CSTADLL

Version 2.5.0

Objective Systems, Inc.

October 2023

CSTADLL

Author's Contact Information. Comments, suggestions, and inquiries regarding CSTADLL or this document may be sent by electronic mail to <info@obj-sys.com>.

Table of Contents

List of Tables

Chapter 1. Namespace Documentation

Com

Namespaces

- struct Com::Objsys

Com::Objsys

Namespaces

- struct Com::Objsys::Csta

Com::Objsys::Csta

Namespaces

- struct Com::Objsys::Csta::Xmled6

Com::Objsys::Csta::Xmled6

Classes

- struct Com::Objsys::Csta::Xmled6::Constants
- struct Com::Objsys::Csta::Xmled6::CSTAContext
- struct Com::Objsys::Csta::Xmled6::CSTAEncDec
- struct Com::Objsys::Csta::Xmled6::CSTAResponseInfo
- struct Com::Objsys::Csta::Xmled6::GenericXMLed6
- struct Com::Objsys::Csta::Xmled6::LicenseException
- struct Com::Objsys::Csta::Xmled6::LicenseHelper
- struct Com::Objsys::Csta::Xmled6::LicenseOptions
- struct Com::Objsys::Csta::Xmled6::PBXSession
- struct Com::Objsys::Csta::Xmled6::PBXSessionException
- struct Com::Objsys::Csta::Xmled6::PBXSessionHelper
- struct Com::Objsys::Csta::Xmled6::PBXSessionHelperEd6
- struct Com::Objsys::Csta::Xmled6::ResetSessionInfo
- struct Com::Objsys::Csta::Xmled6::SocketState
- struct Com::Objsys::Csta::Xmled6::uaSIPInvite

- struct Com::Objsys::Csta::Xmled6::uaXMLed6
- struct Com::Objsys::Csta::Xmled6::XMLParseInfo

Detailed Description

The namespace `Com.Objsys.Csta.Xmled6` contains classes that are specific to XML CSTA edition 6. Most of these classes are generated by XBinder from the CSTA and session management (ECMA-354) XML schema specifications. These generated classes are not documented here, but you can consult the XBinder Java/C# User Guide for information about how XML schema constructions are translated into C# classes.

The namespace also contains several classes that are not generated by XBinder. These classes are the ones documented in this manual.

Definition at line 47 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

com::objsys::xbinder::runtime

System

System::Collections::Generic

System::IO

System::Linq

System::Net

System::Net::Sockets

System::Reflection

System::Runtime::InteropServices

System::Text

System::Threading

System::Threading::Tasks

System::Xml

Chapter 2. CSTAXMLEd6

CSTAXMLEd6 is a Microsoft .NET 4.5 DLL that allows client code to communicate with a PBX or UA device.

The DLL uses the following namespaces:

- `Com.Objsys.Csta.Devices`
- `Com.Objsys.Csta.Xmled6`

The `Com.Objsys.Csta.Devices` namespace contains classes that allow a caller to use specific PBX or UA devices.

The `Com.Objsys.Csta.Xmled(n)` namespaces contain classes that are specific to the indicated edition of XML CSTA. Most of these classes are generated by XBinder from the CSTA and session management (ECMA-354) XML schema specifications. These generated classes are not documented here, but you can consult the XBinder Java/C# User Guide for information about how XML schema constructions are translated into C# classes.

Each namespace also contains several classes that are not generated by ASN1C. These classes are the ones documented in this manual.

A typical way to use the DLL is to use the `PBXSession` class to set up the communication to the PBX or UA device via the constructor. If the PBX or UA will be sending asynchronous data, such as monitor packets, to the client, the `ClientCallback` or `XMLClientCallback` property can be used to define a callback method to receive the asynchronous data. If no callback method is defined, asynchronous data will be ignored.

If the PBX or UA will be sending Call Detail Records Report or Call Detail Records Notification messages to the client, the `CDRCallback` or `XMLCDRCallback` property can be used to define a callback method to receive the messages. If no callback method is defined, Call Detail messages will be ignored.

The CSTADLL kit includes some samples to guide you in writing your own code. The samples are evenly split between those implemented in C# and those implemented in Visual BASIC. Each language has samples for communicating with PBX devices that use BER CSTA and with PBX devices that use XML CSTA.

The classes and methods exposed by the DLL are probably sufficient to handle operations for most PBX or UA devices. But if needed, you can write a class of your own to handle operations for a PBX device that the software doesn't explicitly support. The sample NewPBX shows how this might be accomplished. This sample contains code for a small separate DLL that could be used to support a fictitious PBX device. The assumption in the sample is that this device uses standard messages for all operations except for the initial association messages. These messages are the ones that are most commonly different from one PBX to the next. The NewPBX sample shows how the EncodeACSEConnectionRequest() method within the GenericCSTAp2 class (for BER PBX devices) or the EncodeStartSession() method within the GenericXMLEd4 class (for XML PBX devices) can be overridden in a class that you can write. The override implementation handles the details that are specific to the device.

The DLL can log message traffic between a client program and the PBX or UA device if so desired. The logging is controlled by the `LoggingEnabled` property with the `PBXSessionHelper` class. The logging is off by default. Both of the provided sample clients enable the logging. The log file used is named `cstadll_<program>.log`, where <program> is the name of the executable image that is using the DLL. The location of the log file is the folder where the executable image resides. The default behavior is that if the log file grows to more then 5 Mb, it is copied to `cstadll_<program>.backup.log`, and a new log file is opened. If there is already a file with the backup file name, it is overwritten. That default size of 5 Mb can be modified by using the `MaxLogFileSize` property of the `PBXSessionHelper` class.

If your CSTADLL kit is licensed (i.e., not unlimited), and its license file has a file type of .lic, then you will need to deploy your application with the DLLs Reprise.dll and rlm1212.dll that are in the kit. The file rlm1212.dll is a 32-bit native DLL as opposed to a .NET DLL. As such, if you build your code with a Makefile, you will need to use the `/platform:x86` qualifier to the `csc` or `vbc` command. If you build your code with a Visual Studio project, you will need to use x86 as the target platform instead of AnyCPU. These steps are to ensure proper interfacing to the native 32-bit rlm1212.dll. There is also a 64-bit version of rlm1212.dll available if you prefer to target the x64 platform. If your license file is an osyslic.txt file, these steps are not necessary.

Chapter 3. Class Documentation

ApplicationException class Reference

Com::Objsys::Csta::Xmled6::Constants class Reference

- enum ACSEMessageTypes {
  MakeAssociation,
  ReleaseAssociation
  }
- enum CallbackInvocationMechanisms {
  InvokeCallbackThenPostNextRead,
  PostNextReadThenInvokeCallback
  }
- enum CommunicationTypes {
  RawBER,
  IETFBER,
  SiemensBER,
  RawXML,
  SIPXML
  }
- enum Encoding {
  BER,
  XML
  }
- enum PBXModels {
  Unknown,
  GenericBER,
  GenericIETF,
  Panasonic,
  Alcatel4400,
  AlcatelOXO,
  AlcatelOXE,
  SiemensHicom300,
  SiemensHipath3000,
  SiemensCap,
  TadiranCoral,
  SiemensRealitis,
  SiemensHipath4000,
  UnifyOpenscapeX5,
  PhilipsSopho,
  GenericXML,
  GenericSIP,
  UnifyOpenscapeVoice,
  VodiaSNOMOne,

UnifyOpenscape4000BER,
SamsungSCM,
PanasonicNS
}

- enum XMLSessionMessageTypes {
  StartSession,
  StopSession,
  ResetSession
  }

- const ushort ASN_K_MAXSUBIDS
- const string INVALID_COMM_TYPE
- const string INVALID_PHASE
- const string INVALID_XML_EDITION
- const ushort MAX_NUM_SIZE
- const ushort MAX_RECV_TIMEOUT
- const ushort MAX_SEQUENCE_NUMBER
- const string NO_RESPONSE_FROM_PBX
- const string PBX_UNIVERSAL_FAILURE
- const string ROSE_ENCODE_FAILURE

Public Attributes

- const long MAX_LOGFILE_SIZE

Detailed Description

The Constants class contains some helpful constant and enum definitions.

Definition at line 52 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

enum ACSEMessageTypes

Provides symbolic names for the ACSE message types.

Enumerator:

MakeAssociation

ReleaseAssociation

Definition at line 176 of file Constants.cs

```
{
MakeAssociation,
ReleaseAssociation,
}ACSEMessageTypes;
```

enum CallbackInvocationMechanisms

Indicates how an asynchronous callback method should be invoked. This setting influences how the asynchronous callback methods for monitor event report messages, route messages, and Call Detail Record messages are invoked.

The value InvokeCallbackThenPostNextRead causes the callback method to be invoked before the next read from the PBX or UA is posted to the socket. This setting is the default. With this mechanism callback methods can be easily debugged because new packets from the PBX or UA won't be arriving while debugging of the method is in progress. This mechanism also ensures that messages from the PBX or UA will arrive in a predictable order.

The value PostNextReadThenInvokeCallback causes the callback method to be invoked after the next read from the PBX or UA is posted to the socket. Use of this mechanism is necessary if additional synchronous CSTA messages are going to be sent as part of a callback method's processing. If this mechanism is not used in such a case, the response to the CSTA message sent from the callback method will never be seen because no read to the socket was posted. With that said, however, use this mechanism with EXTREME caution. Because the read to the socket is posted before the event is handled, event n+1 may come in and get handled before event n. You may need to add code to ensure that events get handled in an expected order, if such code is even possible for your situation.

Enumerator:

InvokeCallbackThenPostNextRead

PostNextReadThenInvokeCallback

Definition at line 216 of file Constants.cs

```
{
InvokeCallbackThenPostNextRead,
PostNextReadThenInvokeCallback,
}CallbackInvocationMechanisms;
```

enum CommunicationTypes

Provides symbolic names for different ways of communicating with a PBX or UA. The values of this enum influence how each message exchange with a PBX or UA is handled.

