* hlaObj, void* rel)
- static void **timedOutObjectReflectCb**(const DtStateMsg& msg, DtHlaObject* obj, void* usr)
- static void **doTimeoutsPostDrainCb**(void* usr)

Documentation

class DtReflectedAggregateList: Instances of DtReflectedAggregateList are used to manage remote aggregates from the network

- **`DtReflectedAggregateList(DtExerciseConn* exConn, DtBoolean smoothCapable = DtFalse)`**

  Constructor - obtains the set of object classes to subscribe to from the exConn's fomMapper

- **`DtReflectedAggregateList(DtExerciseConn* exConn, const DtList *handleList, DtBoolean smoothCapable = DtFalse)`**

  Constructor - only subscribe to the RTI::ObjectClassHandles in handleList

- **`virtual ~DtReflectedAggregateList()`**

  destructor

- **`virtual DtReflectedAggregate* lookup(RTI::ObjectHandle id)`**

Look up aggregate by identifier

- **virtual DtReflectedAggregate* lookup(const DtGlobalObjectDesignator& objName)**

  Look up aggregate by name

- **virtual DtReflectedAggregate* first() const**

  Get the first/last entity in the list

- **virtual DtReflectedAggregate* last() const**

- **virtual void addAggregateAdditionCallback(DtAggregateCallbackFunction cb, void* userData)**

  Aggregate added callback registration/deregistration

- **virtual void removeAggregateAdditionCallback(DtAggregateCallbackFunction cb, void* userData)**

- **virtual void addAggregateRemovalCallback(DtAggregateCallbackFunction cb, void* userData)**

  Aggregate removed callback registration/deregistration

- **virtual void removeAggregateRemovalCallback(DtAggregateCallbackFunction cb, void* userData)**

- **DtReflectedAggregateList(const DtReflectedAggregateList& orig)**

  copy constructor

- **DtReflectedAggregateList& operator=(const DtReflectedAggregateList& orig)**

  assignment operator

- **virtual const DtList* dfltHandleList()**

  Asks the FOM Mapper for the DtList of RTI::ObjectClassHandles to use

- **virtual DtReflectedObject* newReflectedObject(DtHlaObject* obj) const**

  Override of virtual function. Calls newReflectedAggregate.

- **virtual DtReflectedAggregate* newReflectedAggregate(DtHlaObject* obj) const**

  Create a remote aggregate object for the HLA object. Caller is responsible for deletion.

- **virtual void removeAndDelete(DtReflectedObject* object, char* msg)**

  Virtual function override. Calls removeAndDelete(DtReflectedAggregate*....)

- **virtual void removeAndDelete(DtReflectedAggregate* aggregate, char* msg = NULL)**

  Default calls down to base class version, but can be overridden

- **virtual void objectAdded(DtReflectedObject* obj)**

  Called from processDiscoverObject() after a remote transmitter has been added to the list. Calls aggregateAdded.

- **virtual void aggregateAdded(DtReflectedAggregate* ent)**

  Called by objectAdded (which is called from processDiscoverObject()) after a remote aggregate has been added to the list. Default implementation prints a message. Subclasses can override this function to add functionality on addition.

- **DtBoolean mySmoothCapable**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file regctlcbinfo.h:

class DtRegControlCbInfo

class DtRegControlCbInfo: Instances of DtRegControlCbInfo are used for providing a callback when the RTI update control services are invoked

---

Public Methods

	DtRegControlCbInfo (RTI::ObjectClassHandle h, DtRegControlCb cb, void* usr) *constructor*
virtual	**~DtRegControlCbInfo** () *destructor*
	DtRegControlCbInfo (const DtRegControlCbInfo& orig) *copy constructor*
DtRegControlCbInfo&	**operator=** (const DtRegControlCbInfo& orig) *assignment operator*
virtual int	**match** (const DtRegControlCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtRegControlCbInfo& other) const *Check whether one callback info is equal to another*
virtual const RTI::ObjectClassHandle&	**objectClass** () const *Get the object class handle*
virtual const DtRegControlCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtRegControlCb&	**cbFunction** () const
inline const RTI::ObjectClassHandle&	**objectClass** () const
inline void*	**userData** () const

Protected Fields

RTI::ObjectClassHandle	**myClass**
DtRegControlCb	**myFunc**
void*	**myUsr**

Documentation

class DtRegControlCbInfo:

Instances of DtRegControlCbInfo are used for providing a callback when the RTI update control services are invoked. This class is a helper class to DtRegControlCbList.

- **`DtRegControlCbInfo(RTI::ObjectClassHandle h, DtRegControlCb cb, void* usr)`**
  constructor
- **`virtual ~DtRegControlCbInfo()`**
  destructor
- **`DtRegControlCbInfo( const DtRegControlCbInfo& orig)`**
  copy constructor
- **`DtRegControlCbInfo& operator=( const DtRegControlCbInfo& orig)`**
  assignment operator
- **`virtual int match(const DtRegControlCbInfo* const other) const`**
  Check whether one callback info is equal to another
- **`virtual int operator==(const DtRegControlCbInfo& other) const`**
  Check whether one callback info is equal to another
- **`virtual const RTI::ObjectClassHandle& objectClass() const`**
  Get the object class handle
- **`virtual const DtRegControlCb& cbFunction() const`**
  Get the callback function
- **`virtual void* userData() const`**
  Get the user data
- **`RTI::ObjectClassHandle myClass`**
- **`DtRegControlCb myFunc`**
- **`void* myUsr`**
- **`inline const DtRegControlCb& cbFunction() const`**
- **`inline const RTI::ObjectClassHandle& objectClass() const`**
- **`inline void* userData() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file regctlcblist.h:

class DtRegControlCbList

Public Methods

	DtRegControlCbList () *default constructor*
virtual	**~DtRegControlCbList** () *destructor*
	DtRegControlCbList (const DtRegControlCbList& orig) *copy constructor*
DtRegControlCbList&	**operator=** (const DtRegControlCbList& orig) *assignment operator*
virtual int	**add** (RTI::ObjectClassHandle h, DtRegControlCb cb, void* usr) *Add/Remove a start updates callback to/from the list*
virtual int	**remove** (RTI::ObjectClassHandle h, DtRegControlCb cb, void* usr)
virtual void	**clear** () *Remove all callbacks from the list*
virtual void	**apply** (RTI::ObjectClassHandle objectClass) *Callbacks are for a particular object class handle*

Protected Fields

DtList **myList**

Protected Methods

void	**dtorClear** () *Remove all callbacks from the list*
void	**ctorCopyList** (const DtRegControlCbList& orig) *Copy another callback list into this object's callback list*

Documentation

`DtRegControlCbList()`

default constructor

`virtual ~DtRegControlCbList()`

destructor

- **`DtRegControlCbList( const DtRegControlCbList& orig)`**

  copy constructor

- **`DtRegControlCbList& operator=( const DtRegControlCbList& orig)`**

  assignment operator

- **`virtual int add(RTI::ObjectClassHandle h, DtRegControlCb cb, void* usr)`**

  Add/Remove a start updates callback to/from the list. Returns whether add/remove took place.

- **`virtual int remove(RTI::ObjectClassHandle h, DtRegControlCb cb, void* usr)`**

- **`virtual void clear()`**

  Remove all callbacks from the list

- **`virtual void apply(RTI::ObjectClassHandle objectClass)`**

  Callbacks are for a particular object class handle. Call any callbacks in the list for the object class handle.

- **`void dtorClear()`**

  Remove all callbacks from the list. Non-virtual since meant to be called from dtor, otherwise should call clear().

- **`void ctorCopyList(const DtRegControlCbList& orig)`**

  Copy another callback list into this object's callback list. Non-virtual since meant to be called from ctor.

- **`DtList myList`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file remobjcbinfo.h:

class DtRemoveObjectCbInfo

Public Methods

	DtRemoveObjectCbInfo (DtRemoveObjectCb cb, void* usr) *constructor*
virtual	**~DtRemoveObjectCbInfo** () *destructor*
	DtRemoveObjectCbInfo (const DtRemoveObjectCbInfo& orig) *copy constructor*
DtRemoveObjectCbInfo&	**operator=** (const DtRemoveObjectCbInfo& orig) *assignment operator*
virtual int	**match** (const DtRemoveObjectCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtRemoveObjectCbInfo& other) const *Check whether one callback info is equal to another*
virtual const DtRemoveObjectCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtRemoveObjectCb&	**cbFunction** () const
inline void*	**userData** () const

Protected Fields

DtRemoveObjectCb	**myFunc**
void*	**myUsr**

Documentation

- **`DtRemoveObjectCbInfo(DtRemoveObjectCb cb, void* usr)`**
  constructor
- **`virtual ~DtRemoveObjectCbInfo()`**
  destructor
- **`DtRemoveObjectCbInfo( const DtRemoveObjectCbInfo& orig)`**
  copy constructor

- **DtRemoveObjectCbInfo& operator=(const DtRemoveObjectCbInfo& orig)**

  assignment operator

- **virtual int match(const DtRemoveObjectCbInfo* const other) const**

  Check whether one callback info is equal to another

- **virtual int operator==(const DtRemoveObjectCbInfo& other) const**

  Check whether one callback info is equal to another

- **virtual const DtRemoveObjectCb& cbFunction() const**

  Get the callback function

- **virtual void* userData() const**

  Get the user data

- **DtRemoveObjectCb myFunc**
- **void* myUsr**
- **inline const DtRemoveObjectCb& cbFunction() const**
- **inline void* userData() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vlutil.h:

endif class DtSelectParam

Public Methods

- **DtSelectParam** ()
- int **select** (DtTime timeout)
- void **addRead** (int fd)
- void **delRead** (int fd)
- int **canRead** (int fd)

Private Fields

- int **maxfd**
- fd_set **read0**
- fd_set **read**

Documentation

- `DtSelectParam()`
- `int select(DtTime timeout)`
- `void addRead(int fd)`
- `void delRead(int fd)`
- `int canRead(int fd)`
- `int maxfd`
- `fd_set read0`
- `fd_set read`

This class has no child classes.

In file socket.h:

class DtSocket

Inheritance:

DtSocket

Public Fields

- int **blocking**
- int **pkts_rcvd**

Public Methods

	DtSocket ()
virtual	**~DtSocket** ()
virtual int	**DtSend** (caddr_t packet, size_t size, DtInetAddr addr = DtUSE_DEFAULT_IP_ADDR)
virtual int	**sendTo** (caddr_t packet, size_t size, DtInetAddr addr = DtUSE_DEFAULT_IP_ADDR)
virtual int	**DtRecv** (caddr_t *buffptr, int *status=NULL) *Depending on the implementation in derived DtSockets, one of the forms of DtRecv may be faster than the other*
virtual int	**DtRecv** (caddr_t buff, size_t)
caddr_t	**DtRecv** () *for those that don't care about length:*
int	**flush** ()
void	**setBundling** (int maxSize)
int	**isBundling** ()
virtual void	**setUnbundling** (int)
virtual DtInetAddr	**lastSourceInetAddr** ()
virtual int	**DtReportPktsRcvd** (void)
virtual void	**DtBecomeNonBlocking** (void)
virtual void	**notifyMe** ()
virtual int	**listenToMulticastGroup** (DtInetAddr)
virtual int	**dropMulticastGroup** (DtInetAddr)

- virtual int **setMcastTtl** (int)
- virtual DtFilterTest* **exerciseId** ()
- virtual DtFilterTest* **protocolVersion** ()
- virtual DtFilterTest* **pduKind** ()
- virtual DtFilterTest* **ipAddr** ()
- virtual int **readFd** ()
  not all DtSocket have associated file descriptors those that do not will have readFd()/writeFd() return -1
- virtual int **writeFd** ()
- virtual int **isOk** ()

Protected Fields

- char **outBuff** [DtMaxPacketSize]
- int **outSize**
- int **bundling**
- int **maxBundleSize**

Protected Methods

- virtual DtBoolean **accept** (char *buf, DtInetAddr iaddr)

Private Fields

- DtFilterTest **version**
- DtFilterTest **exercise**
- DtFilterTest **kind**
- DtFilterTest **addr**

Documentation

- **`DtSocket()`**
- **`virtual ~DtSocket()`**
- **`virtual int DtSend(caddr_t packet, size_t size, DtInetAddr addr = DtUSE_DEFAULT_IP_ADDR)`**
- **`virtual int sendTo(caddr_t packet, size_t size, DtInetAddr addr =`**

DtUSE_DEFAULT_IP_ADDR)

- **virtual int DtRecv(caddr_t *buffptr, int *status=NULL)**

  Depending on the implementation in derived DtSockets, one of the forms of DtRecv may be faster than the other.

  NOTE: if a derived class complains "... hides virtual DtRecv() ...", make sure you have BOTH overloads of DtRecv declared and defined in the subclass.

  DtRecv has an optional status argument that could more elegantly be implemented as different return codes, but this would break older code.

- **virtual int DtRecv(caddr_t buff, size_t)**
- **caddr_t DtRecv()**

  for those that don't care about length:

- **int flush()**
- **void setBundling(int maxSize)**
- **int isBundling()**
- **virtual void setUnbundling(int)**
- **virtual DtInetAddr lastSourceInetAddr()**
- **virtual int DtReportPktsRcvd(void)**
- **virtual void DtBecomeNonBlocking(void)**
- **virtual void notifyMe()**
- **virtual int listenToMulticastGroup(DtInetAddr)**
- **virtual int dropMulticastGroup(DtInetAddr)**
- **virtual int setMcastTtl(int)**
- **virtual DtFilterTest* exerciseId()**
- **virtual DtFilterTest* protocolVersion()**
- **virtual DtFilterTest* pduKind()**
- **virtual DtFilterTest* ipAddr()**
- **int blocking**
- **int pkts_rcvd**
- **virtual int readFd()**

  not all DtSocket have associated file descriptors those that do not will have readFd()/writeFd() return -1

- **virtual int writeFd()**
- **virtual int isOk()**
- **virtual DtBoolean accept(char *buf, DtInetAddr iaddr)**
- **char outBuff[DtMaxPacketSize]**

- **int outSize**
- **int bundling**
- **int maxBundleSize**
- **DtFilterTest version**
- **DtFilterTest exercise**
- **DtFilterTest kind**
- **DtFilterTest addr**

Direct child classes:

DtTcpSocket
DtSocketPairSocket
DtNetSocket
DtClientSocket

Alphabetic index *Hierarchy of classes*

In file tcpsocket.h:

class DtTcpSocket: public DtSocket

Inheritance:

DtTcpSocket - DtSocket

Public Fields

- int **myOk**

Public Methods

- **DtTcpSocket** (int socketfd, struct sockaddr_in addr)
  Server Constructor
- **DtTcpSocket** (DtInetAddr addr, int port=DtDEFAULT_PORT, int nonBlockingConnect=0)
  Client Constructor
- virtual **~DtTcpSocket** ()
  Destructor
- int **sendTo** (caddr_t packet, size_t packetSize, DtInetAddr addr=DtUSE_DEFAULT_IP_ADDR)
- int **DtRecv** (caddr_t *buffptr, int *status=NULL)
- int **DtRecv** (caddr_t buff, size_t bufsize)
- int **readFd** ()
- int **writeFd** ()
- void **DtBecomeNonBlocking** (void)
  designate the socket as non-blocking
- int **isOk** ()
- int **connectionEstablished** ()
  Determines if the tcp socket has connected successfully
- int **sizeOffset** ()
  Get/Set mySizeOffset
- void **setSizeOffset** (int offset)
- int **sizeLength** ()
  Get/Set mySizeLength
- void **setSizeLength** (int length)

Private Fields

- char **myPktBuff** [2000]
- SOCKET **mySockfd**
- int **mySockfd**
- int **mySizeOffset**
- int **mySizeLength**
- static DtBoolean **theAnyInstancesFlag**

Private Methods

- int **recvWithStatus** (caddr_t *buffptr, int *status)

Inherited from DtSocket:

Public Fields

- int **blocking**
- int **pkts_rcvd**

Public Methods

- virtual int **DtSend**(caddr_t packet, size_t size, DtInetAddr addr = DtUSE_DEFAULT_IP_ADDR)
- int **flush**()
- void **setBundling**(int maxSize)
- int **isBundling**()
- virtual void **setUnbundling**(int)
- virtual DtInetAddr **lastSourceInetAddr**()
- virtual int **DtReportPktsRcvd**(void)
- virtual void **notifyMe**()
- virtual int **listenToMulticastGroup**(DtInetAddr)
- virtual int **dropMulticastGroup**(DtInetAddr)
- virtual int **setMcastTtl**(int)
- virtual DtFilterTest* **exerciseId**()
- virtual DtFilterTest* **protocolVersion**()

- virtual DtFilterTest* **pduKind**()
- virtual DtFilterTest* **ipAddr**()

Protected Fields

- char **outBuff**[DtMaxPacketSize]
- int **outSize**
- int **bundling**
- int **maxBundleSize**

Protected Methods

- virtual DtBoolean **accept**(char *buf, DtInetAddr iaddr)

Private Fields

- DtFilterTest **version**
- DtFilterTest **exercise**
- DtFilterTest **kind**
- DtFilterTest **addr**

Documentation

- **`DtTcpSocket(int socketfd, struct sockaddr_in addr)`**
  Server Constructor
- **`DtTcpSocket(DtInetAddr addr, int port=DtDEFAULT_PORT, int nonBlockingConnect=0)`**
  Client Constructor
- **`virtual ~DtTcpSocket()`**
  Destructor
- **`int sendTo(caddr_t packet, size_t packetSize, DtInetAddr addr=DtUSE_DEFAULT_IP_ADDR)`**
- **`int DtRecv(caddr_t *buffptr, int *status=NULL)`**
- **`int DtRecv(caddr_t buff, size_t bufsize)`**
- **`int readFd()`**
- **`int writeFd()`**
- **`void DtBecomeNonBlocking(void)`**
  designate the socket as non-blocking

- **`int isOk()`**
- **`int connectionEstablished()`**

  Determines if the tcp socket has connected successfully
- **`int sizeOffset()`**

  Get/Set mySizeOffset
- **`void setSizeOffset(int offset)`**
- **`int sizeLength()`**

  Get/Set mySizeLength
- **`void setSizeLength(int length)`**
- **`int recvWithStatus(caddr_t *buffptr, int *status)`**
- **`char myPktBuff[2000]`**
- **`SOCKET mySockfd`**
- **`int mySockfd`**
- **`int myOk`**
- **`int mySizeOffset`**
- **`int mySizeLength`**
- **`static DtBoolean theAnyInstancesFlag`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file pairsock.h:

class DtSocketPairSocket: public DtSocket

Inheritance:

DtSocketPairSocket - DtSocket

Public Methods

- **DtSocketPairSocket** ()
- **DtSocketPairSocket** (int sockNum)
- int **sendTo** (caddr_t packet, size_t size, DtInetAddr addr)
- int **recvWithStatus** (caddr_t buffptr, size_t size, int *status=NULL)
- int **DtRecv** (caddr_t *buffptr, int *status)
- int **DtRecv** (caddr_t buff, size_t size)
- void **DtBecomeNonBlocking** (void)
- int **otherFd** ()
- int **readFd** ()
- int **writeFd** ()

Private Fields

- int **myFd**
- int **ofd**
- char **buffer** [1500]

Inherited from DtSocket:

Public Fields

- int **blocking**
- int **pkts_rcvd**

Public Methods

- virtual int **DtSend**(caddr_t packet, size_t size, DtInetAddr addr = DtUSE_DEFAULT_IP_ADDR)
- int **flush**()
- void **setBundling**(int maxSize)
- int **isBundling**()
- virtual void **setUnbundling**(int)
- virtual DtInetAddr **lastSourceInetAddr**()
- virtual int **DtReportPktsRcvd**(void)
- virtual void **notifyMe**()
- virtual int **listenToMulticastGroup**(DtInetAddr)
- virtual int **dropMulticastGroup**(DtInetAddr)
- virtual int **setMcastTtl**(int)
- virtual DtFilterTest* **exerciseId**()
- virtual DtFilterTest* **protocolVersion**()
- virtual DtFilterTest* **pduKind**()
- virtual DtFilterTest* **ipAddr**()
- virtual int **isOk**()

Protected Fields

- char **outBuff**[DtMaxPacketSize]
- int **outSize**
- int **bundling**
- int **maxBundleSize**

Protected Methods

- virtual DtBoolean **accept**(char *buf, DtInetAddr iaddr)

Private Fields

- DtFilterTest **version**
- DtFilterTest **exercise**
- DtFilterTest **kind**

DtFilterTest **addr**

Documentation

- `DtSocketPairSocket()`
- `DtSocketPairSocket(int sockNum)`
- `int sendTo(caddr_t packet, size_t size, DtInetAddr addr)`
- `int recvWithStatus(caddr_t buffptr, size_t size, int *status=NULL)`
- `int DtRecv(caddr_t *buffptr, int *status)`
- `int DtRecv(caddr_t buff, size_t size)`
- `void DtBecomeNonBlocking(void)`
- `int otherFd()`
- `int readFd()`
- `int writeFd()`
- `int myFd`
- `int ofd`
- `char buffer[1500]`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file netsocket.h:

class DtNetSocket: public DtSocket

Inheritance:

DtNetSocket - DtSocket

Public Fields

static DtInitErrorAction	**initErrorAction**
static DtU32	**DtInetBroadcastMask**
int	**udpPort**
char	**outBuf** [2048]
DtInetAddr	**from_addr**

Public Methods

- **DtNetSocket** (int port=DtDEFAULT_PORT, DtU8 flags=DtReadWrite)
- **DtNetSocket** (int port, DtInetAddr destAddr, DtU8 flags)
  No default for flags since 3rd arg is needed to disambiguate
- **DtNetSocket** (int port, const struct in_addr *dest_addr, DtU8 flags=DtReadWrite)
  This constructor is obsolete, and will eventually disappear Use DtInitNetSocket(port, dest_addr->s_addr) instead
- **DtNetSocket** (char* dummy, int sockfd, DtU8 flags = DtReadWrite)
- **DtNetSocket** (int port, DtInetAddr fromAddr, DtInetAddr destAddr, const char *device, DtU8 flags = DtReadWrite)
  It only makes sense to set both destAddr and device on the GT

virtual	**~DtNetSocket** ()
int	**sendTo** (caddr_t packet, size_t size, DtInetAddr addr)
int	**DtRecv** (caddr_t buff, size_t)
int	**DtRecv** (caddr_t *buffptr, int *status=NULL)
int	**DtReportPktsRcvd** (void)
DtInetAddr	**lastSourceInetAddr** ()
void	**DtBecomeNonBlocking** (void)

- void **setUnbundling** (int onOff)
- int **isUnbundling** ()
- int **dataLeftInCurrentBundle** ()
- int **readFd** ()
- int **writeFd** ()
- int **isOk** ()
- void **DtGetBroadcastMask** ()
- inline int **listenToMulticastGroup** (DtInetAddr multicastAddr)
  must be kept around for GT
- int **dropMulticastGroup** (DtInetAddr multicastAddr)
- int **setMcastTtl** (int ttl)
- int **DtGtPacketDequeue** (caddr_t)
- void **DtGtPacketEnqueue** (char *)
- int **DtGtSend** (caddr_t, size_t)
  No, but Ethernet does, and that's the only thing the GT100 will ever work with

Protected Fields

- void* **nextPdu**
- int **sizeLeft**
- int **unbundling**
- int **m_bExternalSocket**
- DtInetAddr **m_devAddr**

Protected Methods

- void **DtInitNetSocket** (int port, DtInetAddr dest_addr, DtU8 rw_flags, const char *device)
- int **recvWithStatus** (caddr_t *buffptr, int *status)
- int **recvFromNet** (caddr_t *buffptr)
- int **recvWithUnbundling** (caddr_t *buffptr)

Private Fields

- DtNetFloat64 **pktBuff8** [DtMaxPacketSize/sizeof(DtNetFloat64)]
- DtNetFloat64 **unbundleBuff8** [DtMaxPacketSize/sizeof(DtNetFloat64)]
- DtU8 **RW_flags**
- int **boundToDestAddr**
- DtInetAddr **destInetAddr**

- int **firstInBundle**
- int **firstExId**
- int **firstProtocol**
- int **ok**
- int **skt_num**
- GtQueueElement* **PacketQueueOut**
- GtQueueElement* **PacketQueueIn**

Private Methods

- int **init_iodev** (const char *)
- int **init_write** (DtInetAddr *dest)
- int **init_read** (DtInetAddr dest)
- int **PrivateListenToMulticastGroup** (DtInetAddr multicastAddr, int on_init=0)
- void **findNextPdu** ()
- unsigned int **currentHeaderMatchesFirstInBundle** ()
- int **DtGtRecv** (caddr_t bufp, size_t bufsize)

Private

- *does DIS have a defined maximum?*

Inherited from DtSocket:

Public Fields

- int **blocking**
- int **pkts_rcvd**

Public Methods

- virtual int **DtSend**(caddr_t packet, size_t size, DtInetAddr addr = DtUSE_DEFAULT_IP_ADDR)
- int **flush**()
- void **setBundling**(int maxSize)
- int **isBundling**()
- virtual void **notifyMe**()

- virtual DtFilterTest* **exerciseId**()
- virtual DtFilterTest* **protocolVersion**()
- virtual DtFilterTest* **pduKind**()
- virtual DtFilterTest* **ipAddr**()

Protected Fields

- char **outBuff**[DtMaxPacketSize]
- int **outSize**
- int **bundling**
- int **maxBundleSize**

Protected Methods

- virtual DtBoolean **accept**(char *buf, DtInetAddr iaddr)

Private Fields

- DtFilterTest **version**
- DtFilterTest **exercise**
- DtFilterTest **kind**
- DtFilterTest **addr**

Documentation

- **`DtNetSocket(int port=DtDEFAULT_PORT, DtU8 flags=DtReadWrite)`**
- **`DtNetSocket(int port, DtInetAddr destAddr, DtU8 flags)`**

  No default for flags since 3rd arg is needed to disambiguate. Note a real loss, however, since any point-to-point NetSocket should probably be write-only anyway.
- **`DtNetSocket(int port, const struct in_addr *dest_addr, DtU8 flags=DtReadWrite)`**

  This constructor is obsolete, and will eventually disappear Use DtInitNetSocket(port, dest_addr->s_addr) instead
- **`DtNetSocket(char* dummy, int sockfd, DtU8 flags = DtReadWrite)`**
- **`DtNetSocket(int port, DtInetAddr fromAddr, DtInetAddr destAddr, const char *device, DtU8 flags = DtReadWrite)`**

  It only makes sense to set both destAddr and device on the GT. Under UNIX (with IP sockets), the device is implied by the destAddr It also only makes sense to speciify from_addr on the _GT_. On UNIX boxes, the from IP address gets filled in automagically.

- **virtual ~DtNetSocket()**
- **int sendTo(caddr_t packet, size_t size, DtInetAddr addr)**
- **int DtRecv(caddr_t buff, size_t)**
- **int DtRecv(caddr_t *buffptr, int *status=NULL)**
- **int DtReportPktsRcvd(void)**
- **DtInetAddr lastSourceInetAddr()**
- **void DtBecomeNonBlocking(void)**
- **void setUnbundling(int onOff)**
- **int isUnbundling()**
- **int dataLeftInCurrentBundle()**
- **int readFd()**
- **int writeFd()**
- **int isOk()**
- **static DtInitErrorAction initErrorAction**
- **static DtU32 DtInetBroadcastMask**
- **void DtGetBroadcastMask()**
- **int udpPort**
- **inline int listenToMulticastGroup(DtInetAddr multicastAddr)**
  must be kept around for GT
- **int dropMulticastGroup(DtInetAddr multicastAddr)**
- **int setMcastTtl(int ttl)**
- **int DtGtPacketDequeue(caddr_t)**
- **void DtGtPacketEnqueue(char *)**
- **void DtInitNetSocket(int port, DtInetAddr dest_addr, DtU8 rw_flags, const char *device)**
- **int recvWithStatus(caddr_t *buffptr, int *status)**
- **int recvFromNet(caddr_t *buffptr)**
- **int recvWithUnbundling(caddr_t *buffptr)**
- **void* nextPdu**
- **int sizeLeft**
- **int unbundling**
- **int m_bExternalSocket**
- **DtInetAddr m_devAddr**

- **DtNetFloat64 pktBuff8[DtMaxPacketSize/sizeof(DtNetFloat64)]**
- **DtNetFloat64 unbundleBuff8[DtMaxPacketSize/sizeof(DtNetFloat64)]**
- **int init_iodev(const char *)**
- **int init_write(DtInetAddr *dest)**
- **int init_read(DtInetAddr dest)**
- **int PrivateListenToMulticastGroup(DtInetAddr multicastAddr, int on_init=0)**
- **void findNextPdu()**
- **unsigned int currentHeaderMatchesFirstInBundle()**
- **DtU8 RW_flags**
- **int boundToDestAddr**
- **DtInetAddr destInetAddr**
- **int firstInBundle**
- **int firstExId**
- **int firstProtocol**
- **int ok**
- **int skt_num**
- **char outBuf[2048]**
- does DIS have a defined maximum?
- **int DtGtSend(caddr_t, size_t)**
  No, but Ethernet does, and that's the only thing the GT100 will ever work with
- **int DtGtRecv(caddr_t bufp, size_t bufsize)**
- **GtQueueElement* PacketQueueOut**
- **GtQueueElement* PacketQueueIn**
- **DtInetAddr from_addr**

Direct child classes:

DtGroupSocket

Alphabetic index *Hierarchy of classes*

In file netsocket.h:

enum DtInitErrorAction

- **DtInitErrorFatal**
- **DtInitErrorWarn**
- **DtInitErrorIgnore**

Documentation

- `DtInitErrorFatal`
- `DtInitErrorWarn`
- `DtInitErrorIgnore`

In file groupsocket.h:

class DtGroupSocket: public DtNetSocket

Inheritance:

DtGroupSocket - DtNetSocket - DtSocket

Public Methods

	DtGroupSocket (int port = DtDEFAULT_PORT, DtU8 flags = DtReadWrite) *Constructor - uses default address for listening*
	DtGroupSocket (int port, DtInetAddr addr, DtU8 flags) *Constructor - uses addr for listening*
virtual	**~DtGroupSocket** () *Destructor*
virtual void	**removeAllRemoteSites** () *Clear the list of remote sites*
virtual int	**DtSend** (caddr_t pkt, size_t size, DtInetAddr addr) *Virtual function override - send to all remote sites*
virtual void	**addRemoteSite** (DtInetAddr addr) *Add/remove a remote site to send to*
virtual void	**removeRemoteSite** (DtInetAddr addr)
virtual int	**sendToAllRemoteSites** (caddr_t pkt, size_t size, DtInetAddr dwSkipAddr = 0) *Send to all remote sites except dwSkipAddr*
virtual const DtList*	**remoteSitesList** () const *Return the remote sites list*
inline const DtList*	**remoteSitesList** () const

Protected Fields

DtList	**myRemoteSites** *DtList of DtInetAddrs (which do not need to be deleted)*

Protected Methods

DtListItem* **findItem** (DtInetAddr addr)

Inherited from DtNetSocket:

Public Fields

- static DtInitErrorAction **initErrorAction**
- static DtU32 **DtInetBroadcastMask**
- int **udpPort**
- char **outBuf**[2048]
- DtInetAddr **from_addr**

Public Methods

- int **sendTo**(caddr_t packet, size_t size, DtInetAddr addr)
- int **DtRecv**(caddr_t buff, size_t)
- int **DtRecv**(caddr_t *buffptr, int *status=NULL)
- int **DtReportPktsRcvd**(void)
- DtInetAddr **lastSourceInetAddr**()
- void **DtBecomeNonBlocking**(void)
- void **setUnbundling**(int onOff)
- int **isUnbundling**()
- int **dataLeftInCurrentBundle**()
- int **readFd**()
- int **writeFd**()
- int **isOk**()
- void **DtGetBroadcastMask**()
- inline int **listenToMulticastGroup**(DtInetAddr multicastAddr)
- int **dropMulticastGroup**(DtInetAddr multicastAddr)
- int **setMcastTtl**(int ttl)
- int **DtGtPacketDequeue**(caddr_t)
- void **DtGtPacketEnqueue**(char *)
- int **DtGtSend**(caddr_t, size_t)

Protected Fields

- void* **nextPdu**
- int **sizeLeft**
- int **unbundling**
- int **m_bExternalSocket**
- DtInetAddr **m_devAddr**

Protected Methods

- void **DtInitNetSocket**(int port, DtInetAddr dest_addr, DtU8 rw_flags, const char *device)
- int **recvWithStatus**(caddr_t *buffptr, int *status)
- int **recvFromNet**(caddr_t *buffptr)
- int **recvWithUnbundling**(caddr_t *buffptr)

Private Fields

- DtNetFloat64 **pktBuff8**[DtMaxPacketSize/sizeof(DtNetFloat64)]
- DtNetFloat64 **unbundleBuff8**[DtMaxPacketSize/sizeof(DtNetFloat64)]
- DtU8 **RW_flags**
- int **boundToDestAddr**
- DtInetAddr **destInetAddr**
- int **firstInBundle**
- int **firstExId**
- int **firstProtocol**
- int **ok**
- int **skt_num**
- GtQueueElement* **PacketQueueOut**
- GtQueueElement* **PacketQueueIn**

Private Methods

- int **init_iodev**(const char *)
- int **init_write**(DtInetAddr *dest)
- int **init_read**(DtInetAddr dest)
- int **PrivateListenToMulticastGroup**(DtInetAddr multicastAddr, int on_init=0)

- void **findNextPdu**()
- unsigned int **currentHeaderMatchesFirstInBundle**()
- int **DtGtRecv**(caddr_t bufp, size_t bufsize)

Private

-

Inherited from DtSocket:

Public Fields

- int **blocking**
- int **pkts_rcvd**

Public Methods

- int **flush**()
- void **setBundling**(int maxSize)
- int **isBundling**()
- virtual void **notifyMe**()
- virtual DtFilterTest* **exerciseId**()
- virtual DtFilterTest* **protocolVersion**()
- virtual DtFilterTest* **pduKind**()
- virtual DtFilterTest* **ipAddr**()

Protected Fields

- char **outBuff**[DtMaxPacketSize]
- int **outSize**
- int **bundling**
- int **maxBundleSize**

Protected Methods

- virtual DtBoolean **accept**(char *buf, DtInetAddr iaddr)

Private Fields

- DtFilterTest **version**
- DtFilterTest **exercise**
- DtFilterTest **kind**
- DtFilterTest **addr**

Documentation

- **`DtGroupSocket(int port = DtDEFAULT_PORT, DtU8 flags = DtReadWrite)`**
  Constructor - uses default address for listening. Will not send to any addresses until addRemoteSite is called.
- **`DtGroupSocket(int port, DtInetAddr addr, DtU8 flags)`**
  Constructor - uses addr for listening. Will not send to any addresses until addRemoteSite is called.
- **`virtual ~DtGroupSocket()`**
  Destructor
- **`virtual void removeAllRemoteSites()`**
  Clear the list of remote sites
- **`virtual int DtSend(caddr_t pkt, size_t size, DtInetAddr addr)`**
  Virtual function override - send to all remote sites
- **`virtual void addRemoteSite(DtInetAddr addr)`**
  Add/remove a remote site to send to
- **`virtual void removeRemoteSite(DtInetAddr addr)`**
- **`virtual int sendToAllRemoteSites( caddr_t pkt, size_t size, DtInetAddr dwSkipAddr = 0)`**
  Send to all remote sites except dwSkipAddr
- **`virtual const DtList* remoteSitesList() const`**
  Return the remote sites list
- **`DtListItem* findItem(DtInetAddr addr)`**
- **`DtList myRemoteSites`**
  DtList of DtInetAddrs (which do not need to be deleted)
- **`inline const DtList* remoteSitesList() const`**

This class has no child classes.

