Provisioning API Reference

The provisioning API consists of these packages:

- com.cisco.provisioning.cpe—Contains the interfaces and methods used to connect to the provisioning API command engine (PACE) and to manage batch operations.
- com.cisco.provisioning.cpe.api—Contains the interfaces and methods used to provision and manage customer premises equipment (CPE), such as high-speed data (HSD) devices, digital set-top box (DSTB) devices, voice capable (xGCP) devices, and computers.
- com.cisco.provisioning.cpe.constants—Contains the interfaces and constants that interpret status information and retrieve values associated with the device types that BPR provisions.
- com.cisco.provisioning.cpe.events—Contains the interfaces and methods used to implement events.

The sections that follow introduce each of the BPR provisioning API packages.

Note You can access complete descriptions of the provisioning API packages through the JavaDoc packaged with the BPR product. For a list of methods deprecated since the CSRC DPR product release, see Appendix B, "Deprecated Methods."

Hierarchy

Table A-1 shows the cisco.com.provisioning.cpe hierarchy.

Table A-1 cisco.com.provisioning.cpe Hierarchy

com.cisco.provisioning.cpe	
Interfaces	Batch
	BatchStatus
	CommandStatus
	CSRCAPI
	CSRCAPIStatus
	PACEConnection
	ProvAPI
	ProvAPIStatus

Table A-1 cisco.com.provisioning.cpe Hierarchy (continued)

com.cisco.provisioning.cpe	
Classes	ActivationMode
	APINames
	ConfirmationMode
	DataType
	PACEConnectionFactory
	PublishingMode
	SNMPVersion
Exceptions	AlreadyPostedException
	APINotFoundException
	BaseException
	BaseRunTimeException
	BPRAuthenticationException
	PACEConnectionException
	ProvisioningException

Table A-2 com.cisco.provisioning.cpe.api Hierarchy

com.cisco.provisioning.cpe.api	
Interfaces	Computer
	Configuration
	CustomCPE
	DeviceSearch
	DOCSIS
	DSTB
	Provisioning
	XGCP

Table A-3 com.cisco.provisioning.cpe.constraints Hierachy

com.cisco.provisioning.cpe.constants	
Interfaces	BatchStatusCodes
	ClassOfServiceKeys
	ClassOfServiceType
	CNRDetailsKeys
	CNRExtensionSettingKeys
	CNRServersKeys
	CommandStatusCodes
	DeviceDetailsKeys
	DeviceDisruptionStatus
	DeviceTypeValues
	DHCPCriteriaKeys
	DHCPOptionKeys
	DocsisDefaultKeys
	ETTHProfileKeys
	IPDeviceKeys
	LicenseCodes
	LicenseDataKeys
	ModemKeys
	PublishDetailsKeys
	RDUDetailsKeys
	SearchType
	ServerDefaultsKeys
	SNMPPropertyKeys
	TechnologyDefaultsKeys
	UserDetailsKeys
	VOIPServiceDetailsKeys
	XGCPCommandStatusCodes
	XGCPProvisioningKeys

com.cisco.provisioning.cpe.events	
Interfaces	BatchEvent
	BatchListener
	COSEvent
	COSListener
	DeviceEvent
	DeviceListener
	DHCPCriteriaEvent
	DHCPCriteriaListener
	ExternalFileEvent
	ExternalFileListener
	MessagingEvent
	MessagingListener
	ProvAPIEvent
	ProvAPIEventListener
	ProvAPIEventManager
	ProvGroupEvent
	ProvGroupListener
	SystemConfigEvent
	SystemConfigListener

com.cisco.provisioning.cpe.events (continued)	
Classes	BatchAdapter
	BatchEventQualifier
	COSAdapter
	COSEventQualifier
	DeviceAdapter
	DeviceEventQualifier
	DHCPCriteriaAdapter
	DHCPCriteriaEventQualifier
	ExternalFileAdapter
	ExternalFileEventQualifier
	MessagingAdapter
	MessagingQualifier
	ProvGroupAdapter
	ProvGroupEventQualifier
	Qualifier
	QualifyAll
	ServerConfigEventType
	SystemConfigAdapter
	SystemConfigEventQualifier
	SystemConfigEventType
Exceptions	RegistrationFailedException

com.cisco.provisioning.cpe Package

The com.cisco.provisioning.cpe package contains the interfaces and methods used to connect to the provisioning API command engine (PACE) and to manage batch operations. Table A-4 lists the com.cisco.provisioning.cpe interfaces.

Table A-4 The com.cisco.provisioning.cpe Package Interfaces

Interface	Description
Batch	Provides access to the provisioning API commands to be executed.
BatchStatus	Provides the return type for a batch operation.
CommandStatus	Provides a standard return type for the individual commands submitted from a batch operation.
CSRCAPI	Is a marker interface that all BPR API interfaces implement.
CSRCAPIStatus	Is implemented by all interfaces that describe messages returned from the provisioning server to the client.
PACEConnection	Contains the commands used to create and manage communication with the provisioning API command engine (PACE).

Table A-4 The com.cisco.provisioning.cpe Package Interfaces

Interface	Description
ProvAPI	Is a marker interface that is implemented by all other BPR API interfaces.
ProvAPIStatus	Is a status interface that all BPR API interfaces implement. It describes messages returned from PACE to the client program.

Table A-5 lists the com.cisco.provisioning.cpe classes.

Table A-5 The com.cisco.provisioning.cpe Classes

Class	Description
ActivationMode	Provides an enumeration of batch activation request states.
APINames	Provides an enumeration of the BPR named API interfaces and is used in conjunction with the Batch.getAPI() command.
ConfirmationMode	Provides an enumeration of batch confirmation request states.
DataType	Provides an enumeration of DataTypes that can be specified as part of a custom property definition.
PACEConnectionFactory	Returns a singleton instance of the PACE connection using peer objects.
PublishingMode	Provides an enumeration of batch Publishing flags.
SNMPVersion	Provides an enumeration of SNMP versions.

Table A-6 lists the cisco.com.provisioning.cpe package exceptions.

Table A-6 The com.cisco.provisioning.cpe Package Exceptions

Exception	Description
ProvisioningException	Used to signal that PACE could not process the posted batch.

Table A-7 lists the com.cisco.provisioning.cpe exceptions.