Enumerator:

RawBER

IETFBER

SiemensBER

RawXML

SIPXML

Definition at line 154 of file Constants.cs

```
{
RawBER,
IETFBER,
SiemensBER,
RawXML,
SIPXML,
}CommunicationTypes;
```

enum Encoding

Provides symbolic names for the mechanisms for encoding CSTA messages.

Enumerator:

BER

XML

Definition at line 167 of file Constants.cs

```
{
BER,
XML,
}Encoding;
```

enum PBXModels

Provides symbolic names for different PBX models.

Enumerator:

Unknown

GenericBER

GenericIETF

Panasonic

Alcatel4400

AlcatelOXO

AlcatelOXE

SiemensHicom300

SiemensHipath3000

SiemensCap

TadiranCoral

SiemensRealitis

SiemensHipath4000

UnifyOpenscapeX5

PhilipsSopho

GenericXML

GenericSIP

UnifyOpenscapeVoice

VodiaSNOMOne

UnifyOpenscape4000BER

SamsungSCM

PanasonicNS

Definition at line 123 of file Constants.cs

```
{
Unknown,
GenericBER,
GenericIETF,
Panasonic,
Alcatel4400,
AlcatelOXO,
AlcatelOXE,
SiemensHicom300,
SiemensHipath3000,
SiemensCap,
TadiranCoral,
SiemensRealitis,
SiemensHipath4000,
UnifyOpenscapeX5,
PhilipsSopho,
GenericXML,
GenericSIP,
UnifyOpenscapeVoice,
VodiaSNOMOne,
UnifyOpenscape4000BER,
SamsungSCM,
```

```
PanasonicNS,
}PBXModels;
```

enum XMLSessionMessageTypes

Provides symbolic names for the XML session management message types.

Enumerator:

StartSession

StopSession

ResetSession

Definition at line 186 of file Constants.cs

```
{
StartSession,
StopSession,
ResetSession,
}XMLSessionMessageTypes;
```

const ushort ASN_K_MAXSUBIDS

Defines the maximum number of sub-ids that an object id can have.

Definition at line 56 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const string INVALID_COMM_TYPE

Defines a common message for detection of an invalid communication type (should never happen).

Definition at line 92 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const string INVALID_PHASE

Defines a common message for detection of an invalid phase (should never happen).

Definition at line 98 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const string INVALID_XML_EDITION

Defines a common message for detection of an invalid XML edition (should never happen).

Definition at line 104 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const ushort MAX_NUM_SIZE

Defines the maximum number of characters that a text representation of a number can have.

Definition at line 62 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const ushort MAX_RECV_TIMEOUT

Defines the maximum amount of time, in milliseconds, to wait for a response to come in from a PBX.

Definition at line 68 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const ushort MAX_SEQUENCE_NUMBER

Defines the maximum value that the sequence number portion of the invoke id can be.

Definition at line 80 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const string NO_RESPONSE_FROM_PBX

Defines a common message for no response received from the PBX.

Definition at line 109 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const string PBX_UNIVERSAL_FAILURE

Defines a common message for an error returned from the PBX.

Definition at line 115 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

const string ROSE_ENCODE_FAILURE

Defines a common message prefix for a failure to encode the ROSE header.

Definition at line 86 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

Member Data Documentation

const long MAX_LOGFILE_SIZE

Defines the maximum size, in bytes, that a log file is allowed to grow to before a new log file is opened.

Definition at line 74 of file Constants.cs

The Documentation for this struct was generated from the following file:

- Constants.cs

Com::Objsys::Csta::Xmled6::CSTAContext class Reference

- static Dictionary< string, CSTAContext > cdrCalls
- static object cdrLockObject
- static object contextByThreadIDLO
- static object contextByThreadNumberLO
- static Dictionary< int, CSTAContext > contextsByThreadID
- static Dictionary< ushort, CSTAContext > contextsByThreadNumber
- static Dictionary< string, string > devicesByXref
- static Dictionary< string, CSTAContext > getSFDCalls
- static object getSFDLockObject
- static Dictionary< string, CSTAContext > kmeOperations
- static Dictionary< string, CSTAContext > monitorCalls
- static object monitorLockObject
- static Dictionary< string, List< string > > monitorsByDevice

- static ushort nextThreadNum
- static object panasonicKmeLockObject
- static Dictionary< string, CSTAContext > snapshotCallCalls
- static object snapshotCallLockObject
- static Dictionary< string, CSTAContext > snapshotDevCalls
- static object snapshotDevLockObject
- static bool threadNumberWrapped

-
-
-
-
-

Private Attributes

- ushort lastSequenceNum
- byte [] responseFromPBX
- List< byte[]> responsesFromPBX
- int threadID
- ushort threadNumber
- Thread threadObject
- EventWaitHandle waitHandle
- EventWaitHandle xmlResetSessionWaitHandle
- string xmlResponseFromPBX
- List< string > xmlResponsesFromPBX
- EventWaitHandle xmlStopSessionWaitHandle

- CSTAContext ( )

- ~CSTAContext ( )

- static CSTAContext Instance ( )

Detailed Description

The CSTAContext class contains information needed to manage the interaction between the thread and the PBX.

Definition at line 54 of file CSTAContext.cs

The Documentation for this struct was generated from the following file:

- CSTAContext.cs

CSTAContext ()

Default constructor.

~CSTAContext ()

Destructor. Removes this instance from the context list upon garbage collection.

static CSTAContext Instance ()

This method will either return an already created context object or will create a new one.

Returns: . A CSTAContext instance.

Com::Objsys::Csta::Xmled6::CSTAEncDec class Reference

- const int DECRYPTION
- const int ENCRYPTION
- uint [] k

- static readonly uint [] bytebit
- static readonly byte [] pc1
- static readonly byte [] pc2
- static uint [][] spbox
- static readonly byte [] totrot

- CSTAEncDec (byte [] key, int dir)
- virtual void ProcessBlock (byte [] inBlock, byte [] outBlock, int offset)

- uint ByteReverse (uint value)
- uint RotateLeft (uint x, uint y)
- uint RotateRight (uint x, uint y)
- void ToBytes (uint ivalue, byte [] b, int offset)
- uint ToInt (byte [] b, int offset)

Com::Objsys::Csta::Xmled6::CSTAResponseInfo class Reference

Private Attributes

- byte [] responseFromPBX

- List< byte[]> responsesFromPBX
- int statusCode
- string statusMessage
- string xmlResponseFromPBX
- List< string > xmlResponsesFromPBX
-
-
-
-
-
-
-
-

Detailed Description

Contains information about a PBX operation that was attempted.

Definition at line 52 of file CSTAResponseInfo.cs

The Documentation for this struct was generated from the following file:

- CSTAResponseInfo.cs

Com::Objsys::Csta::Xmled6::GenericXMLed6 class Reference

Protected Attributes

- PBXSession sessionObject
- CSTAContext threadContext
-
-
- virtual CSTAResponseInfo AcceptCall (ConnectionID callToAccept)
- virtual CSTAResponseInfo AnswerCall (ConnectionID callToAnswer)

- virtual CSTAResponseInfo AnswerCall (ConnectionID callToAnswer, string deviceToLift)
- virtual CSTAResponseInfo ClearConnection (ConnectionID connectionToClear)
- virtual CSTAResponseInfo ClearDoNotDisturb (string targetDevice)
- virtual CSTAResponseInfo ClearForwarding (string fromDevice, ForwardingType fwdType)
- virtual CSTAResponseInfo ClearMessageWaiting (string targetDevice)
- virtual CSTAResponseInfo ConferenceCall (ConnectionID heldCall, ConnectionID activeCall)
- virtual CSTAResponseInfo ConsultationCall (ConnectionID existingCall, string targetDevice)
- virtual CSTAResponseInfo ConsultationCall (ConnectionID existingCall, string targetDevice, ConsultOptions options)
- virtual string EncodeInvokeID (string initialMessage)
- GenericXMLed6 (string pbxSystem, int port)
- GenericXMLed6 (PBXSession sessionObject)
- virtual CSTAResponseInfo GetAgentState (string agentDevice)
- virtual CSTAResponseInfo GetDoNotDisturb (string targetDevice)
- virtual CSTAResponseInfo GetLogicalDevInfo (string targetDevice)
- virtual CSTAResponseInfo GetPhysicalDevInfo (string targetDevice)
- virtual CSTAResponseInfo GetSFDevices (ReqDeviceCategory deviceCategory)
- virtual CSTAResponseInfo GetSFDevices ( )
- virtual CSTAResponseInfo HoldCall (ConnectionID callToHold)
- virtual CSTAResponseInfo MakeCall (string callingDevice, string calledDevice)
- virtual CSTAResponseInfo MakeCall (string callingDevice, string calledDevice, bool autoOriginate)
- virtual CSTAResponseInfo MonitorStart (string deviceToMonitor)
- virtual CSTAResponseInfo MonitorStart (ConnectionID callToMonitor)
- virtual CSTAResponseInfo MonitorStop (string crossRefID)
- virtual CSTAResponseInfo MonitorStopAtDevice (string monitoredDevice)
- virtual CSTAResponseInfo RequestSystemStatus ( )
- virtual CSTAResponseInfo RetrieveCall (ConnectionID callToRetrieve)
- virtual CSTAResponseInfo RingDevice (string targetDevice, string targetRinger, long ringPattern)
- virtual CSTAResponseInfo SendData (IOCrossRefID xref, string text)
- virtual CSTAResponseInfo SendStoredCDR (string cdrCrossRefID)
- virtual CSTAResponseInfo SetAgentState (string agentDevice, ReqAgentState agentState, string agentID)