In file clientsocket.h:

class DtClientSocket: public DtSocket

Inheritance:

DtClientSocket - DtSocket

Public Fields

- endif static DtTime **psMsgTimeout**

Public Methods

	DtClientSocket (int port, int recvQSize = DtCsQDefaultSize, unsigned short csFlags = DtCsFlagsDefault)
	DtClientSocket (int port, DtInetAddr destAddr, int recvQSize = DtCsQDefaultSize, unsigned short csFlags = DtCsFlagsDefault)
	~DtClientSocket ()
int	**sendTo** (caddr_t packet, size_t size, DtInetAddr addr)
int	**DtRecv** (caddr_t, size_t)
int	**DtRecv** (caddr_t *buffptr, int *status=NULL) *this is slower as it does a memcpy()*
int	**listenToMulticastGroup** (DtInetAddr multicastAddr)
int	**dropMulticastGroup** (DtInetAddr multicastAddr)
int	**setMcastTtl** (int ttl)
void	**setUnbundling** (int onOff)
DtInetAddr	**lastSourceInetAddr** () *int isUnbundling(); is not currently implemented*
void	**DtBecomeNonBlocking** (void)
int	**readFd** ()
int	**writeFd** ()
int	**isOk** ()
DtBoolean	**isInternal** ()
void	**setPsMsgTimeout** (int seconds)

- void **notifyMe** ()
- void **disconnectFromServer** ()
- void **setPacketServerInternal** ()
- void **setPacketServerExternal** ()
- void **pingClients** ()
- void **testClients** ()
- void **terminatePacketServer** ()
- void **requestServerState** ()
- void **requestServerCStates** ()
- void **setPacketServerUnbundling** (int onOff)
- DtFilterTest* **exerciseId** ()
- DtFilterTest* **protocolVersion** ()
- DtFilterTest* **pduKind** ()
- DtFilterTest* **ipAddr** ()

Protected Fields

- DtShmMsgQueueReader* **reader**
- DtNetSocket* **netSock**
- DtBoolean **notifyPending**
- DtBoolean **ok**
- int **recvFd**
- int **sendFd**
- long **clientId**

Protected Methods

- void **init** (int port, DtInetAddr destAddr, int recvQSize, unsigned short csFlags)
- int **timedRecv** (caddr_t buf, size_t size, char *psKind)
- void **drainNotifySocket** ()
- void **checkGoAway** ()

Private Fields

- DtCsFilterTest* **exercise**
- DtCsFilterTest* **version**

- DtCsFilterTest* **kind**
- DtCsFilterTest* **addr**
- DtInetAddr **destAddr**
- char **clientSocketPath** [DtCsClientPathSize]
- int **serverPort**
- DtBoolean **internal**
- DtInetAddr **sourceAddr**
- DtNetFloat64 **pktBuff8** [(DtMaxPacketSize + sizeof(DtCsServerPacket) + 7) / sizeof(DtNetFloat64)]
- char* **shmAddr**
- DtCsControlReqMsg* **controlMsg**

Private Methods

- void **processControlMsg** (DtCsMsg *msg)
- DtCsMsg* **getPsMsg** (DtCsMsg *buf, char desiredKind)
- void **cleanup** (void)

Inherited from DtSocket:

Public Fields

- int **blocking**
- int **pkts_rcvd**

Public Methods

- virtual int **DtSend**(caddr_t packet, size_t size, DtInetAddr addr = DtUSE_DEFAULT_IP_ADDR)
- int **flush**()
- void **setBundling**(int maxSize)
- int **isBundling**()
- virtual int **DtReportPktsRcvd**(void)

Protected Fields

- char **outBuff**[DtMaxPacketSize]
- int **outSize**

- int **bundling**
- int **maxBundleSize**

Protected Methods

- virtual DtBoolean **accept**(char *buf, DtInetAddr iaddr)

Documentation

- **`DtClientSocket(int port, int recvQSize = DtCsQDefaultSize, unsigned short csFlags = DtCsFlagsDefault)`**
- **`DtClientSocket(int port, DtInetAddr destAddr, int recvQSize = DtCsQDefaultSize, unsigned short csFlags = DtCsFlagsDefault)`**
- **`~DtClientSocket()`**
- **`int sendTo(caddr_t packet, size_t size, DtInetAddr addr)`**
- **`int DtRecv(caddr_t, size_t)`**
- **`int DtRecv(caddr_t *buffptr, int *status=NULL)`**
  this is slower as it does a memcpy()
- **`int listenToMulticastGroup(DtInetAddr multicastAddr)`**
- **`int dropMulticastGroup(DtInetAddr multicastAddr)`**
- **`int setMcastTtl(int ttl)`**
- **`void setUnbundling(int onOff)`**
- **`DtInetAddr lastSourceInetAddr()`**
  int isUnbundling(); is not currently implemented. The answer depends on the state of the Packet Server.
- **`void DtBecomeNonBlocking(void)`**
- **`int readFd()`**
- **`int writeFd()`**
- **`int isOk()`**
- **`DtBoolean isInternal()`**
- **`void setPsMsgTimeout(int seconds)`**
- **`void notifyMe()`**
- **`void disconnectFromServer()`**
- **`void setPacketServerInternal()`**
- **`void setPacketServerExternal()`**

- void pingClients()
- void testClients()
- void terminatePacketServer()
- void requestServerState()
- void requestServerCStates()
- void setPacketServerUnbundling(int onOff)
- DtFilterTest* exerciseId()
- DtFilterTest* protocolVersion()
- DtFilterTest* pduKind()
- DtFilterTest* ipAddr()
- endif static DtTime psMsgTimeout
- void init(int port, DtInetAddr destAddr, int recvQSize, unsigned short csFlags)
- int timedRecv(caddr_t buf, size_t size, char *psKind)
- void drainNotifySocket()
- void checkGoAway()
- DtShmMsgQueueReader* reader
- DtNetSocket* netSock
- DtBoolean notifyPending
- DtBoolean ok
- int recvFd
- int sendFd
- long clientId
- void processControlMsg(DtCsMsg *msg)
- DtCsMsg* getPsMsg(DtCsMsg *buf, char desiredKind)
- void cleanup(void)
- DtCsFilterTest* exercise
- DtCsFilterTest* version
- DtCsFilterTest* kind
- DtCsFilterTest* addr
- DtInetAddr destAddr
- char clientSocketPath[DtCsClientPathSize]

- `int serverPort`
- `DtBoolean internal`
- `DtInetAddr sourceAddr`
- `DtNetFloat64 pktBuff8[(DtMaxPacketSize + sizeof(DtCsServerPacket) + 7) / sizeof(DtNetFloat64) ]`
- `char* shmAddr`
- `DtCsControlReqMsg* controlMsg`

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file stintcbinfo.h:

class DtStartStopInterCbInfo

class DtStartStopInterCbInfo: Instances of DtStartStopInterCbInfo are used for providing a callback when the RTI's start or stopInteractionGeneration services are invoked

---

Public Methods

	DtStartStopInterCbInfo (RTI::InteractionClassHandle hand, DtStartStopInterCb cb, void* usr) *constructor*
virtual	**~DtStartStopInterCbInfo** () *destructor*
	DtStartStopInterCbInfo (const DtStartStopInterCbInfo& orig) *copy constructor*
DtStartStopInterCbInfo&	**operator=** (const DtStartStopInterCbInfo& orig) *assignment operator*
virtual int	**match** (const DtStartStopInterCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtStartStopInterCbInfo& other) const *Check whether one callback info is equal to another*
virtual const RTI::InteractionClassHandle&	**interClass** () const *Get the inter class handle*
virtual const DtStartStopInterCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtStartStopInterCb&	**cbFunction** () const
inline const RTI::InteractionClassHandle&	**interClass** () const
inline void*	**userData** () const

Protected Fields

RTI::InteractionClassHandle	**myClass**
DtStartStopInterCb	**myFunc**
void*	**myUsr**

Documentation

class DtStartStopInterCbInfo:

Instances of DtStartStopInterCbInfo are used for providing a callback when the RTI's start or stopInteractionGeneration services are invoked. This class is a helper class to DtStartStopInterCbList.

- **`DtStartStopInterCbInfo( RTI::InteractionClassHandle hand, DtStartStopInterCb cb, void* usr)`**

  constructor

- **`virtual ~DtStartStopInterCbInfo()`**

  destructor

- **`DtStartStopInterCbInfo( const DtStartStopInterCbInfo& orig)`**

  copy constructor

- **`DtStartStopInterCbInfo& operator=( const DtStartStopInterCbInfo& orig)`**

  assignment operator

- **`virtual int match(const DtStartStopInterCbInfo* const other) const`**

  Check whether one callback info is equal to another

- **`virtual int operator==(const DtStartStopInterCbInfo& other) const`**

  Check whether one callback info is equal to another

- **`virtual const RTI::InteractionClassHandle& interClass() const`**

  Get the inter class handle

- **`virtual const DtStartStopInterCb& cbFunction() const`**

  Get the callback function

- **`virtual void* userData() const`**

  Get the user data

- **`RTI::InteractionClassHandle myClass`**
- **`DtStartStopInterCb myFunc`**
- **`void* myUsr`**
- **`inline const DtStartStopInterCb& cbFunction() const`**
- **`inline const RTI::InteractionClassHandle& interClass() const`**
- **`inline void* userData() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file tcpforwarder.h:

class DtTcpForwarder

Public Methods

	DtTcpForwarder (int port=DtDEFAULT_PORT)
virtual	**~DtTcpForwarder** ()
void	**sendToAll** (caddr_t packet, size_t size)
void	**readAndForward** (DtTime maxTimeOverAll, DtTime maxTimePerSocket)
DtTcpSocketMgr*	**socketMgr** ()

Private Fields

DtTcpSocketMgr*	**mySocketMgr**
DtList*	**mySockList**
DtTcpSocket*	**myBeginSocket**
DtListItem*	**myBeginItem**

Documentation

- `DtTcpForwarder(int port=DtDEFAULT_PORT)`
- `virtual ~DtTcpForwarder()`
- `void sendToAll(caddr_t packet, size_t size)`
- `void readAndForward(DtTime maxTimeOverAll, DtTime maxTimePerSocket)`
- `DtTcpSocketMgr* socketMgr()`
- `DtTcpSocketMgr* mySocketMgr`
- `DtList* mySockList`
- `DtTcpSocket* myBeginSocket`
- `DtListItem* myBeginItem`

This class has no child classes.

In file tcpsocketmgr.h:

class DtTcpSocketMgr

Public Fields

- DtList* **mySockList**

Public Methods

- **DtTcpSocketMgr** (int port=DtDEFAULT_PORT)
- virtual **~DtTcpSocketMgr** ()
- void **processConnections** ()
- int **isOk** ()
- DtList* **socketList** ()

Private Fields

- int **serverSockfd**
- int **myOk**

Documentation

- `DtTcpSocketMgr(int port=DtDEFAULT_PORT)`
- `virtual ~DtTcpSocketMgr()`
- `void processConnections()`
- `int isOk()`
- `DtList* socketList()`
- `int serverSockfd`
- `DtList* mySockList`
- `int myOk`

This class has no child classes.

In file strtupdcbinfo.h:

class DtUpdateControlCbInfo

class DtUpdateControlCbInfo: Instances of DtUpdateControlCbInfo are used for providing a callback when the RTI update control services are invoked

---

Public Methods

	DtUpdateControlCbInfo (RTI::ObjectClassHandle h, DtUpdateControlCb cb, void* usr) *constructor*
virtual	**~DtUpdateControlCbInfo** () *destructor*
	DtUpdateControlCbInfo (const DtUpdateControlCbInfo& orig) *copy constructor*
DtUpdateControlCbInfo&	**operator=** (const DtUpdateControlCbInfo& orig) *assignment operator*
virtual int	**match** (const DtUpdateControlCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtUpdateControlCbInfo& other) const *Check whether one callback info is equal to another*
virtual const RTI::ObjectClassHandle&	**objectClass** () const *Get the object class handle*
virtual const DtUpdateControlCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtUpdateControlCb&	**cbFunction** () const
inline const RTI::ObjectClassHandle&	**objectClass** () const
inline void*	**userData** () const

Protected Fields

RTI::ObjectClassHandle	**myClass**
DtUpdateControlCb	**myFunc**
void*	**myUsr**

Documentation

class DtUpdateControlCbInfo:

Instances of DtUpdateControlCbInfo are used for providing a callback when the RTI update control services are invoked. This class is a helper class to DtUpdateControlCbList.

- **`DtUpdateControlCbInfo(RTI::ObjectClassHandle h, DtUpdateControlCb cb, void* usr)`**
  constructor
- **`virtual ~DtUpdateControlCbInfo()`**
  destructor
- **`DtUpdateControlCbInfo( const DtUpdateControlCbInfo& orig)`**
  copy constructor
- **`DtUpdateControlCbInfo& operator=( const DtUpdateControlCbInfo& orig)`**
  assignment operator
- **`virtual int match(const DtUpdateControlCbInfo* const other) const`**
  Check whether one callback info is equal to another
- **`virtual int operator==(const DtUpdateControlCbInfo& other) const`**
  Check whether one callback info is equal to another
- **`virtual const RTI::ObjectClassHandle& objectClass() const`**
  Get the object class handle
- **`virtual const DtUpdateControlCb& cbFunction() const`**
  Get the callback function
- **`virtual void* userData() const`**
  Get the user data
- **`RTI::ObjectClassHandle myClass`**
- **`DtUpdateControlCb myFunc`**
- **`void* myUsr`**
- **`inline const DtUpdateControlCb& cbFunction() const`**
- **`inline const RTI::ObjectClassHandle& objectClass() const`**
- **`inline void* userData() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file updctlcbinfo.h:

class DtUpdateControlCbInfo

class DtUpdateControlCbInfo: Instances of DtUpdateControlCbInfo are used for providing a callback when the RTI update control services are invoked

---

Public Methods

	DtUpdateControlCbInfo (RTI::ObjectHandle hand, DtUpdateControlCb cb, void* usr) *constructor*
virtual	**~DtUpdateControlCbInfo** () *destructor*
	DtUpdateControlCbInfo (const DtUpdateControlCbInfo& orig) *copy constructor*
DtUpdateControlCbInfo&	**operator=** (const DtUpdateControlCbInfo& orig) *assignment operator*
virtual int	**match** (const DtUpdateControlCbInfo* const other) const *Check whether one callback info is equal to another*
virtual int	**operator==** (const DtUpdateControlCbInfo& other) const *Check whether one callback info is equal to another*
virtual const RTI::ObjectHandle&	**objectHandle** () const *Get the object handle*
virtual const DtUpdateControlCb&	**cbFunction** () const *Get the callback function*
virtual void*	**userData** () const *Get the user data*
inline const DtUpdateControlCb&	**cbFunction** () const
inline const RTI::ObjectHandle&	**objectHandle** () const

Protected Fields

RTI::ObjectHandle	**myHand**
DtUpdateControlCb	**myFunc**
void*	**myUsr**

Documentation

class DtUpdateControlCbInfo:

Instances of DtUpdateControlCbInfo are used for providing a callback when the RTI update control

services are invoked. This class is a helper class to DtUpdateControlCbList.

- **`DtUpdateControlCbInfo(RTI::ObjectHandle hand, DtUpdateControlCb cb, void* usr)`**
  constructor
- **`virtual ~DtUpdateControlCbInfo()`**
  destructor
- **`DtUpdateControlCbInfo( const DtUpdateControlCbInfo& orig)`**
  copy constructor
- **`DtUpdateControlCbInfo& operator=( const DtUpdateControlCbInfo& orig)`**
  assignment operator
- **`virtual int match(const DtUpdateControlCbInfo* const other) const`**
  Check whether one callback info is equal to another
- **`virtual int operator==(const DtUpdateControlCbInfo& other) const`**
  Check whether one callback info is equal to another
- **`virtual const RTI::ObjectHandle& objectHandle() const`**
  Get the object handle
- **`virtual const DtUpdateControlCb& cbFunction() const`**
  Get the callback function
- **`virtual void* userData() const`**
  Get the user data
- **`RTI::ObjectHandle myHand`**
- **`DtUpdateControlCb myFunc`**
- **`void* myUsr`**
- **`inline const DtUpdateControlCb& cbFunction() const`**
- **`inline const RTI::ObjectHandle& objectHandle() const`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file strtupdcblist.h:

class DtUpdateControlCbList

Public Methods

	DtUpdateControlCbList () *default constructor*
virtual	**~DtUpdateControlCbList** () *destructor*
	DtUpdateControlCbList (const DtUpdateControlCbList& orig) *copy constructor*
DtUpdateControlCbList&	**operator=** (const DtUpdateControlCbList& orig) *assignment operator*
virtual int	**add** (RTI::ObjectClassHandle h, DtUpdateControlCb cb, void* usr) *Add/Remove a start updates callback to/from the list*
virtual int	**remove** (RTI::ObjectClassHandle h, DtUpdateControlCb cb, void* usr)
virtual void	**clear** () *Remove all callbacks from the list*
virtual void	**apply** (RTI::ObjectClassHandle objectClass, const RTI::AttributeHandleSet& hSet) *Callbacks are for a particular object class handle*

Protected Fields

DtList **myList**

Protected Methods

void	**dtorClear** () *Remove all callbacks from the list*
void	**ctorCopyList** (const DtUpdateControlCbList& orig) *Copy another callback list into this object's callback list*

Documentation

`DtUpdateControlCbList()`

default constructor

`virtual ~DtUpdateControlCbList()`

destructor

- **`DtUpdateControlCbList( const DtUpdateControlCbList& orig)`**

  copy constructor

- **`DtUpdateControlCbList& operator=( const DtUpdateControlCbList& orig)`**

  assignment operator

- **`virtual int add(RTI::ObjectClassHandle h, DtUpdateControlCb cb, void* usr)`**

  Add/Remove a start updates callback to/from the list. Returns whether add/remove took place.

- **`virtual int remove(RTI::ObjectClassHandle h, DtUpdateControlCb cb, void* usr)`**

- **`virtual void clear()`**

  Remove all callbacks from the list

- **`virtual void apply(RTI::ObjectClassHandle objectClass, const RTI::AttributeHandleSet& hSet)`**

  Callbacks are for a particular object class handle. Call any callbacks in the list for the object class handle.

- **`void dtorClear()`**

  Remove all callbacks from the list. Non-virtual since meant to be called from dtor, otherwise should call clear().

- **`void ctorCopyList(const DtUpdateControlCbList& orig)`**

  Copy another callback list into this object's callback list. Non-virtual since meant to be called from ctor.

- **`DtList myList`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file updctlcblist.h:

class DtUpdateControlCbList

Public Methods

	DtUpdateControlCbList () *default constructor*
virtual	**~DtUpdateControlCbList** () *destructor*
	DtUpdateControlCbList (const DtUpdateControlCbList& orig) *copy constructor*
DtUpdateControlCbList&	**operator=** (const DtUpdateControlCbList& orig) *assignment operator*
virtual int	**add** (RTI::ObjectHandle hand, DtUpdateControlCb cb, void* usr) *Add/Remove a start updates callback to/from the list*
virtual int	**remove** (RTI::ObjectHandle hand, DtUpdateControlCb cb, void* usr)
virtual void	**clear** () *Remove all callbacks from the list*
virtual void	**apply** (RTI::ObjectHandle handle, const RTI::AttributeHandleSet& hSet) *Callbacks are for a particular object class handle*

Protected Fields

DtList **myList**

Protected Methods

void	**dtorClear** () *Remove all callbacks from the list*
void	**ctorCopyList** (const DtUpdateControlCbList& orig) *Copy another callback list into this object's callback list*

Documentation

`DtUpdateControlCbList()`

default constructor

`virtual ~DtUpdateControlCbList()`

destructor

- **`DtUpdateControlCbList( const DtUpdateControlCbList& orig)`**

  copy constructor

- **`DtUpdateControlCbList& operator=( const DtUpdateControlCbList& orig)`**

  assignment operator

- **`virtual int add(RTI::ObjectHandle hand, DtUpdateControlCb cb, void* usr)`**

  Add/Remove a start updates callback to/from the list. Returns whether add/remove took place.

- **`virtual int remove(RTI::ObjectHandle hand, DtUpdateControlCb cb, void* usr)`**

- **`virtual void clear()`**

  Remove all callbacks from the list

- **`virtual void apply(RTI::ObjectHandle handle, const RTI::AttributeHandleSet& hSet)`**

  Callbacks are for a particular object class handle. Call any callbacks in the list for the object class handle.

- **`void dtorClear()`**

  Remove all callbacks from the list. Non-virtual since meant to be called from dtor, otherwise should call clear().

- **`void ctorCopyList(const DtUpdateControlCbList& orig)`**

  Copy another callback list into this object's callback list. Non-virtual since meant to be called from ctor.

- **`DtList myList`**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file utmview.h:

class DtUtmView

class DtUtmView: Instances of DtUtmView are used to create a UTM coordinate view of the geocentric coordinates in an Entity State Repository

---

Public Methods

	DtUtmView (DtBaseEntityStateRepository* esr) *constructor*
virtual	**~DtUtmView** () *destructor*
	DtUtmView (const DtUtmView& orig) *copy constructor*
DtUtmView&	**operator=** (const DtUtmView& orig) *assignment operator*
virtual void	**setLocation** (const DtUtmCoord& aLocation) *Set/Get the location in UTM coordinates*
virtual const DtUtmCoord&	**location** () const
virtual void	**setVelocity** (const DtUtmCoord& aVelocity) *Set/Get the velocity*
virtual const DtUtmCoord&	**velocity** () const
virtual void	**setOrientation** (const DtTaitBryan& orient) *Set/Get the orientation*
virtual const DtTaitBryan&	**orientation** () const
virtual DtConstDcm	**bodyToLocal** () const *Get the body to local coordinate matrix*
virtual void	**setAcceleration** (const DtUtmCoord& anAcceleration) *Set/Get the acceleration*
virtual const DtUtmCoord&	**acceleration** () const
virtual void	**setRotationalVelocity** (DtConstVector bodyRates) *Set/Get the rotational velocity*
virtual DtConstVector	**rotationalVelocity** () const
inline DtUtmView*	**self** () const

Protected Fields

DtBaseEntityStateRepository*	**myESR**
DtUtmCoord	**myLocalLoc** *used as return values by inspectors*

DtUtmCoord	**myLocalVel**
DtUtmCoord	**myLocalAccel**
DtTaitBryan	**myLocalOrient**
DtDcm	**myBodyToLocal**
DtDcm	**myUtmToGeocMatrix**
DtVector	**myGeocLocForUtmToGeoc**
DtDcm	**myGeocToUtmMatrix**
DtVector	**myGeocLocForGeocToUtm**

Protected Methods

DtUtmView*	**self** () const *Cast off const-ness for trivial internal change*
virtual DtConstDcm	**utmToGeocMatrix** (DtConstVector geocLoc) const *Get the matrix which transforms from UTM to geocentric*
virtual DtConstDcm	**geocToUtmMatrix** (DtConstVector geocLoc) const *Get the matrix which transforms form geocentric to UTM*

Documentation

class DtUtmView: Instances of DtUtmView are used to create a UTM coordinate view of the geocentric coordinates in an Entity State Repository

- **`DtUtmView(DtBaseEntityStateRepository* esr)`**

  constructor

- **`virtual ~DtUtmView()`**

  destructor

- **`DtUtmView(const DtUtmView& orig)`**

  copy constructor

- **`DtUtmView& operator=(const DtUtmView& orig)`**

  assignment operator

- **`virtual void setLocation(const DtUtmCoord& aLocation)`**

  Set/Get the location in UTM coordinates

- **`virtual const DtUtmCoord& location() const`**

- **`virtual void setVelocity(const DtUtmCoord& aVelocity)`**

  Set/Get the velocity

- **`virtual const DtUtmCoord& velocity() const`**

- **virtual void setOrientation(const DtTaitBryan& orient)**

  Set/Get the orientation

- **virtual const DtTaitBryan& orientation() const**

- **virtual DtConstDcm bodyToLocal() const**

  Get the body to local coordinate matrix. This is just a matrix representation of the euler angles returned by orientation().

- **virtual void setAcceleration(const DtUtmCoord& anAcceleration)**

  Set/Get the acceleration

- **virtual const DtUtmCoord& acceleration() const**

- **virtual void setRotationalVelocity(DtConstVector bodyRates)**

  Set/Get the rotational velocity

- **virtual DtConstVector rotationalVelocity() const**

- **DtUtmView* self() const**

  Cast off const-ness for trivial internal change. Non-virtual since can't override on return type.

- **virtual DtConstDcm utmToGeocMatrix(DtConstVector geocLoc) const**

  Get the matrix which transforms from UTM to geocentric

- **virtual DtConstDcm geocToUtmMatrix(DtConstVector geocLoc) const**

  Get the matrix which transforms form geocentric to UTM

- **DtBaseEntityStateRepository* myESR**

- **DtUtmCoord myLocalLoc**

  used as return values by inspectors. Do not necessarily hold current state.

- **DtUtmCoord myLocalVel**

- **DtUtmCoord myLocalAccel**

- **DtTaitBryan myLocalOrient**

- **DtDcm myBodyToLocal**

- **DtDcm myUtmToGeocMatrix**

- **DtVector myGeocLocForUtmToGeoc**

- **DtDcm myGeocToUtmMatrix**

- **DtVector myGeocLocForGeocToUtm**

- **inline DtUtmView* self() const**

This class has no child classes.

Alphabetic index *Hierarchy of classes*

In file vrlfedtime.h:

class DtVrlFedTime: public RTI FedTime

Inheritance:

DtVrlFedTime

Public Methods

	DtVrlFedTime () ---------- *Constructors and Destructors* ----------
	DtVrlFedTime (const double&)
	DtVrlFedTime (const DtVrlFedTime&)
virtual	**~DtVrlFedTime** ()
virtual void	**setZero** () ---------- *Overloaded functions from RTI::FedTime* ----------
virtual RTI::Boolean	**isZero** ()
virtual void	**setEpsilon** ()
virtual void	**setPositiveInfinity** ()
virtual RTI::Boolean	**isPositiveInfinity** ()
virtual int	**encodedLength** () const
virtual void	**encode** (char *buff) const
virtual int	**getPrintableLength** () const
virtual void	**getPrintableString** (char*)
virtual RTI::FedTime&	**operator+=** (const RTI::FedTime&) throw(RTI::InvalidFederationTime) ---------- *Overloaded operators from RTI::FedTime* ----------
virtual RTI::FedTime&	**operator-=** (const RTI::FedTime&) throw(RTI::InvalidFederationTime)
virtual RTI::Boolean	**operator<=** (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)
virtual RTI::Boolean	**operator<** (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)
virtual RTI::Boolean	**operator>=** (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)
virtual RTI::Boolean	**operator>** (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)
virtual RTI::Boolean	**operator==** (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)

- virtual RTI::FedTime& **operator=** (const RTI::FedTime&) throw(RTI::InvalidFederationTime)
- virtual double **getTime** () const
  ---------- *Implementation functions*

- virtual RTI::Boolean **operator==** (const double&) const throw(RTI::InvalidFederationTime)
  ---------- *Implementation operators*

- virtual RTI::Boolean **operator!=** (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)
- virtual RTI::Boolean **operator!=** (const double&) const throw(RTI::InvalidFederationTime)
- virtual RTI::FedTime& **operator=** (const DtVrlFedTime&) throw(RTI::InvalidFederationTime)
- virtual RTI::FedTime& **operator=** (const double&) throw(RTI::InvalidFederationTime)
- virtual RTI::FedTime& **operator*=** (const RTI::FedTime&) throw(RTI::InvalidFederationTime)
- virtual RTI::FedTime& **operator/=** (const RTI::FedTime&) throw(RTI::InvalidFederationTime)
- virtual RTI::FedTime& **operator+=** (const double&) throw(RTI::InvalidFederationTime)
- virtual RTI::FedTime& **operator-=** (const double&) throw(RTI::InvalidFederationTime)
- virtual RTI::FedTime& **operator*=** (const double&) throw(RTI::InvalidFederationTime)
- virtual RTI::FedTime& **operator/=** (const double&) throw(RTI::InvalidFederationTime)
- virtual DtVrlFedTime **operator+** (const RTI::FedTime&) throw(RTI::InvalidFederationTime)
- virtual DtVrlFedTime **operator+** (const double&) throw(RTI::InvalidFederationTime)
- virtual DtVrlFedTime **operator-** (const RTI::FedTime&) throw(RTI::InvalidFederationTime)
- virtual DtVrlFedTime **operator-** (const double&) throw(RTI::InvalidFederationTime)
- virtual DtVrlFedTime **operator*** (const RTI::FedTime&) throw(RTI::InvalidFederationTime)
- virtual DtVrlFedTime **operator*** (const double&) throw(RTI::InvalidFederationTime)
- virtual DtVrlFedTime **operator/** (const RTI::FedTime&) throw(RTI::InvalidFederationTime)
- virtual DtVrlFedTime **operator/** (const double&) throw(RTI::InvalidFederationTime)
- friend ostream& **operator<<** (ostream&, RTI::FedTime&)
  ---------- *Implementation friends*

Private Fields

- double **_fedTime**
  ---------- *Implementation member variables*

- double **_zero**
- double **_epsilon**
- double **_positiveInfinity**

Documentation

- **DtVrlFedTime()**

  ---------- Constructors and Destructors

- **DtVrlFedTime(const double&)**
- **DtVrlFedTime(const DtVrlFedTime&)**
- **virtual ~DtVrlFedTime()**
- **virtual void setZero()**

  ---------- Overloaded functions from RTI::FedTime

- **virtual RTI::Boolean isZero()**
- **virtual void setEpsilon()**
- **virtual void setPositiveInfinity()**
- **virtual RTI::Boolean isPositiveInfinity()**
- **virtual int encodedLength() const**
- **virtual void encode(char *buff) const**
- **virtual int getPrintableLength() const**
- **virtual void getPrintableString(char*)**
- **virtual RTI::FedTime& operator+= (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**

  ---------- Overloaded operators from RTI::FedTime

- **virtual RTI::FedTime& operator-= (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**
- **virtual RTI::Boolean operator<= (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)**
- **virtual RTI::Boolean operator< (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)**
- **virtual RTI::Boolean operator>= (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)**
- **virtual RTI::Boolean operator> (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)**
- **virtual RTI::Boolean operator== (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator= (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**
- **virtual double getTime() const**

  ---------- Implementation functions

- **virtual RTI::Boolean operator== (const double&) const throw(RTI::InvalidFederationTime)**

---------- Implementation operators ----------

- **virtual RTI::Boolean operator!= (const RTI::FedTime&) const throw(RTI::InvalidFederationTime)**
- **virtual RTI::Boolean operator!= (const double&) const throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator= (const DtVrlFedTime&) throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator= (const double&) throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator*= (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator/= (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator+= (const double&) throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator-= (const double&) throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator*= (const double&) throw(RTI::InvalidFederationTime)**
- **virtual RTI::FedTime& operator/= (const double&) throw(RTI::InvalidFederationTime)**
- **virtual DtVrlFedTime operator+ (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**
- **virtual DtVrlFedTime operator+ (const double&) throw(RTI::InvalidFederationTime)**
- **virtual DtVrlFedTime operator- (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**
- **virtual DtVrlFedTime operator- (const double&) throw(RTI::InvalidFederationTime)**
- **virtual DtVrlFedTime operator* (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**
- **virtual DtVrlFedTime operator* (const double&) throw(RTI::InvalidFederationTime)**
- **virtual DtVrlFedTime operator/ (const RTI::FedTime&) throw(RTI::InvalidFederationTime)**
- **virtual DtVrlFedTime operator/ (const double&) throw(RTI::InvalidFederationTime)**
- **friend ostream& operator<< (ostream&, RTI::FedTime&)**

---------- Implementation friends ----------

- **double _fedTime**

---------- Implementation member variables ----------

- **double _zero**
- **double _epsilon**
- **double _positiveInfinity**

This class has no child classes.