Table A-7 The com.cisco.provisioning.cpe Package Runtime Exceptions

Exception	Description
AlreadyPostedException	Used to indicate that this batch has already been posted.
APINotFoundException	Used to signal an attempt to retrieve an API instance S that is not one of the APIs enumerated by APINames class.
BaseException	Allows for localization and exception cascading. This means adding additional explanations to an exception and rethrowing the enhanced exception.
BaseRuntimeException	Defines an exception that allows for localization and exception cascading. Exception cascading is adding additional explanations to an exception and rethrowing the enhanced exception.

Table A-7 The com.cisco.provisioning.cpe Package Runtime Exceptions

Exception	Description
BPRAuthenticationException	Used to indicate an authentication failure while getting an instance of PACEConnection through the PACEConnectionFactory.
PACEConnectionException	Used to indicate an error in getting an instance of PACEConnection through the PACEConnectionFactory.

Batch Interface

The Batch interface gives access to all the provisioning API methods in BPR. Each batch of commands has a unique identifier; therefore, you can post a batch only once. If you try to add calls to a batch that has already been posted or to repost a batch, BPR generates a runtime exception. Table A-8 lists the Batch methods.

Table A-8 Batch Methods

Method	Description
addAPICall()	Adds an API Call object to the list of method calls in a batch. **Note** This method is for BPR internal use only. BPR API clients should not call this method.
createdByClient()	Indicates whether the Batch was created by the client API.
forceBatchReliable()	Forces the batch to be reliable. This indicates that the batch will complete even if the RDU fails while the batch is in the middle of, or waiting to be, processed.
getActivationMode()	Obtains the Activation mode associated with this batch.
getAllAPIs()	Instantiates and returns all available APIs (which ones are available are determined by customer licensing).
getAPI(APINames api)	Instantiates and returns requested API, or throws a runtime exception if the user is not licensed for the requested API.
getAPICall(int index)	Gets an APICall object from the batch's list of method calls. **Note** This method is for BPR internal use only. BPR API clients should not call this method.
getAPICallCount()	Obtains the count of API calls stored in the batch.
getBatchID()	Gets the batch identifier associated with the batch.
getConfirmationMode()	Obtains the Confirmation mode associated with this batch. The returned value is an enumerated type from the Confirmation mode class.
getProvAPI()	Instantiates and returns requested API, or throws a runtime exception if the user is not licensed for the requested API.
getPublishingMode()	Obtains the PublishingMode (enum) associated with this batch.
isBatchReliable()	Indicates whether the Batch will be completed even if the RDU goes down while the batch is processing or waiting to be processed.
post()	Posts this batch of commands for execution.

Table A-8 Batch Methods

Method	Description
post(long timeout)	Posts this batch of commands for execution.
postNoConfirmation()	Posts this command batch for execution.
postWithConfirmation()	Posts this command batch for execution.
wasPosted()	Indicates whether the batch was ever posted.

BatchStatus Interface

The BatchStatus interface provides the standard return type for a batch operation. Table A-9 lists the BatchStatus methods.

Table A-9 BatchStatus Methods

Method	Description
getBatchID()	Returns the associated batch identifier for this status request.
getCommandCount()	Gets the number of commands in a batch that were processed.
getCommandStatus()	Returns the command status for the specified command or returns null if there is no corresponding command.
getFailedCommandIndex()	Gets the index for the specified command that failed. Returns a value of -1 if no command failed.
getFailedCommandStatus()	Gets the command status return for the failed command or returns null value if no command failed.
isWarning()	A helper method.
wasBatchReliable()	Returns true if the batch was treated as reliable by the server.

CommandStatus Interface

The Command Status interface provides the standard return type for the individual operations submitted in a batch, including command error status. For methods that return data to the caller, this interface also provides the data type code and the data itself.

The CommandStatus.getDataTypeCode() function returns status values to indicate that a command returned these data types:

- Byte array data
- List data
- Map of key/value data
- Null
- String data
- Void (the command did not return any data)

Table A-10 lists the CommandStatus methods.

Table A-10 CommandStatus Methods

Method	Description
commandReturnsData()	Indicates whether a command returns data.
getData()	Returns the requested data for this command status.
getDataTypeCode()	Returns the data type code for this command status.

CSRCAPI Interface

The CSRCAPI interface is a marker interface that all the com.cisco.provising.cpe.api interfaces implement. Its subinterfaces are:

- Computer
- Configuration
- CustomCPE
- Device search
- DOCSIS
- DSTB
- Provisioning
- XGCP

ProvAPIStatus Interface

The ProvAPIStatus interface is implemented by all interfaces that describe messages returned from PACE to the client program. This interface provides this status information:

- Status type of the batch operation
- Status type of the command
- Unknown status type
- Provisioning exception type

Table A-11 lists the ProvAPIStatus methods.

Table A-11 ProvAPIStatus Methods

Method	Description
getBatchID()	Returns the batch identifier.
getErrorMessage()	Obtains an error message that describes a problem encountered in processing.
getStatusCode()	Obtains the status code for the results of processing.
getStatusType()	Returns the type of interface that the implementing object represents.
isError()	Determines whether a status code is an error message.
isSystemError()	Determines whether a status code is a system error message.

PACEConnection Interface

The PACEConnection interface contains the commands used to create and manage communication with the provisioning API command engine (PACE). Table A-12 lists the PACEConnection methods.

Table A-12 PACEConnection Methods

Method	Description
dropConnection()	Drop an RDU connection.
getErrorString()	Return the error string from the last call to addListener() or removeListener().
getHost()	Returns the hostname of the RDU to which you are connecting.
getPort()	Returns the port number of the RDU to which you are connecting.
getReturnString()	Return the status of the last Connection method invocation.
isAlive()	Returns true if the connection to the RDU is open, otherwise false.
joinBatch(java.lang.String batchID)	Joins a batch that has been executed.
joinBatch(java.lang.String batchID, long timeout)	Joins a batch that has been executed.
newBatch()	Gets a new batch with a guaranteed globally-unique batch identifier.
newBatch(ActivationMode activation)	Gets a new batch with a guaranteed globally-unique batch ID.
newBatch(ActivationMode activation, ConfirmationMode confirmation)	Gets a new batch with a guaranteed globally-unique batch ID.
newBatch(ActivationMode activation, ConfirmationMode confimation, PublishingMode publishing)	Gets a new batch with a guaranteed globally-unique batch ID.
newBatch(ActivationMode activation, PublishingMode publishing)	Gets a new batch with a guaranteed globally-unique batch ID.
newBatch(PublishingMode publishing)	Gets a new batch with a guaranteed globally-unique batch ID.
newBatch(String batchID)	Gets a new batch, using a user-supplied batch ID, which must be unique (or the batch will fail to execute when it is posted).
newBatch(String batchID, ActivationMode activation)	Gets a new batch, using a user-supplied batch ID, which must be unique (or the batch will fail to execute when it is posted).
newBatch(String batchID, ActivationMode activation, ConfirmationMode confirmation)	Gets a new batch, using a user-supplied batch ID, which must be unique (or the batch will fail to execute when it is posted).
newBatch(java.lang.String batchID, ActivationMode activation, ConfirmationMode confirmation, PublishingMode publishing)	Gets a new batch, using a user-supplied batch ID, which must be unique (or the batch will fail to execute when it is posted).