- virtual CSTAResponseInfo SetAgentState (string agentDevice, ReqAgentState agentState)
- virtual CSTAResponseInfo SetDisplay (string targetDevice, string text)
- virtual CSTAResponseInfo SetDoNotDisturb (string targetDevice)
- virtual CSTAResponseInfo SetForwarding (string fromDevice, ForwardingType fwdType, string toDevice)
- virtual CSTAResponseInfo SetMessageWaiting (string targetDevice)
- virtual CSTAResponseInfo SingleStepTransfer (ConnectionID callToTransfer, string transferToDevice)
- virtual CSTAResponseInfo SnapshotCall (ConnectionID callToSnapshot)
- virtual CSTAResponseInfo SnapshotDevice (string deviceToSnapshot)
- virtual CSTAResponseInfo StartCDRTransmission (CDRTransferMode transferMode)
- virtual CSTAResponseInfo StartDataPath (string targetDevice)
- virtual CSTAResponseInfo StartSession ( )
- virtual CSTAResponseInfo StartSession (string applicationID)
- virtual CSTAResponseInfo StopCDRTransmission (string cdrCrossRefID)
- virtual CSTAResponseInfo StopDataPath (IOCrossRefID xref)
- virtual CSTAResponseInfo StopRing (string targetDevice, string targetRinger, long ringPattern)
- virtual CSTAResponseInfo StopSession ( )
- virtual CSTAResponseInfo TransferCall (ConnectionID initiatedCall, ConnectionID originalCall)

- virtual string EncodeAcceptCall (CSTAResponseInfo response, ConnectionID callToAccept)
- virtual string EncodeAnswerCall (CSTAResponseInfo response, ConnectionID callToAnswer)
- virtual string EncodeAnswerCall (CSTAResponseInfo response, ConnectionID callToAnswer, string deviceToLift)
- virtual string EncodeClearConnection (CSTAResponseInfo response, ConnectionID connectionToClear)
- virtual string EncodeConferenceCall (CSTAResponseInfo response, ConnectionID heldCall, ConnectionID activeCall)
- virtual string EncodeConsultationCall (CSTAResponseInfo response, ConnectionID existingCall, string targetDevice, ConsultOptions options)
- virtual string EncodeGetAgentState (CSTAResponseInfo response, string targetDevice)
- virtual string EncodeGetDND (CSTAResponseInfo response, string targetDevice)
- virtual string EncodeGetLogicalDevInfo (CSTAResponseInfo response, string targetDevice)
- virtual string EncodeGetPhysicalDevInfo (CSTAResponseInfo response, string targetDevice)
- virtual string EncodeGetSFDevices (CSTAResponseInfo response, ReqDeviceCategory category)
- virtual string EncodeHoldCall (CSTAResponseInfo response, ConnectionID callToHold)

- virtual string EncodeMakeCall (CSTAResponseInfo response, string callingDevice, string calledDevice, bool autoOriginate)
- virtual string EncodeMonitorStart (CSTAResponseInfo response, string targetDevice)
- virtual string EncodeMonitorStart (CSTAResponseInfo response, ConnectionID targetCall)
- virtual string EncodeMonitorStop (CSTAResponseInfo response, string xref)
- virtual string EncodeRequestSystemStatus (CSTAResponseInfo response)
- virtual string EncodeRetrieveCall (CSTAResponseInfo response, ConnectionID callToRetrieve)
- virtual string EncodeSendData (CSTAResponseInfo response, IOCrossRefID xref, string strText)
- virtual string EncodeSendStoredCDR (CSTAResponseInfo response, string cdrCrossRefID)
- virtual string EncodeSetAgentState (CSTAResponseInfo response, string agentDevice, ReqAgentState agentState, string agentID)
- virtual string EncodeSetDisplay (CSTAResponseInfo response, string targetDevice, string text)
- virtual string EncodeSetDND (CSTAResponseInfo response, string targetDevice, bool dndOn)
- virtual string EncodeSetMsgWaiting (CSTAResponseInfo response, string targetDevice, bool indicatorOn)
- virtual string EncodeSetOrClearFwdRequest (CSTAResponseInfo response, string fromDevice, bool fwdOn, ForwardingType fwdType, string toDevice)
- virtual string EncodeSetRingerStatus (CSTAResponseInfo response, string targetDevice, string targetRinger, RingMode rm, long ringPattern)
- virtual string EncodeSingleStepTransfer (CSTAResponseInfo response, ConnectionID callToTransfer, string transferToDevice)
- virtual string EncodeSnapshotCall (CSTAResponseInfo response, ConnectionID callToSnapshot)
- virtual string EncodeSnapshotDevice (CSTAResponseInfo response, string targetDevice)
- virtual string EncodeStartCDRTrans (CSTAResponseInfo response, CDRTransferMode transferMode)
- virtual string EncodeStartDataPath (CSTAResponseInfo response, string targetDevice)
- virtual string EncodeStartSession (CSTAResponseInfo response, string applicationID)
- virtual string EncodeStopCDRTrans (CSTAResponseInfo response, string cdrCrossRefID)
- virtual string EncodeStopDataPath (CSTAResponseInfo response, IOCrossRefID xref)
- virtual string EncodeStopSession (CSTAResponseInfo response)
- virtual string EncodeTransferCall (CSTAResponseInfo response, ConnectionID initiatedCall, ConnectionID originalCall)
- byte [] RemoveBOM (byte [] message)
- string EncodeResetSession (CSTAResponseInfo response)

- int InterpretResetSessionResponse (CSTAResponseInfo response)
- void InterpretStartSessionResponse (CSTAResponseInfo response)
- void InterpretStopSessionResponse (CSTAResponseInfo response)
- void InterpretXMLResponse (CSTAResponseInfo response)
- static void ResetSession (object arg)

Detailed Description

Implements CSTA phase 3 operations using XML edition 6. Note that most PBXes don't support all CSTA messages, so some methods in this class may result in an error status being returned by your PBX.

Definition at line 52 of file GenericXMLed6.cs

The Documentation for this struct was generated from the following file:

- GenericXMLed6.cs

virtual CSTAResponseInfo AcceptCall (ConnectionID callToAccept)

Accepts a call.

Table 3.1. Parameters

callToAccept	The ConnectionID of the call to accept.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo AnswerCall (ConnectionID callToAnswer)

Answers a call.

Table 3.2. Parameters

callToAnswer	The ConnectionID of the call to answer.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo AnswerCall (ConnectionID callToAnswer, string deviceToLift)

Answers a call.

Table 3.3. Parameters

callToAnswer	ConnectionID of an existing call (such as initiated through MakeCall()).
deviceToLift	The device (e.g., "800") that is to answer the call.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo ClearConnection (ConnectionID connectionToClear)

Clears a connection.

Table 3.4. Parameters

connectionToClear	The ConnectionID of the connection to clear.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo ClearDoNotDisturb (string targetDevice)

Turns off the Do Not Disturb functionality for a phone.

Table 3.5. Parameters

targetDevice	The device for which the Do Not Disturb functionality is to be turned off.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo ClearForwarding (string fromDevice, ForwardingType fwdType)

Clears the forwarding feature for a phone.

Table 3.6. Parameters

fromDevice	The device for which forwarding is to be cleared. This is the device from which calls are being forwarded to a different device.
fwdType	The type of forwarding to clear.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo ClearMessageWaiting (string targetDevice)

Turns off the message waiting indicator on a device's display.

Table 3.7. Parameters

targetDevice	The device for which the indicator is to be turned off.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo ConferenceCall (ConnectionID heldCall, ConnectionID activeCall)

Brings a held call into conference with an active call.

Table 3.8. Parameters

heldCall	The held call to be brought into conference.
activeCall	The active call.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo ConsultationCall (ConnectionID existingCall, string targetDevice)

Instruct the PBX or UA to do a consultation call.

Table 3.9. Parameters

existingCall	The connection id of the call for which the consultation call will be made.
targetDevice	Identifier (e.g., phone number) of the device that is the target of the consultation call.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo ConsultationCall (ConnectionID existingCall, string targetDevice, ConsultOptions options)

Instruct the PBX or UA to do a consultation call.

Table 3.10. Parameters

existingCall	The connection id of the call for which the consultation call will be made.
targetDevice	Identifier (e.g., phone number) of the device that is the target of the consultation call.
options	A ConsultOptions object.

Returns: . A CSTAResponseInfo object.

virtual string EncodeInvokeID (string initialMessage)

This method prepends an invoke ID to an already encoded XML CSTA message.

Table 3.11. Parameters

initialMessage	The XML CSTA message without the invoke ID.

Returns: . The message with the invoke ID prepended.

GenericXMLed6 (string pbxSystem, int port)

Constructs an instance associated with the given PBX identifier and port.

Table 3.12. Parameters

pbxSystem	Well-known name or IP address of the PBX.
port	Port on which the PBX listens for CSTA messages.

GenericXMLed6 (PBXSession sessionObject)

Constructs an instance associated with the given PBXSession object.

Table 3.13. Parameters

sessionObject	A PBXSession object.

virtual CSTAResponseInfo GetAgentState (string agent-Device)

Gets the state of the agent associated with a device.

Table 3.14. Parameters

agentDevice	The device associated with the agent.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo GetDoNotDisturb (string targetDevice)

Gets the Do Not Disturb setting for a phone.

Table 3.15. Parameters

targetDevice	The phone for which the Do Not Disturb setting is desired.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo GetLogicalDevInfo (string targetDevice)

Gets information about the logical element of a device.

Table 3.16. Parameters

targetDevice	The device for which the information is desired.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo GetPhysicalDevInfo (string targetDevice)

Gets information about the physical element of a device.

Table 3.17. Parameters

targetDevice	The device for which the information is desired.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo GetSFDevices (ReqDeviceCategory deviceCategory)

Sends a Get Switching Function Devices request to the PBX or UA.

Table 3.18. Parameters

deviceCategory	The category of device for which the list is desired.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo GetSFDevices ()

Sends a Get Switching Function Devices request to the PBX or UA.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo HoldCall (ConnectionID callToHold)

Instruct the PBX or UA to hold a call.

Table 3.19. Parameters

callToHold	The ConnectionID of the call to be held.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo MakeCall (string callingDevice, string calledDevice)

Instruct the PBX or UA to place a call.

Table 3.20. Parameters

callingDevice	Identifier (e.g., phone number) of the device making the call.
calledDevice	Identifier (e.g., phone number) of the device being called.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo MakeCall (string callingDevice, string calledDevice, bool autoOriginate)

Instruct the PBX or UA to place a call.