- #define **DtAbsNotifyObserver_H_**
- #define **DtAntennaPatternRecord_H_**
- #define **DtApproximator_H_**
- #define **DtArtPartSpec_H_**
- const int **DtMAX_PARAMS_PER_PART**
- typedef struct DtArtPartSpec int **partType**
- int **attachedType**
- int **params**[DtMAX_PARAMS_PER_PART]
- **DtArtPartSpec**
- const int **DtAttachedToBase**
- #define **DtNoArtParts**((DtArtPartSpec*)
- #define **DtNullArtPartSpec**
- #define **DtArtPart_H_**
- typedef struct DtInternalArtParam float **myValue**
  Used only for internal representation of Art Param information
- int **myParamNum**
- int **myNext**
- DtTime **myTimeSet**
- **DtInternalArtParam**
- #define **DtArtPartThresholder_H_**
- #define **DtAttachedPartList_H_**
- const int **DtNoAttachedParts**[1]
  This constant can be passed to a VR-Link function that expects a list of attached part stations to indicate an empty list
- #define **DtBaseExerciseConn_H_**
- #define **DtBaseHashKey_H_**
- #define **DtBaseMsg_H_**
- typedef struct DtBufferPlaceHolder* **DtBufferPtr**
- #define **DtUSE_INTERNAL_BUFFER_REP**
- #define **DtUSE_INTERNAL_BUFFER**(DtBufferPtr(NULL)
- #define **DtBaseReflectedEntity_H_**
- #define **DtBeamAntennaPatternRecord_H_**
- #define **DtCartesianView_H_**
- #define **_CLIENTSOCKET_H_**
- #define **DtCsFlagsDefault**(DtCsFlagsReadWrite | DtCsFlagsServerChooseDropMode)
- #define **DtCsQDefaultSize**
- #define **COLLISION_H_**
- #define **_DATUMIDTYPE_H_**
  This file provides protocol independence for Collision Interactions
- #define **DtTYPE_NOT_DEFINED**
- #define **DtTYPE_BYTE_ARRAY**
- #define **DtTYPE_FLOAT32**
- #define **DtTYPE_FLOAT64**
- #define **DtTYPE_INT32**
- #define **DtTYPE_UNSIGNED32**

- #define **DtTYPE_STRING**
- int **DtDatumId2Type**(DtDatumParam id)
- int **DtAddDatumId2TypeMapping**(DtDatumParam id, int dataType, int warn = 0)
- #define **DtDatumRecordSet_H_**
- #define **DtNullDatumParam**(0)
- #define **DtNullVarDatumSpec**
- #define **DtFIXED**
  For backwards compatibility:
- #define **DtVAR**
- #define **DtDATUM_VALID**
- #define **DtDATUM_BAD_LENGTH**
- #define **DtDATUM_BAD_ID**
- #define **DtDATUM_BAD_INDEX**
- #define **DtDcm_H_**
- typedef const DtDcm& **DtConstDcm**
- typedef DtDcm& **DtDcmRef**
- #define **DtDeadReckoner_H_**
- #define **DtDesignatorRepository_H_**
- #define **DtDirectedRotRate_H_**
- #define **DtEmbeddedSystemRepository_H_**
- #define **DtEmitterSystemRepository_H_**
- #define **_ENUM2STRING_H_**
- const char* **DtEnumString**(DtPduKind kind)
- const char* **DtAddEnumString**(DtPduKind kind, const char *str, int warn = 0)
- #define **DtADD_ENUM_STRING**(k)
- #define **Euler_H_**
- void **DtEuler_to_BodyToRef**(const DtTaitBryan &euler, DtDcmRef body2ref)
  Given Euler angles between a body and a reference frame, produce the transformation matrix that converts from the body frame to the reference frame (or visa-versa)
- void **DtEuler_to_RefToBody**(const DtTaitBryan &euler, DtDcmRef ref2body)
- void **DtBodyToRef_to_Euler**(DtConstDcm body2ref, DtTaitBryan *euler)
  Given the transformation matrix that converts from a body frame to a reference frame (or visa-versa), produce the Euler angles between the body and the reference frames
- void **DtRefToBody_to_Euler**(DtConstDcm ref2body, DtTaitBryan *euler)
- void **DtEulerToEuler**(const DtTaitBryan &in, DtConstDcm in2out, DtTaitBryan *out)
  Given Euler angles with respect to reference frame "in", and the transformation matrix between frame "in" and frame öut", produce the Euler angles with repect to reference frame öut"
- #define **DtFedReader_H_**
- #define **DtFedTarget_H_**
- #define **DtFilename_H_**
- #define **DtGeodeticCoord_H_**
- void **DtGeocToGeod**(DtConstVector in, DtGeodeticCoord& out)
  Conversions between geocentric and geodetic: Use of these functions is discouraged in favor of DtGeodeticCoord member functions which perform conversions
- void **DtGeodToGeoc**(const DtGeodeticCoord& in, DtVectorRef out)
- typedef struct DtSpheroid DtFloat64 **semiMajor**
  Spheroid
- DtFloat64 **semiMinor**
- **DtSpheroid**

- typedef struct DtMapDatum DtSpheroid — **spheroid**
- DtFloat64 — **datumShift**[3]
- — **DtMapDatum**
- void — **DtUseMapDatum**(const DtMapDatum* md)
  Set and Get current map datum NOTE: map datum memory passed to DtUseMapDatum() must remain valid until next change of datum
- const DtMapDatum* — **DtCurrentMapDatum**()
- #define — **DtWGS84Spheroid**
  Here are a few of the most important spheroids; we should put all 13 in
- #define — **DtInternationalSpheroid**
- #define — **DtClark1866Spheroid**
- const DtMapDatum — **DtED50**
- const DtMapDatum — **DtNASANAD27**
- const DtMapDatum — **DtNASBNAD27**
- const DtMapDatum — **DtCONUSNAD27**
- #define — **DtGlobalIdHashlist_H_**
- typedef DtGlobalIdHashlist — **DtGlobalIdHashList**
- #define — **DtGroupSocket_H_**
- #define — **DtHashItem_H_**
- #define — **DtHashlist_H_**
- *Include obsolete old version for backwards compatibility*
- static const int — **dfltNumBuckets**
  Not static member since g++ doesn't like to use static members as default arguments
- typedef DtHashlist — **DtHashList**
- #define — **INTERACTION_H_**
- #define — **DtIntHashKey_H_**
  This file is used as a convenience to group interaction concepts together
- #define — **DtLgrEndPdu_H_**
- #define — **DtLgrEofActionPdu_H_**
- #define — **DtLgrHeartbeatPdu_H_**
- #define — **DtLgrPausePdu_H_**
- #define — **DtLgrPlayPdu_H_**
- #define — **DtLgrPlayActionPdu_H_**
- #define — **DtLgrPbFilePdu_H_**
- #define — **DtLgrQuitPdu_H_**
- #define — **DtLgrRecordActionPdu_H_**
- #define — **DtLgrRecFilePdu_H_**
- #define — **DtLgrRecordPdu_H_**
- #define — **DtLgrSetTimePdu_H_**
- #define — **DtLgrSkipTimePdu_H_**
- #define — **DtLgrStartPdu_H_**
- #define — **DtLgrStopPdu_H_**
- #define — **DtLgrTimeoutPdu_H_**
- #define — **DtLgrTimeScalePdu_H_**
- #define — **_LibMatrix_h**
- #define — **DtRad2Deg**(x)
- #define — **DtDeg2Rad**(x)

#define **DtEPico**
A selection of constants useful as tolerance and threshold levels, etc

#define **DtENano**

#define **DtEMicro**

#define **DtEMilli**

#define **DtECenti**

#define **DtEDeci**

#define **DtEHalf**

#define **DtENineTenths**

#define **DtEOne**

#define **DtEDeca**

#define **DtEHecto**

#define **DtEKilo**

#define **DtEMega**

#define **DtEGiga**

#define **DtEq**(x, y, e)
A macro to make use of these tolerances

#define **DtSetVec**(v, x, y, z)
Vector routines

void **DtVecInit**(DtVectorRef v)

void **DtVecCopy**(DtConstVector from, DtVectorRef to)
Initialize to all zeros

void **DtVecNormalize**(DtConstVector v, DtVectorRef result)

int **DtVecCheck**(DtConstVector v)

void **DtVecScale**(DtConstVector v, DtReal scale_factor, DtVectorRef result)
Is this vector normalized?

void **DtVecNeg**(DtConstVector v1, DtVectorRef result)

void **DtVecAdd**(DtConstVector v1, DtConstVector v2, DtVectorRef result)
result = -vec

void **DtVecSub**(DtConstVector v1, DtConstVector v2, DtVectorRef result)

DtReal **DtVecCosProd**(DtConstVector v1, DtConstVector v2)
result = v1 - v2

DtReal **DtVecDotProd**(DtConstVector v1, DtConstVector v2)
cos of angle between vecs

void **DtVecDump**(const char *str, DtConstVector v)

void **DtVecCrossProd**(DtConstVector v1, DtConstVector v2, DtVectorRef result)

DtReal **DtVecBoxProd**(DtConstVector v1, DtConstVector v2, DtConstVector v3)
sweep v1 to v2

sweep v1 to v2 to v3

double **DtDcmDcmRotAngle**(DtConstDcm a, DtConstDcm b)
Direction Cosine Matrix routines

void **DtDcmDcmRotAxis**(DtConstDcm a2ref, DtConstDcm b2ref, DtVectorRef n)

void **DtDcmInit**(DtDcmRef m)

void **DtDcmIdentInit**(DtDcmRef m)
Initialize to all zeros

int **DtDcmCheck**(DtConstDcm m)

void **DtDcmFixMatrix**(DtDcmRef mat, int axis)
Is this matrix orthonormal?

DtReal **DtDcmDeterminant**(DtConstDcm m)

void **DtDcmAdd**(DtConstDcm m1, DtConstDcm m2, DtDcmRef result)

void **DtDcmSub**(DtConstDcm m1, DtConstDcm m2, DtDcmRef result)
result = m1 + m2

- void **DtDcmDcmMul**(DtConstDcm m1, DtConstDcm m2, DtDcmRef result)
  result = m1 - m2
- void **DtDcmTranspose**(DtConstDcm m, DtDcmRef result)
  result = m1 x m2
- void **DtDcmAdjugate**(DtConstDcm m, DtDcmRef result)
- void **DtDcmInverse**(DtConstDcm m, DtDcmRef result)
- void **DtDcmScale**(DtConstDcm mat, DtReal scale_factor, DtDcmRef result)
- void **DtDcmDump**(const char *str, DtConstDcm mat)
- void **DtDcmRotInit**(DtDcmRef m, DtReal theta, int rot_axis)
  Rotation is counterclockwise when viewed a positive axis
- void **DtDcmRotInit2**(DtDcmRef m, DtReal sin_theta, int rot_axis)
- void **DtDcmTrigInit**(DtDcmRef m, DtReal sin_theta, DtReal cos_theta, int rot_axis)
  `$ only pos rots?`
- void **DtDcmForm**(DtReal *axis, DtReal angle, DtDcmRef C, DtReal *sin_ang, DtReal *cos_ang)
- void **DtDcmLevelInit**(DtDcmRef m, DtConstVector v)
- *y axis given, x axis parallel to ground*
- void **DtDcmVecMul**(DtConstDcm m, DtConstVector v, DtVectorRef result)
  Combined vector/matrix routines
- void **DtVecDcmMul**(DtConstVector v, DtConstDcm m, DtVectorRef result)
- typedef DtReal **DtEParm**[4]
  Euler parameter routines
- extern void **DtElrCopy**(DtEParm from, DtEParm to)
- extern void **DtElrElrCat**(DtEParm A_e_B0, DtEParm B0_e_B, DtEParm A_e_B)
- extern void **DtElrForm**(DtEParm axis, DtReal angle, DtEParm result)
- extern void **DtElrIdent**(DtEParm e)
- extern void **DtElrToMat**(DtEParm A_e_B, DtDcmRef A_c_B)
- extern void **DtElrTranspose**(DtEParm e, DtEParm result)
- extern void **DtDcmToElr**(DtConstDcm C, DtEParm e)
- extern void **DtVecElrTransform**(DtEParm v_A, DtEParm A_to_B, DtEParm v_B)
- typedef struct DtReal **DtQ0**
  Quaternion routines
- DtReal **DtQ1**
- DtReal **DtQ2**
- DtReal **DtQ3**
- **DtQuaternion**
- void **DtQuatQuatMul**(DtQuaternion *qresult, DtQuaternion *DtQ1, DtQuaternion *DtQ2)
- void **DtDcmFromQuaternion**(DtDcmRef m, DtQuaternion *q)
- void **DtQuaternionDump**(DtQuaternion *q)
  *void DtQuatScale(DtQuaternion *qresult, DtQuaternion *q, DtReal delta_t);*
- void **DtQuaternionFromTaitBryan**(DtQuaternion *q, const DtTaitBryan &tb)
- void **DtBodyToMat**(DtBodyRates av, DtConstDcm HcW, DtDcmRef cDot)
- void **DtTbToMat**(const DtTaitBryan &tb, DtDcmRef HcW)
- *Periodic modulus*
- double **DtModPer**(double x, double incl, double excl)
  For x with period ABS(incl-excl), return DtModPerLo(x, incl, excl) if incl < excl return DtModPerHi(x, excl, incl) if incl > excl
- double **DtModPerZero**(double x, double per)
  For x with period per, return 0 <= x < per
- double **DtModPerLo**(double x, double min, double max)
  For x with period (max-min), return min <= x < max

- double — **DtModPerHi**(double x, double min, double max)
  For x with period (max-min), return min < x <= max
- DtDirectedRotRate — **DtDirectRotRate**(DtConstVector pqr)
  directed rotation rates
- void — **DtMakeRotationDcm**(DtDirectedRotRate drr, double dt, DtDcmRef m)
- DtReal — **DtMM_Mult**(DtConstDcm m1, DtConstDcm m2, DtDcmRef result, generic_mm_multflag flag=TRANSPOSE_NONE)
- #define — **ADD**
  *Generic Vector Sum :Ans=A+s*B*
- #define — **SUBTRACT**
- void — **DtSum**(DtConstVector a, DtConstVector b, DtReal s, DtVectorRef ans)
- #define — **ZERO**
  generic initialize matrix routines
- #define — **IDENTITY**
- void — **DtM_Init**(DtConstDcm input, DtDcmRef result, int flag)
- void — **DtSetSym**(DtDcmRef v, DtReal aa, DtReal ab, DtReal ac, DtReal bb, DtReal bc, DtReal cc)
- void — **DtVV_Product**(DtConstVector a, DtConstVector b, DtDcmRef ans)
- void — **DtCreateCross**(DtConstVector r, DtDcmRef ans)
- void — **DtM**(DtConstVector v, DtDcmRef m)
- void — **DtRotate**(DtDcmRef ans, DtConstVector direction, double theta)
- double — **DtVecInfNorm**(DtConstVector v1)
- typedef DtReal — **DtThreeIndex**[3][3][3]
  Unobvious Constants
- typedef — **DtReal**(*DtThreeIndexPtr)
- extern DtThreeIndex — **DtLevi_Civita**
- double — **DtDistance**(DtConstVector p1, DtConstVector p2)
  additional Routines
- double — **DtDistSqr**(DtConstVector p1, DtConstVector p2)
  ||p1-p2||_{2}
- double — **DtMax**(double x, double y)
- double — **DtMin**(double x, double y)
- double — **DtLimit**(double x, double hi, double lo)
- #define — **_MachineTypes_h**
- *defined(__cplusplus), not __cplusplus, because __SUNPRO_CC defines it to be empty*
- typedef unsigned char — **DtU8**
- typedef unsigned short — **DtU16**
- typedef unsigned int — **DtU32**
- typedef int — **DtInt32**
- typedef unsigned long — **DtU32**
- typedef long int — **DtInt32**
- typedef char — **DtInt8**
- typedef short int — **DtInt16**
- typedef float — **DtFloat32**
- typedef double — **DtFloat64**
- typedef DtFloat64 — **DtReal**
- #define — **__declspec**(dllexport)
  Windows requires extra work to export from DLLs
- #define — **__declspec**(dllimport)
- #define — **__declspec**(dllexport)

- #define **__declspec**(dllimport)
- #define **__declspec**(dllexport)
- #define **__declspec**(dllimport)
- #define **__declspec**(dllexport)
- #define **__declspec**(dllimport)
- #define **define**
- #define **define**
- extern C int **getopt**(int argc, char** argv, char* opts)
- extern char* **optarg**
- extern int **optind**
- extern int **opterr**
- #define **MathTypes_H_**
- #define **SQR**(x)
- #define **MIN**(x, y)
- #define **MAX**(x, y)
- #define **LIMIT**(low, x, high)
- *For DtVector*
- *For DtDcm*
- *For DtTaitBryan*
- typedef DtVector **DtBodyRates**
  Angular rotation about body X, Y, Z axes
- *For DtDirectedRotRate and DtBodyRates*
- #define **DtRead**((DtU8)
- #define **DtWrite**((DtU8)
- #define **DtReadWrite**(DtRead | DtWrite)
- #define **DtQuiet**((DtU8)
- #define **NULL**
- **********
- int **datalen**
- caddr_t **data**
- **GtQueueElement**
- *The functionallity of the older constructor DtNetSocket(const char *device, int port) has been replaced by the call DtNetSocket(int port, DtInetBroadcastOfDevice(char *device));*
- #define **DtNetTimeStamp_H_**
  example device names are "ec0" and "le0"
- typedef DtU32 **DtTimeStamp**
  Host representation of a timeStamp
- #define **_NetTypes**
- typedef DtU8 **DtNetU8**
- typedef DtInt8 **DtNetInt8**
- typedef DtU16 **DtNetU16**
- typedef DtU32 **DtNetU32**
- typedef DtInt16 **DtNetInt16**

- typedef DtInt32 **DtNetInt32**
- typedef DtFloat32 **DtNetFloat32**
- typedef DtFloat64 **DtNetFloat64**
- #define **DtGlobalObjectDesignatorList_H_**
- *class DtGlobalObjectDesignatorList: Instances of DtGlobalObjectDesignatorList are list of globally significant object identifiers*
- #define **DtObjectIdHashKey_H_**
- #define **DtObjectIdHashlist_H_**
- typedef DtObjectIdHashlist **DtObjectIdHashList**
- #define **_PAIRSOCK_H_**
- #define **_PRINT_UTIL_H_**
- extern FILE* **DtDumpFile**
- void **DtDumpInteger**(const char *label, long i)
- void **DtDumpFloat**(const char *label, double f)
- void **DtDumpString**(const char *label, const char *string)
- void **DtDumpEnum**(const char *label, const char *enumStr, int enumNum)
- void **DtDumpHexBytes**(const char *label, const void *bytes, int size)
- void **DtDumpHexWord**(const char *label, DtU32 word)
- void **DtDumpBoolean**(const char *label, DtBoolean val, int onlyIfTrue = 0)
- #define **_ProcessControl_h**
- void **DtSleep**(DtTime sleep_period)
- char* **DtPollInputLine**(void)
  polls line-buffered standard input returns NULL if no input returns internal buffer of line if have input exits on ^D
- #define **DtRadioReceiverRepository_H_**
- #define **DtRadioTransmitterRepository_H_**
- #define **DtReflectedObjectCallbackInfo_H_**
- typedef void **(*DtReflectedObjectCallbackFunction)**(DtReflectedObject* obj, void* userData)
  Callback function that takes a DtReflectedObject and user data
- #define **SCRIPT_H_**
- typedef struct DtTime **t**
- DtViewControlPdu* **pdu**
- **StealthInstruction**
- #define **SIGNAL_INTER_H_**
- #define **DtSmoother_H_**
  This file provides protocol independence for Signal Interactions
- #define **DtSmoothOriWithDr_H_**
- #define **DtSmoothPosWithDr_H_**
- #define **DtSmoothPos_H_**
- #define **_Socket_h**
  DtSocket is a misnomer This provides net-level Packet IO Sometimes, it is implemented with sockets, sometimes not
- #define **DtDEFAULT_PORT**
- #define **DtUSE_DEFAULT_IP_ADDR**(DtInetAddr(0)
- #define **DtSOCKET_PACKET**
- #define **DtSOCKET_NO_PACKET**
- #define **DtSOCKET_SERVER_MSG**

- #define **DtSOCKET_FILTERED_PACKET**
- DtSocket* **DtNewDfltSocket**(int port, DtInetAddr destAddr = 0)
- #define **DtString_H_**
- #define **DtStringHashKey_H_**
- #define **DtTaitBryan_H_**
- #define **DtTcpForwarder_H_**
- #define **DtTcpSocket_H_**
- #define **DtTcpSocketMgr_H_**
- #define **DtThresholder_H_**
- #define **TopoCoord_H_**
- void **DtLatLon_to_GeocToTopo**(const DtGeodeticCoord& g, DtDcmRef geocToTopo)
  The matrix produced by DtLatLon_to_GeocToTopo can be used to rotate a geocentric vector to a topographic vector like this: DtDcmVecMul(geocToTopo, Vgeoc, Vtopo);
- void **DtGeocToTopoTransform**(double latitude, double longitude, DtCoordTransform* geoc2topo)
  This function can initialize a DtCoordTransform so that it can provide transformations between the geocentric coordinate system, and a particular topographic coordinate system defined by latitude and longitude
- #define **DtTopoView_H_**
- #define **DtTimeStampType_H_**
- #define **DtUtmCoord_H_**
- void **DtGeodToUtm**(const DtGeodeticCoord& in, DtUtmCoord& out)
  Use of these functions is discouraged in favor of DtUtmCoord member functions which perform conversions
- void **DtUtmToGeod**(const DtUtmCoord& utm, DtGeodeticCoord& out)
- void **DtGeocToUtm**(DtConstVector in, DtUtmCoord& out)
  Use of these functions is discouraged in favor of DtUtmCoord member functions which perform conversions
- void **DtUtmToGeoc**(const DtUtmCoord& in, DtVectorRef out)
- DtConstDcm **DtNedToEnuDcm**()
- DtConstDcm **DtEnuToNedDcm**()
- double **DtUtmGridDeclination**(const DtUtmCoord& utm)
  Use of these functions is discouraged in favor of DtUtmCoord member functions which perform conversions
- double **DtUtmGridDeclination**(double lat, double lon)
- void **DtGridDeclinationDcm**(double gridDecl, DtDcmRef utm2ned)
- typedef struct double **deg**
- typedef struct double **min**
- typedef struct double **sec**
- DtPolarDirection **direction**
- **DtDegMinSec**

Some constants:

- const DtDegMinSec **DtUtmLatRef1_1**
  Origin of the IITSEC '92 terrain database
- const DtDegMinSec **DtUtmLonRef1_1**
- const DtDegMinSec **DtUtmLatRef_IITSEC_93**
  Origin of the IITSEC '94 & '95 terrain database
- const DtDegMinSec **DtUtmLonRef_IITSEC_93**
- const DtDegMinSec **DtUtmLonRefZone10**
- #define **DtUtmView_H_**
- #define **DtVariableDatumRecord_H_**

#define **DT_LITTLE_ENDIAN**

#define **DT_FUNCTION_MUST_RETURN**

#define **DT_HAVE_COPYSIGN**

#define **DT_HAVE_AINT**

#define **DT_HAVE_CPP_STRINGIZE**

#define **DT_HAVE_STRTOUL**

#define **DT_CANT_HAVE_NET_UNIONS**

#define **DT_HAVE_ATT_SELECT**
Both of these next two should NOT be true (but both MAY be false)

#define **DT_HAVE_BSD_SELECT**

#define **DT_HAVE_SELECT_PROTO**

#define **DT_NO_CONSTPTRPTR**

#define **DT_HAVE_GETTIMEOFDAY**

#define **DT_HAVE_GETTIMEOFDAY_PROTO**

#define **DT_HAVE_SYSV_IPC**

#define **DT_HAVE_SYSV_IPC_PROTOS**

#define **DT_HAVE_SEMUN**

#define **DT_HAVE_SOCKET_PROTOS**

#define **DT_HAVE_WRITEV_PROTO**

#define **DT_HAVE_GETHOSTID**

#define **DT_HAVE_GETHOSTID_PROTO**

#define **DT_HAVE_STD_H**

#define **DT_HAVE_GETOPT_H**

#define **DT_HAVE_SOCKIO_H**

#define **DT_HAVE_ARPA_INET_H**

#define **DT_NO_MULTICAST**

#define **DT_FD_ZERO_USES_MEMSET**

#define **DT_HAVE_BZERO_PROTO_IN_MEMORY_H**

#define **DT_HAVE_BZERO_PROTO_IN_CC_LIBC_H**

#define **DT_HAVE_BZERO_PROTO_IN_BSTRING_H**

#define **DT_HAVE_FILE_H**

#define **DT_HAVE_UNISTD_H**
these were added while doing the MSVC Dos port

#define **DT_HAVE_MESSAGE_SOCKETS**

#define **DT_HAVE_PARAM_H**

#define **DT_HAVE_NET_IF_H**

#define **DT_HAVE_SYS_UIO_H**

#define **DT_HAVE_DAEMON**

#define **DT_HAVE_NUMERIC_CONSTANTS**

#define **DT_HAVE_PROCESS_H**

#define **DT_CANT_CAST_TO_CONST_2DARRAY**

#define **REUSE_ALLOWED**

#define **DT_HAVE_DRAND_48**

#define **DT_NO_RANDOM**

#define **DT_NO_RAND**

#define **M_LN2**

- #define **M_PI**
- #define **M_SQRT2**
- #define **M_E**
- #define **M_LOG2E**
- #define **M_LOG10E**
- #define **M_LN10**
- #define **M_PI_2**
- #define **M_PI_4**
- #define **M_1_PI**
- #define **M_2_PI**
- #define **M_2_SQRTPI**
- typedef char* **caddr_t**
- #define **DtList_H_**
- #define **looptlist**(theData, type, list)
- #define **looptlist2**(theData, type, list)
- #define **_VLPRINT_H**
- #define **DtNIFatal**
  Notification Levels
- #define **DtNIWarn**
- #define **DtNIInfo**
- #define **DtNIVerbose**
- extern int **DtNotifyLevel**

default is DtNIInfo

- void **DtFatalPerror**(const char *s, ...)
- void **DtWarnPerror**(const char *s, ...)
- void **DtFatalNetPerror**(const char *s, ...)
- void **DtWarnNetPerror**(const char *s, ...)
- void **DtWarn**(const char *s, ...)
- void **DtInfo**(const char *s, ...)
- void **DtVerbose**(const char *s, ...)
- void **setFileOutput**(const char *filename, int con=0, int son=0)
- #define **VL_TIME_H_**
- typedef DtReal **DtTime**
- const double **DtHour2Sec**
- const double **DtSec2Hour**
- void **DtTimeInit**(void)
  Initialize all VR-Link time functionality This must be called before any other time-related functions

Simulation Time Calls

- DtTime **DtGetSimTime**(void)
  Returns the last value passed to DtSetSimTime - the current value of VR-Link simulation time
- void **DtIsSimTimeSet**(void)
  Abort if DtSetSimTime was never called
- DtTime **DtGetElapsedRealTime**(void)
  Return real seconds elapsed since DtTimeInit was called
- DtTime **DtGetApproximateElapsedRealTime**(void)
  Returns the value of DtGetElapsedRealTime at the time that DtSetSimTime was lask invoked

- DtTime — **DtAbsRealTime**(void)
  Return current wallclock time - seconds since Jan 1, 1970
- extern int — **DtUseSimTimeForApproxRealTime**
- #define — **_VLUTIL_H_**
- #define — **DtDELETE**(x)
- #define — **DtISON**(a, x)
- #define — **DtISOFF**(a, x)
- #define — **DtTURNON**(a, x)
- #define — **DtTURNOFF**(a, x)
- #define — **DtMIN**(x, y)
- #define — **DtMAX**(x, y)
- #define — **DtLIMIT**(low, x, hi)
- #define — **DtABS**(x)
- #define — **DtSQR**(x)
- #define — **DtNUMBER**(foo)
- #define — **DtOFFSET**(type, field)
- #define — **DtSIZEOF**(type, field)
- inline int — **DtROUND_TO_MULT**(int num, int mult)
  Rounds num up to a multiple of mult
- #define — **DtATOI**(s)
- char* — **DtDeg2DmsString**(double val, char *buff, int prec)
- char* — **DtDeg2DmsString**(double val)
- extern C void — **bzero**(char *b, int length)
- int — **DtSelect**(int n, fd_set *i, fd_set *o, fd_set *e, DtTime t)
- inline void — **DtClose**(int &fd)
- else inline void — **DtClose**(int &fd)
- char* — **DtToLower**(char* str)
  Convert str to all lowercase
- char* — **DtToUpper**(char* str)
  Convert str to all uppercase
- typedef DtU32 — **DtInetAddr**
  ---------- *inet utils* ----------
- const DtInetAddr — **DtInetAddrAny**
- const DtInetAddr — **loopbackAddr**
- const char* — **DtInetAddrString**(DtInetAddr addr)
  converts ulong to internet dot notation (returns internal static buffer)
- *converts to internet dot notation to ulong in host order*
- DtInetAddr — **DtStringToInetAddr**(char *str)
- DtInetAddr — **DtStringToInetAddr**(const char *str)
- DtInetAddr — **DtInetBroadcastOfDevice**(const char *device)
- DtInetAddr — **DtInetAddrOfDevice**(const char *device)
- DtInetAddr — **DtNetMaskOfDevice**(const char *device)
- DtInetAddr — **DtFirstHostInetAddr**()
- int — **DtIsMulticastAddr**(DtInetAddr addr)
- #define — **DtVector_H_**
- typedef const DtVector& — **DtConstVector**
- typedef DtVector& — **DtVectorRef**

#define **ACK_INTERACTION_H_**

#define **DtAcknowledgePdu_H_**
This file provides protocol independence for Acknowledge Interactions

#define **DtAcknowledgeRPdu_H_**

#define **DtActionRequestPdu_H_**

#define **ACTION_REQUEST_INTERACTION_H_**

#define **DtActionRequestRPdu_H_**
This file provides protocol independence for ActionRequest Interactions

#define **ACTION_RESPONSE_H_**

#define **DtActionResponseRPdu_H_**
This file provides protocol independence for ActionResponse Interactions

#define **DtSilentAggregateList_H_**

#define **DtAggregateEntityDecoder_H_**

#define **DtDEFINE_SIMPLE_AGGREGATE_ATTR_DECODER**(\ attrName, netType, inspector)

#define **DtAggregateEntityEncoder_H_**

#define **DtDEFINE_SIMPLE_AGG_ENTITY_ATTR_CHECKER**(\ attrName, inspector)

#define **DtSilentEntityList_H_**

#define **AGGREGATE_PUB_H_**

#define **DtAggregateStateRepository_H_**
This file provides protocol independence for local aggregates

#define **DtAggregateStatePdu_H_**

#define **DtArealObjectStatePdu_H_**

#define **DtArtPartList_H_**

#define **DtBaseEmbeddedSystemDecoder_H_**

#define **DtDECLARE_EMBEDSYS_ATTR_DECODER**(attrName)
This is used to declare attribute decoding functions which are members of the class

#define **DtDEFINE_SIMPLE_BASE_EMBEDSYS_ATTR_DECODER**(\ attrName, netType, inspector)

#define **DtBaseEmbeddedSystemEncoder_H_**

#define **DtDECLARE_BASE_EMBEDSYS_ATTR_CHECKER**(attrName)
This is used to declare static member functions for determining whether or not an attribute needs to be encoded

#define **DtDECLARE_BASE_EMBEDSYS_ATTR_ENCODER**(attrName)

#define **DtDEFINE_SIMPLE_BASE_EMBEDSYS_ATTR_CHECKER**(\ attrName, inspector)

#define **DtBaseEmitterBeamDecoder_H_**

#define **DtDECLARE_EMITTERBEAM_ATTR_DECODER**(attrName)
This is used to declare attribute decoding functions which are members of the class

#define **DtDEFINE_SIMPLE_BASE_EMITTERBEAM_ATTR_DECODER**(\ attrName, netType, inspector)

#define **DtBaseEmitterBeamEncoder_H_**

#define **DtDECLARE_BASE_EMITTERBEAM_ATTR_CHECKER**(attrName)
This is used to declare static member functions for determining whether or not an attribute needs to be encoded

#define **DtDECLARE_BASE_EMITTERBEAM_ATTR_ENCODER**(attrName)

#define **DtDEFINE_SIMPLE_BASE_EMITTERBEAM_ATTR_CHECKER**(\ attrName, inspector)

#define **DtBaseEntityDecoder_H_**

#define **DtDECLARE_ENTITY_ATTR_DECODER**(attrName)
This is used to declare attribute decoding functions which are members of the class

#define **DtDECLARE_AGGREGATE_ATTR_DECODER**(attrName)

#define **DtDECLARE_BASE_ENTITY_ATTR_DECODER**(attrName)

#define **DtDEFINE_SIMPLE_BASE_ENTITY_ATTR_DECODER**(\ attrName, netType, inspector)

#define **DtBaseEntityEncoder_H_**

#define **DtDECLARE_ENTITY_ATTR_CHECKER**(attrName)
This is used to declare static member functions for determining whether or not an attribute needs to be encoded

#define **DtDECLARE_ENTITY_ATTR_ENCODER**(attrName)

#define **DtDECLARE_AGGREGATE_ATTR_CHECKER**(attrName)

#define **DtDECLARE_AGGREGATE_ATTR_ENCODER**(attrName)

#define **DtDECLARE_BASE_ENTITY_ATTR_CHECKER**(attrName)

#define **DtDECLARE_BASE_ENTITY_ATTR_ENCODER**(attrName)