Table A-12 PACEConnection Methods (continued)

Method	Description
newBatch(java.lang.String batchID, ActivationMode activation, PublishingMode publishing)	Gets a new batch, using a user-supplied batch ID, which must be unique (or the batch will fail to execute when it is posted).
newBatch(java.lang.String batchID, PublishingMode publishing)	Gets a new batch, using a user-supplied batch ID, which must be unique (or the batch will fail to execute when it is posted).
openConnection()	Opens a connection to the RDU.
postBatch(Batch batch)	Posts a batch of commands for execution.
postBatch(Batch batch, long timeout)	Posts a batch for execution.
postBatchNoStatus(Batch batch)	Posts a batch for execution. No status is returned for this batch.
releaseConnection()	Releases an RDU connect.

ActivationMode Class

Use the Activation mode class for an enumeration of batch activation request states. This class has three fields:

- AUTOMATIC—Perform all necessary steps for activation of the devices in the batch, including device disruption.
- NO_ACTIVATION—Do not activate the devices in the batch. A new configuration is generated.

Table A-13 lists the ActivationModeClass methods.

Table A-13 ActivationMode Class Methods

Method	Description
getNamed()	Returns the ActivationMode object with the selected name, or it returns null if the name does not exist.
getValued()	Returns the ActivationMode object with the selected value, or it returns null if the value does not exist.

APINames Class

Use this class, in conjunction with the Batch.getAPI() method, for an enumeration of the BPR named API interfaces.

Table A-14 lists the fields for the API names class.

Table A-14 APINames Class Fields

Field	Description
Computer	References the com.cisco.provisioning.cpe.api.Computer interface.
Configuration	References the com.cisco.provisioning.cpe.api.Configuration interface.
CustomCPE	References the com.cisco.provisioning.cpe.api.CustomCPE interface.
DeviceSearch	References the com.cisco.provisioning.cpe.api.DeviceSearch interface.
DOCSIS	References the com.cisco.provisioning.cpe.api.DOCSIS interface.
DSTB	References the com.cisco.provisioning.cpe.api.DSTB interface.
Provisioning	References the com.cisco.provisioning.cpe.api.Provisioning interface.
XGCP	References the com.cisco.provisioning.cpe.api.XGCP interface.

Table A-15 lists the APINames methods.

Table A-15 APINames Methods

Method	Description
getNamed()	Returns the APIName object with the selected name, or it returns null value if the name does not exist.
getValued()	Returns the APIName object with the selected value, or it returns null value if the value does not exist.

ConfirmationMode Class

The ConfirmationMode class is used for an enumeration of batch confirmation request states. Table A-16 lists the ConfirmationMode class fields.

Table A-16 ConfirmationMode Class Fields

Class	Description
CUSTOM_CONFIRMATION	If disruption error status is set, the batch status code will be set to BatchStatusCodes.BATCH_FAILED_DISRUPTION, and the whole batch will roll back.
NO_CONFIRMATION	If disruption fails, does not confirm device reboot and returns with a BatchStatusCode value of BATCH_WARNING. NO_CONFIRMATION is the default.

Table A-17 lists the ConfirmationMode methods.

Table A-17 ConfirmationMode Methods

Method	Description
getNamed()	Returns the APIName object with the selected name, or it returns null value if the name does not exist.
getValued()	Returns the APIName object with the selected value, or it returns null value if the value does not exist.

DataType Class

The DataType class provides an enumeration of the data types that can be specified as part of a custom property definition. Table A-18 identifies the fields implemented by the DataType class and Table A-19 lists the DataType methods.

Table A-18 DataType Fields

Field	Description
BOOLEAN	References an object of type Boolean.
BYTE	References an object of type Byte.
BYTE_ARRAY	References an object of type Byte_Array.
CHARACTER	References an object of type Character.
DOUBLE	References an object of type Double.
FLOAT	References an object of type Float.
INTEGER	References an object of type Integer.
LIST	References an object of type List.
LONG	References an object of type Long.
MAP	References an object of type Map.
SHORT	References an object of type Short.
STRING	References an object of type String.

Table A-19 DataTypeClass Methods

Method	Description
getNamed()	Returns the DataType object with the selected name (or null if it does not exist).
getValued()	Returns the DataType object with the selected value (or null if it does not exist).
readExternal()	Identifies an externalizable interface.
writeExternal()	Identifies an externalizable interface.

PACEConnectionFactory Class

Use the PACE connection factory class to return singleton instances using peer objects. Table A-20 lists the PACEConnectionFactory methods.

Table A-20 PACEConnectionFactory Methods

Method	Description
containsInstance(InetAddress address, int port)	Returns true if a singleton instance exists on the given port and host.
containsInstance(String host, int port)	Returns true if a singleton instance exists on the given port and host.
getInstance(InetAddress address, int port)	Returns the singleton instance of the anonymous PACEConnection on the given port and host.
getInstance(InetAddress address, int port, String username, String password)	Returns the singleton instance of the PACEConnection on the given port and host.
getInstance(com.cisco.csrc.messages.Peer peer)	Returns the singleton instance of the PACEConnection on the given Peer.
getInstance(com.cisco.csrc.messages.Peer peer, String username, String password)	Returns the singleton instance of the PACEConnection on the given Peer.
getInstance(String host, int port)	Returns the singleton instance of the PACEConnection on the given port and host.
getInstance(String host, int port, String username, String password)	Returns the singleton instance of the PACEConnection on the given port and host.
getInstance(String host, int port, String username, String password, boolean authenticateImmediately)	Returns the singleton instance of the PACEConnection on the given port and host.

SNMPVersion Class

The SNMPVersion class provides an enumeration of SNMP versions. Table A-21 lists the SNMPVersion methods.