Table 3.21. Parameters

callingDevice	Identifier (e.g., phone number) of the device making the call.
calledDevice	Identifier (e.g., phone number) of the device being called.
autoOriginate	If true, the call will be answered automatically (if the PBX supports this feature). If false, the called device will alert.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo MonitorStart (string deviceToMonitor)

Issues a MonitorStart request to the PBX or UA to monitor a device.

Table 3.22. Parameters

deviceToMonitor	Identifier (e.g., telephone number) of the device to monitor.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo MonitorStart (ConnectionID callToMonitor)

Issues a MonitorStart request to the PBX or UA to monitor a call.

Table 3.23. Parameters

callToMonitor	The call to monitor.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo MonitorStop (string crossRefID)

Stop a previously started PBX or UA monitor request.

Table 3.24. Parameters

crossRefID	The cross reference id of the monitor request as a MonitorCrossRefID object.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo MonitorStopAtDevice (string monitoredDevice)

This method stops all monitors active against the indicated device, regardless of what thread started the monitor. The method will only stop monitors started through the MonitorStart() method.

Table 3.25. Parameters

monitoredDevice	The monitored device (e.g., extension).

Returns: . If no problems are encountered, the method returns a CSTAResponseInfo object containing the response from the PBX for the LAST MonitorStop message.

If any problems are encountered, the method returns a CSTAResponseInfo object containing information about the error, including any response from the PBX for the problematic MonitorStop message.

virtual CSTAResponseInfo RequestSystemStatus ()

Retrieves a system status from the PBX or UA.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo RetrieveCall (ConnectionID callToRetrieve)

Retrieves a held call.

Table 3.26. Parameters

callToRetrieve	The ConnectionID of the call to retrieve.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo RingDevice (string targetDevice, string targetRinger, long ringPattern)

Causes a telephony device to ring.

Table 3.27. Parameters

targetDevice	The device to ring.
targetRinger	The id of the ringer to use for the ring. This argument can be specified as a character string (e.g, "abc"), a hex string (e.g, "'010A05'H"), or a binary string (e.g, "'000000010000101000000101'B").
ringPattern	The indicator of the ring pattern to use.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SendData (IOCrossRefID xref, string text)

Sends a text message to a telephony device.

Table 3.28. Parameters

xref	An IOCrossRefID object, most likely obtained by a previous call to StartDataPath.
text	The text to send to the telephony device.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SendStoredCDR (string cdr-CrossRefID)

Issues a SendStoredCallDetailRecords request to the PBX or UA. A CDR callback method (see PBXSession.XMLCDRCallback) must be defined in order to receive CDR messages.

Table 3.29. Parameters

cdrCrossRefID	The CDR cross reference id that was returned in the response to a previously issued StartC-DRTransmission() call.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SetAgentState (string agent-Device, ReqAgentState agentState, string agentID)

Sets the state of an agent associated with a device.

Table 3.30. Parameters

agentDevice	The device associated with the agent.
agentState	The desired state for the agent.
agentID	The agent id.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SetAgentState (string agent-Device, ReqAgentState agentState)

Sets the state of an agent associated with a device.

Table 3.31. Parameters

agentDevice	The device associated with the agent.
agentState	The desired state for the agent.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SetDisplay (string targetDevice, string text)

Sends text to a telephony device's display

Table 3.32. Parameters

targetDevice	The device to which the text is to be sent.
text	The text to be sent.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SetDoNotDisturb (string targetDevice)

Sets the Do Not Disturb feature for a phone.

Table 3.33. Parameters

targetDevice	The device for which Do Not Disturb is to be set.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SetForwarding (string fromDevice, ForwardingType fwdType, string toDevice)

Sets the forwarding feature for a phone.

Table 3.34. Parameters

fromDevice	The device for which forwarding is to be set. This is the device from which calls are to be forwarded to a different device.
fwdType	The type of forwarding to set.
toDevice	The device to which calls are to be forwarded from the device indicated by the fromDevice parameter.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SetMessageWaiting (string targetDevice)

Turns on the message waiting indicator on a device's display.

Table 3.35. Parameters

targetDevice	The device for which the indicator is to be turned on.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SingleStepTransfer (ConnectionID callToTransfer, string transferToDevice)

Perform a single step transfer.

Table 3.36. Parameters

callToTransfer	A ConnectionID object that identifies the call to be transferred.
transferToDevice	The device (e.g., "101") to which the call is to be transferred.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SnapshotCall (ConnectionID callToSnapshot)

Instruct the PBX or UA to take a snapshot of a call.

Table 3.37. Parameters

callToSnapshot	The ConnectionID of the call for which the snapshot is desired.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo SnapshotDevice (string deviceToSnapshot)

Instruct the PBX or UA to take a snapshot of calls active at a device.

Table 3.38. Parameters

deviceToSnapshot	Identifier (e.g., phone number) of the device for which the snapshot is desired.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo StartCDRTransmission (CDRTransferMode transferMode)

Issues a StartCallDetailRecordsTransmission request to the PBX or UA. A CDR callback method (see PBXSession.XMLCDRCallback) must be defined in order to receive CDR messages.

Table 3.39. Parameters

transferMode	Indicates how the PBX is to transfer the CDR information.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo StartDataPath (string targetDevice)

Opens up a data path to a specified device.

Table 3.40. Parameters

targetDevice	Specifies the device to which a data path is to be opened.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo StartSession ()

Establish a session with the PBX, using "CSTADLL" as the application identifier.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo StartSession (string applicationID)

Establish a session with the PBX.

Table 3.41. Parameters

applicationID	A free text string to identify the application.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo StopCDRTransmission (string cdrCrossRefID)

Issues a StopCallDetailRecordsTransmission request to the PBX or UA.

Table 3.42. Parameters

cdrCrossRefID	The CDR cross reference id that was returned in the response to a previously issued StartCDRTransmission() call.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo StopDataPath (IOCrossRefID xref)

Stops a previously established data path

Table 3.43. Parameters

xref	An IOCrossRefID object, most likely obtained from a previous call to StartDataPath.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo StopRing (string targetDevice, string targetRinger, long ringPattern)

Stops a ringer on a telephony device.

Table 3.44. Parameters

targetDevice	The device for which the ringer is to stop.
targetRinger	The id of the ringer to stop. This argument can be specified as a character string (e.g, "abc"), a hex string (e.g, "'010A05'H"), or a binary string (e.g, "'000000010000101000000101'B").
ringPattern	The indicator of the ring pattern to stop.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo StopSession ()

Stops a session with a PBX. The TCP/IP connection to the PBX will be terminated.

Returns: . A CSTAResponseInfo object.

virtual CSTAResponseInfo TransferCall (ConnectionID initiatedCall, ConnectionID originalCall)

Transfers a call. A consultation call must be done before calling this method.

Table 3.45. Parameters

initiatedCall	ConnectionID of the new call initiated by the consultation call. The initiatedCall member of the ConsultationCallResponse_ELEM class, for example, contains this ConnectionID.

originalCall	ConnectionID of the original call. The somewhat confusingly named callingDevice member of the MakeCallResponse_ELEM class contains this ConnectionID, as does the established-Connection member of the EstablishedEvent_ELEM class.

Returns: . A CSTAResponseInfo object.

virtual string EncodeAcceptCall (CSTAResponseInfo response, ConnectionID callToAccept)

Encodes an AcceptCall message.

Table 3.46. Parameters

response	A CSTAResponseInfo object.
callToAccept	The ConnectionID of the call to accept.

Returns: . The encoded message.

virtual string EncodeAnswerCall (CSTAResponseInfo response, ConnectionID callToAnswer)

Encodes an AnswerCall message.

Table 3.47. Parameters

response	A CSTAResponseInfo object.
callToAnswer	The ConnectionID of the call to answer.

Returns: . The encoded message.

virtual string EncodeAnswerCall (CSTAResponseInfo response, ConnectionID callToAnswer, string deviceToLift)

Encodes an AnswerCall message.

Table 3.48. Parameters

response	A CSTAResponseInfo object.
callToAnswer	The ConnectionID of the call to answer.
deviceToLift	The device (e.g., "800") that is to answer the call.

Returns: . The encoded message.

virtual string EncodeClearConnection (CSTAResponseInfo response, ConnectionID connectionToClear)

Encodes a ClearConnection message.

Table 3.49. Parameters

response	A CSTAResponseInfo object.
connectionToClear	The ConnectionID of the connection to clear.

Returns: . The encoded message.

virtual string EncodeConferenceCall (CSTAResponseInfo response, ConnectionID heldCall, ConnectionID activeCall)

Encodes a ConferenceCall message.

Table 3.50. Parameters

response	A CSTA ResponseInfo object.
heldCall	ConnectionID of the held call to be retrieved.
activeCall	ConnectionID of the active call.

Returns: . The length of the encoded message, or -1 if an error occurred.

virtual string EncodeConsultationCall (CSTAResponseInfo response, ConnectionID existingCall, string targetDevice, ConsultOptions options)

Encodes a ConsultationCall message.

Table 3.51. Parameters

response	A CSTAResponseInfo object.
existingCall	The connection id of the call for which the consultation call will be made.
targetDevice	Identifier (e.g., phone number) of the device that is the target of the consultation call.
options	A ConsultOptions object.

Returns: . The encoded message.

virtual string EncodeGetAgentState (CSTAResponseInfo response, string targetDevice)

Encodes a GetAgentState message.

Table 3.52. Parameters

response	A CSTAResponseInfo object.
targetDevice	The device whose agent state is desired.

Returns: . The encoded message.

virtual string EncodeGetDND (CSTAResponseInfo response, string targetDevice)

Encodes a GetDoNotDisturb message.

Table 3.53. Parameters

response	A CSTAResponseInfo object.
targetDevice	The phone for which the Do Not Disturb setting is desired.

Returns: . The encoded message.

virtual string EncodeGetLogicalDevInfo (CSTAResponseInfo response, string targetDevice)

Encodes a GetLogicalDeviceInformation message.

Table 3.54. Parameters

response	A CSTAResponseInfo object.
targetDevice	The device for which the information is needed.

Returns: . The encoded message.

virtual string EncodeGetPhysicalDevInfo (CSTAResponseInfo response, string targetDevice)

Encodes a GetPhysicalDeviceInformation message.