#define **DtDEFINE_SIMPLE_BASE_ENTITY_ATTR_CHECKER**(\ attrName, inspector)

#define **DtBaseEntityStateRepository_H_**

#define **DtBaseReflectedAggregate_H_**

#define **_COLLPDU_H_**

#define **DtCollisionElasticPdu_H_**

#define **COMMENT_H_**

#define **DtCommentPdu_H_**
This file provides protocol independence for Comment Interactions

#define **DtCommentRPdu_H_**

#define **DtCreateEntityRPdu_H_**

#define **CREATEENTINTER_H_**

#define **DtCreateEntityPdu_H_**
This file provides protocol independence for CreateEntity Interactions

#define **DtAggregateStateDecoder_H_**

#define **DtAggregateStateEncoder_H_**

#define **DtDisAggregatePublisher_H_**

#define **DATA_H_**

#define **DtDataPdu_H_**
This file provides protocol independence for Comment Interactions

#define **DATA_QUERY_H_**

#define **DtDataQueryRPdu_H_**
This file provides protocol independence for ActionResponse Interactions

#define **DtDataRPdu_H_**

#define **DtDatumMap_H_**

#define **DtStateDecoderFactory_H_**

#define **DtEntityStateDecoder_H_**

#define **DtEntityEncoder_H_**

#define **DtDisEntityPublisher_H_**

#define **DtDesignatorPdu_H_**

#define **DtDesignatorDecoder_H_**

#define **DtDECLARE_DESIGNATOR_ATTR_DECODER**(attrName)
This is used to declare attribute decoding functions which are members of the class

#define **DtDEFINE_SIMPLE_DESIGNATOR_ATTR_DECODER**(\ attrName, netType, inspector)

#define **DtDesignatorEncoder_H_**

#define **DtDECLARE_DESIGNATOR_ATTR_CHECKER**(attrName)
This is used to declare static member functions for determining whether or not an attribute needs to be encoded

#define **DtDECLARE_DESIGNATOR_ATTR_ENCODER**(attrName)

#define **DtDEFINE_SIMPLE_DESIGNATOR_ATTR_CHECKER**(\ attrName, inspector)

#define **DETONATION_INTERACTION_H_**

#define **_detonationPdu_h**
This file provides protocol independence for Detonation Interactions

#define **DtDisExerciseConn_H_**

- #define **DtDInteraction_H_**
- #define **DtDirectRtiAmbassador_H_**
- #define **_DISENUMS_H_**
- const DtPduKind **DtNPCollisionPduKind**
- const DtPduKind **DtNPFieldPduKind**
- const DtPduKind **DtViewControlPduKind**
- const DtPduKind **DtMOPMigratorPduKind**
- const DtPduKind **DtNPSurfaceContactPduKind**
- const DtPduKind **DtGenericPduKind**
- const DtPduKind **DtLgrControlPduKind**
- #define **DtViewControlKind**
  backwards compatible
- #define **DtRadioOther**
- typedef DtRadioDetailedModulationAmplitude **DtRadioDetailedModulation**
- typedef DtDetailedModulationUnmodulated **DetailedModulationUnmodulated**
- typedef DtEncodingClass **DtRadioSignalEncoding**
- const DtEncodingClass **DtEncodedVoice**
- const DtEncodingClass **DtRawBinary**
- const DtEncodingClass **DtApplicationSpecific**
- const DtEncodingClass **DtPreRecordedVoice**
- #define **DtEXTEND_BIT_SHIFT_ENUM**(attr, val)
  This macro creates definitions that look like this: const DtAppearAttributeBitShift DtAppDensityBitShift = DtAppearAttributeBitShift(16);
  - **DtEXTEND_BIT_SHIFT_ENUM**(ActiveState, 23)
  - **DtEXTEND_BIT_SHIFT_ENUM**(Launcher, 16)
  - **DtEXTEND_BIT_SHIFT_ENUM**(CamoType, 17)
  - **DtEXTEND_BIT_SHIFT_ENUM**(Concealed, 19)
  - **DtEXTEND_BIT_SHIFT_ENUM**(Tent, 24)
  - **DtEXTEND_BIT_SHIFT_ENUM**(Ramp, 25)
  - **DtEXTEND_BIT_SHIFT_ENUM**(AfterBurner, 16)
  - **DtEXTEND_BIT_SHIFT_ENUM**(LaunchFlash, 16)
  - **DtEXTEND_BIT_SHIFT_ENUM**(LifeformState, 16)
  - **DtEXTEND_BIT_SHIFT_ENUM**(Weapon1, 24)
  - **DtEXTEND_BIT_SHIFT_ENUM**(Weapon2, 26)
  - **DtEXTEND_BIT_SHIFT_ENUM**(Density, 16)
  - **DtEXTEND_BIT_SHIFT_ENUM**(MinefieldType, 0)
    Minefield appearance
  - **DtEXTEND_BIT_SHIFT_ENUM**(MinefieldActiveStatus, 2)
  - **DtEXTEND_BIT_SHIFT_ENUM**(MinefieldLane, 3)
  - **DtEXTEND_BIT_SHIFT_ENUM**(MinefieldState, 13)
  - **DtEXTEND_BIT_SHIFT_ENUM**(ObjectPercentComplete, 0)
    Point, Linear, and Areal Object apearance
  - **DtEXTEND_BIT_SHIFT_ENUM**(ObjectDamage, 8)
  - **DtEXTEND_BIT_SHIFT_ENUM**(ObjectPredistributed, 10)
  - **DtEXTEND_BIT_SHIFT_ENUM**(ObjectState, 11)
  - **DtEXTEND_BIT_SHIFT_ENUM**(ObjectSmoking, 12)
  - **DtEXTEND_BIT_SHIFT_ENUM**(ObjectFlaming, 13)
  - *Point object specific appearance*
  - **DtEXTEND_BIT_SHIFT_ENUM**(ObjectAbatisBreach, 0)
    This is for the log crib, abatis, vehicle defilade, and infantry fighting position breach appearance

- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectBurstOpacity, 0)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectBurstSize, 8)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectBurstHeight, 16)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectBurstNumOfBursts, 24)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectBurstChemicalSmoke, 30)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectCraterSize, 0)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectRibbonBridgeNumOfSegments, 0)

Linear object specific appearance

- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectTankDitchBreach, 0)
  This is for the tank ditch and concertina wire breach appearance
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectTankDitchBreachLength, 16)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectTankDitchBreachLocation, 24)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectExhaustSmokeOpacity, 0)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectExhaustSmokeAttached, 8)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectExhaustSmokeChemical, 9)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectMinefieldLaneMarkerVisibleSide, 0)
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectMinefieldBreach, 0)
  Areal object specific appearance
- **DtEXTEND_BIT_SHIFT_ENUM**(ObjectMinefieldMineCount, 16)
- #define **DtEXTEND_ATTR_ENUM**(attr, val)
  This macro creates definitions that look like this: const DtAppearAttribute DtAppDensity = DtAppearAttribute(16 << DtAppDensityBitShift);
- **DtEXTEND_ATTR_ENUM**(DtAppActiveState, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppLauncher, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppCamoType, 0x3)
- **DtEXTEND_ATTR_ENUM**(DtAppConcealed, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppTent, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppRamp, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppAfterBurner, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppLaunchFlash, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppLifeformState, 0xf)
- **DtEXTEND_ATTR_ENUM**(DtAppWeapon1, 0x3)
- **DtEXTEND_ATTR_ENUM**(DtAppWeapon2, 0x3)
- **DtEXTEND_ATTR_ENUM**(DtAppDensity, 0xf)
- **DtEXTEND_ATTR_ENUM**(DtAppMinefieldType, 0x3)
  Minefield appearance
- **DtEXTEND_ATTR_ENUM**(DtAppMinefieldActiveStatus, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppMinefieldLane, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppMinefieldState, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppObjectPercentComplete, 0xff)
  Point, Linear, and Areal Object apearance
- **DtEXTEND_ATTR_ENUM**(DtAppObjectDamage, 0x3)
- **DtEXTEND_ATTR_ENUM**(DtAppObjectPredistributed, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppObjectState, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppObjectSmoking, 0x1)
- **DtEXTEND_ATTR_ENUM**(DtAppObjectFlaming, 0x1)

Point object specific appearance

- **DtEXTEND_ATTR_ENUM**(DtAppObjectAbatisBreach, 0x3)
  This is for the log crib, abatis, vehicle defilade, and infantry fighting position breach appearance
- **DtEXTEND_ATTR_ENUM**(DtAppObjectBurstOpacity, 0xff)
- **DtEXTEND_ATTR_ENUM**(DtAppObjectBurstSize, 0xff)
- **DtEXTEND_ATTR_ENUM**(DtAppObjectBurstHeight, 0xff)

- DtEXTEND_ATTR_ENUM(DtAppObjectBurstNumOfBursts, 0x3f)
- DtEXTEND_ATTR_ENUM(DtAppObjectBurstChemicalSmoke, 0x3)
- DtEXTEND_ATTR_ENUM(DtAppObjectCraterSize, 0xff)
- DtEXTEND_ATTR_ENUM(DtAppObjectRibbonBridgeNumOfSegments, 0xff)
-

Linear object specific appearance

- DtEXTEND_ATTR_ENUM(DtAppObjectTankDitchBreach, 0x3)
  This is for the tank ditch and concertina wire breach appearance
- DtEXTEND_ATTR_ENUM(DtAppObjectTankDitchBreachLength, 0xff)
- DtEXTEND_ATTR_ENUM(DtAppObjectTankDitchBreachLocation, 0xff)
- DtEXTEND_ATTR_ENUM(DtAppObjectExhaustSmokeOpacity, 0xff)
- DtEXTEND_ATTR_ENUM(DtAppObjectExhaustSmokeAttached, 0x1)
- DtEXTEND_ATTR_ENUM(DtAppObjectExhaustSmokeChemical, 0x3)
- DtEXTEND_ATTR_ENUM(DtAppObjectMinefieldLaneMarkerVisibleSide, 0x3)
- DtEXTEND_ATTR_ENUM(DtAppObjectMinefieldBreach, 0x3)
  Areal object specific appearance
- DtEXTEND_ATTR_ENUM(DtAppObjectMinefieldMineCount, 0xffff)
- #define **DtEXTEND_ATTR_VAL_ENUM**(name, val, attr)
- DtEXTEND_ATTR_VAL_ENUM(DtPaintSchemeUniformBitVal, 0, PaintScheme)
- DtEXTEND_ATTR_VAL_ENUM(DtPaintSchemeCamouflageBitVal, 1, PaintScheme)
- DtEXTEND_ATTR_VAL_ENUM(DtMobilityKillNoneBitVal, 0, MobilityKill)
- DtEXTEND_ATTR_VAL_ENUM(DtMobilityKillDisabledBitVal, 1, MobilityKill)
- DtEXTEND_ATTR_VAL_ENUM(DtFirePowerKillNoneBitVal, 0, FirePowerKill)
- DtEXTEND_ATTR_VAL_ENUM(DtFirePowerKillDisabledBitVal, 1, FirePowerKill)
- DtEXTEND_ATTR_VAL_ENUM(DtDamageNoneBitVal, 0, Damage)
- DtEXTEND_ATTR_VAL_ENUM(DtDamageSlightBitVal, 1, Damage)
- DtEXTEND_ATTR_VAL_ENUM(DtDamageModerateBitVal, 2, Damage)
- DtEXTEND_ATTR_VAL_ENUM(DtDamageDestroyedBitVal, 3, Damage)
- DtEXTEND_ATTR_VAL_ENUM(DtSmokeNoneBitVal, 0, Smoke)
- DtEXTEND_ATTR_VAL_ENUM(DtSmokeRisingBitVal, 1, Smoke)
- DtEXTEND_ATTR_VAL_ENUM(DtSmokeEngineBitVal, 2, Smoke)
- DtEXTEND_ATTR_VAL_ENUM(DtSmokeEngineAndPlumeBitVal, 3, Smoke)
- DtEXTEND_ATTR_VAL_ENUM(DtTrailingEffectsNoneBitVal, 0, TrailingEffects)
- DtEXTEND_ATTR_VAL_ENUM(DtTrailingEffectsSmallBitVal, 1, TrailingEffects)
- DtEXTEND_ATTR_VAL_ENUM(DtTrailingEffectsMediumBitVal, 2, TrailingEffects)
- DtEXTEND_ATTR_VAL_ENUM(DtTrailingEffectsLargeBitVal, 3, TrailingEffects)
- DtEXTEND_ATTR_VAL_ENUM(DtHatchNABitVal, 0, Hatch)
- DtEXTEND_ATTR_VAL_ENUM(DtHatchClosedBitVal, 1, Hatch)
- DtEXTEND_ATTR_VAL_ENUM(DtHatchPoppedBitVal, 2, Hatch)
- DtEXTEND_ATTR_VAL_ENUM(DtHatchPoppedAndPersonBitVal, 3, Hatch)
- DtEXTEND_ATTR_VAL_ENUM(DtHatchOpenBitVal, 4, Hatch)
- DtEXTEND_ATTR_VAL_ENUM(DtHatchOpenAndPersonBitVal, 5, Hatch)
- DtEXTEND_ATTR_VAL_ENUM(DtLightsNoneBitVal, 0, Lights)
- DtEXTEND_ATTR_VAL_ENUM(DtLightsRunningBitVal, 1, Lights)
- DtEXTEND_ATTR_VAL_ENUM(DtLightsNavigationBitVal, 2, Lights)
- DtEXTEND_ATTR_VAL_ENUM(DtLightsFormationBitVal, 3, Lights)
- DtEXTEND_ATTR_VAL_ENUM(DtFlamingNoBitVal, 0, Flaming)
- DtEXTEND_ATTR_VAL_ENUM(DtFlamingYesBitVal, 1, Flaming)
- DtEXTEND_ATTR_VAL_ENUM(DtNotFrozenBitVal, 0, Frozen)
- DtEXTEND_ATTR_VAL_ENUM(DtFrozenBitVal, 1, Frozen)
- DtEXTEND_ATTR_VAL_ENUM(DtPowerPlantOffBitVal, 0, PowerPlantStatus)
- DtEXTEND_ATTR_VAL_ENUM(DtPowerPlantOnBitVal, 1, PowerPlantStatus)
- DtEXTEND_ATTR_VAL_ENUM(DtActiveBitVal, 0, ActiveState)

- **DtEXTEND_ATTR_VAL_ENUM**(DtDeactivatedBitVal, 1, ActiveState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLauncherNotRaisedBitVal, 0, Launcher)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLauncherRaisedBitVal, 1, Launcher)
- **DtEXTEND_ATTR_VAL_ENUM**(DtDesertCamouflageBitVal, 0, CamoType)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWinterCamouflageBitVal, 1, CamoType)
- **DtEXTEND_ATTR_VAL_ENUM**(DtForestCamouflageBitVal, 2, CamoType)
- **DtEXTEND_ATTR_VAL_ENUM**(DtNotConcealedBitVal, 0, Concealed)
- **DtEXTEND_ATTR_VAL_ENUM**(DtConcealedBitVal, 1, Concealed)
- **DtEXTEND_ATTR_VAL_ENUM**(DtTentNotExtendedBitVal, 0, Tent)
- **DtEXTEND_ATTR_VAL_ENUM**(DtTentExtendedBitVal, 1, Tent)
- **DtEXTEND_ATTR_VAL_ENUM**(DtRampUpBitVal, 0, Ramp)
- **DtEXTEND_ATTR_VAL_ENUM**(DtRampDownBitVal, 1, Ramp)
- **DtEXTEND_ATTR_VAL_ENUM**(DtAfterBurnerNotOnBitVal, 0, AfterBurner)
- **DtEXTEND_ATTR_VAL_ENUM**(DtAfterBurnerOnBitVal, 1, AfterBurner)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLaunchFlashNotPresentBitVal, 0, LaunchFlash)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLaunchFlashPresentBitVal, 1, LaunchFlash)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformNABitVal, 0, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformStandingBitVal, 1, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformWalkingBitVal, 2, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformRunningBitVal, 3, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformKneelingBitVal, 4, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformProneBitVal, 5, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformCrawlingBitVal, 6, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformSwimmingBitVal, 7, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformParachutingBitVal, 8, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformJumpingBitVal, 9, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformSittingBitVal, 10, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformSquattingBitVal, 11, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformCrouchingBitVal, 12, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtLifeformWadingBitVal, 13, LifeformState)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWeapon1None, 0, Weapon1)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWeapon1Stowed, 1, Weapon1)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWeapon1Deployed, 2, Weapon1)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWeapon1InFirePosition, 3, Weapon1)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWeapon2None, 0, Weapon2)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWeapon2Stowed, 1, Weapon2)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWeapon2Deployed, 2, Weapon2)
- **DtEXTEND_ATTR_VAL_ENUM**(DtWeapon2InFirePosition, 3, Weapon2)
- **DtEXTEND_ATTR_VAL_ENUM**(DtDensityClearBitVal, 0, Density)
- **DtEXTEND_ATTR_VAL_ENUM**(DtDensityHazyBitVal, 1, Density)
- **DtEXTEND_ATTR_VAL_ENUM**(DtDensityDenseBitVal, 2, Density)
- **DtEXTEND_ATTR_VAL_ENUM**(DtDensityVeryDenseBitVal, 3, Density)
- **DtEXTEND_ATTR_VAL_ENUM**(DtDensityOpaqueBitVal, 4, Density)
- **DtEXTEND_ATTR_VAL_ENUM**(DtMixedMinefield, 0, MinefieldType)

  Minefield appearance
- **DtEXTEND_ATTR_VAL_ENUM**(DtPureAntiPersonnelMinefield, 1, MinefieldType)
- **DtEXTEND_ATTR_VAL_ENUM**(DtPureAntiTankMinefield, 2, MinefieldType)
- **DtEXTEND_ATTR_VAL_ENUM**(DtActiveMinefield, 0, MinefieldActiveStatus)
- **DtEXTEND_ATTR_VAL_ENUM**(DtInactiveMinefield, 1, MinefieldActiveStatus)
- **DtEXTEND_ATTR_VAL_ENUM**(DtMinefieldHasActiveLane, 0, MinefieldLane)
- **DtEXTEND_ATTR_VAL_ENUM**(DtMinefieldHasInactiveLane, 1, MinefieldLane)
- **DtEXTEND_ATTR_VAL_ENUM**(DtActiveMinefieldState, 0, MinefieldState)

- DtEXTEND_ATTR_VAL_ENUM(DtDeactivatedMinefieldState, 1, MinefieldState)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectNoDamage, 0, ObjectDamage)

  Point, Linear, and Areal Object apearance
- DtEXTEND_ATTR_VAL_ENUM(DtObjectDamaged, 1, ObjectDamage)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectDestroyed, 2, ObjectDamage)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectCreatedDuringExercise, 0, ObjectPredistributed)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectPredistributed, 1, ObjectPredistributed)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectActive, 0, ObjectState)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectDeactivated, 1, ObjectState)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectNotSmoking, 0, ObjectSmoking)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectSmoking, 1, ObjectSmoking)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectNotFlaming, 0, ObjectFlaming)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectFlaming, 1, ObjectFlaming)

Point object specific appearance

- DtEXTEND_ATTR_VAL_ENUM(DtObjectAbatisNoBreaching, 0, ObjectAbatisBreach)

  This is for the log crib, abatis, vehicle defilade, and infantry fighting position breach appearance
- DtEXTEND_ATTR_VAL_ENUM(DtObjectAbatisBreached, 1, ObjectAbatisBreach)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectAbatisCleared, 2, ObjectAbatisBreach)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectBurstOtherSmoke, 0, ObjectBurstChemicalSmoke)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectBurstHydrochloricSmoke, 1, ObjectBurstChemicalSmoke)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectBurstWhitePhosphorousSmoke, 2, ObjectBurstChemicalSmoke)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectBurstRedPhosphorousSmoke, 3, ObjectBurstChemicalSmoke)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectTankDitchBreachNoBreaching, 0, ObjectTankDitchBreach)

  Linear object specific appearance
- DtEXTEND_ATTR_VAL_ENUM(DtObjectTankDitchBreachSlightBreaching, 1, ObjectTankDitchBreach)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectTankDitchBreachModerateBreaching, 2, ObjectTankDitchBreach)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectTankDitchBreachCleared, 3, ObjectTankDitchBreach)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectExhaustSmokeNotAttached, 0, ObjectExhaustSmokeAttached)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectExhaustSmokeIsAttached, 1, ObjectExhaustSmokeAttached)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectExhaustOtherSmoke, 0, ObjectExhaustSmokeChemical)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectExhaustHydrochloricSmoke, 1, ObjectExhaustSmokeChemical)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectExhaustWhitePhosphorousSmoke, 2, ObjectExhaustSmokeChemical)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectExhaustRedPhosphorousSmoke, 3, ObjectExhaustSmokeChemical)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectMinefieldLeftSideVisible, 0, ObjectMinefieldLaneMarkerVisibleSide)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectMinefieldRightSideVisible, 1, ObjectMinefieldLaneMarkerVisibleSide)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectMinefieldBothSidesVisible, 2, ObjectMinefieldLaneMarkerVisibleSide)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectMinefieldNoBreaching, 0, ObjectMinefieldBreach)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectMinefieldBreached, 1, ObjectMinefieldBreach)
- DtEXTEND_ATTR_VAL_ENUM(DtObjectMinefieldCleared, 2, ObjectMinefieldBreach)
- #define **DtDisPduFactory_H_**
- #define **DtReflectedEmitterSystemList_H_**

- #define **DtReflectedAggregateList_H_**
- #define **DtReflectedAggregate_H_**
- #define **DtReflectedEntity_H_**
- #define **DtReflectedObject_H_**
- #define **DtReflectedEmitterSystem_H_**
- #define **DtReflectedEntityList_H_**
- #define **DtReflectedObjectList_H_**
- #define **DtDiscoverObjectCbInfo_H_**
- typedef void **(*DtDiscoverObjectCb)**(DtHlaObject* obj, void* usr)
  class DtDiscoverObjectCbInfo: Instances of DtDiscoverObjectCbInfo are used for providing a callback when an object which meets the federate's subscription criteria is discovered
- #define **dDtStateMsg_H_**
- #define **DtElectromagneticEmissionPdu_H_**
- #define **DtEmissionPdu_H_**
- #define **DtEmitterSystemDecoder_H_**
- #define **DtDECLARE_EMITTERSYS_ATTR_DECODER**(attrName)
  This is used to declare attribute decoding functions which are members of the class
- #define **DtDEFINE_SIMPLE_EMITTERSYS_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtEmitterSystemEncoder_H_**
- #define **DtDECLARE_EMITTERSYS_ATTR_CHECKER**(attrName)
  This is used to declare static member functions for determining whether or not an attribute needs to be encoded
- #define **DtDECLARE_EMITTERSYS_ATTR_ENCODER**(attrName)
- #define **DtDEFINE_SIMPLE_EMITTERSYS_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtEmitterBeamRepository_H_**
- typedef DtGlobalObjectDesignator **DtTrackJamDesignator**
- typedef DtGlobalObjectDesignatorList **DtTrackJamDesignatorList**
- #define **DtEmptyFomMapper_H_**
- #define **DtStateEncoderFactory_H_**
- #define **ENTITY_PUB_H_**
- #define **DtEntityStateRepository_H_**
  This file provides protocol independence for Local Entities
- #define **_ENTITYTYPES_H_**
- #define **DtSameEntityKind**(a,b)
- #define **DtIsLandPlatform**(a)
- #define **DtIsLandLifeForm**(a)
- #define **DtIsAirPlatform**(a)
- #define **DtIsTank**(a)
- #define **DtIsRotaryWing**(a)
- #define **DtIsFixedWing**(a)
- #define **antiPersonell**
- #define **DtEntityIsAntiPersonnel**(a)
- #define **DtEnvironmentEntityDecoder_H_**
- #define **DtDEFINE_SIMPLE_ENVENT_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtEnvironmentEntityEncoder_H_**
- #define **DtDEFINE_SIMPLE_ENVENT_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtEnvironmentTypeRecord_H_**
- #define **DtEnvironmentalProcessPdu_H_**

- #define **DtEntityStatePdu_H_**
- #define **DtEntityStateUpdatePdu_H_**
- #define **DtEventId_H_**
- typedef DtEventId **DtEventID**
- #define **EVENTREPORT_H_**
- #define **DtEventReportPdu_H_**
  This file provides protocol independence for EventReport Interactions
- #define **DtEventReptRPdu_H_**
- #define **EXERCISE_CONNECTION_H_**
- #define **FIRE_INTERACTION_H_**
  This file provides protocol independence for exercise connections
- #define **DtFirePdu_H_**
  This file provides protocol independence for Fire Interactions
- #define **DtFom_H_**
- #define **DtFomMapper_H_**
- #define **DtBasicFixedWingAircraftRecord_H_**
- #define **DtBasicRotorWingAircraftRecord_H_**
- #define **DtBasicGroundCombatSoldierRecord_H_**
- #define **DtBasicGroundCombatVehicleRecord_H_**
- #define **DtEnhancedFixedWingAircraftRecord_H_**
- #define **DtEnhancedRotorWingAircraftRecord_H_**
- #define **DtEnhancedGroundCombatSoldierRecord_H_**
- #define **DtEnhancedGroundCombatVehicleRecord_H_**
- #define **DtGroundLogisticsVehicleRecord_H_**
- #define **DtGenericObjectDecoder_H_**
- #define **DtGenericObjectEncoder_H_**
- #define **DtGenericObjectPublisher_H_**
- #define **DtGenericStateRepository_H_**
- #define **DtGlobalObjectDesignator_H_**
- typedef DtObjectId **DtGlobalObjectDesignator**
- typedef DtString **DtGlobalObjectDesignator**
- #define **DtGridAxisRecordFixed_H_**
- #define **DtGridAxisRecordVariable_H_**
- #define **DtGriddedDataPdu_H_**
- #define **DtGridDataRecord_H_**
- #define **DtGridDataRecordType0_H_**
- #define **DtGridDataRecordType1_H_**
- #define **DtGridDataRecordType2_H_**
- #define **DtAcknowledgeDecoder_H_**
- #define **DtDECLARE_ACK_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_ACK_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtDEFINE_SIMPLE_ACK_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtAcknowledgeEncoder_H_**
- #define **DtDECLARE_ACK_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_ACK_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtAcknowledgeInteraction_H_**

class DtAcknowledgeInteraction: Instances of DtAcknowledgeInteraction are used to represent Acknowledge Interactions in the RPR ROM

typedef void **(*DtAcknowledgeInteractionCb)**(DtAcknowledgeInteraction* inter, void* usr)
Acknowledge interaction callback function signature

#define **DtActionRequestDecoder_H_**

#define **DtDECLARE_ACTREQ_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_ACTREQ_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_ACTREQ_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtActionRequestEncoder_H_**

#define **DtDECLARE_ACTREQ_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_ACTREQ_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtActionRequestInteraction_H_**

typedef void **(*DtActionRequestInteractionCb)**(DtActionRequestInteraction* inter, void* usr)
ActionRequest interaction callback function signature

#define **DtActionRequestToObjDecoder_H_**

#define **DtDECLARE_ACTREQOBJ_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_ACTREQOBJ_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_ACTREQOBJ_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtActionRequestToObjEncoder_H_**

#define **DtDECLARE_ACTREQOBJ_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_ACTREQOBJ_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtActionRequestToObjInter_H_**

class DtActionRequestToObjInter: Instances of DtActionRequestToObjInter are used to represent ActionRequestToObject Interactions in the RPR FOM

typedef void **(*DtActionRequestToObjInterCb)**(DtActionRequestToObjInter* inter, void* usr)
ActionRequestToObject interaction callback function signature

#define **DtActionResponseDecoder_H_**

#define **DtDECLARE_ACTRESP_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_ACTRESP_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_ACTRESP_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtActionResponseEncoder_H_**

#define **DtDECLARE_ACTRESP_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_ACTRESP_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtActionResponseInteraction_H_**

typedef void **(*DtActionResponseInteractionCb)**(DtActionResponseInteraction* inter, void* usr)
ActionResponse interaction callback function signature

#define **DtActionResponseFromObjDecoder_H_**

#define **DtDECLARE_ACTRESPOBJ_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_ACTRESPOBJ_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_ACTRESPOBJ_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

- #define **DtActionResponseFromObjEncoder_H_**
- #define **DtDECLARE_ACTRSPOBJ_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_ACTRSPOBJ_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtActionResponseFromObjInter_H_**
- *class DtActionResponseFromObjInter: Instances of DtActionResponseFromObjInter are used to represent ActionResponseFromObject Interactions in the RPR FOM*
- typedef void **(*DtActionResponseFromObjInterCb)**(DtActionResponseFromObjInter* inter, void* usr)
  ActionResponseFromObject interaction callback function signature
- inline void DtActionResponseFromObjInter:: **setActionResult**(DtResponseFlag response)
- #define **DtAggregatePublisher_H_**
- #define **DtAppSpecificRadioSignalDecoder_H_**
- #define **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER**(paramName, \ netType, mutator)
- #define **DtDEFINE_SIMPLE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtAppSpecificRadioSignalEncoder_H_**
- #define **DtDECLARE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_APP_SPECIFIC_RADIO_SIGNAL_PARAM_ENCODER**(paramName, \ netType, inspector)
- #define **DtAttributeChangeReqDec_H_**
- #define **DtDECLARE_ATTCHGREQ_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_ATTCHGREQ_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtAttributeChangeReqEnc_H_**
- #define **DtDECLARE_ATTCHGREQ_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_ATTCHGREQ_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtAttributeChangeReqInter_H_**
- *class DtAttributeChangeReqInter: Instances of DtAttributeChangeReqInter are used to represent AttributeChangeRequest Interactions in the RPR FOM*
- typedef void **(*DtAttributeChangeReqInterCb)**(DtAttributeChangeReqInter* inter, void* usr)
  AttributeChangeRequest interaction callback function signature
- #define **DtAttributeChangeResDec_H_**
- #define **DtDECLARE_ATTCHGRES_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_ATTCHGRES_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtDEFINE_SIMPLE_ATTCHGRES_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtAttributeChangeResEnc_H_**
- #define **DtDECLARE_ATTCHGRES_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_ATTCHGRES_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtAttributeChangeResInter_H_**
- *class DtAttributeChangeResInter: Instances of DtAttributeChangeResInter are used to represent AttributeChangeResult Interactions in the RPR FOM*
- typedef void **(*DtAttributeChangeResInterCb)**(DtAttributeChangeResInter* inter, void* usr)
  AttributeChangeResult interaction callback function signature

- #define **DtCollisionDecoder_H_**
- #define **DtDECLARE_COLLISION_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_COLLISION_PARAM_DECODER**(paramName, netType, mutator)
- #define **DtDEFINE_SIMPLE_COLLISION_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtCollisionEncoder_H_**
- #define **DtDECLARE_COLLISION_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_COLLISION_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtCollisionInteraction_H_**
- *class DtCollisionInteraction: Instances of DtCollisionInteraction are used to represent Collision Interactions in the RPR FOM*
- typedef void **(*DtCollisionInteractionCb)**(DtCollisionInteraction* inter, void* usr)
  Collision interaction callback function signature
- #define **DtCommentDecoder_H_**
- #define **DtDECLARE_COMMENT_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_COMMENT_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtCommentEncoder_H_**
- #define **DtDECLARE_COMMENT_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_COMMENT_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtCommentInteraction_H_**
- typedef void **(*DtCommentInteractionCb)**(DtCommentInteraction* inter, void* usr)
  Comment interaction callback function signature
- #define **DtCreateEntityDecoder_H_**
- #define **DtDECLARE_CREATENT_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_CREATENT_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtDEFINE_SIMPLE_CREATENT_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtCreateEntityEncoder_H_**
- #define **DtDECLARE_CREATENT_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_CREATENT_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtCreateEntityInteraction_H_**
- *class DtCreateEntityInteraction: Instances of DtCreateEntityInteraction are used to represent CreateEntity Interactions in the RPR FOM*
- typedef void **(*DtCreateEntityInteractionCb)**(DtCreateEntityInteraction* inter, void* usr)
  CreateEntity interaction callback function signature
- #define **DtCreateObjReqDecoder_H_**
- #define **DtDECLARE_CRTOBJREQ_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_CRTOBJREQ_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtCreateObjReqEncoder_H_**
- #define **DtDECLARE_CRTOBJREQ_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_CRTOBJREQ_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtCreateObjReqInter_H_**

class DtCreateObjReqInter: Instances of DtCreateObjReqInter are used to represent CreateObjectRequest Interactions in the RPR FOM

typedef void **(*DtCreateObjReqInterCb)**(DtCreateObjReqInter* inter, void* usr)
CreateObjectRequest interaction callback function signature

#define **DtCreateObjectResultDecoder_H_**

#define **DtDECLARE_CREATOBJRES_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_CREATOBJRES_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_CREATOBJRES_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtCreateObjectResultEncoder_H_**

#define **DtDECLARE_CRTOBJRES_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_CRTOBJRES_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtCreateObjectResultInter_H_**

class DtCreateObjectResultInter: Instances of DtCreateObjectResultInter are used to represent CreateObjectResult Interactions in the RPR FOM

typedef void **(*DtCreateObjectResultInterCb)**(DtCreateObjectResultInter* inter, void* usr)
CreateObjectResult interaction callback function signature

inline DtCreateObjectResult DtCreateObjectResultInter:: **createObjectResult**() const

#define **DtDataDecoder_H_**

#define **DtDECLARE_DATA_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_DATA_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_DATA_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtDataEncoder_H_**

#define **DtDECLARE_DATA_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_DATA_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtDataInteraction_H_**

typedef void **(*DtDataInteractionCb)**(DtDataInteraction* inter, void* usr)
Data interaction callback function signature