The SNMPVersion class implements this field:

- V1—References SNMP version 1.

Table A-21 SNMPVersion Methods

Method	Description
getNamed()	Returns the SNMPVersion object with the selected name (or null if the object does not exist).
getValued()	Returns the SNMPVersion object with the selected value (or null if the object does not exist).

Exceptions

The com.cisco.provisioning.cpe package implements these exceptions to handle processing of errors and other abnormal conditions:

- ProvisioningException—Thrown as a result of calling Batch.post(), and signals that PACE could not process the posted batch due to catastrophic failure.
- BaseException—A BaseException is a BPR defined exception that allows for localization and exception cascading. Exception cascading is adding additional explanations to an exception and rethrowing the enhanced exception.
- BaseRunTimeException—A BaseRunTimeException is a BPR defined exception that allows for localization and exception cascading. Exception cascading is adding additional explanations to an exception and rethrowing the enhanced exception.

The com.cisco.provisioning.cpe package implements these runtime exceptions to handle processing errors and other abnormal conditions:

- AlreadyPostedException—Thrown when an attempt is made to repost an already posted batch. The caller must get a new batch instance.
- APINotFoundException—Thrown when an attempt is made to retrieve an API Instance object that is not one of the APIs enumerated by APINames.
- PACEConnectionException—This exception can occur when getting an instance of PACEConnection through the PACEConnectionFactory.
- BPRAuthenticationException—Thrown when getting an instance of PACEConnection through the PACEConnectionFactory.

com.cisco.provisioning.cpe.api Package

The com.cisco.provisioning.cpe.api package contains the interfaces and commands used to manage these device types:

- High-speed data (HSD) devices
- Digital set-top box (DSTB) devices
- Voice capable Gateway Control Protocol (xGCP) devices
- Computers
- Custom CPE devices

Table A-22 lists the com.cisco.provisioning.cpe.api package interfaces.

Table A-22 com.cisco.provisioning.cpe.api Package Interfaces

Interface	Description
Computer	Provides methods that manipulate provisioned computers.
Configuration	Contains configuration operations common to all sectors of BPR.
CustomCPE	Provides commands that manipulate CPE for proprietary devices.
DeviceSearch	Provides the methods that search for multiple CPE devices and retrieve a list of matching device identifiers.
DOCSIS	Provides methods that manipulate provisioned DOCSIS modems.

Table A-22 com.cisco.provisioning.cpe.api Package Interfaces (continued)

Interface	Description
DSTB	Provides methods that manipulate provisioned DSTB devices.
Provisioning	Contains methods that manipulate provisioned CPE devices in the BPR system.
XGCP	Contains methods that configure batch operations associated with xGCP devices.

The following sections discuss the com.cisco.provisioning.cpe.api interfaces.

Computer Interface

The computer interface contains methods that manipulate provisioned computers. Table A-23 lists the Computer methods.

Table A-23 Computer Methods

Method	Description
addComputer()	Adds a computer.
addComputerByIPAddress()	Adds a computer identified by its IP address.
changeComputerClassOfService()	Changes a computer's class of service.
changeComputerDefaults()	Changes a computer's defaults
changeComputerDHCPCriteria()	Changes a computer DHCP criteria definition.
changeComputerFQDN()	Changes the FQDN of a computer.
changeComputerMACAddress()	Changes a computer's MAC address.
changeComputerOwnerID()	Changes a computer's owner identifier (ownership).
changeComputerProperties()	Changes a computer's properties.
deleteComputer()	Deletes a computer.
getAllComputers()	Gets the list of all computers.
getComputerDefaults()	Gets the defaults for a computer.
getDetailsForComputer()	Gets the device details for a given computer.
getDetailsForComputerByIPAddress()	Gets the device details for a given computer, identified by IP address.

Configuration Interface

Use the Configuration interface to get and set default configuration parameters for CPE devices. In the BPR, API methods are defined with arguments and properties. Arguments indicate the most commonly used fields that an API call must specify. In many cases, you can set arguments to null. You must specify properties in the form of a map containing key/value pairs.

The Configuration interface provides commonly used configuration methods. Table A-24 lists the Configuration methods.

Table A-24 Configuration Methods

Method	Description
addCustomPropertyDefinition()	Adds a custom property definition.
addExternalFile()	Adds an external file.
addLicenseKey()	Adds a license key.
addUser()	Adds a user.
changeDPEDefaults()	Changes DPE default properties.
changeExtensionPointSettings()	Changes Network Registrar extension point settings.
changePublishingPluginSettings()	Changes the publishing plug-in default settings.
changeRDUDefaults()	Changes RDU default properties.
changeSystemDefaults()	Changes system default properties.
changeUser()	Changes user properties.
deleteCNR()	Deletes a Network Registrar server.
deleteDPE()	Deletes a DPE server.
deleteExternalFile()	Deletes an external file.
deleteUser()	Deletes a user.
disablePublishingPlugin()	Disables a publishing plug-in.
enablePublishingPlugin()	Enables a publishing plug-in.
getAllCNRs()	Retrieves a list of all currently registered Network Registrar servers.
getAllSystemPropertyDefinitions()	Retrieves a map of all currently defined custom property definitions.
getAllCustomPropertyDefinitions()	Gets a map of all custom property definitions currently defined in BPR.
getAllDPEs()	Retrieves a list of all currently registered DPE servers.
getAllProvGroups()	Retrieves a list of all provisioning groups.
getAllRDUs()	Retrieves a list of all currently registered RDUs.
getAllSystemPropertyDefinitions()	Gets a Map of all BPR pre-defined property definitions.
getAllUsers()	Retrieves a list of all users.
getCNRDefaults()	Retrieves a map of Network Registrar default properties.
getCNRDetails()	Retrieves the details for a Network Registrar server.
getCNRsByProvGroup()	Retrieves a list of all Network Registrar servers in a provisioning group.
getDPEDefaults()	Retrieves a map of DPE default properties.
getDPEDetails()	Retrieve the details for a DPE server.
getDPEsByProvGroup()	Retrieves a list of the DPE servers in a provisioning group.
getExtensionPointSettings()	Retrieves a map of extension point settings.
getExternalFile()	Retrieves an external file.
getLicenseKeyData()	Retrieves a list of maps of license key data.