Table 3.55. Parameters

response	A CSTAResponseInfo object.
targetDevice	The device for which the information is needed.

Returns: . The encoded message.

virtual string EncodeGetSFDevices (CSTAResponseInfo response, ReqDeviceCategory category)

Encodes a GetSwitchingFunctionDevices message.

Table 3.56. Parameters

response	A CSTAResponseInfo object.
category	The category of device for which the list is desired.

Returns: . The encoded message.

virtual string EncodeHoldCall (CSTAResponseInfo response, ConnectionID callToHold)

Encodes a HoldCall message.

Table 3.57. Parameters

response	A CSTAResponseInfo object.
callToHold	The ConnectionID object for the call to put on hold.

Returns: . The encoded message.

virtual string EncodeMakeCall (CSTAResponseInfo response, string callingDevice, string calledDevice, bool autoOriginate)

Encodes a MakeCall message.

Table 3.58. Parameters

response	A CSTAResponseInfo object.
callingDevice	The device that is making the call.
calledDevice	The device that is being called.
autoOriginate	If true, the call will be answered automatically (if the PBX supports this feature). If false, the called device will alert.

Returns: . The encoded message.

virtual string EncodeMonitorStart (CSTAResponseInfo response, string targetDevice)

Encodes a MonitorStart message to monitor a device.

Table 3.59. Parameters

response	A CSTAResponseInfo object.
targetDevice	Identifier (e.g., telephone number) of the device to monitor.

Returns: . The encoded message.

virtual string EncodeMonitorStart (CSTAResponseInfo response, ConnectionID targetCall)

Encodes a MonitorStart message to monitor a call.

Table 3.60. Parameters

response	A CSTAResponseInfo object.
targetCall	The call to monitor.

Returns: . The encoded message.

virtual string EncodeMonitorStop (CSTAResponseInfo response, string xref)

Encodes a MonitorStop message.

Table 3.61. Parameters

response	A CSTAResponseInfo object.
xref	The cross reference id of the monitor request as a MonitorCrossRefID object.

Returns: . The encoded message.

virtual string EncodeRequestSystemStatus (CSTAResponseInfo response)

Encodes a RequestSystemStatus message.

Table 3.62. Parameters

response	A CSTAResponseInfo object.

Returns: . The encoded message.

virtual string EncodeRetrieveCall (CSTAResponseInfo response, ConnectionID callToRetrieve)

Encodes a RetrieveCall message.

Table 3.63. Parameters

response	A CSTAResponseInfo object.
callToRetrieve	The ConnectionID of the call to retrieve.

Returns: . The encoded message.

virtual string EncodeSendData (CSTAResponseInfo response, IOCrossRefID xref, string strText)

Encodes a SendData message.

Table 3.64. Parameters

response	A CSTAResponseInfo object.
xref	An IOCrossRefID object, most likely obtained by a previous call to StartDataPath.
strText	The text to send to the telephony device.

Returns: . The encoded message.

virtual string EncodeSendStoredCDR (CSTAResponseInfo response, string cdrCrossRefID)

Encodes a SendStoredCallDetailRecords message.

Table 3.65. Parameters

response	A CSTAResponseInfo object.
cdrCrossRefID	The CDR cross reference id that was returned in the response to a previously issued StartCDRTransmission() call.

Returns: . The encoded message.

virtual string EncodeSetAgentState (CSTAResponseInfo response, string agentDevice, ReqAgentState agentState, string agentID)

Encodes a SetAgentState message.

Table 3.66. Parameters

response	A CSTAResponseInfo object.
agentDevice	The device associated with the agent.
agentState	An ReqAgentState object indicating the desired state of the agent.
agentID	The agent id.

Returns: . The encoded message.

virtual string EncodeSetDisplay (CSTAResponseInfo response, string targetDevice, string text)

Encodes a Set Display message.

Table 3.67. Parameters

response	A CSTAResponseInfo object.
targetDevice	The device to which the text is to be sent.
text	The text to be sent.

Returns: . The encoded message.

virtual string EncodeSetDND (CSTAResponseInfo response, string targetDevice, bool dndOn)

Encodes a SetDoNotDisturb message.

Table 3.68. Parameters

response	A CSTAResponseInfo object.
targetDevice	The device for which Do Not Disturb is to be set or cleared.
dndOn	If true, indicates that Do Not Disturb is to be turned on. If false, indicates that Do Not Disturb is to be turned off.

Returns: . The encoded message.

virtual string EncodeSetMsgWaiting (CSTAResponseInfo response, string targetDevice, bool indicatorOn)

Encodes a SetMessageWaiting message.

Table 3.69. Parameters

response	A CSTAResponseInfo object.
targetDevice	The device for which the message waiting indicator is to be turned on or off.
indicatorOn	If true, indicates that the message waiting indicator is to be turned on. If false, indicates that the message waiting indicator is to be turned off.

Returns: . The encoded message.

virtual string EncodeSetOrClearFwdRequest (CSTAResponseInfo response, string fromDevice, bool fwdOn, ForwardingType fwdType, string toDevice)

Encodes a SetForwarding message to set or clear forwarding.

Table 3.70. Parameters

response	A CSTAResponseInfo object.
fromDevice	The device for which forwarding is to be set or cleared. In the case of setting forwarding, this is the device from which calls are to be forwarded to a different device. In the case of clearing forwarding, this is the device from which calls are currently being forwarded to a different device.
fwdOn	If true, forwarding will be turned on. If false, forwarding will be turned off.
fwdType	The type of forwarding to set or clear.
toDevice	The device to which calls are to be forwarded from the device indicated by the fromDevice parameter. If forwarding is being cleared, this parameter is ignored and can be set to null.

Returns: . The encoded message.

virtual string EncodeSetRingerStatus (CSTAResponseInfo response, string targetDevice, string targetRinger, RingMode rm, long ringPattern)

Encodes a SetRingerStatus message.

Table 3.71. Parameters

response	A CSTAResponseInfo object.
targetDevice	The device to ring.
targetRinger	The id of the ringer to use for the ring. This argument can be specified as a character string (e.g, "abc"), a hex string (e.g, '"010A05'H"), or a binary string (e.g, '"000000010000101000000101'B").
rm	A RingMode instance that indicates either ringing or notRinging.
ringPattern	The indicator of the ring pattern to use.

Returns: . The encoded message.

virtual string EncodeSingleStepTransfer (CSTAResponseInfo response, ConnectionID callToTransfer, string transferToDevice)

Encodes a SingleStepTransfer message.

Table 3.72. Parameters

response	A CSTAResponseInfo object.
callToTransfer	A ConnectionID object that identifies the call to be transferred.
transferToDevice	The device (e.g., "101") to which the call is to be transferred.

Returns: . The encoded message.

virtual string EncodeSnapshotCall (CSTAResponseInfo response, ConnectionID callToSnapshot)

Encodes a SnapshotCall message.

Table 3.73. Parameters

response	A CSTAResponseInfo object.
callToSnapshot	The ConnectionID of the call for which the snapshot is desired.

Returns: . The encoded message.

virtual string EncodeSnapshotDevice (CSTAResponseInfo response, string targetDevice)

Encodes a SnapshotDevice message.

Table 3.74. Parameters

response	A CSTAResponseInfo object.
targetDevice	Identifier (e.g., phone number) of the device for which the snapshot is desired.

Returns: . The encoded message.

virtual string EncodeStartCDRTrans (CSTAResponseInfo response, CDRTransferMode transferMode)

Encodes a StartCallDetailRecordsTransmission message.

Table 3.75. Parameters

response	A CSTAResponseInfo object.
transferMode	The mode the PBX or UA is to use to transfer call detail records.

Returns: . The encoded message.

virtual string EncodeStartDataPath (CSTAResponseInfo response, string targetDevice)

Encodes a StartDataPath message.

Table 3.76. Parameters

response	A CSTAResponseInfo object.
targetDevice	The device against which a data path is to be started.

Returns: . The encoded message.

virtual string EncodeStartSession (CSTAResponseInfo response, string applicationID)

Encodes a StartApplicationSession message.

Table 3.77. Parameters

response	A CSTAResponseInfo object.
applicationID	The application id to be encoded into the message.

Returns: . The encoded message.

virtual string EncodeStopCDRTrans (CSTAResponseInfo response, string cdrCrossRefID)

Encodes a StopCallDetailRecordsTransmission message.

Table 3.78. Parameters

response	A CSTAResponseInfo object.
cdrCrossRefID	The CDR cross reference id that was returned in the response to a previously issued StartCDRTransmission() call.

Returns: . The encoded message.

virtual string EncodeStopDataPath (CSTAResponseInfo response, IOCrossRefID xref)

Encodes a StopDataPath message.

Table 3.79. Parameters

response	A CSTAResponseInfo object.
xref	An IOCrossRefID object, most likely obtained from a previous call to StartDataPath.

Returns: . The encoded message.

virtual string EncodeStopSession (CSTAResponseInfo response)

Encodes a StopApplicationSession message.

Table 3.80. Parameters

response	A CSTAResponseInfo object.

Returns: . The encoded message.

virtual string EncodeTransferCall (CSTAResponseInfo response, ConnectionID initiatedCall, ConnectionID originalCall)

Encodes a TransferCall message.

Table 3.81. Parameters

response	A CSTAResponseInfo object.
initiatedCall	ConnectionID of the new call initiated by the consultation call. The initiatedCall member of the ConsultationCallResponse_ELEM class, for example, contains this ConnectionID.
originalCall	ConnectionID of the original call. The somewhat confusingly named callingDevice member of the MakeCallResponse_ELEM class contains this ConnectionID, as does the established-Connection member of the EstablishedEvent_ELEM class.

Returns: . The encoded message.

Com::Objsys::Csta::Xmled6::LicenseException class Reference

- LicenseException (string message)

Detailed Description

Defines an exception that occurs while trying to find license information.