#define **DtDataQueryDecoder_H_**

#define **DtDECLARE_DATAQ_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_DATAQ_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_DATAQ_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtDataQueryEncoder_H_**

#define **DtDECLARE_DATAQ_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_DATAQ_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtDataQueryInteraction_H_**

class DtDataQueryInteraction: Instances of DtDataQueryInteraction are used to represent DataQuery Interactions in the RPR FOM

typedef void **(*DtDataQueryInteractionCb)**(DtDataQueryInteraction* inter, void* usr)
DataQuery interaction callback function signature

#define **DtDatabaseIndexRadioSignalDecoder_H_**

- #define **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_DATABASE_RADIO_SIGNAL_PARAM_DECODER**(paramName, \ netType, mutator)
- #define **DtDEFINE_SIMPLE_DATABASE_RADIO_SIGNAL_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtDatabaseIndexRadioSignalEncoder_H_**
- #define **DtDECLARE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_DATABASE_RADIO_SIGNAL_PARAM_ENCODER**(paramName, \ netType, inspector)
- #define **DtDesignatorPublisher_H_**
- #define **DtDetonationDecoder_H_**
- #define **DtDECLARE_DETONATION_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_DETONATION_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtDEFINE_SIMPLE_DETONATION_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtDetonationEncoder_H_**
- #define **DtDECLARE_DETONATION_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_DETONATION_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtDetonationInteraction_H_**
- *class DtDetonationInteraction: Instances of DtDetonationInteraction are used to represent Detonation Interactions in the RPR ROM*
- typedef void **(*DtDetonationInteractionCb)**(DtDetonationInteraction* inter, void* usr)
  Detonation interaction callback function signature
- #define **DtEmitterBeamPublisher_H_**
- #define **DtEmitterSystemPublisher_H_**
- #define **DtEncodedAudioRadioSignalDecoder_H_**
- #define **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER**(paramName, \ netType, mutator)
- #define **DtDEFINE_SIMPLE_ENC_AUDIO_RADIO_SIGNAL_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtEncodedAudioRadioSignalEncoder_H_**
- #define **DtDECLARE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_ENC_AUDIO_RADIO_SIGNAL_PARAM_ENCODER**(paramName, \ netType, inspector)
- #define **DtEntityPublisher_H_**
- static inline const DtEntityType& **gGuiseSameAsType**()
- #define **gGuiseSameAsType**()
- ifdef __GNUC__ inline const DtEntityType& **gGuiseSameAsType**()
- #define **DtEventReportDecoder_H_**
- #define **DtDECLARE_EVENTREP_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_EVENTREP_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtDEFINE_SIMPLE_EVENTREP_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtEventReportEncoder_H_**
- #define **DtDECLARE_EVENTREP_PARAM_ENCODER**(paramName)

- #define **DtDEFINE_SIMPLE_EVENTREP_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtEventReportInteraction_H_**
- typedef void **(*DtEventReportInteractionCb)**(DtEventReportInteraction* inter, void* usr)
  EventReport interaction callback function signature
- #define **DtExerciseConn_H_**
- typedef DtVrlFederateAmbassador* **(*DtFedAmbCreator)**()
- typedef DtVrlRtiAmbassador* **(*DtRtiAmbCreator)**()
- typedef DtFomMapper* **(*DtFomMapperCreator)**(void* usr)
- typedef void **(*DtFomMapperDeletor)**(DtFomMapper*)
- #define **DtFireDecoder_H_**
- #define **DtDECLARE_FIRE_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_FIRE_PARAM_DECODER**(paramName, netType, mutator)
- #define **DtDEFINE_SIMPLE_FIRE_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtFireEncoder_H_**
- #define **DtDECLARE_FIRE_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_FIRE_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtFireInteraction_H_**
- *class DtFireInteraction: Instances of DtFireInteraction are used to represent Fire Interactions in the RPR FOM*
- typedef void **(*DtFireInteractionCb)**(DtFireInteraction* inter, void* usr)
  Fire interaction callback function signature
- #define **DtHInteraction_H_**
- #define **DtInteractionFactory_H_**
- #define **DtInteractionWithData_H_**
- #define **DtInteractionWithEncDec_H_**
- #define **DtHlaObject_H_**
- #define **DtHlaObjectFactory_H_**
- #define **DtHlaObjectWithStateRep_H_**
- #define **DtLgrControlDecoder_H_**
- #define **DtDECLARE_LGR_CONTROL_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtLgrControlEncoder_H_**
- #define **DtDECLARE_LGR_CONTROL_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_LGR_CONTROL_PARAM_ENCODER**(paramName, netType, inspector)
- #define **DtLgrControlInteraction_H_**
- *class DtLgrControlInteraction: Instances of DtLgrControlInteraction are used to represent LgrControl Interactions in HLA*
- typedef void **(*DtLgrControlInteractionCb)**(DtLgrControlInteraction* inter, void* usr)
  LgrControl interaction callback function signature
- #define **DtObjectPublisher_H_**
- #define **_HOSTSTRUCTS_H_**
- typedef DtU16 **DtExerciseId**
- DtU16 **DtHost**
- **DtSimulationAddress**

- if i386 const DtU16 **DtDeferedEntityId**
- const DtU16 **DtDeferedEntityId**
- *CJW: THIS IS A BUG*
- #define **DtAny**
- #define **DtRadioReceiverPublisher_H_**
- #define **DtRadioTransmitterPublisher_H_**
- #define **DtRawBinaryRadioSignalDecoder_H_**
- #define **DtDECLARE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER**(paramName, \ netType, mutator)
- #define **DtDEFINE_SIMPLE_RAW_BIN_RADIO_SIGNAL_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)
- #define **DtRawBinaryRadioSignalEncoder_H_**
- #define **DtDECLARE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER**(paramName)
- #define **DtDEFINE_SIMPLE_RAW_BINARY_RADIO_SIGNAL_PARAM_ENCODER**(paramName, \ netType, inspector)
- #define **DtReflectedEmitterBeamList_H_**
- #define **DtReflectedDesignatorList_H_**
- #define **DtReflectedEmitterSystemList_H_**
- #define **DtReflectedAggregateList_H_**
- #define **DtReflectedEmitterBeam_H_**
- typedef void **(*DtEmitterBeamCallbackFunction)**(DtReflectedEmitterBeam* obj, void* userData)
- #define **DtReflectedDesignator_H_**
- typedef void **(*DtDesignatorCallbackFunction)**(DtReflectedDesignator* obj, void* userData)
- #define **DtReflectedAggregate_H_**
- typedef void **(*DtAggregateCallbackFunction)**(DtReflectedAggregate* obj, void* userData)
- #define **DtReflectedEntity_H_**
- typedef void **(*DtEntityCallbackFunction)**(DtReflectedEntity* obj, void* userData)
- #define **DtReflectedObject_H_**
- #define **DtReflectedEmitterSystem_H_**
- typedef void **(*DtEmitterSystemCallbackFunction)**(DtReflectedEmitterSystem* obj, void* userData)
- #define **DtReflectedEntityList_H_**
- #define **DtReflectedRadioReceiver_H_**
- typedef void **(*DtRadioReceiverCallbackFunction)**(DtReflectedRadioReceiver* obj, void* userData)
- #define **DtReflectedRadioTransmitter_H_**
- typedef void **(*DtRadioTransmitterCallbackFunction)**(DtReflectedRadioTransmitter* obj, void* userData)
- #define **DtReflectedObjectList_H_**
- #define **DtReflectedRadioReceiverList_H_**
- #define **DtReflectedRadioTransmitterList_H_**
- #define **DtRemoveObjectRequestDecoder_H_**
- #define **DtDECLARE_REMOBJREQ_PARAM_DECODER**(paramName)
  This can be used to declare parameter processor functions which are static members of a class
- #define **DtDEFINE_SIMPLE_REMOBJREQ_PARAM_DECODER**(\ paramName, netType, mutator)
- #define **DtDEFINE_SIMPLE_REMOBJREQ_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtRemoveObjectRequestEncoder_H_**

#define **DtDECLARE_REMOBJREQ_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_REMOBJREQ_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtRemoveObjectRequestInter_H_**

class DtRemoveObjectRequestInter: Instances of DtRemoveObjectRequestInter are used to represent RemoveObjectRequest Interactions in the RPR FOM

typedef void **(*DtRemoveObjectRequestInterCb)**(DtRemoveObjectRequestInter* inter, void* usr)
RemoveObjectRequest interaction callback function signature

#define **DtRemoveObjectResultDecoder_H_**

#define **DtDECLARE_REMOBJRES_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_REMOBJRES_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_REMOBJRES_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtRemoveObjectResultEncoder_H_**

#define **DtDECLARE_REMOBJRES_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_REMOBJRES_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtRemoveObjectResultInter_H_**

class DtRemoveObjectResultInter: Instances of DtRemoveObjectResultInter are used to represent RemoveObjectResult Interactions in the RPR FOM

typedef void **(*DtRemoveObjectResultInterCb)**(DtRemoveObjectResultInter* inter, void* usr)
RemoveObjectResult interaction callback function signature

inline DtRemoveObjectResult DtRemoveObjectResultInter:: **removeObjectResult**() const

#define **DtRemoveEntityDecoder_H_**

#define **DtDECLARE_REMOVENT_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_REMOVENT_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_REMOVENT_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtRemoveEntityEncoder_H_**

#define **DtDECLARE_REMOVENT_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_REMOVENT_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtRemoveEntityInteraction_H_**

class DtRemoveEntityInteraction: Instances of DtRemoveEntityInteraction are used to represent RemoveEntity Interactions in the RPR FOM

typedef void **(*DtRemoveEntityInteractionCb)**(DtRemoveEntityInteraction* inter, void* usr)
RemoveEntity interaction callback function signature

#define **DtRepairCompleteDecoder_H_**

#define **DtDECLARE_REPCOMP_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_REPCOMP_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_REPCOMP_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtRepairCompleteEncoder_H_**

#define **DtDECLARE_REPCOMP_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_REPCOMP_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtRepairCompleteInteraction_H_**

class DtRepairCompleteInteraction: Instances of DtRepairCompleteInteraction are used to represent RepairComplete Interactions in the RPR FOM

typedef void **(*DtRepairCompleteInteractionCb)**(DtRepairCompleteInteraction* inter, void* usr)
RepairComplete interaction callback function signature

inline const DtGlobalObjectDesignator& DtRepairCompleteInteraction:: **receivingId**() const

inline const DtGlobalObjectDesignator& DtRepairCompleteInteraction:: **repairingId**() const

#define **DtRepairResponseDecoder_H_**

#define **DtDECLARE_REPRESP_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_REPRESP_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_REPRESP_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtRepairResponseEncoder_H_**

#define **DtDECLARE_REPRESP_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_REPRESP_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtRepairResponseInteraction_H_**

class DtRepairResponseInteraction: Instances of DtRepairResponseInteraction are used to represent RepairResponse Interactions in the RPR FOM

typedef void **(*DtRepairResponseInteractionCb)**(DtRepairResponseInteraction* inter, void* usr)
RepairResponse interaction callback function signature

inline const DtGlobalObjectDesignator& DtRepairResponseInteraction:: **receivingId**() const

inline const DtGlobalObjectDesignator& DtRepairResponseInteraction:: **repairingId**() const

#define **DtResupplyCancelDecoder_H_**

#define **DtDECLARE_RESCANC_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_RESCANC_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_RESCANC_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtResupplyCancelEncoder_H_**

#define **DtDECLARE_RESCANC_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_RESCANC_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtResupplyCancelInteraction_H_**

class DtResupplyCancelInteraction: Instances of DtResupplyCancelInteraction are used to represent ResupplyCancel Interactions in the RPR FOM

typedef void **(*DtResupplyCancelInteractionCb)**(DtResupplyCancelInteraction* inter, void* usr)
ResupplyCancel interaction callback function signature

inline const DtGlobalObjectDesignator& DtResupplyCancelInteraction:: **receivingId**() const

inline const DtGlobalObjectDesignator& DtResupplyCancelInteraction:: **supplyingId**() const

#define **DtResupplyOfferDecoder_H_**

#define **DtDECLARE_RESOFF_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_RESOFF_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_RESOFF_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtResupplyOfferEncoder_H_**

#define **DtDECLARE_RESOFF_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_RESOFF_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtResupplyOfferInteraction_H_**

#define **DtSUPPLY_VALID**

#define **DtSUPPLY_BAD_INDEX**

class DtResupplyOfferInteraction: Instances of DtResupplyOfferInteraction are used to represent ResupplyOffer Interactions in the RPR FOM

typedef void **(*DtResupplyOfferInteractionCb)**(DtResupplyOfferInteraction* inter, void* usr)
ResupplyOffer interaction callback function signature

inline const DtGlobalObjectDesignator& DtResupplyOfferInteraction:: **receivingId**() const

inline const DtGlobalObjectDesignator& DtResupplyOfferInteraction:: **supplyingId**() const

#define **DtResupplyReceivedDecoder_H_**

#define **DtDECLARE_RESREC_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_RESREC_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_RESREC_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtResupplyReceivedEncoder_H_**

#define **DtDECLARE_RESREC_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_RESREC_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtResupplyRecInteraction_H_**

#define **DtSUPPLY_VALID**

#define **DtSUPPLY_BAD_INDEX**

class DtResupplyRecInteraction: Instances of DtResupplyRecInteraction are used to represent ResupplyReceived Interactions in the RPR FOM

typedef void **(*DtResupplyRecInteractionCb)**(DtResupplyRecInteraction* inter, void* usr)
ResupplyReceived interaction callback function signature

inline const DtGlobalObjectDesignator& DtResupplyRecInteraction:: **receivingId**() const

inline const DtGlobalObjectDesignator& DtResupplyRecInteraction:: **supplyingId**() const

#define **DtServiceRequestDecoder_H_**

#define **DtDECLARE_SERVREQ_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_SERVREQ_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_SERVREQ_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtServiceRequestEncoder_H_**

#define **DtDECLARE_SERVREQ_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_SERVREQ_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtServiceRequestInteraction_H_**

#define **DtSUPPLY_VALID**

#define **DtSUPPLY_BAD_INDEX**

class DtServiceRequestInteraction: Instances of DtServiceRequestInteraction are used to represent ServiceRequest Interactions in the RPR FOM

typedef void **(*DtServiceRequestInteractionCb)**(DtServiceRequestInteraction* inter, void* usr)
ServiceRequest interaction callback function signature

inline const DtGlobalObjectDesignator& DtServiceRequestInteraction:: **requestingId**() const

inline const DtGlobalObjectDesignator& DtServiceRequestInteraction:: **servicingId**() const

#define **DtSetDataDecoder_H_**

#define **DtDECLARE_SETDATA_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_SETDATA_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_SETDATA_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtSetDataEncoder_H_**

#define **DtDECLARE_SETDATA_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_SETDATA_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtSetDataInteraction_H_**

typedef void **(*DtSetDataInteractionCb)**(DtSetDataInteraction* inter, void* usr)
SetData interaction callback function signature

#define **DtSignalInteraction_H_**

class DtSignalInteraction: Instances of DtSignalInteraction are

typedef void **(*DtSignalInteractionCb)**(DtSignalInteraction* inter, void* usr)
Collision interaction callback function signature

#define **DtStartResumeDecoder_H_**

#define **DtDECLARE_START_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_START_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_START_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtStartResumeEncoder_H_**

#define **DtDECLARE_START_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_START_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtStartResumeInteraction_H_**

class DtStartResumeInteraction: Instances of DtStartResumeInteraction are used to represent StartResume Interactions in the RPR ROM

typedef void **(*DtStartResumeInteractionCb)**(DtStartResumeInteraction* inter, void* usr)
StartResume interaction callback function signature

#define **DtHlaStateDecoder_H_**

inline DtHlaStateDecoder::DtAttributeDecoder DtHlaStateDecoder:: **decoder**(int handle) const

#define **DtADD_ATTR_DECODER**(attrName)
This macro is used to add attribute decoder functions (functions that decode individual attributes) to a DtHlaStateDecoder

#define **DtADD_ATTR_DECODER_QUIET**(attrName)
Same as above, but don't complain if name is not found

#define **DtDECLARE_ATTR_DECODER**(stateRepClass, attrName)
This macro is used to declare attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes) to a DtHlaStateDecoder

#define **DtDEFINE_SIMPLE_ATTR_DECODER_WITH_CAST**(\ decClass, stateRepClass, attrName, netType, mutator, castExpr)

#define **DtDEFINE_SIMPLE_ATTR_DECODER**(\ /// decClass, stateRepClass, attrName, netType, mutator)

#define **DtHlaStateEncoder_H_**

#define **DtADD_ATTR_ENCODER**(attrName)
This macro is used to add attribute encoder functions (functions that encode individual attributes) to a DtHlaStateEncoder

#define **DtADD_ATTR_ENCODER_QUIET**(attrName)

#define **DtADD_ATTR_CHECKER**(attrName)

#define **DtADD_ATTR_CHECKER_QUIET**(attrName)

#define **DtDECLARE_ATTR_ENCODER**(stateRepClass, attrName)

#define **DtDECLARE_ATTR_CHECKER**(stateRepClass, attrName)

#define **DtDEFINE_SIMPLE_ATTR_ENCODER**(\ encClass, stateRepClass, attrName, netType, inspector)

#define **hDtStateMsg_H_**

#define **ObjectHandle**

#define **DtStopFreezeDecoder_H_**

#define **DtDECLARE_STOP_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtDEFINE_SIMPLE_STOP_PARAM_DECODER**(\ paramName, netType, mutator)

#define **DtDEFINE_SIMPLE_STOP_PARAM_DECODER_WITH_CAST**(\ paramName, netType, mutator, castExpr)

#define **DtStopFreezeEncoder_H_**

#define **DtDECLARE_STOP_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_STOP_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtStopFreezeInteraction_H_**

class DtStopFreezeInteraction: Instances of DtStopFreezeInteraction are used to represent StopFreeze Interactions in the RPR ROM

typedef void **(*DtStopFreezeInteractionCb)**(DtStopFreezeInteraction* inter, void* usr)
StopFreeze interaction callback function signature

#define **DtUnknownInteraction_H_**

class DtUnknownInteraction: Instances of DtUnknownInteraction are used to represent Interactions for which there is no derived DtInteraction

typedef void **(*DtUnknownInteractionCb)**(DtUnknownInteraction* inter, void* usr)
Unknown interaction callback function signature

#define **DtViewControlDecoder_H_**

#define **DtDECLARE_VIEW_CONTROL_PARAM_DECODER**(paramName)
This can be used to declare parameter processor functions which are static members of a class

#define **DtViewControlEncoder_H_**

#define **DtDECLARE_VIEW_CONTROL_PARAM_ENCODER**(paramName)

#define **DtDEFINE_SIMPLE_VIEW_CONTROL_PARAM_ENCODER**(paramName, netType, inspector)

#define **DtViewControlInteraction_H_**

class DtViewControlInteraction: Instances of DtViewControlInteraction are used to represent ViewControl Interactions in the RPR FOM

typedef void **(*DtViewControlInteractionCb)**(DtViewControlInteraction* inter, void* usr)
ViewControl interaction callback function signature

#define **DtIffAtcNavaidsLayerHeader_H_**

#define **DtIffAtcNavaidsOperational_H_**

#define **DtIffAtcNavaidsSecondOperational_H_**

#define **DtIffAtcNavaidsParameter_H_**

#define **DtIffAtcNavaidsPdu_H_**

#define **DtIffAtcNavaidsSystemID_H_**

#define **DtInterClassDesc_H_**

#define **DtInteractionDecoderFactory_H_**

#define **DtInteractionEncoderFactory_H_**

#define **DtIntercomControlPdu_H_**

#define **DtIntercomEntityDestRec_H_**

#define **DtIntercomGroupAsgmtRec_H_**

#define **DtIntercomGroupDestRec_H_**

#define **DtIntercomParmRec_H_**

#define **DtIntercomSignalPdu_H_**

#define **DtInteractionDecoder_H_**

#define **DtADD_PARAM_DECODER**(attrName)
This can be used to add a parameter decoder function from a routine

#define **DtDECLARE_PARAM_DECODER**(RepT, paramName)

#define **DtDEFINE_SIMPLE_PARAM_DECODER_WITH_CAST**(decClass, interClass, \
paramName, netType, mutator, castExpr)

#define **DtDEFINE_SIMPLE_PARAM_DECODER**(decClass, interClass, \ /// paramName, netType, mutator)

#define **DtInteractionEncoder_H_**

define DtADD_PARAM_ENCODER(attrName)(`attrName, (DtParameterEncoder) encodeattrName)(interClass, paramName)` **(const interClass& inter, RTI::ParameterHandleValuePairSet* params, RTI::ParameterHandle paramHandle)**(encClass, interClass, paramName, netType, inspector)

#define **DtInteractionManager_H_**

#define **DtIsGroupOfPdu_H_**

#define **DtIsPartOfPdu_H_**

#define **DtJammerBeamDecoder_H_**

#define **DtDECLARE_JAMMERBEAM_ATTR_DECODER**(attrName)
This is used to declare attribute decoding functions which are members of the class

#define **DtDEFINE_SIMPLE_BASE_JAMMERBEAM_ATTR_DECODER**(\ attrName, netType, inspector)

#define **DtJammerBeamEncoder_H_**

#define **DtDECLARE_JAMMERBEAM_ATTR_CHECKER**(attrName)
This is used to declare static member functions for determining whether or not an attribute needs to be encoded

#define **DtDECLARE_JAMMERBEAM_ATTR_ENCODER**(attrName)

#define **DtDEFINE_SIMPLE_JAMMERBEAM_ATTR_CHECKER**(\ attrName, inspector)

#define **DtLaserPdu_H_**

#define **LGR_CONTROL_INTERACTION_H_**

#define **DtLgrControlPdu_H_**
This file provides protocol independence for LgrControl Interactions

#define **DtLeAppearancePdu_H_**

#define **DtLeArticulatedPartsPdu_H_**

#define **DtLeDeadReckoningParameters_H_**

#define **DtLeDetonationPdu_H_**

#define **DtLeEntityID_H_**

#define **DtLeEntityOrientation_H_**

- #define **DtLeEntityLinearVelocity_H_**
- #define **DtLeFirePdu_H_**
- #define **DtLeEntityLocationEntityCoordinates_H_**
- #define **DtLeEntityLocationWorldCoordinates_H_**
- #define **DtLeOrientationError_H_**
- #define **DtLePositionError_H_**
- #define **DtLeTspiPdu_H_**
- #define **DtLinearObjectStatePdu_H_**
- #define **DtLinearSegmentParameter_H_**
- #define **DtLocalHlaObjectManager_H_**
- #define **DtMessagePdu_H_**
- #define **DtMilitaryEntityDecoder_H_**
- #define **DtDEFINE_SIMPLE_MIL_ENTITY_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtMilitaryEntityEncoder_H_**
- #define **DtDEFINE_SIMPLE_MIL_ENTITY_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtMilitaryPlatformEntityDecoder_H_**
- #define **DtDEFINE_SIMPLE_MIL_PLAT_ENTITY_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtMilitaryPlatformEntityEncoder_H_**
- #define **DtDEFINE_SIMPLE_MIL_PLAT_ENTITY_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtMinefieldDataPdu_H_**
- #define **DtMinefieldPerimeterRecord_H_**
- #define **DtMinefieldQueryPdu_H_**
- #define **DtMinefieldResponseNackPdu_H_**
- #define **DtMinefieldStatePdu_H_**
- #define **DtMunitionEntityDecoder_H_**
- #define **DtDEFINE_SIMPLE_MUN_ENTITY_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtMunitionEntityEncoder_H_**
- #define **DtDEFINE_SIMPLE_MUN_ENTITY_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtNetBoolean_H_**
- typedef DtNetU32 **DtNetBoolean**
- #define **_NETSTRUCTS_H_**

For backwards compatibility

- typedef DtNetU8 **DtNet8Bits**
- typedef DtNetU8 **DtNet16Bits**[2]
- typedef DtNetU8 **DtNet32Bits**[4]
- typedef DtNetU8 **DtNet64Bits**[8]
- typedef DtNetU32 **DtNetObjectId**
- typedef DtNetObjectId **DtNetGlobalObjectDesignator**
- typedef DtVector **DtNet64Vector**
- typedef struct DtNetRadioEntityType DtNetU8 **kind**
- DtNetU8 **domain**
- DtNetU16 **country**
- DtNetU8 **category**
- DtNetU8 **nomenclatureVersion**

DtNetU16 **nomenclature**

DtNetRadioEntityType

typedef struct DtNetEntityType DtNetU8 **DtEntityKind**

DtNetU8 **DtDomain**

DtNetU16 **DtCountry**

DtNetU8 **DtCategory**

DtNetU8 **DtSubCategory**

DtNetU8 **DtSpecific**

DtNetU8 **DtExtra**

DtNetEntityType

typedef struct DtNetBurstDescriptor DtNetEntityType **DtMunition**

DtNetU16 **DtWarhead**

DtNetU16 **DtFuze**

DtNetU16 **DtQuantity**

DtNetU16 **DtRate**

DtNetBurstDescriptor

typedef struct DtNetSimulationAddress DtNetU16 **DtSite**

DtNetU16 **DtHost**

DtNetSimulationAddress

typedef struct DtNetEntityIdentifier DtNetSimulationAddress **DtSimulator**

DtNetU16 **DtEntity**

DtNetEntityIdentifier

typedef struct DtNetEventId DtNetSimulationAddress **DtSimulator**

DtNetU16 **DtEvent**

DtNetEventId

typedef struct DtNetEulerAngles DtNetFloat32 **psi**

DtNetFloat32 **theta**

DtNetFloat32 **phi**

DtNetEulerAngles

typedef DtNetFloat32 **DtNet32Vector**[3]

typedef DtNetFloat64 **DtNet64Vector**[3]

typedef struct DtNetArtParamRecord DtNetEnum **partType**

DtNetU8 **change**

DtNetU16 **partAttachedTo**

DtNetU32 **paramType**

DtNetFloat32 **value**

DtNetArtParamRecord

typedef struct DtNetArtParamRecord DtNetU8 **artOrAttached**

DtNetU8 **DtChange**

DtNetU16 **DtChange**

DtNetU16 **DtAttachedTo**

DtNetU32 **DtType**

DtNet64Bits **DtValue**

DtNetArtParamRecord

typedef DtNetArtParamRecord **DtArticulatedParts**

typedef DtNetArtParamRecord	**DtNetArtPartRecord**
typedef struct DtNetAttachedPart DtNetU8	**myArtOrAttached**
DtNetU8	**myChange**
DtNetU16	**myAttachedTo**
DtNetU32	**myStation**
DtNetEntityType	**myType**
	DtNetAttachedPart
typedef struct DtNetAngularVelocity DtNetFloat32	**wx**
DtNetFloat32	**wy**
DtNetFloat32	**wz**
	DtNetAngularVelocity
typedef struct DtNetLinearVelocity DtNetFloat32	**DtX**
DtNetFloat32	**DtY**
DtNetFloat32	**DtZ**
	DtNetLinearVelocity
typedef struct DtNetWorldCoordinates DtNetFloat64	**DtX**
DtNetFloat64	**DtY**
DtNetFloat64	**DtZ**
	DtNetWorldCoordinates
typedef struct DtNetEntityCoordinates DtNetFloat32	**DtX**
DtNetFloat32	**DtY**
DtNetFloat32	**DtZ**
	DtNetEntityCoordinates
typedef struct DtNetTopoUpCoord DtNetFloat32	**East**
DtNetFloat32	**North**
DtNetFloat32	**Up**
	DtNetTopoUpCoord
typedef struct DtNetInt32	**hour**
DtNetU32	**hourFrac**
	DtNetClockTime
DtTime	**DtClockTime2AbsTime**(const DtNetClockTime *clock)
void	**DtAbsTime2ClockTime**(DtTime absTime, DtNetClockTime *clock)
#define	**DtObjClassDesc_H_**
#define	**DtObjectType_H_**
#define	**DtObjectReflectAttributeValuesCbInfo_H_**
typedef void	**(*DtObjectReflectAttributeValuesCb)**(const DtStateMsg& stateMsg, DtHlaObject* obj, void* usr) *Object reflect attribute values callback signature*
#define	**_parseEnum_h_**
char*	**DtEnumString**(DtDetonatorFuze param)
char*	**DtEnumString**(DtWarheadType param)
char*	**DtEnumString**(DtRepairType param)
char*	**DtEnumString**(DtCollisionType param)
char*	**DtEnumString**(DtForceType param)
char*	**DtEnumString**(DtDetonationResult param)
char*	**DtEnumString**(DtProtocolFamily param)

- char* **DtEnumString**(DtRepairResult param)
- char* **DtEnumString**(DtServiceType param)
- char* **DtEnumString**(DtStopReason param)
- char* **DtEnumString**(DtAcknowledgeFlag param)
- char* **DtEnumString**(DtResponseFlag param)
- char* **DtEnumString**(DtActionId param)
- char* **DtEnumString**(DtRequestStatus param)
- char* **DtEnumString**(DtEventType param)
- char* **DtEnumString**(DtDatumParam param)
- char* **DtEnumString**(DtEmissionPduType param)
- char* **DtEnumString**(DtEmitterSysName param)
- char* **DtEnumString**(DtEmissionFunc param)
- char* **DtEnumString**(DtBeamFunc param)
- char* **DtEnumString**(DtTrackingJamming param)
- char* **DtEnumString**(DtReferenceSys param)
- char* **DtEnumString**(DtDesignatorCodeName param)
- char* **DtEnumString**(DtDesignatorCode param)
- char* **DtEnumString**(DtRadioTransmitState param)
- char* **DtEnumString**(DtRadioInputSource param)
- char* **DtEnumString**(DtAntennaPatternType param)
- char* **DtEnumString**(DtRadioSpreadSpectrum param)
- char* **DtEnumString**(DtRadioMajorModulation param)
- char* **DtEnumString**(DtRadioDetailedModulationAmplitude param)
- char* **DtEnumString**(DtDetailedModulationAmplitudeAngle param)
- char* **DtEnumString**(DtDetailedModulationAngle param)
- char* **DtEnumString**(DtDetailedModulationCombination param)
- char* **DtEnumString**(DtDetailedModulationUnmodulated param)
- char* **DtEnumString**(DtDetailedModulationType param)
- char* **DtEnumString**(DtRadioSystem param)
- char* **DtEnumString**(DtCryptoMode param)
- char* **DtEnumString**(DtRadioCryptoSystem param)
- char* **DtEnumString**(DtEncodingClass param)
- char* **DtEnumString**(DtRadioReceiverState param)
- char* **DtEnumString**(DtTacticalDataLinkType param)
- char* **DtEnumString**(DtUserProtocol param)
- char* **DtEnumString**(DtEncodingType param)
- char* **DtEnumString**(DtCreateObjectResult param)
- char* **DtEnumString**(DtRemoveObjectResult param)
- char* **DtEnumString**(DtAttributeChangeResult param)
- char* **DtEnumString**(DtDeadReckonTypes param)
- char* **DtEnumString**(DtViewerCommand param)
- char* **DtEnumString**(DtLgrCommand param)
- char* **DtEnumString**(DtAppearAttributeBitShift param)
- char* **DtEnumString**(DtAppearAttribute param)

- char* **DtEnumString**(DtAppearAttrVal param)
- char* **DtEnumString**(DtDamageState param)
- char* **DtEnumString**(DtTrailState param)
- char* **DtEnumString**(DtHatchState param)
- char* **DtEnumString**(DtLightState param)
- char* **DtEnumString**(DtLifeformState param)
- char* **DtEnumString**(DtCamouflageType param)
- char* **DtEnumString**(DtWeaponState param)
- char* **DtEnumString**(DtCapability param)
- char* **DtEnumString**(DtDensityType param)
- char* **DtEnumString**(DtArtParamType param)
- char* **DtEnumString**(DtArtPartType param)
- char* **DtEnumString**(DtIntercomCtlType param)
- char* **DtEnumString**(DtIntercomCommClass param)
- char* **DtEnumString**(DtIntercomCommType param)
- char* **DtEnumString**(DtIntercomCmdType param)
- char* **DtEnumString**(DtIntercomTransLineState param)
- char* **DtEnumString**(DtIntercomDestLineStateCmd param)
- char* **DtEnumString**(DtIntercomCtlRecType param)
- char* **DtEnumString**(DtEnvironmentalModelType param)
- char* **DtEnumString**(DtEnvironmentalStatusBit param)
- char* **DtEnumString**(DtGridDataRecType param)
- char* **DtEnumString**(DtGridAxisType param)
- char* **DtEnumString**(DtIffAtcNavaidsSysName param)
- char* **DtEnumString**(DtIffAtcNavaidsSysType param)
- char* **DtEnumString**(DtIffAtcNavaidsStatusBit param)
- char* **DtEnumString**(DtIffAtcNavaidsSysMode param)
- char* **DtEnumString**(DtIffAtcNavaidsChangeOptionsBit param)
- char* **DtEnumString**(DtIffAtcNavaidsParameterBit param)
- char* **DtEnumString**(DtIffAtcNavaidsAltParm4 param)
- char* **DtEnumString**(DtIffAtcNavaidsModifier param)
- char* **DtEnumString**(DtIffAtcNavaidsLayerSpecificInfo param)
- char* **DtEnumString**(DtIffAtcNavaidsParm1 param)
- char* **DtEnumString**(DtIffAtcNavaidsParm2 param)
- char* **DtEnumString**(DtIffAtcNavaidsApplicableModes param)
- char* **DtEnumString**(DtUnderwaterAcousticUpdate param)
- char* **DtEnumString**(DtUnderwaterAcousticSystemName param)
- char* **DtEnumString**(DtUnderwaterAcousticFunction param)
- char* **DtEnumString**(DtUnderwaterAcousticActiveEmissionParmIndex param)
- char* **DtEnumString**(DtUnderwaterAcousticScanPattern param)
- char* **DtEnumString**(DtUnderwaterAcousticPassiveParmIndex param)
- char* **DtEnumString**(DtUnderwaterAcousticPropulsionPlantConfig param)
- char* **DtEnumString**(DtUnderwaterAcousticHullMountedMaskerStatus param)
- char* **DtEnumString**(DtUnderwaterAcousticAdditionalPassiveActivityParmIndex param)