Table A-24 Configuration Methods (continued)

Method	Description
getMatchingExternalFilenames()	Retrieves a list of known matching external filenames.
getPublishingPlugins()	Retrieves a list of publishing plug-ins.
getPublishingPluginSettings()	Retrieves the default settings for a specified plug-in.
getRDUDefaults()	Retrieves a map of RDU default properties.
getRDUDetails()	Retrieves the details for an RDU server.
getSystemDefaults()	Retrieves the default system properties.
getUserDetails()	Retrieves the details for a given user.
removeCustomPropertyDefinition()	Removes a custom property definition.
replaceExternalFile()	Replaces an external file.

CustomCPE Interface

Use the CustomCPE interface to manage proprietary technologies and equipment. Table A-25 lists the CustomCPE methods.

Table A-25 CustomCPE Methods

Method	Description
addCustomCPE()	Adds a custom CPE device using a MAC address.
addCustomCPEByIPAddress()	Adds a custom CPE device using an IP address.
addCustomCPEType()	Adds a custom CPE device type.
changeCustomCPEClassOfService()	Changes the class of service for a custom CPE device.
changeCustomCPECPEDHCPCriteria()	Changes the CPE DHCP criteria of a custom CPE device.
changeCustomCPEDefaults()	Changes the defaults for custom CPE devices.
changeCustomCPEDHCPCriteria()	Changes the DHCP criteria for a custom CPE device.
changeCustomCPEFQDN()	Changes the FQDN for a custom CPE device.
changeCustomCPEMACAddress()	Changes the MAC address for a custom CPE device.
changeCustomCPEOwnerID()	Changes the owned identifier for a custom CPE device.
changeCustomCPEProperties()	Changes the properties for a custom CPE device.
changeCustomCPETypeProperties()	Changes the property type for a custom CPE device.
deleteCustomCPE()	Deletes a custom CPE device.
deleteCustomCPEType()	Deletes a custom CPE device type.
getAllCustomCPEs()	Retrieves a list of all custom CPE devices.
getAllCustomCPETypes()	Gets the list of all custom CPE types.
getCustomCPEDefaults()	Retrieves the defaults for custom CPE devices.
getCustomCPETypeDetails()	Retrieves the details for a custom CPE type.

Table A-25 CustomCPE Methods (continued)

Method	Description
getDetailsForCustomCPE()	Retrieves the details for a custom CPE device by identifier.
getDetailsForCustomCPEByIPAddress()	Retrieves the details for a custom CPE device by IP address.

DeviceSearch Interface

Use the Device Search interface to search for multiple CPE devices and to retrieve a list of matching device identifiers. Table A-26 lists the DeviceSearch methods.

Table A-26 DeviceSearch Methods

Method	Description
getIPDevicesByClassOfService()	Looks up the IP devices with the designated class of service name.
getIPDevicesByCPEDHCPCriteria()	Looks up the IP devices with the designated CPE DHCP criteria name.
getIPDevicesByDHCPCriteria()	Looks up the IP devices with the designated DHCP criteria name.
getIPDevicesByFQDN()	Searches for IP devices by fully qualified domain name (FQDN).
getIPDevicesByIPAddress()	Looks up the IP devices whose IP addresses match the supplied ipAddress.
getIPDevicesByMACAddress()	Searches for IP devices by MAC address.

DOCSIS Interface

Use the DOCSIS interface to manage DOCSIS devices. Table A-27 lists the DOCSIS methods.

Table A-27 DOCSIS Methods

Method	Description
addDOCSISModem()	Adds a DOCSIS modem.
addDOCSISModemByIPAddress()	Adds a DOCSIS modem identified by its IP address.
changeDOCSISDefaults()	Changes DOCSIS default properties.
changeDOCSISModemClassOfService()	Changes a DOCSIS modem's class of service.
changeDOCSISModemCPEDHCPCriteria()	Changes a DOCSIS modem’s CPE DHCP criteria.
changeDOCSISModemDHCPCriteria()	Changes a DOCSIS modem’s DHCP criteria.
changeDOCSISModemFQDN()	Changes a DOCSIS modem's FQDN.
changeDOCSISModemMACAddress()	Changes a DOCSIS modem's MAC address.
changeDOCSISModemOwnerID()	Changes a DOCSIS modem's owner identifier.

Table A-27 DOCSIS Methods (continued)

Method	Description
changeDOCSISModemProperties()	Changes a DOCSIS modem's properties.
deleteDOCSISModem()	Deletes a DOCSIS modem.
getAllDOCSISModems()	Gets the list of DOCSIS modems
getDetailsForDOCSISModem()	Gets the device details for a given DOCSIS modem.
getDetailsForDOCSISModemByIPAddress()	Gets the device details for a given DOCSIS modem, identified by IP address.
getDOCSISDefaults()	Gets the DOCSIS defaults.

DSTB Interface

Use the DSTB interface to configure digital set-top box components for BPR and to retrieve information about the components from BPR. Table A-28 lists the DSTB methods.

Table A-28 DSTB Methods

Method	Description
addDSTB()	Adds a DSTB.
addDSTBByIPAddress()	Adds a DSTB identified by its IP address.
changeDSTBClassOfService()	Changes a DSTB's class of service.
changeDSTBDefaults()	Changes a DSTB's default properties.
changeDSTBDHCPCriteria()	Changes a DSTB's DHCP criteria.
changeDSTBFQDN()	Changes the FQDN of a DSTB.
changeDSTBMACAddress()	Changes a DSTB's MAC address.
changeDSTBOwnerID()	Changes the owner identifier of a DSTB.
changeDSTBProperties()	Changes the properties of a DSTB.
changeDSTBRelatedDOCSISModemID()	Changes the modem identifier of a DSTB's related DOCSIS modem.
deleteDSTB()	Deletes a DSTB and optionally deletes related DOCSIS modems related to the DSTB.
getAllDSTBs()	Gets the list of all DSTBs.
getDetailsForDSTB()	Gets the device details for a given DSTB.
getDetailsForDSTBByIPAddress()	Gets the device details for a given DSTB, identified by IP address.
getDSTBDefaults()	Retrieve the DSTB defaults in a map.

Provisioning Interface

Use the Provisioning interface to manipulate provisioned customer premises equipment (CPE) in the BPR system.

Note The Provisioning interface in CSRC DPR contained calls now deprecated in BPR. See Appendix B, "Deprecated Methods" for the list of deprecated calls.

Table A-29 lists the Provisioning methods.