Definition at line 53 of file LicenseException.cs

The Documentation for this struct was generated from the following file:

- LicenseException.cs

Com::Objsys::Csta::Xmled6::LicenseHelper class Reference

- static object licenseLO
- static ushort sessionCount

- static bool CheckLicenseSettings (LicenseOptions.BERPhases phase)
- static bool CheckLicenseSettings (LicenseOptions.XMLEditions edition)
- static void FreeLicense (bool close)
- static string GenNotEnabledMsg (Constants.Encoding encoding, ushort phaseOrEdition)
- static void HandleException (PBXSession sessionObject, string text)

static bool CheckLicenseSettings (LicenseOptions.BERPhases phase)

Returns true if the indicated BER phase is enabled in the license, false otherwise.

Table 3.82. Parameters

phase	Indicates the BER phase number.

Returns: . True if the capability is enabled in the license, false otherwise.

static bool CheckLicenseSettings (LicenseOptions.XMLEditions edition)

Returns true if the indicated XML edition is enabled in the license, false otherwise.

Table 3.83. Parameters

edition	Indicates the XML edition number.

Returns: . True if the capability is enabled in the license, false otherwise.

static void FreeLicense (bool close)

This method checks in a license.

Table 3.84. Parameters

close	Indicates whether to close the RLM handle if RLM is being used.

static string GenNotEnabledMsg (Constants.Encoding encoding, ushort phaseOrEdition)

Returns a string indicating that a particular capability is not enabled in the user's CSTADLL license.

Table 3.85. Parameters

encoding	Indicates BER or XML encoding.
phaseOrEdition	Indicates the phase number (for BER) or the edition number (for XML)

Returns: . The "not enabled" message as a string.

static void HandleException (PBXSession sessionObject, string text)

This method either throws an exception or invokes an exception callback.

Table 3.86. Parameters

sessionObject	The PBXSession object.
text	The text of the exception message.

Com::Objsys::Csta::Xmled6::LicenseOptions class Reference

- enum BERPhases {
  BERPhase1,
  BERPhase2,
  BERPhase3
  }
- enum XMLEditions {
  XMLEdition3,
  XMLEdition4,
  XMLEdition5,
  XMLEdition6
  }

Private Attributes

- bool [] enabledBERPhases
- bool [] enabledXMLEditions
-
-

Detailed Description

This class holds booleans that define what capabilities are defined in the license.

Definition at line 54 of file LicenseOptions.cs

The Documentation for this struct was generated from the following file:

- LicenseOptions.cs

enum BERPhases

Values indices into the array of booleans that defines what BER phases are enabled in the license.

Enumerator:

BERPhase1

BERPhase2

BERPhase3

Definition at line 60 of file LicenseOptions.cs

```
{
BERPhase1,
BERPhase2,
BERPhase3,
}BERPhases;
```

enum XMLEditions

Values indices into the array of booleans that defines what XML editions are enabled in the license.

Enumerator:

XMLEdition3

XMLEdition4

XMLEdition5

XMLEdition6

Definition at line 83 of file LicenseOptions.cs

```
{
XMLEdition3,
XMLEdition4,
XMLEdition5,
XMLEdition6,
}XMLEditions;
```

Com::Objsys::Csta::Xmled6::PBXSession class Reference

Private Attributes

- bool asyncReadInProgress
- Constants.CallbackInvocationMechanisms callbackInvocationMechanism
- AsyncCallback cdrCallback
- AsyncCallback clientCallback
- Constants.CommunicationTypes commType

- bool connected
- ConnectionCallback connectionLostCallback
- bool debugMode
- string deviceType
- bool discardOldResponses
- AsyncExceptionCallback exceptionCallback
- long lastInvokeId
- List< long > lateInvokeIds
- int maxReceiveTimeout
- Constants.Encoding messageEncoding
- Constants.PBXModels pbxModel
- Socket pbxSocket
- string pbxSystem
- ushort phase
- int port
- AsyncCallback systemStatusCallback
- string uaSIPContact
- XMLAsyncCallback xmlCDRCallback
- XMLAsyncCallback xmlClientCallback
- ushort xmlEdition
- bool xmlImmediateSystemStatus
- string xmlLastInvokeId
- List< string > xmlLateInvokeIds
- CSTAContext xmlResetSessionContext
- EventWaitHandle xmlResetSessionTerminate
- string xmlSessionID
- CSTAContext xmlStopSessionContext
- XMLAsyncCallback xmlSystemStatusCallback
-

-
-
-
-
-

- static Dictionary< IntPtr, PBXSession > sessionList
- static object sessionLockObject

- delegate void AsyncCallback (PBXSession sessionObject, byte [] asyncData)
- delegate void AsyncExceptionCallback (PBXSession sessionObject, ApplicationException exception)
- void Close (CSTAContext threadContext)
- delegate void ConnectionCallback (PBXSession sessionObject)
- void Open (CSTAContext threadContext)
- PBXSession (string pbxSystem, int port)
- SocketState SendACSEMessage (byte [] message, int messageLength, Constants.ACSEMessageTypes messageType, CSTAContext threadContext)
- void SendMessage (byte [] message, int messageLength, CSTAContext threadContext)
- void SendMessage (string messageType, byte [] message, int messageLength, CSTAContext threadContext)
- void SendXMLMessage (string strMessage, CSTAContext threadContext)
- void SendXMLMessage (string messageType, string strMessage, CSTAContext threadContext)
- SocketState SendXMLSession (string strMessage, Constants.XMLSessionMessageTypes enmMessageType, CSTAContext threadContext)
- void WaitForROSEResponse (CSTAContext threadContext)
- void WaitForXMLResponse (CSTAContext threadContext)
- delegate void XMLAsyncCallback (PBXSession sessionObject, string message)

- static void Init ( )

- void AlcatelInit (CSTAContext threadContext)
- void Connect (CSTAContext threadContext)
- void SendMessageInternal (byte [] message, int messageLength, CSTAContext threadContext)
- void SendSIPAck (SocketState ss)

- void SendSIPBye (CSTAContext threadContext)
- void SendSIPHeaderBlock (string strHeaderBlock, CSTAContext threadContext)
- void SendSIPInfo (CSTAContext threadContext, int contentLength, bool ssResponse)
- SocketState SendSIPInvite (CSTAContext threadContext, string xmlMessage, uaSIPInvite inviteObject)
- void WaitForResetSessionResponse (CSTAContext threadContext)
- void WaitForStopSessionResponse (CSTAContext threadContext)
- void WaitForResponse (CSTAContext threadContext, EventWaitHandle waitHandle)

Detailed Description

This class manages communication with a PBX. One instance of this class should be created for each PBX with which a CSTADLL client application needs to exchange CSTA messages.

The CSTA worker classes (e.g., Alcatel4400, PanasonicNCP) hold a reference to a PBXSession object. If the constructor for the worker class that takes a PBX identification and a PBX port is used, a PBXSession object is created. Alternatively, the client application can create a PBXSession instance and pass a reference to the instance to the other worker class constructor signature.

Only one PBXSession instance for a PBX/port combination should be created. The behavior is undefined if multiple PBXSession instances are created for the same PBX and port.

Definition at line 74 of file PBXSession.cs

The Documentation for this struct was generated from the following file:

- PBXSession.cs

delegate void AsyncCallback (PBXSession sessionObject, byte[] asyncData)

Declaration of a callback function to be invoked when one of the following messages is received: a monitor event report message, a route message, a CDR Report message, a CDR Notification message, or a system status request.

Table 3.87. Parameters

sessionObject	The session object for the PBX that generated the asynchronous message.
asyncData	The data received asynchronously from the PBX.

delegate void AsyncExceptionCallback (PBXSession sessionObject, ApplicationException exception)

Declaration of a callback function to be invoked if a condition is encountered in the asynchronous I/O handler that would otherwise result in an exception being thrown. Note that in a couple of cases the asynchronous code will still throw an exception, even if this callback is defined.

Table 3.88. Parameters

sessionObject	The session object for the PBX that sent a packet that triggered an exception condition.
exception	The ApplicationException object that would have been thrown in the asynchronous I/O handling code if this callback were not defined.

void Close (CSTAContext threadContext)

Terminates the session to the PBX. This method can be used to terminate sessions with PBX devices that don't accept ACSE release association requests.

Table 3.89. Parameters

threadContext	The context object for the caling thread.

delegate void ConnectionCallback (PBXSession sessionObject)

Declaration of a callback function to be invoked if the connection to the PBX is lost.

Table 3.90. Parameters

sessionObject	The session object for the PBX whose connection was lost.

void Open (CSTAContext threadContext)

This method can be used to establish communication with a PBX device before any messages are actually sent to the device. TCP/IP connectivity is established and an asynchronous read is started to receive messages sent from the PBX.

Table 3.91. Parameters

threadContext	The thread context object.

PBXSession (string pbxSystem, int port)

Constructs a PBXSession object.

Table 3.92. Parameters

pbxSystem	The name or IP address of the PBX system.
port	The port on the PBX system to which the client is connecting.

SocketState SendACSEMessage (byte[] message, int messageLength, Constants.ACSEMessageTypes messageType, CSTAContext threadContext)

This method sends an ACSE message (either Make Association or Release Association) to the PBX and receives the response. This operation is done synchronously. If the Make Association needs to be done (usually it does), it must be done before any threads for sending and receiving CSTA messages are started.

This method is only intended to be used by client code that encodes its own ACSEMakeAssociation or ACSEReleaseAssociation message. Most clients can probably use the MakeACSEAssociation() and ReleaseACSEAssociation() methods that are in each phase's helper classes.

Table 3.93. Parameters

message	An encoded ACSE Make Association or Release Association message.
messageLength	The length of the encoded message.
messageType	A constant telling whether the message is an ACSE Make Association or an ACSE Release Association.
threadContext	The thread context object.

Returns: . A populated SocketState instance.

void SendMessage (byte[] message, int messageLength, CSTAContext threadContext)

This method sends a message to the PBX using TCP/IP.

Table 3.94. Parameters

message	Byte array containing the encoded message to send.
messageLength	The length of the encoded message.
threadContext	The thread context object.

void SendMessage (string messageType, byte[] message, int messageLength, CSTAContext threadContext)

This method sends a message to the PBX using TCP/IP.