- char* **DtEnumString**(DtTransferType param)
- char* **DtEnumString**(DtRequiredReliabilityService param)
- char* **DtEnumString**(DtNature param)
- char* **DtEnumString**(DtPosition param)
- char* **DtEnumString**(DtStationName param)
- char* **DtEnumString**(DtObjectKind param)
- char* **DtEnumString**(DtModificationField param)
- char* **DtEnumString**(DtMinefieldDataFilterField param)
- char* **DtEnumString**(DtMinefieldFusing param)
- char* **DtEnumString**(DtMinefieldAntiHandlingDevice param)
- char* **DtEnumString**(DtSensorTypeCategory param)
- char* **DtEnumString**(DtOpticalSensorTypeSubcategory param)
- char* **DtEnumString**(DtFLIR_SensorTypeSubcategory param)
- char* **DtEnumString**(DtRADAR_SensorTypeSubcategory param)
- char* **DtEnumString**(DtMagneticSensorTypeSubcategory param)
- char* **DtEnumString**(DtLaserSensorTypeSubcategory param)
- char* **DtEnumString**(DtSONARSensorTypeSubcategory param)
- char* **DtEnumString**(DtPhysicalSensorTypeSubcategory param)
- char* **DtEnumString**(DtMultispectralSensorTypeSubcategory param)
- char* **DtEnumString**(DtMinefieldAlgae param)
- char* **DtEnumString**(DtMinefieldPaintScheme param)
- char* **DtEnumString**(DtMinefieldProtocol param)
- char* **DtEnumString**(DtSubProtocol param)
- char* **DtEnumString**(DtCharacterSet param)
- char* **DtEnumString**(DtLE_TSPI_FieldFlag param)
- char* **DtEnumString**(DtLE_AppearanceField1Flag param)
- char* **DtEnumString**(DtLE_AppearanceField2Flag param)
- char* **DtEnumString**(DtLE_FireFieldFlag param)
- char* **DtEnumString**(DtLE_DetonFieldFlag param)
- char* **DtEnumString**(DtLE_DetonField2Flag param)
- char* **DtEnumString**(DtGroupedEntityCategory param)
- char* **DtEnumString**(DtRestStatus param)
- char* **DtEnumString**(DtAggregateState param)
- char* **DtEnumString**(DtAggregateFormation param)
- #define **DtPdu_H_**
- #define **DtSTATUS_OK**
- #define **DtSTATUS_NOT_OK**
- extern int **DtProtocolVersionToSend**
- extern int **DtProtocolVersionToRecvMin**
- extern int **DtProtocolVersionToRecvMax**
- #define **netPdu**()
- #define **DtPduCallbackInfo_H_**
- #define **DtPduCallbackManager_H_**
- #define **DtPduDatumRecordSet_H_**

- #define **DtPduFactory_H_**
- #define **PDUS_H_**
- #define **DtPduWithData_H_**
- #define **DtPhysicalEntityDecoder_H_**
- #define **DtDEFINE_SIMPLE_PHYS_ENTITY_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtPhysicalEntityEncoder_H_**
- #define **DtDEFINE_SIMPLE_PHYS_ENTITY_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtPointObjectStatePdu_H_**
- #define **DtDataQueryPdu_H_**
- #define **DtRadarBeamDecoder_H_**
- #define **DtDECLARE_RADARBEAM_ATTR_DECODER**(attrName)
  This is used to declare attribute decoding functions which are members of the class
- #define **DtDEFINE_SIMPLE_RADARBEAM_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtRadarBeamEncoder_H_**
- #define **DtDECLARE_RADARBEAM_ATTR_CHECKER**(attrName)
  This is used to declare static member functions for determining whether or not an attribute needs to be encoded
- #define **DtDECLARE_RADARBEAM_ATTR_ENCODER**(attrName)
- #define **DtDEFINE_SIMPLE_RADARBEAM_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtRadioReceiverDecoder_H_**
- #define **DtDECLARE_RADIORECVR_ATTR_DECODER**(attrName)
  This is used to declare attribute decoding functions which are members of the class
- #define **DtDEFINE_SIMPLE_RADIORECVR_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtRadioReceiverEncoder_H_**
- #define **DtDECLARE_RADIORECVR_ATTR_CHECKER**(attrName)
  This is used to declare static member functions for determining whether or not an attribute needs to be encoded
- #define **DtDECLARE_RADIORECVR_ATTR_ENCODER**(attrName)
- #define **DtDEFINE_SIMPLE_RADIORECVR_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtRadioTransmitterDecoder_H_**
- #define **DtDECLARE_RADIOXMIT_ATTR_DECODER**(attrName)
  This is used to declare attribute decoding functions which are members of the class
- #define **DtDEFINE_SIMPLE_RADIOXMIT_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtRadioTransmitterEncoder_H_**
- #define **DtDECLARE_RADIOXMIT_ATTR_CHECKER**(attrName)
  This is used to declare static member functions for determining whether or not an attribute needs to be encoded
- #define **DtDECLARE_RADIOXMIT_ATTR_ENCODER**(attrName)
- #define **DtDEFINE_SIMPLE_RADIOXMIT_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtReceiverPdu_H_**
- #define **DtReceiveInteractionCbInfo_H_**
- typedef void **(*DtReceiveInteractionCb)**(DtInteraction* inter, void* usr)
  class DtReceiveInteractionCbInfo: Instances of DtReceiveInteractionCbInfo are used for providing a callback when an interaction is received from another federate
- #define **DtRecordPdu_H_**
- #define **DtRecordSet_H_**
- #define **DtRecordSetPdu_H_**
- #define **DtRecordQueryPdu_H_**
- #define **DtReflectAttributeValuesCbInfo_H_**
- #define **ObjectHandle**
- #define **DtReflectAttributeValuesCbList_H_**

- #define **ObjectHandle**
- typedef void **(*DtReflectAttributeValuesCb)**(const DtStateMsg& stateMsg, void* usr)
- #define **REFLECTED_EMITTER_LIST_H**
- #define **DtReflectedGenericObject_H_**
  This file provides protocol independence for reflected emitter system lists
- typedef void **(*DtGenericObjectCallbackFunction)**(DtReflectedGenericObject* obj, void* userData)
- #define **DtReflectedGenericObjectList_H_**
- #define **REMOTE_AGGREGATE_LIST_H_**
- #define **REMOTE_AGGREGATE_H_**
  This file provides protocol independence for remote aggregate lists
- #define **REMOTE_ENTITY_H_**
  This file provides protocol independence for reflected aggregates
- #define **REMOTE_OBJ_H_**
  This file provides protocol independence for Reflected Entities
- #define **REFLECTED_EMITTER_H**
  This file provides protocol independence for Reflected Objects
- #define **REMOTE_ENTITY_LIST_H_**
  This file provides protocol independence for reflected emitter systems
- #define **REMOTE_OBJ_LIST_H_**
  This file provides protocol independence for Remote Entity Lists
- #define **DtReflectedHlaObjectManager_H_**
  This file provides protocol independence for Reflected Object Lists
- #define **DtRegControlCbInfo_H_**
- typedef DtRegControlCbInfo **DtStartRegsCbInfo**
- typedef DtRegControlCbInfo **DtStopRegsCbInfo**
- #define **DtRegControlCbList_H_**
- typedef void **(*DtRegControlCb)**(RTI::ObjectClassHandle objectClass, void* usr)
  class DtRegControlCbList: Instances of DtRegControlCbList are used to provide a list of callbacks (DtRegControlCbInfo) which are invoked when a registration control advisory is called by RTI
- typedef DtRegControlCb **DtStartRegCb**
- typedef DtRegControlCb **DtStopRegCb**
- typedef DtRegControlCbList **DtStartRegCbList**
- typedef DtRegControlCbList **DtStopRegCbList**
- #define **REMOVE_ENTITY_H_**
- #define **DtRemoveObjectCbInfo_H_**
  This file provides protocol independence for RemoveEntity Interactions
- #define **DtRemoveEntityRPdu_H_**
- #define **DtRemoveEntityPdu_H_**
- #define **DtRepairResponsePdu_H_**
- #define **REPAIR_COMPLETE_H_**
- #define **REPAIR_RESPONSE_H_**
  This file provides protocol independence for ActionResponse Interactions
- #define **RESUPPLY_H_**
  This file provides protocol independence for ActionResponse Interactions
- #define **DtResupplyCancelPdu_H_**
  This file provides protocol independence for the Resupply Interactions: ResupplyCancel, ResupplyOffer, ResupplyReceived
- #define **DtRprStateDecoderFactory_H_**
- *class DtRprStateDecoderFactory: Instances of DtRprStateDecoderFactory are used to create a Real-time Platform Reference (RPR) Federation Object Model (FOM) decoder*
- #define **DtRprStateDecoderFactory**
  If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprStateDecoderFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools

#define **DtRprStateEncoderFactory_H_**

class DtRprStateEncoderFactory: Instances of DtRprStateEncoderFactory are used to create encoders which know how to encode state repositories for RPR (Real-time Platform Reference) FOM specific objects into an HLA state message

#define **DtRprStateEncoderFactory**
If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprStateEncoderFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools

#define **DtRprFomMapper_H_**

class DtRprFomMapper: Instances of DtRprFomMapper are DtFomMappers configured to map between VR-Link classes and functionality to RPR FOM classes, attributes and parameters

#define **DtRprFomMapper**
If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleFomMapper - a FOM Mapper that contains mappings for only the subset of RPR FOM objects/interactions that are used by most MAK tools

#define **DtRprInteractionDecoderFactory_H_**

class DtRprInteractionDecoderFactory: Instances of DtRprInteractionDecoderFactory are DtInteractionDecoderFactories configured with the VR-Link's default set of Encoders

#define **DtRprInteractionDecoderFactory**
If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprInteractionDecoderFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools

#define **DtRprInteractionEncoderFactory_H_**

class DtRprInteractionEncoderFactory: Instances of DtRprInteractionEncoderFactory are DtInteractionEncoderFactories configured with the VR-Link's default set of Encoders

#define **DtRprInteractionEncoderFactory**
If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprInteractionEncoderFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools

#define **DtRprInteractionFactory_H_**

class DtRprInteractionFactory: Instances of DtRprInteractionFactory are used to create RPR (Real-time Platform Reference) FOM interactions

#define **DtRprInteractionFactory**
If you define DtSIMPLE_RPR to 1, and include this file, then this file defines a class called DtSimpleRprInteractionFactory, which contains mappings for only the subset of RPR FOM that is used by most MAK tools

#define **DtRprHlaObjectFactory_H_**

#define **SERVICE_REQUEST_H_**

#define **SET_DATA_H_**
This file provides protocol independence for ActionResponse Interactions

#define **DtSetDataPdu_H_**
This file provides protocol independence for ActionResponse Interactions

#define **DtSetDataRPdu_H_**

#define **DtSetRecordPdu_H_**

#define **DtSignalPdu_H_**

#define **DtSimanPdu_H_**

#define **DtSimpleRprFomMapper_H_**

#define **DtSIMPLE_RPR_WAS_DEFINED**

- #define **DtRprFomMapper_H_WAS_DEFINED**
- #define **_ACKPRED_H_**
- #define **DtSoldierDecoder_H_**
- #define **DtDEFINE_SIMPLE_SOLDIER_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtSoldierEncoder_H_**
- #define **DtDEFINE_SIMPLE_SOLDIER_ATTR_CHECKER**(\ attrName, inspector)
- #define **START_INTERACTION_H_**
- #define **DtStartResumePdu_H_**
  This file provides protocol independence for StartResume Interactions
- #define **DtStartResumeRPdu_H_**
- #define **STATE_MESSAGE_H_**
- #define **DtStateRepository_H_**
  This file provides protocol independence for state messages
- #define **DtStartStopInterCbInfo_H_**
- #define **DtStartStopInterCbList_H_**
- typedef void **(*DtStartStopInterCb)**(RTI::InteractionClassHandle intClass, void* usr)
  class DtStartStopInterCbList: Instances of DtStartStopInterCbList are used to provide a list of callbacks (DtStartStopInterCbInfo) which are called when the RTI's start or stopInteractionGeneration services are invoked
- #define **DtStopFrzRPdu_H_**
- #define **STOP_INTERACTION_H_**
- #define **DtStopPdu_H_**
  This file provides protocol independence for StopFreeze Interactions
- #define **DtUpdateControlCbInfo103_H_**
- typedef DtUpdateControlCbInfo **DtStartUpdatesCbInfo**
- typedef DtUpdateControlCbInfo **DtStopUpdatesCbInfo**
- #define **DtUpdateControlCbList103_H_**
- typedef void **(*DtUpdateControlCb)**(RTI::ObjectClassHandle objectClass, const RTI::AttributeHandleSet& attributeSet, void* usr)
  class DtUpdateControlCbList: Instances of DtUpdateControlCbList are used to provide a list of callbacks (DtUpdateControlCbInfo) which are called when an object starts updating
- typedef DtUpdateControlCb **DtStartUpdatesCb**
- typedef DtUpdateControlCb **DtStopUpdatesCb**
- typedef DtUpdateControlCbList **DtStartUpdatesCbList**
- typedef DtUpdateControlCbList **DtStopUpdatesCbList**
- #define **DtVectoringNozzleSystem_H_**
- #define **DtSuppEmisEntStatePdu_H_**
- #define **DtPropulsionSystem_H_**
- #define **DtSystemHashKey_H_**
- #define **DtTrackJamDesignatorList_H_**
- #define **DtTrackJamDesignator_H_**
- #define **DtTransferControlRequestPdu_H_**
- #define **DtTransmitterPdu_H_**
- #define **DtTrackJamBeamDecoder_H_**
- #define **DtDECLARE_TRKJAMBEAM_ATTR_DECODER**(attrName)
  This is used to declare attribute decoding functions which are members of the class
- #define **DtDEFINE_SIMPLE_BASE_TRKJAMBEAM_ATTR_DECODER**(\ attrName, netType, inspector)
- #define **DtTrackJamBeamEncoder_H_**
- #define **DtDECLARE_BASE_TRKJAMBEAM_ATTR_CHECKER**(attrName)
  This is used to declare static member functions for determining whether or not an attribute needs to be encoded

- #define **DtDECLARE_BASE_TRKJAMBEAM_ATTR_ENCODER**(attrName)
- #define **DtDEFINE_SIMPLE_BASE_TRKJAMBEAM_ATTR_CHECKER**(\ attrName, inspector)
- #define **DtUnderwaterAcousticBeam_H_**
- #define **DtUnderwaterAcousticEmitterSystem_H_**
- #define **DtUnderwaterAcousticAddtnlPassiveAct_H_**
- #define **DtUnderwaterAcousticPdu_H_**
- #define **DtUnderwaterAcousticShaftSpeed_H_**
- #define **DtUnknownHlaObject_H_**
- #define **DtUnknownPdu_H_**
- #define **DtUpdateControlCbInfo_H_**
- typedef DtUpdateControlCbInfo **DtUpdatesOnCbInfo**
- typedef DtUpdateControlCbInfo **DtUpdatesOnCbInfo**
- #define **DtUpdateControlCbList_H_**
- typedef void **(*DtUpdateControlCb)**(RTI::ObjectHandle handle, const RTI::AttributeHandleSet& attributeSet, void* usr)
  class DtUpdateControlCbList: Instances of DtUpdateControlCbList are used to provide a list of callbacks (DtUpdateControlCbInfo) which are called when an object starts updating
- typedef DtUpdateControlCb **DtUpdatesOnCb**
- typedef DtUpdateControlCb **DtUpdatesOffCb**
- typedef DtUpdateControlCbList **DtUpdatesOnCbList**
- typedef DtUpdateControlCbList **DtUpdatesOffCbList**
- #define **VIEW_CONTROL_INTERACTION_H_**
- #define **DtViewControlPdu_H_**
  This file provides protocol independence for ViewControl Interactions
- DtViewControlPdu* **DtNewViewControlPdu**(DtViewerCommand command)
- #define **DtIGNORE_EAST**(1<<0)
- #define **DtIGNORE_NORTH**(1<<1)
- #define **DtIGNORE_HEIGHT**(1<<2)
- #define **DtIGNORE_HEADING**(1<<3)
- #define **DtIGNORE_PITCH**(1<<4)
- #define **DtIGNORE_ROLL**(1<<5)
- #define **DtIGNORE_EAST_OFFSET**(1<<6)
- #define **DtIGNORE_NORTH_OFFSET**(1<<7)
- #define **DtIGNORE_HEIGHT_OFFSET**(1<<8)
- #define **DtIGNORE_RADIUS**(1<<9)
- #define **DtIGNORE_BODY_HEADING**(1<<10)
- #define **DtIGNORE_HEADING_OFFSET**(1<<11)
- #define **DtIGNORE_PITCH_OFFSET**(1<<12)
- #define **DtIGNORE_FORWARD**(1<<13)
- #define **DtIGNORE_RIGHT**(1<<14)
- #define **DtIGNORE_UP**(1<<15)
- #define **DtIGNORE_BODY_PITCH**(1<<16)
- #define **DtIGNORE_BODY_ROLL**(1<<17)
- #define **DtIGNORE_AZIMUTH_RATE**(1<<18)
- #define **DtIGNORE_RADIAL_OFFSET**(1<<19)
- #define **DtIGNORE_PRIMARY_ID**(1<<20)

#define	**DtIGNORE_SECONDARY_ID**(1<<21)
#define	**DtIGNORE_GROUP_IDS**(1<<22)
#define	**DtVrlFederateAmbassador_H_**
DtVrlFedTime	**operator+** (const double&, const RTI::FedTime&) ---------- *Global operators* ----------
DtVrlFedTime	**operator-** (const double&, const RTI::FedTime&)
DtVrlFedTime	**operator*** (const double&, const RTI::FedTime&)
DtVrlFedTime	**operator/** (const double&, const RTI::FedTime&)
#define	**DtVrlRtiAmbassador_H_**

Alphabetic index *Hierarchy of classes*

In file baseexconn.h:

typedef void (*DtPostDrainCb)(void* usr)

class DtBaseExerciseConn: DtBaseExerciseConn is an abstract base class

---

Documentation

class DtBaseExerciseConn:

DtBaseExerciseConn is an abstract base class. It provides an application's interface or connection to a network.

In file remobjcbinfo.h:

typedef void (*DtRemoveObjectCb)(DtHlaObject* obj, void* usr)

class DtRemoveObjectCbInfo: Instances of DtRemoveObjectCbInfo are used for providing a callback when an object has been removed from the federation or no longer meets the interest criteria of the federate

---

Documentation

class DtRemoveObjectCbInfo:

Instances of DtRemoveObjectCbInfo are used for providing a callback when an object has been removed from the federation or no longer meets the interest criteria of the federate. This class is a helper class to DtRemoveObjectCbList.

In file vlprint.h:

void DtAbort(void)

DtAbort() exits the program

Documentation

DtAbort() exits the program. The user can provide for different behaviour [e.g. calling abort()] by linking with a another definition DtAbort [i.e. It is in its own object file.], but the user's DtAbort() should not return.

In file libmatrix.h:

void DtDcmCopy(DtConstDcm src, DtDcmRef dest)

make orthonormal, pick most important

Documentation

make orthonormal, pick most important. axis

In file libmatrix.h:

double DtDcmDcmRotTrace(DtConstDcm a, DtConstDcm b)

Returns the trace of the rotation matrix, R, between between matricies A and B

Documentation

Returns the trace of the rotation matrix, R, between between matricies A and B. If A is a DCM between frame body and world, and B is that same frame rotated through angle theta about an arbitrary axis, then trace of R [where B = A * R, or R = inv(A) * B] is 2 * cos(theta) + 1.

In file vlprint.h:

void DtFatalError(const char *s, ...)

These functions print messages using stdio depending on the notify level

---

Documentation

These functions print messages using stdio depending on the notify level. Fatal and Warn print to stderr. Info prints to stdout. Fatal funcs call DtAbort at end, even if (notifyLevel < DtNlFatal) Fatal prints are preceded by "**FATAL ERROR**: " the DtFatalError print is followed by \n Perror prints are followed by a call to perror()

In file vlutil.h:

int DtIsInList(const int *lst, int num)

Returns -1 if num is not in lst

Documentation

Returns -1 if num is not in lst. Otherwise, returns the index in the list at which num is first found. lst must be a zero-terminated array of integers.

In file topocoord.h:

void DtLatLon_to_TopoToGeoc(const DtGeodeticCoord& g, DtDcmRef topoToGeoc)

From latitude and longitude, we can produce the rotation matrices that rotate between geocentric and topographic-down orientations

---

Documentation

From latitude and longitude, we can produce the rotation matrices that rotate between geocentric and topographic-down orientations. VR-Link uses post-multiplication for its rotation matrices. For example, the matrix produced by DtLatLon_to_TopoToGeoc can be used to rotate a topographic vector to a geocentric vector like this: DtDcmVecMul(topoToGeoc, Vtopo, Vgeoc);

In file vltime.h:

void DtSetSimTime(DtTime NewSimTime)

Set the current value of VR-Link simulation time

Documentation

Set the current value of VR-Link simulation time. Should be called once per simulation frame to let VR-Link know the value of simulation time for the frame.

In file vlutil.h:

char* DtStrdup(const char* str)

Strdup wrapper

Documentation

Strdup wrapper. Returns NULL if str is NULL. Otherwise returns a newed char* containing a copy of str. Caller deletes using delete[].

In file vltime.h:

DtTime DtTimeNow(void)

Obsolete

Documentation

Obsolete. This function exists for backwards compatibility with older versions of VR-Link. Use DtGetElapsedRealTime instead.

In file utmcoord.h:

void DtUtmInit(const DtDegMinSec& latRef, const DtDegMinSec& lonRef, int standardOffsets=1, int verbose=1)

Sets reference point for all DtUtmCoords Vrlink presently supports only one UTM reference at a time, but the reference point may be reinitialized at any point

---

Documentation

Sets reference point for all DtUtmCoords Vrlink presently supports only one UTM reference at a time, but the reference point may be reinitialized at any point.

If standardOffsets is true, the standard false easting (500,000 m) and false northing (10,000,000 m only when south of equator) will be applied.

DtUtmInit expects lonRef between +/- 180. The UTM zone will be set by the zone that the reference point falls in, where zone N extends from (N*6 - 186) to (N*6 - 180).

In file entitytypes.h:

enum DtMunitionUSAntiArmorUnguided

- **Dt25mm**
- **Dt105m**
- **Dt120mm**

Documentation

- `Dt25mm`
- `Dt105m`
- `Dt120mm`

In file entitytypes.h:

enum DtMunitionUSAntiPersonnelUnguided

Dt7_62mm

Documentation

Dt7_62mm

In file entitytypes.h:

enum DtMunitionUKAntiPer/onnelUnguided

UK

Dt7_62mmUK

Documentation

UK

Dt7_62mmUK

In file entitytypes.h:

enum DtMunitionUSSRAntiPersonnelUnguided

Dt7_62mmUSSR

Documentation

Dt7_62mmUSSR

In file entitytypes.h:

enum DtPlatformAirUSAttack

DtA10

Documentation

DtA10

In file entitytypes.h:

enum DtMunitionUSSRAntiAirGuided

USSR

- **DtAA3Anan**
- **DtAA9Amos**
- **DtSA2Guideline**
- **DtSA3Goa**
- **DtSA4Ganef**
- **DtSA6Gainful**
- **DtSA7Grail**
- **DtSA8Gecko**
- **DtSA9Gaskin**
- **DtSA10Grumble**

Documentation

USSR

- **DtAA3Anan**
- **DtAA9Amos**
- **DtSA2Guideline**
- **DtSA3Goa**
- **DtSA4Ganef**
- **DtSA6Gainful**
- **DtSA7Grail**
- **DtSA8Gecko**
- **DtSA9Gaskin**
- **DtSA10Grumble**

In file entitytypes.h:

enum DtMunitionUSAntiRadarGuided

DtAGM88HARM

Documentation

DtAGM88HARM

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformAirUSAttackHelicopter

- **DtAH64**
- **DtAH1W**
- **DtRAH66**

Documentation

- **`DtAH64`**
- **`DtAH1W`**
- **`DtRAH66`**

In file entitytypes.h:

enum DtMunitionUSAntiAirGuided

US

- **DtSidewinder**
- **DtAMRAAM**
- **DtHawk**
- **DtPhoenix**
- **DtRoland**
- **DtSeaSparrow**
- **DtSparrow**
- **DtStinger**
- **DtPatriot**

Documentation

US

- `DtSidewinder`
- `DtAMRAAM`
- `DtHawk`
- `DtPhoenix`
- `DtRoland`
- `DtSeaSparrow`
- `DtSparrow`
- `DtStinger`
- `DtPatriot`

In file disenums.h:

enum DtMagneticSensorTypeSubcategory

- **DtMagneticGeneric**
- **DtAN_PSS_11**
- **DtAN_PSS_12**
- **DtGSTAMIDS**

Documentation

- `DtMagneticGeneric`
- `DtAN_PSS_11`
- `DtAN_PSS_12`
- `DtGSTAMIDS`

In file disenums.h:

enum DtLaserSensorTypeSubcategory

- **DtLaserGeneric**
- **DtASTAMIDS**

Documentation

- `DtLaserGeneric`
- `DtASTAMIDS`

In file disenums.h:

enum DtFLIR_SensorTypeSubcategory

- **DtGeneric3_5**
- **DtGeneric8_12**
- **DtASTAMIDS_I**
- **DtASTAMIDS_II**
- **DtGSTAMIDS3_5**
- **DtGSTAMIDS8_12**
- **DtHSTAMIDS3_5**
- **DtHSTAMIDS8_12**
- **DtCOBRA3_5**
- **DtCOBRA8_12**

Documentation

- `DtGeneric3_5`
- `DtGeneric8_12`
- `DtASTAMIDS_I`
- `DtASTAMIDS_II`
- `DtGSTAMIDS3_5`
- `DtGSTAMIDS8_12`
- `DtHSTAMIDS3_5`
- `DtHSTAMIDS8_12`
- `DtCOBRA3_5`
- `DtCOBRA8_12`

In file disenums.h:

enum DtRequiredReliabilityService

Reliable service PDU enums

- **DtAcknowledged**
- **DtUnacknowledged**

Documentation

Reliable service PDU enums

- **`DtAcknowledged`**
- **`DtUnacknowledged`**

In file disenums.h:

enum DtRadioDetailedModulationAmplitude

- **DtRadioDetailModOther**
- **DtAfsk**
- **DtAm**
- **DtCw**
- **DtDsb**
- **DtIsb**
- **DtLsb**
- **DtSsbFull**
- **DtSsbReduc**
- **DtUsb**
- **DtVsb**

Documentation

- `DtRadioDetailModOther`
- `DtAfsk`
- `DtAm`
- `DtCw`
- `DtDsb`
- `DtIsb`
- `DtLsb`
- `DtSsbFull`
- `DtSsbReduc`
- `DtUsb`
- `DtVsb`

In file disenums.h:

enum DtStationName

- **DtOtherStationName**
- **DtAircraftWingstation**
- **DtShipsForwardGunmountStarboard**
- **DtShipsForwardGunmountPort**
- **DtShipsForwardGunmountCenterline**
- **DtShipsAftGunmountStarboard**
- **DtShipsAftGunmountPort**
- **DtShipsAftGunmountCenterline**
- **DtForwardTorpedoTube**
- **DtAftTorpedoTube**
- **DtBombBay**
- **DtCargoBay**
- **DtTruckBed**
- **DtTrailerBed**
- **DtWellDeck**
- **DtOnStation_RNG_BRG**
- **DtOnStation_XYZ**

Documentation

- `DtOtherStationName`
- `DtAircraftWingstation`
- `DtShipsForwardGunmountStarboard`
- `DtShipsForwardGunmountPort`
- `DtShipsForwardGunmountCenterline`
- `DtShipsAftGunmountStarboard`

- **DtShipsAftGunmountPort**
- **DtShipsAftGunmountCenterline**
- **DtForwardTorpedoTube**
- **DtAftTorpedoTube**
- **DtBombBay**
- **DtCargoBay**
- **DtTruckBed**
- **DtTrailerBed**
- **DtWellDeck**
- **DtOnStation_RNG_BRG**
- **DtOnStation_XYZ**

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformSurface

- **DtCarrier**
- **DtCommandShip**
- **DtGuidedMissleCruiser**
- **DtGuidedMissleDestroyer**
- **DtDestroyer**
- **DtGuidedMissleFrigate**
- **DtPatrolCraft**
- **DtMineCounterMeasureShip**
- **DtDockLandingShip**
- **DtTankLandingShip**
- **DtLandingCraft**
- **DtFrigate**
- **DtBattleship**
- **DtHeavyCruiser**
- **DtDestroyerTender**
- **DtAmphibiousAssaultShip**
- **DtAmphibiousCargoShip**
- **DtAmphibiousTransportDock**
- **DtAmmunitionShip**
- **DtCombatStoresShip**

Documentation

- `DtCarrier`
- `DtCommandShip`
- `DtGuidedMissleCruiser`

- **DtGuidedMissleDestroyer**
- **DtDestroyer**
- **DtGuidedMissleFrigate**
- **DtPatrolCraft**
- **DtMineCounterMeasureShip**
- **DtDockLandingShip**
- **DtTankLandingShip**
- **DtLandingCraft**
- **DtFrigate**
- **DtBattleship**
- **DtHeavyCruiser**
- **DtDestroyerTender**
- **DtAmphibiousAssaultShip**
- **DtAmphibiousCargoShip**
- **DtAmphibiousTransportDock**
- **DtAmmunitionShip**
- **DtCombatStoresShip**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtCapability

- **DtAmmunitionSupplyCap**
- **DtFuelSupplyCap**
- **DtRecoveryCap**
- **DtRepairCap**

Documentation

- `DtAmmunitionSupplyCap`
- `DtFuelSupplyCap`
- `DtRecoveryCap`
- `DtRepairCap`

In file disenums.h:

enum DtDetailedModulationCombination

- **DtRadioDetailModCombo**
- **DtAmplitudeAnglePulse**

Documentation

- `DtRadioDetailModCombo`
- `DtAmplitudeAnglePulse`

In file entitytypes.h:

enum DtMunitionDomain

- **DtMunitionDomainOther**
- **DtAntiAir**
- **DtAntiArmor**
- **DtAntiGuidedMunition**
- **DtAntiRadar**
- **DtAntiSatellite**
- **DtAntiShip**
- **DtAntiSubmarine**
- **DtAntiPersonnel**
- **DtBattlefieldSupport**
- **DtStrategic**

Documentation

- `DtMunitionDomainOther`
- `DtAntiAir`
- `DtAntiArmor`
- `DtAntiGuidedMunition`
- `DtAntiRadar`
- `DtAntiSatellite`
- `DtAntiShip`
- `DtAntiSubmarine`
- `DtAntiPersonnel`
- `DtBattlefieldSupport`
- `DtStrategic`

In file entitytypes.h:

enum DtMunitionAntiArmor

- **DtAntiArmorGuided**
- **DtAntiArmorUnguided**

Documentation

- `DtAntiArmorGuided`
- `DtAntiArmorUnguided`

In file disenums.h:

enum DtMinefieldAntiHandlingDevice

- **DtNoAntiHandlingDevice**
- **DtAntiHandlingDevice**

Documentation

- `DtNoAntiHandlingDevice`
- `DtAntiHandlingDevice`

In file entitytypes.h:

enum DtMunitionAntiPersonnel

- **DtAntiPersonnelGuided**
- **DtAntiPersonnelUnguided**

Documentation

- `DtAntiPersonnelGuided`
- `DtAntiPersonnelUnguided`

In file entitytypes.h:

enum DtMunitionAntiRadar

- **DtAntiRadarGuided**
- **DtAntiRadarUnguided**

Documentation

- `DtAntiRadarGuided`
- `DtAntiRadarUnguided`

In file entitytypes.h:

enum DtMunitionAntiShip

- **DtAntiShipGuided**
- **DtAntiShipUnguided**

Documentation

- `DtAntiShipGuided`
- `DtAntiShipUnguided`

In file entitytypes.h:

enum DtMunitionAntiSubmarine

- **DtAntiSubmarineGuided**
- **DtAntiSubmarineUnguided**

Documentation

- `DtAntiSubmarineGuided`
- `DtAntiSubmarineUnguided`

In file entitytypes.h:

enum DtPlatformAir

- **DtFighter**
- **DtAttack**
- **DtBomber**
- **DtCargo**
- **DtAntiSubmarinePatrol**
- **DtAttackHelicopter**
- **DtUtilityHelicopter**
- **DtSeaGoingHelicopter**
- **DtElectronicWarfare**
- **DtUnmannedAirVehicle**
- **DtReconnaisance**
- **DtSpecialOperations**

Documentation

- `DtFighter`
- `DtAttack`
- `DtBomber`
- `DtCargo`
- `DtAntiSubmarinePatrol`
- `DtAttackHelicopter`
- `DtUtilityHelicopter`
- `DtSeaGoingHelicopter`
- `DtElectronicWarfare`
- `DtUnmannedAirVehicle`
- `DtReconnaisance`

DtSpecialOperations

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtArtParamType

Articulated parts

- **DtApPosition**
- **DtApPositionRate**
- **DtApExtension**
- **DtApExtensionRate**
- **DtApX**
- **DtApXRate**
- **DtApY**
- **DtApYRate**
- **DtApZ**
- **DtApZRate**
- **DtApAzimuth**
- **DtApAzimuthRate**
- **DtApElevation**
- **DtApElevationRate**
- **DtApRotation**
- **DtApRotationRate**