Table A-29 Provisioning Methods

Method	Description
addClassOfService()	Adds a class of service.
addDHCPCriteria()	Adds client class and scope selection tag definitions.
changeClassOfServiceProperties()	Changes a class of service's properties.
changeDHCPCriteriaClientClass()	Changes a DHCP criteria client class definition.
changeDHCPCriteriaExcludeSelectionTags()	Changes a DHCP criteria exclude selection tag definition.
changeDHCPCriteriaIncludeSelectionTags()	Changes a DHCP criteria include selection tag definition.
changeDHCPCriteriaProperties()	Changes a DHCP criteria property definition.
deleteClassOfService()	Deletes a class of service.
deleteDHCPCriteria()	Deletes a DHCP criteria definition.
deleteIPDevice()	Deletes an IP device (regardless of type).
generateConfiguration()	Generates a configuration for the device specified.
getAllClassesOfService()	Gets a list of the names of all classes of service, for a given type.
getAllDHCPCriterias()	Gets a list of all DHCP criteria definitions.
getClassOfServiceProperties()	Gets the properties for the given class of service.
getDetailsForIPDevice()	Gets the device details for a given IP device instance.
getDetailsForIPDeviceByIPAddress()	Gets the IP device details for a given IP address.
getDHCPCriteriaDetails()	Gets the details for a DHCP criteria definition.
getIPDeviceDetailsList()	Gets a list of IP device details for the list of deviceID strings supplied.
getIPDeviceDetailsListByIPAddress()	Gets a list of IP device details for the list of ipAddress strings supplied.
getIPDeviceForIPAddress()	Looks up the IP device for the specified IP address.
getIPDevicesBehindModem()	Lists all of the devices currently located behind the specified modem.
getIPDevicesByOwnerID()	Gets the list of IP devices for a specified owner ID.
resetIPDevice()	Resets (reboots) an IP device.

XGCP Interface

Use the XGCP interface to manage batch operations associated with xGCP devices. This interface contains the methods used to configure xGCP information for BPR and to retrieve xGCP information from BPR. Table A-30 lists the XGCP methods.

Table A-30 XGCP Methods

Method	Description
addXGCPService()	Assigns an xGCP service to a specified port on a modem.
changeXGCPServiceCmsFQDN()	Changes the FQDN of the call management system (CMS) associated with an xGCP service.
changeXGCPServiceCmsPortNumber()	Changes the CMS port number associated with an xGCP service.
changeXGCPServiceCmsQualifier()	Changes the CMS Qualifier associated with an xGCP service.
changeXGCPServiceProperties()	Changes properties associated with an xGCP service.
changeXGCPServiceTelephoneNumber()	Changes the telephone number associated with an xGCP service.
deleteXGCPService()	Deletes the service from the specified port.
getXGCPServiceDetails()	Gets the xGCP port information.

com.cisco.provisioning.cpe.constants Package

The com.cisco.provisioning.cpe.constants package contains the interfaces and constants that interpret status information and retrieve values associated with the device types that BPR provisions. Table A-31 lists the interfaces in the com.cisco.provisioning.cpe.constants package.

Table A-31 com.cisco.provisioning.cpe.constants Package

Interface	Description
BatchStatusCodes	Specifies constants that are used to interpret returned status information for a batch operation of commands.
ClassOfServiceKeys	Specifies constant strings that are used as keys in class of service objects.
ClassOfServiceType	Specifies the class of service types supported by BPR.
CNRDetailsKeys	Specifies constant strings that are used as keys in map objects returned by queries to CNR.
CNRExtensionSettingKeys	Specifies constant strings that are used as keys in map objects by the changeExtensionPointSettings() method and returned by the getExtensionPointSettings() method.
CNRServersKeys	Specifies constant strings that are used as system default keys for a BPR configuration of Network Registrar.

Table A-31 com.cisco.provisioning.cpe.constants Package (continued)

Interface	Description
CommandStatusCodes	Specifies constants that are used to interpret the returned command status.
DeviceDetailsKeys	Specifies character strings that are used as keys to retrieve values associated with IP devices, for example an IP address or a MAC address.
DeviceDisruptionStatus	Provides the different status codes returned by a device disruption.
DeviceTypeValues	Specifies the allowed values for a device type.
DHCPCriteriaKeys	Specifies constant strings that are used as keys in DHCPCriteria objects.
DHCPOptionKeys	Specifies constants that are used as keys in map objects to set and get properties that apply to BPR DHCP configuration and BPR client policy support.
DocisisDefaultKeys	Specifies the constant strings that are used as keys in map objects to set and get properties that apply to DOCSIS and RDU defaults.
DPEDetailsKeys	Specifies constants that are used as keys in map objects returned by query methods on DPE.
IPDeviceKeys	Specifies constant strings that are used as keys in properties map objects for IP device provisioning interfaces.
LicenseCodes	Specifies constant strings that are used as algorithms needed by the write method for license key data.
LicenseDataKeys	Specifies constant strings that are used as keys in map objects returned by the query method for license key data.
ModemKeys	Specifies constants that are used as keys in map objects returned by query methods on DOCSIS interfaces.
PublishDetailsKeys	Specifies constant strings, which are used as keys in Map objects returned by query methods on Publish objects.
RDUDetailsKeys	Specifies constants that are used as keys in map objects returned by query methods on the RDU.
SearchType	Enumerates the search types supported by BPR.
ServerDefaultKeys	Specifies the constants that servers use.
SNMPPropertyKeys	Specifies character strings that are used as keys to store and retrieve values associated with devices that respond to simple network management protocol (SNMP) requests, for example their read and write community strings.
TechnologyDefaultsKeys	Specifies constants that are used as keys in map objects to get and set properties that apply to technology defaults.
UserDetailsKeys	Specifies constants that are used as keys in map objects returned by query methods on user objects.
VOIPServiceDetailsKeys	Specifies character strings that are used as keys to retrieve values associated with xGCP devices, for example, an FQDN, a port number, or a telephone number.

Table A-31 com.cisco.provisioning.cpe.constants Package (continued)

Interface	Description
XGCPCommandStatusCodes	Specifies constants that are used to interpret returned xGCP command and query status information.
XGCPProvisioningKeys	Specifies the constants that are used to interpret and set the values associated with xGCP devices, for example, the SGCP version string.

The following sections provide a summary of the interfaces in this package.

BatchStatusCodes Interface

Use the Batch Status Codes interface to obtain status information about a batch operation. Table A-32 lists the BatchStatus.getStatusCode values.