Table 3.95. Parameters

messageType	A string token to help identify the message in the CSTADLL log file.
message	Byte array containing the encoded message to send.
messageLength	The length of the encoded message.

threadContext	The thread context object.

void SendXMLMessage (string strMessage, CSTAContext threadContext)

This method sends an XML message to the PBX using TCP/IP.

Table 3.96. Parameters

strMessage	The XML message to send.
threadContext	The thread context object.

void SendXMLMessage (string messageType, string strMessage, CSTAContext threadContext)

This method sends an XML message to the PBX using TCP/IP.

Table 3.97. Parameters

messageType	A string token to help identify the message in the CSTADLL log file.
strMessage	The XML message to send.
threadContext	The thread context object.

SocketState SendXMLSession (string strMessage, Constants.XMLSessionMessageTypes enmMessageType, CSTAContext threadContext)

This method sends an XML session management (ECMA-354) message to the PBX.

Table 3.98. Parameters

strMessage	The text of the XML message to send.
enmMessageType	A constant indicating what kind of session management message is being sent.
threadContext	The thread context object.

Returns: . A populated SocketState instance if the message is a StartSession message. Null if the message is StopSession or ResetSession.

void WaitForROSEResponse (CSTAContext threadContext)

This method waits for a response to a CSTA message sent with a ROSE header.

Table 3.99. Parameters

threadContext	The CSTAContext object associated with the calling thread.

void WaitForXMLResponse (CSTAContext threadContext)

This method waits for a response to an XML CSTA message.

Table 3.100. Parameters

threadContext	The CSTAContext object associated with the calling thread.

delegate void XMLAsyncCallback (PBXSession sessionObject, string message)

Declaration of a callback function to be invoked when an asynchronous XML monitor event or route messae is received.

Table 3.101. Parameters

sessionObject	The session object for the PBX or UA that generated the asynchronous message.
message	The text of the message received asynchronously from the PBX or UA.

static void Init ()

Performs license initialization from the generated objects. This method gets called instead of SetKey() or SetKey2(). This is here in case a user just wants to use the generated classes and not the helper classes.

void AlcatelInit (CSTAContext threadContext)

Sends the initialization byte to an Alcatel PBX and receives the response.

Table 3.102. Parameters

threadContext	A context object.

void Connect (CSTAContext threadContext)

This method establishes TCP/IP connectivity to a PBX without starting an asynchronous read to receive messages sent by the PBX. If the PBX is an XML PBX or UA that immediately sends a System Status message once TCP/IP connectivity is established, this message will be received, and a response will be sent.

Table 3.103. Parameters

threadContext	A CSTAContext object.

void SendMessageInternal (byte[] message, int messageLength, CSTAContext threadContext)

This method sends a message to the PBX using TCP/IP.

Table 3.104. Parameters

message	Byte array containing the encoded message to send.
messageLength	The length of the encoded message.
threadContext	The thread context object.

void SendSIPAck (SocketState ss)

This method sends a SIP ACK header block to a UA using TCP/IP.

Table 3.105. Parameters

ss	The SocketState object.

void SendSIPBye (CSTAContext threadContext)

This method sends a SIP BYE header block to a UA using TCP/IP.

Table 3.106. Parameters

threadContext	The CSTAContext object.

void SendSIPHeaderBlock (string strHeaderBlock, CSTAContext threadContext)

This method sends a SIP header block to a UA using TCP/IP.

Table 3.107. Parameters

strHeaderBlock	The SIP header block to send.
threadContext	The thread context object.

void SendSIPInfo (CSTAContext threadContext, int contentLength, bool ssResponse)

This method sends a SIP INFO header block to a UA using TCP/IP

Table 3.108. Parameters

threadContext	The CSTAContext object for the thread.
contentLength	The length of the XML message that will follow this header block.
ssResponse	Indicates whether the message that will follow this INFO block is a System Status Response message.

void WaitForResetSessionResponse (CSTAContext threadContext)

This method waits for a response to a ResetSession message.

Table 3.109. Parameters

threadContext	The CSTAContext object associated with the calling thread.

void WaitForStopSessionResponse (CSTAContext threadContext)

This method waits for a response to a StopSession message.

Table 3.110. Parameters

threadContext	The CSTAContext object associated with the calling thread.

Com::Objsys::Csta::Xmled6::PBXSessionException class Reference

- PBXSessionException (string message)

Detailed Description

Defines an exception that occurs while communicating with a PBX.

Definition at line 52 of file PBXSessionException.cs

The Documentation for this struct was generated from the following file:

- PBXSessionException.cs

Com::Objsys::Csta::Xmled6::PBXSessionHelper class Reference

-
-
-
-
-

- static List< string > routeMessageTags
- static object routeMessageTagsLO

- static void HandleException (PBXSession sessionObject, string text)
- static void HandleSocketException (SocketException se, PBXSession sessionObject, Socket pbxSocket, ushort location)
- static void QueueNextRead (bool newSocketState, SocketState ss, PBXSession sessionObject)
- static void Read_Callback (IAsyncResult ar)

- static void HandleSIP (PBXSession sessionObject, SocketState ss, Socket pbxSocket, IAsyncResult ar)
- static void HandleXML (PBXSession sessionObject, SocketState ss, Socket pbxSocket, IAsyncResult ar)

Detailed Description

This class holds static properties that affect all PBX sessions.

Definition at line 57 of file PBXSessionHelper.cs

The Documentation for this struct was generated from the following file:

- PBXSessionHelper.cs

Com::Objsys::Csta::Xmled6::PBXSessionHelperEd6 class Reference

- static string EncodeSSResponse (PBXSession sessionObject)

- static void HandleXMLEd6 (PBXSession sessionObject, SocketState ss, Socket pbxSocket, IAsyncResult ar)
- static XMLParseInfo ParseXML (SocketState ss, PBXSession sessionObject)
- static string EncodeCDRNotificationAck (PBXSession sessionObject)
- static string EncodeCDRReportAck (PBXSession sessionObject)

Detailed Description

This class contains utility methods used by PBXSessionHelper for XML edition 6.

Definition at line 50 of file PBXSessionHelperEd6.cs

The Documentation for this struct was generated from the following file:

- PBXSessionHelperEd6.cs

static string EncodeSSResponse (PBXSession sessionObject)

Encodes an empty XML edition 6 SystemStatusResponse message.

Table 3.111. Parameters

sessionObject	The PBXSession object that describes the session with the PBX.

Returns: . The encoded SystemStatusResponse message.

static void HandleXMLEd6 (PBXSession sessionObject, SocketState ss, Socket pbxSocket, IAsyncResult ar)

Handles an XML edition 6 message.

Table 3.112. Parameters

sessionObject	The PBXSession object.
ss	The SocketState object.
pbxSocket	The Socket object.
ar	Object that conforms to the IAsyncResult interface.

static XMLParseInfo ParseXML (SocketState ss, PBXSession sessionObject)

Does initial parsing of an XML CSTA edition 6 message.

Table 3.113. Parameters

ss	The SocketState object that contains the message.
sessionObject	The PBXSession object that describes the session with the PBX.

Returns: . An XMLParseInfo object.

Com::Objsys::Csta::Xmled6::ResetSessionInfo class Reference

Private Attributes

- int initialSleepTime
- PBXSession sessionObject
-
-

Detailed Description

This class provides information that needs to be passed to the thread that periodically sends an XML ResetSession message to the PBX.

Definition at line 52 of file ResetSessionInfo.cs

The Documentation for this struct was generated from the following file:

- ResetSessionInfo.cs

Com::Objsys::Csta::Xmled6::SocketState class Reference

Private Attributes

- byte [] ackBuffer
- int ackLength
- int allocatedLength
- uint asn1Tag
- int bytesRequested
- uint classForm

- Socket comSocket
- int currentHeaderBegin
- int fragmentLength
- uint idCode
- bool ietfLengthRequested
- object invokeID
- bool isIetfLengthComplete
- bool isLengthComplete
- bool isLengthStarted
- bool isMessageComplete
- bool isSIPBlockComplete
- bool isTagComplete
- bool isTagStarted
- bool isXMLPrefixComplete
- bool lastSIPCRLF
- byte [] readBuffer
- List< byte[]> readBuffers
- bool secondIetfByteRequested
- int sipContentLength
- string sipMessageType
- uint sipSequenceNumber
- ushort tagReadCount
- CSTAContext threadContext
- ushort threadNumber
- int totalLength
- byte [] xmlHeader
-
-
-

- const ushort ASN_K_MEMBUFINITIAL

- void Reset ( )
- SocketState ( )
- void ValidateBufferLength ( )

Detailed Description

This class contains the response received from the PBX and state information about the exchange with the PBX that is used internally by CSTADLL.

Definition at line 57 of file SocketState.cs

The Documentation for this struct was generated from the following file:

- SocketState.cs

const ushort ASN_K_MEMBUFINITIAL

The amount of memory we initially allocate for the receive buffer.

Definition at line 450 of file SocketState.cs

The Documentation for this struct was generated from the following file:

- SocketState.cs

void Reset ()

Resets the object for re-use.

SocketState ()

Default constructor. Creates an EventWaitHandle object.

void ValidateBufferLength ()

Checks to see if the read buffer is large enough to receive the expected fragment. If it isn't, we reallocate.

Com::Objsys::Csta::Xmled6::uaSIPInvite class Reference

Private Attributes

- string callId
- string contact
- string from
- string inviteTarget

- ushort maxForwards
- string via

Detailed Description

This class holds information needed in order to do a SIP INVITE to a UA for a uaCSTA session.

Definition at line 52 of file uaSIPInvite.cs

The Documentation for this struct was generated from the following file:

- uaSIPInvite.cs

Com::Objsys::Csta::Xmled6::uaXMLed6 class Reference

- virtual void Bye ( )
- virtual CSTAResponseInfo Invite (uaSIPInvite inviteObject)
- sealed override CSTAResponseInfo StartSession (string applicationID)
- sealed override CSTAResponseInfo StartSession ( )
- sealed override CSTAResponseInfo StopSession ( )
- uaXMLed6 (string ua, int port)
- uaXMLed6 (PBXSession sessionObject)

Detailed Description

Implements uaCSTA phase 3 operations using XML edition 6. Note that most PBXes don't support all CSTA messages, so some methods in this class may result in an error status being returned by your PBX.