Documentation

Articulated parts

- `DtApPosition`
- `DtApPositionRate`
- `DtApExtension`
- `DtApExtensionRate`
- `DtApX`
- `DtApXRate`

- **DtApY**
- **DtApYRate**
- **DtApZ**
- **DtApZRate**
- **DtApAzimuth**
- **DtApAzimuthRate**
- **DtApElevation**
- **DtApElevationRate**
- **DtApRotation**
- **DtApRotationRate**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtAppearAttributeBitShift

- **DtAppPaintSchemeBitShift**
- **DtAppMobilityKillBitShift**
- **DtAppFirePowerKillBitShift**
- **DtAppDamageBitShift**
- **DtAppSmokeBitShift**
- **DtAppTrailingEffectsBitShift**
- **DtAppHatchBitShift**
- **DtAppLightsBitShift**
- **DtAppFlamingBitShift**
- **DtAppFrozenBitShift**
- **DtAppPowerPlantStatusBitShift**
- **DtAppDummyBitShift**

Documentation

- `DtAppPaintSchemeBitShift`
- `DtAppMobilityKillBitShift`
- `DtAppFirePowerKillBitShift`
- `DtAppDamageBitShift`
- `DtAppSmokeBitShift`
- `DtAppTrailingEffectsBitShift`
- `DtAppHatchBitShift`
- `DtAppLightsBitShift`
- `DtAppFlamingBitShift`
- `DtAppFrozenBitShift`
- `DtAppPowerPlantStatusBitShift`

DtAppDummyBitShift

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtTransferType

Transfer Control Request PDU enums

- **DtOtherTransfer**
- **DtCtlReqTransferEntity**
- **DtAppReqTransferEntity**
- **DtReqExchangeEntity**
- **DtCtlReqTransferEnvProc**
- **DtAppReqTransferEnvProc**
- **DtReqExchangeEnvProc**

Documentation

Transfer Control Request PDU enums

- `DtOtherTransfer`
- `DtCtlReqTransferEntity`
- `DtAppReqTransferEntity`
- `DtReqExchangeEntity`
- `DtCtlReqTransferEnvProc`
- `DtAppReqTransferEnvProc`
- `DtReqExchangeEnvProc`

In file entitytypes.h:

enum DtPlatformLand

- **DtTank**
- **DtArmoredFightingVehicle**
- **DtArmoredUtilityVehicle**
- **DtSelfPropelledArtillery**
- **DtTowedArtillery**
- **DtSmallWheeledUtilityVehicle**
- **DtLargeWheeledUtilityVehicle**
- **DtSmallTrackedUtilityVehicle**
- **DtLargeTrackedUtilityVehicle**
- **DtMortar**

Documentation

- `DtTank`
- `DtArmoredFightingVehicle`
- `DtArmoredUtilityVehicle`
- `DtSelfPropelledArtillery`
- `DtTowedArtillery`
- `DtSmallWheeledUtilityVehicle`
- `DtLargeWheeledUtilityVehicle`
- `DtSmallTrackedUtilityVehicle`
- `DtLargeTrackedUtilityVehicle`
- `DtMortar`

In file entitytypes.h:

enum DtNumberOfIndividualLifeForms

- **DtFireTeam**
- **DtSquad**
- **DtPlatoon**
- **DtCompany**
- **DtBattalion**
- **DtRegiment**
- **DtDivision**
- **DtCorps**
- **DtArmy**

Documentation

- `DtFireTeam`
- `DtSquad`
- `DtPlatoon`
- `DtCompany`
- `DtBattalion`
- `DtRegiment`
- `DtDivision`
- `DtCorps`
- `DtArmy`

In file entitytypes.h:

enum DtMunitionUSBattlefieldSupportGuided

DtBGM109Tomahawk

Documentation

DtBGM109Tomahawk

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtMunitionUSAntiArmorGuided

- **DtBGM71TOW**
- **DtHellfire**
- **DtMaverick**

Documentation

- `DtBGM71TOW`
- `DtHellfire`
- `DtMaverick`

In file entitytypes.h:

enum DtPlatformLandUSSRArmoredFightingVehicle

- **DtBMP1**
- **DtBMP2**
- **DtBRDM1**
- **DtMTLB**

Documentation

- `DtBMP1`
- `DtBMP2`
- `DtBRDM1`
- `DtMTLB`

In file entitytypes.h:

enum DtF16

- **DtBasicF16**
- **DtF16C**

Documentation

- `DtBasicF16`
- `DtF16C`

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtGroupedEntityCategory

IsGroupOf PDU enums

- **DtUndefinedGroupEntityCategory**
- **DtBasicGroundCombatVehicle**
- **DtEnhancedGroundCombatVehicle**
- **DtBasicGroundCombatSoldier**
- **DtEnhancedGroundCombatSoldier**
- **DtBasicRotorWingAircraft**
- **DtEnhancedRotorWingAircraft**
- **DtBasicFixedWingAircraft**
- **DtEnhancedFixedWingAircraft**
- **DtGroundLogisticsVehicle**

Documentation

IsGroupOf PDU enums

- `DtUndefinedGroupEntityCategory`
- `DtBasicGroundCombatVehicle`
- `DtEnhancedGroundCombatVehicle`
- `DtBasicGroundCombatSoldier`
- `DtEnhancedGroundCombatSoldier`
- `DtBasicRotorWingAircraft`
- `DtEnhancedRotorWingAircraft`
- `DtBasicFixedWingAircraft`
- `DtEnhancedFixedWingAircraft`
- `DtGroundLogisticsVehicle`

In file entitytypes.h:

enum DtT72Specific

DtBasicT72

Documentation

DtBasicT72

In file entitytypes.h:

enum DtMunitionUSBattlefieldSupportUnguided

- **DtBattlefield20mm**
- **DtBattlefield30mm**
- **DtBattlefield105mm**
- **DtBattlefield120mm**

Documentation

- `DtBattlefield20mm`
- `DtBattlefield30mm`
- `DtBattlefield105mm`
- `DtBattlefield120mm`

In file entitytypes.h:

enum DtMunitionUKBattlefieldSupportUnguided

DtBattlefield105mmUK

Documentation

DtBattlefield105mmUK

In file entitytypes.h:

enum DtMunitionBattlefieldSupport

- **DtBattlefieldSupportGuided**
- **DtBattlefieldSupportUnguided**

Documentation

- `DtBattlefieldSupportGuided`
- `DtBattlefieldSupportUnguided`

In file entitytypes.h:

enum DtLHASpecific

- **DtTarawa**
- **DtSaipan**
- **DtBelleauWood**
- **DtNassau**
- **DtPeleliu**

Documentation

- `DtTarawa`
- `DtSaipan`
- `DtBelleauWood`
- `DtNassau`
- `DtPeleliu`

In file disenums.h:

enum DtOpticalSensorTypeSubcategory

- **DtUnaidedEyeActivelySearching**
- **DtUnaidedEyeNotActivelySearching**
- **DtBinoculars**
- **DtImageIntensifier**
- **DtHMMWV_OccActivelySearching**
- **DtHMMWV_OccNotActivelySearching**
- **DtTruckOccActivelySearching**
- **DtTruckOccNotActivelySearching**
- **DtTrackedVehicleOccClosedHatchActSearching**
- **DtTrackedVehicleOccClosedHatchNotActSearching**
- **DtTrackedVehicleOccOpenHatchActSearching**
- **DtTrackedVehicleOccOpenHatchNotActSearching**

Documentation

- **`DtUnaidedEyeActivelySearching`**
- **`DtUnaidedEyeNotActivelySearching`**
- **`DtBinoculars`**
- **`DtImageIntensifier`**
- **`DtHMMWV_OccActivelySearching`**
- **`DtHMMWV_OccNotActivelySearching`**
- **`DtTruckOccActivelySearching`**
- **`DtTruckOccNotActivelySearching`**
- **`DtTrackedVehicleOccClosedHatchActSearching`**
- **`DtTrackedVehicleOccClosedHatchNotActSearching`**
- **`DtTrackedVehicleOccOpenHatchActSearching`**

DtTrackedVehicleOccOpenHatchNotActSearching

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtMinefieldPaintScheme

- **DtOtherPaintScheme**
- **DtStandardPaintScheme**
- **DtCamouflageDesertPaintScheme**
- **DtCamouflageJunglePaintScheme**
- **DtCamouflageSnowPaintScheme**
- **DtCamouflageGravelPaintScheme**
- **DtCamouflagePavementPaintScheme**
- **DtCamouflageSandPaintScheme**
- **DtNaturalWoodPaintScheme**
- **DtClearPaintScheme**
- **DtRedPaintScheme**
- **DtBluePaintScheme**
- **DtGreenPaintScheme**
- **DtOlivePaintScheme**
- **DtWhitePaintScheme**
- **DtTanPaintScheme**
- **DtBlackPaintScheme**
- **DtYellowPaintScheme**
- **DtBrownPaintScheme**

Documentation

- `DtOtherPaintScheme`
- `DtStandardPaintScheme`
- `DtCamouflageDesertPaintScheme`
- `DtCamouflageJunglePaintScheme`

- **DtCamouflageSnowPaintScheme**
- **DtCamouflageGravelPaintScheme**
- **DtCamouflagePavementPaintScheme**
- **DtCamouflageSandPaintScheme**
- **DtNaturalWoodPaintScheme**
- **DtClearPaintScheme**
- **DtRedPaintScheme**
- **DtBluePaintScheme**
- **DtGreenPaintScheme**
- **DtOlivePaintScheme**
- **DtWhitePaintScheme**
- **DtTanPaintScheme**
- **DtBlackPaintScheme**
- **DtYellowPaintScheme**
- **DtBrownPaintScheme**

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtCulturalFeatureLand

- **DtBridgeConcreteTwoLane**
- **DtBuildingOneStory**
- **DtBuildingTwoStory**
- **DtBuildingThreeStory**
- **DtBuildingFourStory**
- **DtDragonsTeeth**

Documentation

- `DtBridgeConcreteTwoLane`
- `DtBuildingOneStory`
- `DtBuildingTwoStory`
- `DtBuildingThreeStory`
- `DtBuildingFourStory`
- `DtDragonsTeeth`

In file entitytypes.h:

enum DtPlatformAirUSCargo

- **DtC130**
- **DtKC130**

Documentation

- `DtC130`
- `DtKC130`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformSurfaceUSCarrier

DtCV59

Documentation

`DtCV59`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformLandUKTank

UK

DtChallengerMBT

Documentation

UK

`DtChallengerMBT`

In file disenums.h:

enum DtEmissionPduType

- **DtStateUpdate**
- **DtChangeDataUpdate**

Documentation

- `DtStateUpdate`
- `DtChangeDataUpdate`

In file entitytypes.h:

enum DtPlatformLandAnySmallWheeledUtilityVehicle

Any Country

DtCivilianCar

Documentation

Any Country

`DtCivilianCar`

In file entitytypes.h:

enum DtEnvironmentalAir

- **DtEnvironmentalDomainAirOther**
- **DtSmoke**
- **DtFog**
- **DtCloud**
- **DtCloudWithRainFalling**
- **DtCloudWithSnowFalling**

Documentation

- `DtEnvironmentalDomainAirOther`
- `DtSmoke`
- `DtFog`
- `DtCloud`
- `DtCloudWithRainFalling`
- `DtCloudWithSnowFalling`

In file disenums.h:

enum DtMinefieldFusing

- **DtNoFuse**
- **DtOtherFuse**
- **DtPressureFuse**
- **DtMagneticFuse**
- **DtTiltRodFuse**
- **DtCommandFuse**
- **DtTripWireFuse**

Documentation

- `DtNoFuse`
- `DtOtherFuse`
- `DtPressureFuse`
- `DtMagneticFuse`
- `DtTiltRodFuse`
- `DtCommandFuse`
- `DtTripWireFuse`

In file disenums.h:

enum DtNature

IsPartOf PDU enums

- **DtOtherNature**
- **DtHostFireableMunition**
- **DtMunitionCarriedAsCargo**
- **DtFuelCarriedAsCarge**
- **DtGunmountAttachedToHost**
- **DtComputerGeneratedForcesCarriedAsCargo**
- **DtVehicleCarriedAsCargo**
- **DtEmitterMountedOnHost**
- **DtMobileCommandAndControlEntityCarriedAboardHost**
- **DtEntityStationedAt**
- **DtTeamMemberInFormationWith**

Documentation

IsPartOf PDU enums

- `DtOtherNature`
- `DtHostFireableMunition`
- `DtMunitionCarriedAsCargo`
- `DtFuelCarriedAsCarge`
- `DtGunmountAttachedToHost`
- `DtComputerGeneratedForcesCarriedAsCargo`
- `DtVehicleCarriedAsCargo`
- `DtEmitterMountedOnHost`
- `DtMobileCommandAndControlEntityCarriedAboardHost`
- `DtEntityStationedAt`
- `DtTeamMemberInFormationWith`

In file disenums.h:

enum DtDetailedModulationUnmodulated

- **DtRadioDetailModUnmod**
- **DtContinuous**

Documentation

- `DtRadioDetailModUnmod`
- `DtContinuous`

In file entitytypes.h:

enum DtCountryCode

- **DtCountryCodeOther**
- **DtFrance**
- **DtIraq**
- **DtUnionOfSovietSocialistRepublics**
- **DtUnitedKingdom**
- **DtUnitedStates**

Documentation

- `DtCountryCodeOther`
- `DtFrance`
- `DtIraq`
- `DtUnionOfSovietSocialistRepublics`
- `DtUnitedKingdom`
- `DtUnitedStates`

In file entitytypes.h:

enum DtCulturalFeatureDomain

- **DtCulturalFeatureDomainOther**
- **DtCulturalFeatureDomainLand**
- **DtCulturalFeatureDomainAir**
- **DtCulturalFeatureDomainSurface**
- **DtCulturalFeatureDomainSubsurface**
- **DtCulturalFeatureDomainSpace**

Documentation

- `DtCulturalFeatureDomainOther`
- `DtCulturalFeatureDomainLand`
- `DtCulturalFeatureDomainAir`
- `DtCulturalFeatureDomainSurface`
- `DtCulturalFeatureDomainSubsurface`
- `DtCulturalFeatureDomainSpace`

In file entitytypes.h:

enum DtEntityKind

- **DtEntityKindOther**
- **DtPlatform**
- **DtMunition**
- **DtLifeForm**
- **DtEnvironmental**
- **DtCulturalFeature**
- **DtSupply**

Documentation

- `DtEntityKindOther`
- `DtPlatform`
- `DtMunition`
- `DtLifeForm`
- `DtEnvironmental`
- `DtCulturalFeature`
- `DtSupply`

In file entitytypes.h:

enum DtPlatformLandUSSRSelfTowedArtillery

DtD20

Documentation

`DtD20`

In file disenums.h:

enum DtDetailedModulationAmplitudeAngle

- **DtRadioModAmpAngOther**
- **DtDetailedAmplitudeAndAngle**

Documentation

- `DtRadioModAmpAngOther`
- `DtDetailedAmplitudeAndAngle`

In file entitytypes.h:

enum DtLifeFormLand

DtDismountedInfantry

Documentation

`DtDismountedInfantry`

In file utmcoord.h:

enum DtPolarDirection

- **DtEast**
- **DtWest**
- **DtNorth**
- **DtSouth**

Documentation

- `DtEast`
- `DtWest`
- `DtNorth`
- `DtSouth`

In file disenums.h:

enum DtArtPartType

- **DtEntityBase**
- **DtRudder**
- **DtLeftFlap**
- **DtRightFlap**
- **DtLeftAileron**
- **DtRightAileron**
- **DtPeriscope**
- **DtGenericAntenna**
- **DtSnorkel**
- **DtLandingGear**
- **DtTailHook**
- **DtSpeedBrake**
- **DtLeftWeaponBayDoor**
- **DtRightWeaponBayDoor**
- **DtTankAPCHatch**
- **DtWingSweep**
- **DtPrimaryTurret1**
- **DtPrimaryTurret2**
- **DtPrimaryTurret3**
- **DtPrimaryTurret4**
- **DtPrimaryTurret5**
- **DtPrimaryTurret6**
- **DtPrimaryTurret7**
- **DtPrimaryTurret8**
- **DtPrimaryTurret9**
- **DtPrimaryTurret10**

- DtPrimaryGun1
- DtPrimaryGun2
- DtPrimaryGun3
- DtPrimaryGun4
- DtPrimaryGun5
- DtPrimaryGun6
- DtPrimaryGun7
- DtPrimaryGun8
- DtPrimaryGun9
- DtPrimaryGun10
- DtPrimaryLauncher1
- DtPrimaryLauncher2
- DtPrimaryLauncher3
- DtPrimaryLauncher4
- DtPrimaryLauncher5
- DtPrimaryLauncher6
- DtPrimaryLauncher7
- DtPrimaryLauncher8
- DtPrimaryLauncher9
- DtPrimaryLauncher10
- DtPrimaryDefenseSys1
- DtPrimaryDefenseSys2
- DtPrimaryDefenseSys3
- DtPrimaryDefenseSys4
- DtPrimaryDefenseSys5
- DtPrimaryDefenseSys6
- DtPrimaryDefenseSys7
- DtPrimaryDefenseSys8
- DtPrimaryDefenseSys9
- DtPrimaryDefenseSys10
- DtPrimaryRadar1
- DtPrimaryRadar2

- DtPrimaryRadar3
- DtPrimaryRadar4
- DtPrimaryRadar5
- DtPrimaryRadar6
- DtPrimaryRadar7
- DtPrimaryRadar8
- DtPrimaryRadar9
- DtPrimaryRadar10
- DtSecondaryTurret1
- DtSecondaryTurret2
- DtSecondaryTurret3
- DtSecondaryTurret4
- DtSecondaryTurret5
- DtSecondaryTurret6
- DtSecondaryTurret7
- DtSecondaryTurret8
- DtSecondaryTurret9
- DtSecondaryTurret10
- DtSecondaryGun1
- DtSecondaryGun2
- DtSecondaryGun3
- DtSecondaryGun4
- DtSecondaryGun5
- DtSecondaryGun6
- DtSecondaryGun7
- DtSecondaryGun8
- DtSecondaryGun9
- DtSecondaryGun10
- DtSecondaryLauncher1
- DtSecondaryLauncher2
- DtSecondaryLauncher3
- DtSecondaryLauncher4

- **DtSecondaryLauncher5**
- **DtSecondaryLauncher6**
- **DtSecondaryLauncher7**
- **DtSecondaryLauncher8**
- **DtSecondaryLauncher9**
- **DtSecondaryLauncher10**
- **DtSecondaryDefenseSys1**
- **DtSecondaryDefenseSys2**
- **DtSecondaryDefenseSys3**
- **DtSecondaryDefenseSys4**
- **DtSecondaryDefenseSys5**
- **DtSecondaryDefenseSys6**
- **DtSecondaryDefenseSys7**
- **DtSecondaryDefenseSys8**
- **DtSecondaryDefenseSys9**
- **DtSecondaryDefenseSys10**
- **DtSecondaryRadar1**
- **DtSecondaryRadar2**
- **DtSecondaryRadar3**
- **DtSecondaryRadar4**
- **DtSecondaryRadar5**
- **DtSecondaryRadar6**
- **DtSecondaryRadar7**
- **DtSecondaryRadar8**
- **DtSecondaryRadar9**
- **DtSecondaryRadar10**

Documentation

- `DtEntityBase`
- `DtRudder`
- `DtLeftFlap`

- **DtRightFlap**
- **DtLeftAileron**
- **DtRightAileron**
- **DtPeriscope**
- **DtGenericAntenna**
- **DtSnorkel**
- **DtLandingGear**
- **DtTailHook**
- **DtSpeedBrake**
- **DtLeftWeaponBayDoor**
- **DtRightWeaponBayDoor**
- **DtTankAPCHatch**
- **DtWingSweep**
- **DtPrimaryTurret1**
- **DtPrimaryTurret2**
- **DtPrimaryTurret3**
- **DtPrimaryTurret4**
- **DtPrimaryTurret5**
- **DtPrimaryTurret6**
- **DtPrimaryTurret7**
- **DtPrimaryTurret8**
- **DtPrimaryTurret9**
- **DtPrimaryTurret10**
- **DtPrimaryGun1**
- **DtPrimaryGun2**
- **DtPrimaryGun3**
- **DtPrimaryGun4**
- **DtPrimaryGun5**
- **DtPrimaryGun6**
- **DtPrimaryGun7**
- **DtPrimaryGun8**
- **DtPrimaryGun9**

- **DtPrimaryGun10**
- **DtPrimaryLauncher1**
- **DtPrimaryLauncher2**
- **DtPrimaryLauncher3**
- **DtPrimaryLauncher4**
- **DtPrimaryLauncher5**
- **DtPrimaryLauncher6**
- **DtPrimaryLauncher7**
- **DtPrimaryLauncher8**
- **DtPrimaryLauncher9**
- **DtPrimaryLauncher10**
- **DtPrimaryDefenseSys1**
- **DtPrimaryDefenseSys2**
- **DtPrimaryDefenseSys3**
- **DtPrimaryDefenseSys4**
- **DtPrimaryDefenseSys5**
- **DtPrimaryDefenseSys6**
- **DtPrimaryDefenseSys7**
- **DtPrimaryDefenseSys8**
- **DtPrimaryDefenseSys9**
- **DtPrimaryDefenseSys10**
- **DtPrimaryRadar1**
- **DtPrimaryRadar2**
- **DtPrimaryRadar3**
- **DtPrimaryRadar4**
- **DtPrimaryRadar5**
- **DtPrimaryRadar6**
- **DtPrimaryRadar7**
- **DtPrimaryRadar8**
- **DtPrimaryRadar9**
- **DtPrimaryRadar10**
- **DtSecondaryTurret1**

- **DtSecondaryTurret2**
- **DtSecondaryTurret3**
- **DtSecondaryTurret4**
- **DtSecondaryTurret5**
- **DtSecondaryTurret6**
- **DtSecondaryTurret7**
- **DtSecondaryTurret8**
- **DtSecondaryTurret9**
- **DtSecondaryTurret10**
- **DtSecondaryGun1**
- **DtSecondaryGun2**
- **DtSecondaryGun3**
- **DtSecondaryGun4**
- **DtSecondaryGun5**
- **DtSecondaryGun6**
- **DtSecondaryGun7**
- **DtSecondaryGun8**
- **DtSecondaryGun9**
- **DtSecondaryGun10**
- **DtSecondaryLauncher1**
- **DtSecondaryLauncher2**
- **DtSecondaryLauncher3**
- **DtSecondaryLauncher4**
- **DtSecondaryLauncher5**
- **DtSecondaryLauncher6**
- **DtSecondaryLauncher7**
- **DtSecondaryLauncher8**
- **DtSecondaryLauncher9**
- **DtSecondaryLauncher10**
- **DtSecondaryDefenseSys1**
- **DtSecondaryDefenseSys2**
- **DtSecondaryDefenseSys3**

- **DtSecondaryDefenseSys4**
- **DtSecondaryDefenseSys5**
- **DtSecondaryDefenseSys6**
- **DtSecondaryDefenseSys7**
- **DtSecondaryDefenseSys8**
- **DtSecondaryDefenseSys9**
- **DtSecondaryDefenseSys10**
- **DtSecondaryRadar1**
- **DtSecondaryRadar2**
- **DtSecondaryRadar3**
- **DtSecondaryRadar4**
- **DtSecondaryRadar5**
- **DtSecondaryRadar6**
- **DtSecondaryRadar7**
- **DtSecondaryRadar8**
- **DtSecondaryRadar9**
- **DtSecondaryRadar10**

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtEnvironmentalModelType

Environmental Process PDU enums

DtEnvModelInvalid

Documentation

Environmental Process PDU enums

`DtEnvModelInvalid`

In file entitytypes.h:

enum DtEnvironmentalDomain

- **DtEnvironmentalDomainOther**
- **DtEnvironmentalDomainLand**
- **DtEnvironmentalDomainAir**
- **DtEnvironmentalDomainSurface**
- **DtEnvironmentalDomainSubsurface**
- **DtEnvironmentalDomainSpace**

Documentation

- `DtEnvironmentalDomainOther`
- `DtEnvironmentalDomainLand`
- `DtEnvironmentalDomainAir`
- `DtEnvironmentalDomainSurface`
- `DtEnvironmentalDomainSubsurface`
- `DtEnvironmentalDomainSpace`

In file disenums.h:

enum DtEnvironmentalStatusBit

- **DtEnvironmentalStatusLast**
- **DtEnvironmentalStatusOn**

Documentation

- `DtEnvironmentalStatusLast`
- `DtEnvironmentalStatusOn`

In file entitytypes.h:

enum DtPlatformAirUSFighter

- **DtF117**
- **DtF16**
- **DtF15**
- **DtF15E**
- **DtF18**

Documentation

- `DtF117`
- `DtF16`
- `DtF15`
- `DtF15E`
- `DtF18`

In file entitytypes.h:

enum DtPlatformSurfaceUSGuidedMissleFrigate

DtFFG7

Documentation

DtFFG7

In file disenums.h:

enum DtSensorTypeCategory

- **DtOtherSensorTypeCategory**
- **DtOpticalSensorType**
- **DtFLIR_SensorType**
- **DtRADAR_SensorType**
- **DtMagneticSensorType**
- **DtLaserSensorType**
- **DtSONAR_SensorType**
- **DtPhysicalSensorType**
- **DtMultispectralSensorType**

Documentation

- `DtOtherSensorTypeCategory`
- `DtOpticalSensorType`
- `DtFLIR_SensorType`
- `DtRADAR_SensorType`
- `DtMagneticSensorType`
- `DtLaserSensorType`
- `DtSONAR_SensorType`
- `DtPhysicalSensorType`
- `DtMultispectralSensorType`

In file disenums.h:

enum DtDetailedModulationAngle

- **DtRadioModeAngleOther**
- **DtFm**
- **DtFsk**
- **DtPm**

Documentation

- `DtRadioModeAngleOther`
- `DtFm`
- `DtFsk`
- `DtPm`

In file disenums.h:

enum DtRadioSpreadSpectrum

- **DtRadioSpreadOther**
- **DtFrequncyHopping**
- **DtPseudoNoise**
- **DtTimeHopping**
- **DtPseudoNoiseFrequencyHopping**
- **DtFrequencyHoppingTimeHopping**
- **DtPseudoNoiseTimeHopping**
- **DtPseudoNoiseFrequencyHoppingTimeHopping**

Documentation

- `DtRadioSpreadOther`
- `DtFrequncyHopping`
- `DtPseudoNoise`
- `DtTimeHopping`
- `DtPseudoNoiseFrequencyHopping`
- `DtFrequencyHoppingTimeHopping`
- `DtPseudoNoiseTimeHopping`
- `DtPseudoNoiseFrequencyHoppingTimeHopping`

In file entitytypes.h:

enum DtPlatformLandUSSRLargeWheeledUtilityVehicle

- **DtZIL157**
- **DtGAZ66**
- **DtUral375**

Documentation

- `DtZIL157`
- `DtGAZ66`
- `DtUral375`

In file disenums.h:

enum DtRADAR_SensorTypeSubcategory

- **DtRADAR_Generic**
- **DtGenericGPR**
- **DtGSTAMIDS_I**
- **DtGSTAMIDS_II**
- **DtHSTAMIDS_I**
- **DtHSTAMIDS_II**

Documentation

- `DtRADAR_Generic`
- `DtGenericGPR`
- `DtGSTAMIDS_I`
- `DtGSTAMIDS_II`
- `DtHSTAMIDS_I`
- `DtHSTAMIDS_II`

In file entitytypes.h:

enum DtPlatformAirFranceUtilityHelicopter

France

DtGazelle

Documentation

France

DtGazelle

In file disenums.h:

enum DtPhysicalSensorTypeSubcategory

- **DtGenericProbe**
- **DtProbeMetalContent**
- **DtProbeNoMetalContent**

Documentation

- `DtGenericProbe`
- `DtProbeMetalContent`
- `DtProbeNoMetalContent`

In file disenums.h:

enum DtGridAxisType

- **DtGridAxisFixed**
- **DtGridAxisVariable**

Documentation

- **`DtGridAxisFixed`**
- **`DtGridAxisVariable`**

In file disenums.h:

enum DtGridDataRecType

Gridded data PDU enums

- **DtGridDataRecType0**
- **DtGridDataRecType1**
- **DtGridDataRecType2**

Documentation

Gridded data PDU enums

- **`DtGridDataRecType0`**
- **`DtGridDataRecType1`**
- **`DtGridDataRecType2`**

In file entitytypes.h:

enum DtPlatformLandUSSmallWheeledUtilityVehicle

- **DtUSCivilianCar**
- **DtHMMWV**

Documentation

- `DtUSCivilianCar`
- `DtHMMWV`

In file disenums.h:

enum DtMinefieldProtocol

- **DtHeartbeatMode**
- **DtQRP_Mode**

Documentation

- `DtHeartbeatMode`
- `DtQRP_Mode`

In file disenums.h:

enum DtMinefieldAlgae

- **DtNoAlgae**
- **DtLightAlgae**
- **DtModerateAlgae**
- **DtHeavyAlgae**

Documentation

- `DtNoAlgae`
- `DtLightAlgae`
- `DtModerateAlgae`
- `DtHeavyAlgae`

In file disenums.h:

enum DtIffAtcNavaidsSysName

IFF/ATC/NAVAIDS PDU enums

- **DtIffAtcNavaidsOtherSysName**
- **DtIffAtcNavaidsMarkX**
- **DtIffAtcNavaidsMarkXII**
- **DtIffAtcNavaidsATCRBS**
- **DtIffAtcNavaidsSoviet**
- **DtIffAtcNavaidsModeS**
- **DtIffAtcNavaidsMarkX_XII_ATCRBS**
- **DtIffAtcNavaidsMarkX_XII_ATCRBS_ModeS**

Documentation

IFF/ATC/NAVAIDS PDU enums

- `DtIffAtcNavaidsOtherSysName`
- `DtIffAtcNavaidsMarkX`
- `DtIffAtcNavaidsMarkXII`
- `DtIffAtcNavaidsATCRBS`
- `DtIffAtcNavaidsSoviet`
- `DtIffAtcNavaidsModeS`
- `DtIffAtcNavaidsMarkX_XII_ATCRBS`
- `DtIffAtcNavaidsMarkX_XII_ATCRBS_ModeS`

In file disenums.h:

enum DtIffAtcNavaidsAltParm4

- **DtIffAtcNavaidsAltParm4Other**
- **DtIffAtcNavaidsAltParm4Valid**
- **DtIffAtcNavaidsAltParm4Invalid**
- **DtIffAtcNavaidsAltParm4NoResponse**

Documentation

- `DtIffAtcNavaidsAltParm4Other`
- `DtIffAtcNavaidsAltParm4Valid`
- `DtIffAtcNavaidsAltParm4Invalid`
- `DtIffAtcNavaidsAltParm4NoResponse`

In file disenums.h:

enum DtIffAtcNavaidsChangeOptionsBit

- **DtIffAtcNavaidsChangeIndicator**
- **DtIffAtcNavaidsAlternateMode4**
- **DtIffAtcNavaidsAlternateMode5C**

Documentation

- `DtIffAtcNavaidsChangeIndicator`
- `DtIffAtcNavaidsAlternateMode4`
- `DtIffAtcNavaidsAlternateMode5C`

In file disenums.h:

enum DtIffAtcNavaidsApplicableModes

DtIffAtcNavaidsAppModeInvalid

Documentation

`DtIffAtcNavaidsAppModeInvalid`

In file disenums.h:

enum DtIffAtcNavaidsSysType

- **DtIffAtcNavaidsOtherSysType**
- **DtIffAtcNavaidsTransponder**
- **DtIffAtcNavaidsInterrogator**
- **DtIffAtcNavaidsSovietTransponder**
- **DtIffAtcNavaidsSovietInterrogator**

Documentation

- `DtIffAtcNavaidsOtherSysType`
- `DtIffAtcNavaidsTransponder`
- `DtIffAtcNavaidsInterrogator`
- `DtIffAtcNavaidsSovietTransponder`
- `DtIffAtcNavaidsSovietInterrogator`

In file disenums.h:

enum DtIffAtcNavaidsLayerSpecificInfo

DtIffAtcNavaidsLayerInfoInvalid

Documentation

`DtIffAtcNavaidsLayerInfoInvalid`

In file disenums.h:

enum DtIffAtcNavaidsSysMode

DtIffAtcNavaidsModeInvalid

Documentation

`DtIffAtcNavaidsModeInvalid`

In file disenums.h:

enum DtIffAtcNavaidsModifier

DtIffAtcNavaidsModifierInvalid

Documentation

`DtIffAtcNavaidsModifierInvalid`

In file disenums.h:

enum DtIffAtcNavaidsStatusBit

- **DtIffAtcNavaidsSystemOnOff**
- **DtIffAtcNavaidsParameter1**
- **DtIffAtcNavaidsParameter2**
- **DtIffAtcNavaidsParameter3**
- **DtIffAtcNavaidsParameter4**
- **DtIffAtcNavaidsParameter5**
- **DtIffAtcNavaidsParameter6**
- **DtIffAtcNavaidsOperationalStatus**

Documentation

- `DtIffAtcNavaidsSystemOnOff`
- `DtIffAtcNavaidsParameter1`
- `DtIffAtcNavaidsParameter2`
- `DtIffAtcNavaidsParameter3`
- `DtIffAtcNavaidsParameter4`
- `DtIffAtcNavaidsParameter5`
- `DtIffAtcNavaidsParameter6`
- `DtIffAtcNavaidsOperationalStatus`