Table A-32 BatchStatusCodes Values

Value	Description
BATCH_BEING_PROCESSED	Return value from BatchStatus.getStatusCode().
BATCH_CANNOT_USE_ANONYMOUS_ CONNECTION	Return value from BatchStatus.getStatusCode().
BATCH_COMPLETED	Return value from BatchStatus.getStatusCode().
BATCH_DUPLICATE_DROPPED	Return value from BatchStatus.getStatusCode().
BATCH_DUPLICATE_ID	Return value from BatchStatus.getStatusCode().
BATCH_ERROR	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_ACTIVATE	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_COMMIT	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_CONFIGURATION_GENERATION	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_DISRUPTION	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_NO_DISRUPT_ACTIVATION	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_PREPARE	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_PUBLISH	Return value from BatchStatus.getStatusCode().

Table A-32 BatchStatusCodes Values (continued)

Value	Description
BATCH_FAILED_QUERY	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_VALIDATION	Return value from BatchStatus.getStatusCode().
BATCH_FAILED_WRITE	Return value from BatchStatus.getStatusCode().
BATCH_ILLEGAL_MULTIPLE_ACTIVATIONS	Return value from BatchStatus.getStatusCode().
BATCH_INVALID_LICENSES	Return value from BatchStatus.getStatusCode().
BATCH_PASSED_VALIDATION	Return value from BatchStatus.getStatusCode().
BATCH_POST_TIMEOUT	Return value from BatchStatus.getStatusCode().
BATCH_RDU_BUSY	Return value from BatchStatus.getStatusCode().
BATCH_SYSTEM_ERROR	Return value from BatchStatus.getStatusCode().
BATCH_WARNING	Return value from BatchStatus.getStatusCode().

ClassOfServiceKeys Interface

The ClassOfServiceKeys interface specifies constant strings that are used as keys in ClassOfService objects. Table A-33 lists the class of service keys.

Table A-33 ClassOfServiceKeys

Key	Description
COS_DOCSIS_FILE	DOCSIS class of service file.

ClassOfServiceType Interface

The ClassOfServiceType interface provides constants that reference the classes of service that BPR can store. Table A-34 lists the class of service types.

Table A-34 Class of Service Types

Type	Description
COMPUTER	References a Computer class of service.
CUSTOMCPE	References a CustomCPE class of service.

Table A-34 Class of Service Types (continued)

Type	Description
DOCSIS	References a DOCSIS class of service.
DSTB	References a DSTB class of service.

CNRDetailsKeys Interface

The CNRDetails interface specifies constants that are used as keys in map objects returned by query methods on Network Registrar. Table A-35 lists the CNR details keys.

Table A-35 CNRDetailsKeys

Key	Description
CNR_BATCHES	Version number for the queried Network Registrar extension point/server.
CNR_HEALTH	Health of the queried Network Registrar extension point/server.
CNR_HOST_ID	Host identifier for the queried Network Registrar extension point/server.
CNR_HOST_PORT	Port number for the queried Network Registrar extension point/server.
CNR_PROPERTIES	Properties for the queried Network Registrar extension point/server.
CNR_PROV_GROUP_ID	Provisioning group identifier for the queried Network Registrar extension point/server.
CNR_STATE	State of the queried Network Registrar extension point/server.
CNR_UPTIME	Uptime for the queried Network Registrar extension point/server.
CNR_VERSION	Software version of the queried Network Registrar extension point/server.
DPES_STATUS_KEY	Identifies DPEs and their status.

CNRExtensionSettingKeys Interface

The CNRExtensionSettingKeys interface specifies constant strings that can be used as keys in the map objects used by change Extension Point Settings() and returned by getExtensionPointSettings(). Table A-36 lists the CNR extension setting keys.

Table A-36 CNRExtensionSettingKeys

Key	Description
CNR_ATTRIBUTES_TO_READ_FROM_ENVIRONMENT_DICTIONARY	List of all the attributes to be pulled from Network Registrar environment dictionary.
CNR_ATTRIBUTES_TO_READ_FROM_REQUEST_DICTIONARY	List of all the attributes to be pulled from Network Registrar request dictionary.

CNRServersKeys Interface

Use the CNRServersKeys interface to specify the constants that are used as system default keys during BPR configuration. Table A-37 lists the CNRServersKeys.

Table A-37 CNRServersKeys

Key	Description
CNR_CLIENT_PORT	The listening port of the API when communicating with the Network Registrar servers.
CNR_NUMBER_RETRIES	The number of retries when communicating with the Network Registrar servers.
CNR_SERVERS_LIST	Returns a list of Network Registrar servers to query.
CNR_TIMEOUT	The time in milliseconds between retries when communicating with the Network Registrar servers.
USE_CLIENT_ID	If true, uses the client ID instead of the MAC address.

CommandStatusCodes Interface

The Command Status Codes interface specifies a number of constants that are used to interpret the return status of HSD commands. The CommandStatus.getStatusCode() function returns values to indicate that HSD command processing succeeded or that a command processing error occurred. Table A-38 lists CommandStatusCodes values.

Table A-38 CommandStatusCodes Values

Value	Description
CMD_DB_LOCK	Return value from CommandStatus.getStatusCode().
CMD_ERROR	Return value from CommandStatus.getStatusCode().
CMD_ERROR_ADD_USER	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CLASS_OF_SERVICE_EXISTS	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CLASS_OF_SERVICE_ UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CNR_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_COMPUTER_CLASS_OF_ SERVICE_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_COMPUTERS_BEHIND_ MODEM	Return value from CommandStatus.getStatusCode().
CMD_ERROR_SYSTEM_PROPERTY_ REDEFINITION	Return value from CommandStatus.getStatusCode().

Table A-38 CommandStatusCodes Values (continued)

Value	Description
CMD_ERROR_CUSTOM_CPE_CLASS_OF_ SERVICE_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CUSTOM_CPE_TYPE_EXISTS	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CUSTOM_CPE_TYPE_ UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CUSTOM_CPES_BEHIND_ MODEM	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CUSTOM_PROPERTY_ILLEGAL _NAME	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CUSTOM_PROPERTY_ REDEFINITION	Return value from CommandStatus.getStatusCode().
CMD_ERROR_CUSTOM_PROPERTY_ UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DATA_UNCHANGED	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DB_UNREACHABLE	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICE_FQDN_EXISTS	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICE_FQDN_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICE_HOSTID_EXISTS	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICE_HOSTID_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICE_ID_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICE_MAC_EXISTS	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICE_SEARCH_NO_MATCH	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICE_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DEVICEID_INVALID	Return value from CommandStatus.getStatusCode(), representing a command with an invalid deviceID
CMD_ERROR_DHCP_CRITERIA_EXISTS	Return value from CommandStatus.getStatusCode(). Indicates that a DHCP Criteria with the specified name already exists in the database.