Definition at line 47 of file uaXMLed6.cs

The Documentation for this struct was generated from the following file:

- uaXMLed6.cs

virtual void Bye ()

Sends a SIP BYE message to a UA and closes TCP/IP communication with the UA.

virtual CSTAResponseInfo Invite (uaSIPInvite inviteObject)

Sends a SIP INVITE message to a User Agent.

Table 3.114. Parameters

inviteObject	A uaSIPInvite object.

Returns: . A CSTAResponseInfo object.

sealed override CSTAResponseInfo StartSession (string applicationID)

For regular XML CSTA, the StartSession() method starts communication with a PBX. But for uaCSTA with SIP, its use is not valid. The Invite() method must be used instead. This method will throw an exception.

Table 3.115. Parameters

applicationID	Ignored.

Returns: . Nothing. An exception is thrown.

sealed override CSTAResponseInfo StartSession ()

For regular XML CSTA, the StartSession() method starts communication with a PBX. But for uaCSTA with SIP, its use is not valid. The Invite() method must be used instead. This method will throw an exception.

Returns: . Nothing. An exception is thrown.

sealed override CSTAResponseInfo StopSession ()

For regular XML CSTA, the Stp[Session() method stops communication with a PBX. But for uaCSTA with SIP, its use is not valid. The Bye() method must be used instead. This method will throw an exception.

Returns: .

uaXMLed6 (string ua, int port)

Constructs an instance associated with the given UA identifier and port.

Table 3.116. Parameters

ua	Well-known name or IP address of the SIP user agent.
port	Port on which the UA listens for uaCSTA messages.

uaXMLed6 (PBXSession sessionObject)

Constructs an instance associated with the given PBXSession object.

Table 3.117. Parameters

sessionObject	A PBXSession object.

Com::Objsys::Csta::Xmled6::XMLParseInfo class Reference

Private Attributes

- string invokeID
- bool isCDRNotification
- bool isCDRReport
- bool isMonitorEvent
- bool isResetSessionResponse
- bool isResponse
- bool isRouteMessage
- bool isStopSessionResponse
- bool isSystemStatus
- string messageTag
- string messageText
- CSTAContext threadContext
-
-
-
-
-

-
-
-
-
-
-
-

Chapter 4. File Documentation

_SeqOfFloatLicProductInfo.cs File Reference

Detailed Description

Definition in file _SeqOfFloatLicProductInfo.cs

Alcatel4400.cs File Reference

Detailed Description

Definition in file Alcatel4400.cs

AlcatelOXE.cs File Reference

Detailed Description

Definition in file AlcatelOXE.cs

AlcatelOXO.cs File Reference

Detailed Description

Definition in file AlcatelOXO.cs

Constants.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::Constants

Namespaces

- struct Com::Objsys::Csta::Xmled6
- struct System
- struct System::Collections::Generic
- struct System::Text

Detailed Description

Definition in file Constants.cs

CSTAContext.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::CSTAContext

Namespaces

- struct Com::Objsys::Csta::Xmled6
- struct System::Threading

Detailed Description

Definition in file CSTAContext.cs

CSTAEncDec.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::CSTAEncDec

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file CSTAEncDec.cs

CSTAResponseInfo.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::CSTAResponseInfo

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file CSTAResponseInfo.cs

FloatLicInfo.cs File Reference

Detailed Description

Definition in file FloatLicInfo.cs

FloatLicProductInfo.cs File Reference

Detailed Description

Definition in file FloatLicProductInfo.cs

GenericXMLed6.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::GenericXMLed6

Namespaces

- struct Com::Objsys::Csta::Xmled6
- struct com::objsys::xbinder::runtime
- struct System::IO
- struct System::Linq
- struct System::Threading::Tasks
- struct System::Xml

Detailed Description

Definition in file GenericXMLed6.cs

LicenseBitFlags.cs File Reference

Detailed Description

Definition in file LicenseBitFlags.cs

LicenseChoice.cs File Reference

Detailed Description

Definition in file LicenseChoice.cs

LicenseChoice_hosts.cs File Reference

Detailed Description

Definition in file LicenseChoice_hosts.cs

LicenseData.cs File Reference

Detailed Description

Definition in file LicenseData.cs

LicenseData_licProcIds.cs File Reference

Detailed Description

Definition in file LicenseData_licProcIds.cs

LicensedProduct.cs File Reference

Detailed Description

Definition in file LicensedProduct.cs

LicenseException.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::LicenseException

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file LicenseException.cs

LicenseHelper.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::LicenseHelper

Namespaces

- struct Com::Objsys::Csta::Xmled6
- struct System::Reflection

Detailed Description

Definition in file LicenseHelper.cs

LicenseHost.cs File Reference

Detailed Description

Definition in file LicenseHost.cs

LicenseHost_id.cs File Reference

Detailed Description

Definition in file LicenseHost_id.cs

LicenseOptions.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::LicenseOptions

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file LicenseOptions.cs

LicenseUserInfo.cs File Reference

Detailed Description

Definition in file LicenseUserInfo.cs

LicenseValidityPeriod.cs File Reference

Detailed Description

Definition in file LicenseValidityPeriod.cs

PanasonicKXNS.cs File Reference

Detailed Description

Definition in file PanasonicKXNS.cs

PanasonicKXTDA.cs File Reference

Detailed Description

Definition in file PanasonicKXTDA.cs

PanasonicKXTDE.cs File Reference

Detailed Description

Definition in file PanasonicKXTDE.cs

PanasonicNCP.cs File Reference

Detailed Description

Definition in file PanasonicNCP.cs

PanasonicNXS.cs File Reference

Detailed Description

Definition in file PanasonicNXS.cs

PBXSession.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::PBXSession

Namespaces

- struct Com::Objsys::Csta::Xmled6
- struct System::Net
- struct System::Net::Sockets

Detailed Description

Definition in file PBXSession.cs

PBXSessionException.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::PBXSessionException

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file PBXSessionException.cs

PBXSessionHelper.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::PBXSessionHelper

Namespaces

- struct Com::Objsys::Csta::Xmled6
- struct System::Runtime::InteropServices

Detailed Description

Definition in file PBXSessionHelper.cs

PBXSessionHelperEd3.cs File Reference

Detailed Description

Definition in file PBXSessionHelperEd3.cs

PBXSessionHelperEd4.cs File Reference

Detailed Description

Definition in file PBXSessionHelperEd4.cs

PBXSessionHelperEd5.cs File Reference

Detailed Description

Definition in file PBXSessionHelperEd5.cs

PBXSessionHelperEd6.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::PBXSessionHelperEd6

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file PBXSessionHelperEd6.cs

PBXSessionHelperPhase1.cs File Reference

Detailed Description

Definition in file PBXSessionHelperPhase1.cs

PBXSessionHelperPhase2.cs File Reference

Detailed Description

Definition in file PBXSessionHelperPhase2.cs

PBXSessionHelperPhase3.cs File Reference

Detailed Description

Definition in file PBXSessionHelperPhase3.cs

PhilipsSopho.cs File Reference

Detailed Description

Definition in file PhilipsSopho.cs

ResetSessionInfo.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::ResetSessionInfo

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file ResetSessionInfo.cs

ROSEParseInfo.cs File Reference

Detailed Description

Definition in file ROSEParseInfo.cs

RunTimeFloatLicInfo.cs File Reference

Detailed Description

Definition in file RunTimeFloatLicInfo.cs

RunTimeLicAckResp.cs File Reference

Detailed Description

Definition in file RunTimeLicAckResp.cs

RunTimeLicCheckInReq.cs File Reference

Detailed Description

Definition in file RunTimeLicCheckInReq.cs

RunTimeLicCheckOutReq.cs File Reference

Detailed Description

Definition in file RunTimeLicCheckOutReq.cs

RunTimeLicCheckOutResp.cs File Reference

Detailed Description

Definition in file RunTimeLicCheckOutResp.cs

RunTimeLicPIDUpdateReq.cs File Reference

Detailed Description

Definition in file RunTimeLicPIDUpdateReq.cs

SamsungSCM.cs File Reference

Detailed Description

Definition in file SamsungSCM.cs

SiemensCap.cs File Reference

Detailed Description

Definition in file SiemensCap.cs

SiemensHicom300.cs File Reference

Detailed Description

Definition in file SiemensHicom300.cs

SiemensHipath3000p2.cs File Reference

Detailed Description

Definition in file SiemensHipath3000p2.cs

SiemensHipath3000p3.cs File Reference

Detailed Description

Definition in file SiemensHipath3000p3.cs

SiemensHipath4000.cs File Reference

Detailed Description

Definition in file SiemensHipath4000.cs

SiemensRealitis.cs File Reference

Detailed Description

Definition in file SiemensRealitis.cs

SocketState.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::SocketState

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file SocketState.cs

TadiranCoral.cs File Reference

Detailed Description

Definition in file TadiranCoral.cs

uaSIPInvite.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::uaSIPInvite

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file uaSIPInvite.cs

uaXMLed6.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::uaXMLed6

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file uaXMLed6.cs

UnifyOpenscape4000BER.cs File Reference

Detailed Description

Definition in file UnifyOpenscape4000BER.cs

UnifyOpenscapeVoice.cs File Reference

Detailed Description

Definition in file UnifyOpenscapeVoice.cs

UnifyOpenscapeX5.cs File Reference

Detailed Description

Definition in file UnifyOpenscapeX5.cs

Version.cs File Reference

Detailed Description

Definition in file Version.cs

VodiaSNOMOne.cs File Reference

Detailed Description

Definition in file VodiaSNOMOne.cs

XMLParseInfo.cs File Reference

Classes

- struct Com::Objsys::Csta::Xmled6::XMLParseInfo

Namespaces

- struct Com::Objsys::Csta::Xmled6

Detailed Description

Definition in file XMLParseInfo.cs