In file disenums.h:

enum DtIffAtcNavaidsParameterBit

- **DtIffAtcNavaidsParameterTCAS**
- **DtIffAtcNavaidsParameterStatus**
- **DtIffAtcNavaidsParameterDamage**
- **DtIffAtcNavaidsParameterMalfunction**

Documentation

- `DtIffAtcNavaidsParameterTCAS`
- `DtIffAtcNavaidsParameterStatus`
- `DtIffAtcNavaidsParameterDamage`
- `DtIffAtcNavaidsParameterMalfunction`

In file disenums.h:

enum DtIffAtcNavaidsParm1

DtIffAtcNavaidsParm1Invalid

Documentation

`DtIffAtcNavaidsParm1Invalid`

In file disenums.h:

enum DtIffAtcNavaidsParm2

DtIffAtcNavaidsParm2Invalid

Documentation

`DtIffAtcNavaidsParm2Invalid`

In file disenums.h:

enum DtPosition

- **DtOtherPosition**
- **DtOnTopOf**
- **DtInsideOf**

Documentation

- `DtOtherPosition`
- `DtOnTopOf`
- `DtInsideOf`

In file disenums.h:

enum DtIntercomCtlType

Intercom Control PDU enums

- **DtIntercomCtlStatus**
- **DtIntercomReqAck**
- **DtIntercomReqNoAck**
- **DtIntercomAck**
- **DtIntercomNack**

Documentation

Intercom Control PDU enums

- `DtIntercomCtlStatus`
- `DtIntercomReqAck`
- `DtIntercomReqNoAck`
- `DtIntercomAck`
- `DtIntercomNack`

In file disenums.h:

enum DtIntercomCmdType

- **DtIntercomNoCmd**
- **DtIntercomCmdStatus**
- **DtIntercomConnect**
- **DtIntercomDisconnect**
- **DtIntercomReset**
- **DtIntercomOn**
- **DtIntercomOff**

Documentation

- `DtIntercomNoCmd`
- `DtIntercomCmdStatus`
- `DtIntercomConnect`
- `DtIntercomDisconnect`
- `DtIntercomReset`
- `DtIntercomOn`
- `DtIntercomOff`

In file disenums.h:

enum DtIntercomCtlRecType

- **DtIntercomEntDestRec**
- **DtIntercomGrpDestRec**
- **DtIntercomGrpAsgmtRec**

Documentation

- `DtIntercomEntDestRec`
- `DtIntercomGrpDestRec`
- `DtIntercomGrpAsgmtRec`

In file disenums.h:

enum DtIntercomCommType

- **DtIntercomFDX**
- **DtIntercomHDX_RecOnly**
- **DtIntercomHDX_TransOnly**
- **DtIntercomHDX**

Documentation

- `DtIntercomFDX`
- `DtIntercomHDX_RecOnly`
- `DtIntercomHDX_TransOnly`
- `DtIntercomHDX`

In file disenums.h:

enum DtIntercomDestLineStateCmd

- **DtIntercomLS_NoCmd**
- **DtIntercomSetLS_Trans**
- **DtIntercomSetLS_NotTrans**
- **DtIntercomRetLocLS_Ctl**

Documentation

- `DtIntercomLS_NoCmd`
- `DtIntercomSetLS_Trans`
- `DtIntercomSetLS_NotTrans`
- `DtIntercomRetLocLS_Ctl`

In file disenums.h:

enum DtIntercomTransLineState

- **DtIntercomNotApplicable**
- **DtIntercomNotTransmitting**
- **DtIntercomTransmitting**

Documentation

- `DtIntercomNotApplicable`
- `DtIntercomNotTransmitting`
- `DtIntercomTransmitting`

In file disenums.h:

enum DtIntercomCommClass

- **DtIntercomSimCommChnl**
- **DtIntercomSimCommSupChnl**

Documentation

- `DtIntercomSimCommChnl`
- `DtIntercomSimCommSupChnl`

In file disenums.h:

enum DtLE_AppearanceField1Flag

- **DtLE_AppearanceFlagField2**
- **DtLE_AppearanceIR**
- **DtLE_AppearanceVisual**
- **DtLE_AppearanceCapabilities**
- **DtLE_AppearanceMarking**
- **DtLE_AppearanceAlternativeType**
- **DtLE_AppearanceType**
- **DtLE_AppearanceForceID**

Documentation

- `DtLE_AppearanceFlagField2`
- `DtLE_AppearanceIR`
- `DtLE_AppearanceVisual`
- `DtLE_AppearanceCapabilities`
- `DtLE_AppearanceMarking`
- `DtLE_AppearanceAlternativeType`
- `DtLE_AppearanceType`
- `DtLE_AppearanceForceID`

In file disenums.h:

enum DtLE_AppearanceField2Flag

- **DtLE_AppearanceAudio**
- **DtLE_AppearanceEM**

Documentation

- `DtLE_AppearanceAudio`
- `DtLE_AppearanceEM`

In file disenums.h:

enum DtLE_DetonFieldFlag

- **DtLE_DetonField2**
- **DtLE_DetonLocationEntityCoords**
- **DtLE_DetonBurstQuantityRate**
- **DtLE_DetonBurstWarheadFuse**
- **DtLE_DetonEventSiteApp**
- **DtLE_DetonMunitionID**
- **DtLE_DetonMunitionSiteApp**
- **DtLE_DetonTargetID**

Documentation

- `DtLE_DetonField2`
- `DtLE_DetonLocationEntityCoords`
- `DtLE_DetonBurstQuantityRate`
- `DtLE_DetonBurstWarheadFuse`
- `DtLE_DetonEventSiteApp`
- `DtLE_DetonMunitionID`
- `DtLE_DetonMunitionSiteApp`
- `DtLE_DetonTargetID`

In file disenums.h:

enum DtLE_DetonField2Flag

- **DtLE_DetonEventNumber**
- **DtLE_DetonMunitionOrientation**

Documentation

- `DtLE_DetonEventNumber`
- `DtLE_DetonMunitionOrientation`

In file disenums.h:

enum DtLE_FireFieldFlag

- **DtLE_FireLocation**
- **DtLE_FireBurstQuantityRate**
- **DtLE_FireBurstWarheadFuse**
- **DtLE_FireEventSiteApp**
- **DtLE_FireMunitionID**
- **DtLE_FireMunitionSiteApp**
- **DtLE_FireTargetID**

Documentation

- `DtLE_FireLocation`
- `DtLE_FireBurstQuantityRate`
- `DtLE_FireBurstWarheadFuse`
- `DtLE_FireEventSiteApp`
- `DtLE_FireMunitionID`
- `DtLE_FireMunitionSiteApp`
- `DtLE_FireTargetID`

In file disenums.h:

enum DtLE_TSPI_FieldFlag

- **DtLE_TSPI_Reserved**
- **DtLE_TSPI_SystemSpecific**
- **DtLE_TSPI_MeasuredSpeed**
- **DtLE_TSPI_DeadReckoning**
- **DtLE_TSPI_OrientationError**
- **DtLE_TSPI_PositionError**
- **DtLE_TSPI_EntityOrientation**
- **DtLE_TSPI_EntityLinearVelocity**

Documentation

- `DtLE_TSPI_Reserved`
- `DtLE_TSPI_SystemSpecific`
- `DtLE_TSPI_MeasuredSpeed`
- `DtLE_TSPI_DeadReckoning`
- `DtLE_TSPI_OrientationError`
- `DtLE_TSPI_PositionError`
- `DtLE_TSPI_EntityOrientation`
- `DtLE_TSPI_EntityLinearVelocity`

In file lcpdu.h:

enum DtLgrProcess

Receiving application may filter lcPdu based on this start-up flag

---

- **DtLGR_NONE**
- **DtLGR_MINE**
- **DtLGR_ALL**

Documentation

Receiving application may filter lcPdu based on this start-up flag

- **`DtLGR_NONE`**
- **`DtLGR_MINE`**
- **`DtLGR_ALL`**

In file lcpdu.h:

enum DtLgrDirection

Play forwards or backwards when in LGR_PLAY mode

- **DtLGR_STOPPED**
- **DtLGR_NOPKT**
- **DtLGR_FORWARDS**
- **DtLGR_BACKWARDS**

Documentation

Play forwards or backwards when in LGR_PLAY mode

- `DtLGR_STOPPED`
- `DtLGR_NOPKT`
- `DtLGR_FORWARDS`
- `DtLGR_BACKWARDS`

In file lcpdu.h:

enum DtLgrFilePosition

Location status

- **DtLGR_NOFILE**
- **DtLGR_START**
- **DtLGR_MIDDLE**
- **DtLGR_END**

Documentation

Location status

- **DtLGR_NOFILE**
- **DtLGR_START**
- **DtLGR_MIDDLE**
- **DtLGR_END**

In file lcpdu.h:

enum DtLgrStyle

- **DtLGR_PRISTINE**
- **DtLGR_RVA**

Documentation

- **DtLGR_PRISTINE**
- **DtLGR_RVA**

In file entitytypes.h:

enum DtSizeQualifier

- **DtOther**
- **DtVerySmall**
- **DtSmall**
- **DtMedium**
- **DtLarge**
- **DtVeryLarge**

Documentation

- `DtOther`
- `DtVerySmall`
- `DtSmall`
- `DtMedium`
- `DtLarge`
- `DtVeryLarge`

In file entitytypes.h:

enum DtLifeFormDomain

- **DtLifeFormDomainOther**
- **DtLifeFormDomainLand**
- **DtLifeFormDomainAir**
- **DtLifeFormDomainSurface**
- **DtLifeFormDomainSubsurface**
- **DtLifeFormDomainSpace**

Documentation

- `DtLifeFormDomainOther`
- `DtLifeFormDomainLand`
- `DtLifeFormDomainAir`
- `DtLifeFormDomainSurface`
- `DtLifeFormDomainSubsurface`
- `DtLifeFormDomainSpace`

In file entitytypes.h:

enum DtPlatformLandUSSelfPropelledArtillery

DtM109

Documentation

`DtM109`

In file entitytypes.h:

enum DtPlatformLandUSArmoredFightingVehicle

- **DtM2**
- **DtM3**
- **DtM113**

Documentation

- `DtM2`
- `DtM3`
- `DtM113`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformLandUSTank

US

DtM1

Documentation

US

DtM1

In file entitytypes.h:

enum DtPlatformLandUSArmoredUtilityVehicle

DtM88

Documentation

DtM88

In file entitytypes.h:

enum DtPlatformAirUSSpecialOperations

- **DtMH60**
- **DtMH53**

Documentation

- `DtMH60`
- `DtMH53`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformAirUSSRAttackHelicopter

- **DtMI28**
- **DtMI24**

Documentation

- `DtMI28`
- `DtMI24`

Alphabetic index *Hierarchy of classes*

In file proccontrol.h:

extern DtMaximumSocketNumber

the following is used as the first argument to the PC version of select

---

Documentation

the following is used as the first argument to the PC version of select. It cannot take a 0 like unix versions. I should be able to get this number programatically.

In file entitytypes.h:

enum DtPlatformAirUSSRFighter

- **DtMiG29**
- **DtMiG23**

Documentation

- `DtMiG29`
- `DtMiG23`

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtMinefieldDataFilterField

Minefield PDU enums

- **DtMineGroundBurialDepthOffset**
- **DtMineWaterBurialDepthOffset**
- **DtMineSnowBurialDepthOffset**
- **DtMineOrientation**
- **DtMineThermalContrast**
- **DtMineReflectance**
- **DtMineEmplacementAge**
- **DtMineTripDetonationWireLocation**
- **DtMineFusing**
- **DtMineScalarDetectionCoefficient**
- **DtMinePaintScheme**

Documentation

Minefield PDU enums

- `DtMineGroundBurialDepthOffset`
- `DtMineWaterBurialDepthOffset`
- `DtMineSnowBurialDepthOffset`
- `DtMineOrientation`
- `DtMineThermalContrast`
- `DtMineReflectance`
- `DtMineEmplacementAge`
- `DtMineTripDetonationWireLocation`
- `DtMineFusing`
- `DtMineScalarDetectionCoefficient`
- `DtMinePaintScheme`

In file entitytypes.h:

enum DtMunitionUSAntiSubmarineGuided

DtMk48AntiSubmarine

Documentation

`DtMk48AntiSubmarine`

In file disenums.h:

enum DtModificationField

- **DtModLocation**
- **DtModOrientation**

Documentation

- `DtModLocation`
- `DtModOrientation`

In file disenums.h:

enum DtMultispectralSensorTypeSubcategory

DtMultispectralGeneric

Documentation

`DtMultispectralGeneric`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtMunitionAntiAir

- **DtMunitionGuided**
- **DtMunitionUnguided**

Documentation

- `DtMunitionGuided`
- `DtMunitionUnguided`

In file netboolean.h:

typedef DtNetU8 DtNetBoolean

class DtNetBoolean: Instances of DtNetBoolean are network representations of booleans

---

Documentation

class DtNetBoolean:

Instances of DtNetBoolean are network representations of booleans. The only thing that's required is that they can be cast to DtBooleans. This class has NO virtual functions since it is a NetStruct and we rely on its size and byte layout.

In file disenums.h:

enum DtTrackingJamming

- **DtNotTrackingAllTargets**
- **DtTrackingAllTargets**

Documentation

- `DtNotTrackingAllTargets`
- `DtTrackingAllTargets`

In file entitytypes.h:

enum DtPlatformAirUSObservationHelicopter

DtOH58

Documentation

`DtOH58`

In file disenums.h:

enum DtObjectKind

Environmental process PDU enums

- **DtOtherObjectKind**
- **DtObstacle**
- **DtPreparedPosition**
- **DtObjectCulturalFeature**
- **DtPassageway**
- **DtTacticalSmoke**
- **DtObstacleMarker**
- **DtObstacleBreach**

Documentation

Environmental process PDU enums

- `DtOtherObjectKind`
- `DtObstacle`
- `DtPreparedPosition`
- `DtObjectCulturalFeature`
- `DtPassageway`
- `DtTacticalSmoke`
- `DtObstacleMarker`
- `DtObstacleBreach`

In file entitytypes.h:

enum DtPlatformDomain

- **DtPlatformDomainOther**
- **DtPlatformDomainLand**
- **DtPlatformDomainAir**
- **DtPlatformDomainSurface**
- **DtPlatformDomainSubsurface**
- **DtPlatformDomainSpace**

Documentation

- `DtPlatformDomainOther`
- `DtPlatformDomainLand`
- `DtPlatformDomainAir`
- `DtPlatformDomainSurface`
- `DtPlatformDomainSubsurface`
- `DtPlatformDomainSpace`

In file entitytypes.h:

enum DtPlatformLandUSSRSelfPropelledArtillery

- **DtSS1ScudMissileLauncher**
- **DtZSU23_4**
- **DtSA6Launcher**
- **DtSAMSite**
- **DtSA10FlapLidRadar**
- **DtSA8LauncherRadar**

Documentation

- `DtSS1ScudMissileLauncher`
- `DtZSU23_4`
- `DtSA6Launcher`
- `DtSAMSite`
- `DtSA10FlapLidRadar`
- `DtSA8LauncherRadar`

In file entitytypes.h:

enum DtPlatformSubsurfaceUSStrategicMissleSubmarine

DtSBBNOhio

Documentation

`DtSBBNOhio`

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtSONARSensorTypeSubcategory

DtSONAR_Generic

Documentation

`DtSONAR_Generic`

In file entitytypes.h:

enum DtPlatformAirUSSRAttack

DtSU25

Documentation

`DtSU25`

In file disenums.h:

enum DtRestStatus

- **DtSoldierNotRested**
- **DtSoldier1HourSleep**
- **DtSoldier2HourSleep**
- **DtSoldier3HourSleep**
- **DtSoldier4HourSleep**
- **DtSoldier5HourSleep**
- **DtSoldier6HourSleep**
- **DtSoldier7HourSleep**
- **DtSoldierFullyRested**

Documentation

- `DtSoldierNotRested`
- `DtSoldier1HourSleep`
- `DtSoldier2HourSleep`
- `DtSoldier3HourSleep`
- `DtSoldier4HourSleep`
- `DtSoldier5HourSleep`
- `DtSoldier6HourSleep`
- `DtSoldier7HourSleep`
- `DtSoldierFullyRested`

In file entitytypes.h:

enum DtAH1WSpecific

DtSpecificAH1W

Documentation

`DtSpecificAH1W`

In file entitytypes.h:

enum DtCV59Specific

DtSpecificCV59

Documentation

`DtSpecificCV59`

In file entitytypes.h:

enum DtFFG7Specific

DtSpecificFFG7

Documentation

DtSpecificFFG7

In file entitytypes.h:

enum DtSA10Specific

DtSpecificSA10

Documentation

`DtSpecificSA10`

In file entitytypes.h:

enum DtSA8Specific

DtSpecificSA8

Documentation

DtSpecificSA8

In file disenums.h:

enum DtSubProtocol

Live entity protocol PDU enums

DtSubProtocolInvalid

Documentation

Live entity protocol PDU enums

`DtSubProtocolInvalid`

In file entitytypes.h:

enum DtSupplyCategories

- **DtSupplyOther**
- **DtSupplyFuels**
- **DtSupplyOils**
- **DtSupplyLubricants**
- **DtSupplyFood**
- **DtSupplySpareParts**
- **DtSupplyPersonnel**

Documentation

- `DtSupplyOther`
- `DtSupplyFuels`
- `DtSupplyOils`
- `DtSupplyLubricants`
- `DtSupplyFood`
- `DtSupplySpareParts`
- `DtSupplyPersonnel`

In file entitytypes.h:

enum DtPlatformLandUSSRTank

USSR

- **DtT72**
- **DtT64**

Documentation

USSR

- `DtT72`
- `DtT64`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformAirUSSRBomber

- **DtTU26**
- **DtTU95**

Documentation

- `DtTU26`
- `DtTU95`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformSurfaceUSAmphibiousAssaultShip

DtTarawaClass

Documentation

`DtTarawaClass`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtAegisCruiserSpecific

- **DtTiconderoga**
- **DtYorktown**
- **DtVincennes**
- **DtValleyForge**
- **DtThomasSGates**

Documentation

- `DtTiconderoga`
- `DtYorktown`
- `DtVincennes`
- `DtValleyForge`
- `DtThomasSGates`

In file entitytypes.h:

enum DtPlatformSurfaceUSGuidedMissleCruiser

DtTiconderogaClass

Documentation

`DtTiconderogaClass`

Alphabetic index *Hierarchy of classes*

In file entitytypes.h:

enum DtPlatformAirUSUnmannedAirVehicle

DtUAV

Documentation

DtUAV

Alphabetic index *Hierarchy of classes*

In file disenums.h:

enum DtUnderwaterAcousticSystemName

- **DtUnderwaterAcousticOtherSystem**
- **DtUnderwaterAcousticAN_BQQ_5**
- **DtUnderwaterAcousticAN_SSQ_62**
- **DtUnderwaterAcousticAN_SQS_23**
- **DtUnderwaterAcousticAN_SQS_26**
- **DtUnderwaterAcousticAN_SQS_53**
- **DtUnderwaterAcousticALFS**
- **DtUnderwaterAcousticLFA**
- **DtUnderwaterAcousticAN_AQS_901**
- **DtUnderwaterAcousticAN_AQS_902**

Documentation

- `DtUnderwaterAcousticOtherSystem`
- `DtUnderwaterAcousticAN_BQQ_5`
- `DtUnderwaterAcousticAN_SSQ_62`
- `DtUnderwaterAcousticAN_SQS_23`
- `DtUnderwaterAcousticAN_SQS_26`
- `DtUnderwaterAcousticAN_SQS_53`
- `DtUnderwaterAcousticALFS`
- `DtUnderwaterAcousticLFA`
- `DtUnderwaterAcousticAN_AQS_901`
- `DtUnderwaterAcousticAN_AQS_902`

In file disenums.h:

enum DtUnderwaterAcousticActiveEmissionParmIndex

DtUnderwaterAcousticActiveOtherIndex

Documentation

`DtUnderwaterAcousticActiveOtherIndex`

In file disenums.h:

enum DtUnderwaterAcousticPropulsionPlantConfig

- **DtUnderwaterAcousticOtherConfig**
- **DtUnderwaterAcousticDiesel_electric**
- **DtUnderwaterAcousticDiesel**
- **DtUnderwaterAcousticBattery**
- **DtUnderwaterAcousticNuclear**
- **DtUnderwaterAcousticTurbineReduction**
- **DtUnderwaterAcousticSteam**
- **DtUnderwaterAcousticGasTurbine**

Documentation

- `DtUnderwaterAcousticOtherConfig`
- `DtUnderwaterAcousticDiesel_electric`
- `DtUnderwaterAcousticDiesel`
- `DtUnderwaterAcousticBattery`
- `DtUnderwaterAcousticNuclear`
- `DtUnderwaterAcousticTurbineReduction`
- `DtUnderwaterAcousticSteam`
- `DtUnderwaterAcousticGasTurbine`

In file disenums.h:

enum DtUnderwaterAcousticUpdate

Underwater accoustic PDU enums

- **DtUnderwaterAcousticStateUpdate**
- **DtUnderwaterAcousticChangedDataUpdate**

Documentation

Underwater accoustic PDU enums

- **`DtUnderwaterAcousticStateUpdate`**
- **`DtUnderwaterAcousticChangedDataUpdate`**

In file disenums.h:

enum DtUnderwaterAcousticScanPattern

- **DtUnderwaterAcousticScanPatternNotUsed**
- **DtUnderwaterAcousticConical**
- **DtUnderwaterAcousticHelical**
- **DtUnderwaterAcousticRaster**
- **DtUnderwaterAcousticSectorSearch**
- **DtUnderwaterAcousticContinuousSearch**

Documentation

- `DtUnderwaterAcousticScanPatternNotUsed`
- `DtUnderwaterAcousticConical`
- `DtUnderwaterAcousticHelical`
- `DtUnderwaterAcousticRaster`
- `DtUnderwaterAcousticSectorSearch`
- `DtUnderwaterAcousticContinuousSearch`

In file disenums.h:

enum DtUnderwaterAcousticAdditionalPassiveActivityParmIndex

- **DtUnderwaterAcousticOtherAdditionalIndex**
- **DtUnderwaterAcousticGrahamsMKV_CoffeeMaker**

Documentation

- `DtUnderwaterAcousticOtherAdditionalIndex`
- `DtUnderwaterAcousticGrahamsMKV_CoffeeMaker`

In file disenums.h:

enum DtUnderwaterAcousticHullMountedMaskerStatus

- **DtUnderwaterAcousticHullMountedMaskerOff**
- **DtUnderwaterAcousticHullMountedMaskerOn**

Documentation

- `DtUnderwaterAcousticHullMountedMaskerOff`
- `DtUnderwaterAcousticHullMountedMaskerOn`

In file disenums.h:

enum DtUnderwaterAcousticFunction

- **DtUnderwaterAcousticOtherFunction**
- **DtUnderwaterAcousticPlatformSearch_detect_track**
- **DtUnderwaterAcousticNavigation**
- **DtUnderwaterAcousticMineHunting**
- **DtUnderwaterAcousticWeaponSearch_detect_track_detect**

Documentation

- `DtUnderwaterAcousticOtherFunction`
- `DtUnderwaterAcousticPlatformSearch_detect_track`
- `DtUnderwaterAcousticNavigation`
- `DtUnderwaterAcousticMineHunting`
- `DtUnderwaterAcousticWeaponSearch_detect_track_detect`

In file disenums.h:

enum DtUnderwaterAcousticPassiveParmIndex

DtUnderwaterAcousticOtherPassiveIndex

Documentation

`DtUnderwaterAcousticOtherPassiveIndex`

In file utmcoord.h:

const DtDegMinSec DtUtmLatRefZone10

The standard UTM reference for a zone is the point along the zone's central meridian at the equator

---

Documentation

The standard UTM reference for a zone is the point along the zone's central meridian at the equator. Standard UTM positions also use the standard false easting and false easting (see DtUtmInit)

This is the standard UTM reference position for UTM zone 10

In file geodcoord.h:

const DtMapDatum DtWGS84

Documentation

...and Mapping Datums

In file hoststructs.h:

typedef struct DtSimulationAddress DtU16 DtSite

for backwards compatibility

- **DtkObjectIsUnknown**
- **DtkObjectIsDis**
- **DtkObjectIsHla**

Documentation

for backwards compatibility

- **`DtkObjectIsUnknown`**
- **`DtkObjectIsDis`**
- **`DtkObjectIsHla`**

In file libmatrix.h:

enum generic_mm_multflag

Generic DcmDcm Multiply routine

- **TRANSPOSE_NONE**
- **TRANSPOSE_M1**
- **TRANSPOSE_M2**
- **TRANSPOSE_M1_AND_M2**

Documentation

Generic DcmDcm Multiply routine

- **`TRANSPOSE_NONE`**
- **`TRANSPOSE_M1`**
- **`TRANSPOSE_M2`**
- **`TRANSPOSE_M1_AND_M2`**

In file vlconfig.h:

#define DT_NEED_DLL_DECLARATIONS

If building a DLL MachineTypes

Documentation

If building a DLL MachineTypes.h turn on __declspec(...)

In file vlutil.h:

#define DtATOI(s)

some versions of strtol returns INT_MAX for input larger than 0x7fffffff

---

Documentation

some versions of strtol returns INT_MAX for input larger than 0x7fffffff. As long as strtoul accepts and properly processes negative input, this macro will work for both unsigned and signed ints (as other strtol do).

In file netsocket.h:

#define DtBindToDestAddr((DtU8)

setting this bit in the flags constructor argument will cause the NetSocket to ignore all traffic (incl

---

Documentation

setting this bit in the flags constructor argument will cause the NetSocket to ignore all traffic (incl. broadcast when dest_addr is multicast and point-to-point) besides that to dest_addr.

In file aggentdec.h:

#define DtDEFINE_SIMPLE_AGGREGATE_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file aggentenc.h:

#define DtDEFINE_SIMPLE_AGG_ENTITY_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file hstateencoder.h:

#define DtDEFINE_SIMPLE_ATTR_CHECKER(\ /// encClass, stateRepClass, attrName, inspector)

This macro is used to define simple attribute checker functions (members of a derived DtHlaStateEncoder that check whether a particular attribute is out of date

---

Documentation

This macro is used to define simple attribute checker functions (members of a derived DtHlaStateEncoder that check whether a particular attribute is out of date. The simple implementation is to just compare the values returned by the two StateReps' inspector functions.

In file baseembeddec.h:

#define DtDEFINE_SIMPLE_BASE_EMBEDSYS_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file baseembedenc.h:

#define DtDEFINE_SIMPLE_BASE_EMBEDSYS_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file baseemittrdec.h:

#define DtDEFINE_SIMPLE_BASE_EMITTERBEAM_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file baseemittrenc.h:

#define DtDEFINE_SIMPLE_BASE_EMITTERBEAM_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file baseentdec.h:

#define DtDEFINE_SIMPLE_BASE_ENTITY_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file baseentenc.h:

#define DtDEFINE_SIMPLE_BASE_ENTITY_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file jammerbeamdec.h:

#define DtDEFINE_SIMPLE_BASE_JAMMERBEAM_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file trkjambeamdec.h:

#define DtDEFINE_SIMPLE_BASE_TRKJAMBEAM_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file trkjambeamenc.h:

#define DtDEFINE_SIMPLE_BASE_TRKJAMBEAM_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file designatordec.h:

#define DtDEFINE_SIMPLE_DESIGNATOR_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file designatorenc.h:

#define DtDEFINE_SIMPLE_DESIGNATOR_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file emittersysdec.h:

#define DtDEFINE_SIMPLE_EMITTERSYS_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file emittersysenc.h:

#define DtDEFINE_SIMPLE_EMITTERSYS_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file envdec.h:

#define DtDEFINE_SIMPLE_ENVENT_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file envenc.h:

#define DtDEFINE_SIMPLE_ENVENT_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file jammerbeamenc.h:

#define DtDEFINE_SIMPLE_JAMMERBEAM_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file milentdec.h:

#define DtDEFINE_SIMPLE_MIL_ENTITY_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file milentenc.h:

#define DtDEFINE_SIMPLE_MIL_ENTITY_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file milplatdec.h:

#define DtDEFINE_SIMPLE_MIL_PLAT_ENTITY_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file milplatenc.h:

#define DtDEFINE_SIMPLE_MIL_PLAT_ENTITY_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file munentdec.h:

#define DtDEFINE_SIMPLE_MUN_ENTITY_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file munentenc.h:

#define DtDEFINE_SIMPLE_MUN_ENTITY_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file physentdec.h:

#define DtDEFINE_SIMPLE_PHYS_ENTITY_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file physentenc.h:

#define DtDEFINE_SIMPLE_PHYS_ENTITY_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file radarbeamdec.h:

#define DtDEFINE_SIMPLE_RADARBEAM_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file radarbeamenc.h:

#define DtDEFINE_SIMPLE_RADARBEAM_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file radiorecvrdec.h:

#define DtDEFINE_SIMPLE_RADIORECVR_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file radiorecvrenc.h:

#define DtDEFINE_SIMPLE_RADIORECVR_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file radioxmitdec.h:

#define DtDEFINE_SIMPLE_RADIOXMIT_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file radioxmitenc.h:

#define DtDEFINE_SIMPLE_RADIOXMIT_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file soldierdec.h:

#define DtDEFINE_SIMPLE_SOLDIER_ATTR_DECODER_WITH_CAST(\ /// attrName, netType, mutator, castExpr)

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes)

Documentation

These macros are used to define simple attribute decoder functions (members of a derived DtHlaStateDecoder that decode individual attributes). The simple decoding is just to convert the value pulled out of the ahvps to a host type using the cast expression castExpr, and pass it to the mutator function.

In file soldierenc.h:

#define DtDEFINE_SIMPLE_SOLDIER_ATTR_ENCODER(\ /// attrName, netType, inspector)

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes)

Documentation

This macro is used to define simple attribute encoder functions (members of a derived DtHlaStateEncoder that encode individual attributes). The simple encoding is just to implicitly convert the value returned by the inspector to a netType, and add it to the ahvps.

In file disenums.h:

#define DtEXTEND_ENUM(enumType, name, val)

This file should contain only enumeration definitions

Documentation

This file should contain only enumeration definitions. This file is processed by a program that generates the files parseEnum.cc and parseEnum.h. Therefore all enums must conform to the format of the ones already in this file. No comments on the same line as a member of the enum, and all enumeration names and member names must start with Dt. -Len

---

In file netstructs.h:

#define DtMAX_OBJECT_NAME_LENGTH

This class is basically a hack so that the simple encoders and decoders will work for DtGlobalObjectDesignators

---

Documentation

This class is basically a hack so that the simple encoders and decoders will work for DtGlobalObjectDesignators. You can cast a char* to a DtNetGlobalObjectDesignator* (and vice versa), and you can assign a char* to a DtNetGlobalObjectDesignator. It probably shouldn't be used for anything else.

In file socket.h:

#define DtMaxPacketSize

IP + UDP headers are 20 + 8

Documentation

IP + UDP headers are 20 + 8. Ethernet MTU is 1500 (or 1492 for 802.3 encapsulation), thus PDU limits will typically be 1472 or 1464 FDDI MTU is 4352 (PDU limit 4324) IEEE 802.5 token ring MTU is 4464 (PDU limit of 4436)

In file libmatrix.h:

#define DtNeg(x)

Returns x if x is less than zero

Documentation

Returns x if x is less than zero.

In file nullfedamb.h:

#define DtNullFederateAmbassador_H_

Adapted from the DMSO RTI 1

Documentation

Adapted from the DMSO RTI 1.3.4 file NullFederateAmbassador.hh. This class provides empty definitions for the pure virtual functions declared as members of RTI::FederateAmbassador.

In file libmatrix.h:

#define DtPos(x)

Returns x if greater than zero

Documentation

Returns x if greater than zero.

In file simrprfommap.h:

#define DtSIMPLE_RPR

This header file provides a definition for the class DtSimpleRprFomMapper, a FOM Mapper that includes mappings for only the subset of the RPR FOM (with MAK extensions) that is used by most MAK applications

---

Documentation

This header file provides a definition for the class DtSimpleRprFomMapper, a FOM Mapper that includes mappings for only the subset of the RPR FOM (with MAK extensions) that is used by most MAK applications. It includes all entity classes, aggregates, emissions, as well as fire, detonate, collision, stealth control and logger control interactions. Using this FOM Mapper rather than the default RPR FOM Mapper significantly cuts down executable size.

In file libmatrix.h:

#define DtSgn(x)

Returns the sign of x

Documentation

Returns the sign of x.

In file libmatrix.h:

#define DtSquare(x)

Returns the square of x

Documentation

Returns the square of x.

In file vrlfedtime.h:

#define DtVrlFedTime_H_

This file is based on MyFedTime

Documentation

This file is based on MyFedTime.h, which appears in the DMSO RTI MyFedTime example in RTI_HOME/lang/C++/demo/MyFedTime.

In file vlconfig.h:

#define VLCONFIG_INCLUDED

File - libsrc/md/c-NT

Documentation

File - libsrc/md/c-NT.h (vlConfig.h for win32)

In file netsocket.h:

#define _NetSocket_h

class NetSocket NetSocket provides several overloaded constructors that allow the user to specify UDP port, device, and destination address in any combination except port and destination address together

---

Documentation

class NetSocket

NetSocket provides several overloaded constructors that allow the user to specify UDP port, device, and destination address in any combination except port and destination address together.

In addition, each constructor has an optional flags argument, which may be set to DtRead (read only), DtWrite (write only) or both (default, DtRead | DtWrite).

The default destination address is broadcast. The default network interface device is the first appropriate one located by the constructor.

In file vllist.h:

#define looplist(theData, list)

Macro to loop through the list

Documentation

Macro to loop through the list.

In file hexerciseconn.h:

#define what()

Macro definition so that exception

Documentation

Macro definition so that exception.what() will work with pre-NG RTI.