Table A-38 CommandStatusCodes Values (continued)

Value	Description
CMD_ERROR_DHCP_CRITERIA_UNKNOWN	Return value from CommandStatus.getStatusCode(). Indicates that a DHCP Criteria with the specified name is not present in the database.
CMD_ERROR_DISABLE_PLUGIN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DOCSIS_CLASS_OF_SERVICE_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DPE_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_DSTB_CLASS_OF_SERVICE_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_ENABLE	Return value from CommandStatus.getStatusCode().
CMD_ERROR_ENABLE_PLUGIN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_EVAL_LICENSE_EXTENDER_IS_LESS_THAN_EXISTING_EVAL_LICENSE_EXTENDER	Return value from CommandStatus.getStatusCode(). Indicates that the evaluation license extender is less than the previous evaluation extender.
CMD_ERROR_EVAL_LICENSE_REPLACING_EXISTING_PERM_LICENSE	Return value from CommandStatus.getStatusCode(). Indicates that an evaluation license key cannot replace an existing permanent license.
CMD_ERROR_EVALUATION_HAS_EXPIRED	Return value from CommandStatus.getStatusCode(). Indicates that the encrypted key's evaluation duration has expired.
CMD_ERROR_EXTERNAL_FILE_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_FILENAME_INVALID_REGEX	Return value from CommandStatus.getStatusCode(), representing a command with an invalid filename regex.
CMD_ERROR_FQDN_INVALID	Return value from CommandStatus.getStatusCode().
CMD_ERROR_GET_ALL_USERS	Return value from CommandStatus.getStatusCode().
CMD_ERROR_HOSTID_INVALID	Return value from CommandStatus.getStatusCode(), representing a command with an invalid hostID.
CMD_ERROR_ILLEGAL_PARAM_VALUE	Return value from CommandStatus.getStatusCode().

Table A-38 CommandStatusCodes Values (continued)

Value	Description
CMD_ERROR_INTERNAL	Return value from CommandStatus.getStatusCode().
CMD_ERROR_INVALID_DEVICE_ TECHNOLOGY	Return value from CommandStatus.getStatusCode().
CMD_ERROR_INVALID_LICENSE_KEY	Return value from CommandStatus.getStatusCode()., Represents failure of the addLicense() command in a batch.
CMD_ERROR_IPADDR_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_IPADDRESS_INVALID	Return value from CommandStatus.getStatusCode(), representing a command with an invalid IPADDRESS parameter.
CMD_ERROR_LEASE_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_LICENSE_IS_INVALID	Return value from CommandStatus.getStatusCode(). Indicates that the decoded data (from decoding the encrypted key) is invalid.
CMD_ERROR_LICENSE_IS_NOT_PERM_OR_ EVAL	Return value from CommandStatus.getStatusCode(). Indicates that the encrypted key does not have permanent or evaluation license.
CMD_ERROR_LICENSE_IS_NULL	Return value from CommandStatus.getStatusCode(). Indicates that the encrypted key is null or empty.
CMD_ERROR_LIST_PLUGIN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_MAC_AND_FQDN_BOTH_NULL	Return value from CommandStatus.getStatusCode().
CMD_ERROR_MACADDRESS_INVALID	Return value from CommandStatus.getStatusCode(), representing a command with an invalid MACADDRESS parameter.
CMD_ERROR_MTAS_BEHIND_MODEM	Return value from CommandStatus.getStatusCode().
CMD_ERROR_OWNERID_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_PERM_LICENSE_KEY_ ALREADY_EXISTS	Return value from CommandStatus.getStatusCode(). Indicates that the permanent license key already exists.
CMD_ERROR_PLUGIN_UNKNOWN	Return value from CommandStatus.getStatusCode().

Table A-38 CommandStatusCodes Values (continued)

Value	Description
CMD_ERROR_PREPARE	Return value from CommandStatus.getStatusCode().
CMD_ERROR_PROPERTY_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_PROVISIONING_GROUP_ UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_QUERY	Return value from CommandStatus.getStatusCode().
CMD_ERROR_RDU_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_ROLLBACK_ENABLE	Return value from CommandStatus.getStatusCode().
CMD_ERROR_ROLLBACK_PREPARE	Return value from CommandStatus.getStatusCode().
CMD_ERROR_SEARCH_NO_MATCH	Return value from CommandStatus.getStatusCode().
CMD_ERROR_SYSTEM_PROPERTY_REDEFINI TION	Return value from CommandStatus.getStatusCode().
CMD_ERROR_TOO_MANY_IN_LIST	Return value from CommandStatus.getStatusCode().
CMD_ERROR_UNKNOWN_TECHNOLOGY_ NAME	Return value from CommandStatus.getStatusCode(). Indicates that the encrypted key does not have a recognizable technology name.
CMD_ERROR_USER_ADMIN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_USER_ALREADY_EXISTS	Return value from CommandStatus.getStatusCode().
CMD_ERROR_USER_IS_NOT_ADMIN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_USER_MODIFICATION_ERROR	Return value from CommandStatus.getStatusCode().
CMD_ERROR_USER_UNKNOWN	Return value from CommandStatus.getStatusCode().
CMD_ERROR_VALIDATE	Return value from CommandStatus.getStatusCode().
CMD_ERROR_WRITE	Return value from CommandStatus.getStatusCode().
CMD_OK	Return value from CommandStatus.getStatusCode().
CMD_RDU_BUSY	Return value from CommandStatus.getStatusCode().

Table A-38 CommandStatusCodes Values (continued)

Value	Description
CMD_ROLLBACK_PREV_CMD_FAILED	Return value from CommandStatus.getStatusCode().
CMD_ROLLBACK_SUBS_CMD_FAILED	Return value from CommandStatus.getStatusCode().

DeviceDetailsKeys Interface

The Device Details Keys interface specifies the character strings that are used as keys to retrieve values associated with IP devices. This interface defines keys that provide detailed information about IP devices that you are querying. This interface also defines keys that provide information related to a specific technology. Table A-39 lists the device details keys.

Table A-39 DeviceDetailsKeys

Key	Description
CLIENT_CLASS	DHCP criteria for the queried device.
CLIENT_ID	Retrieves the client identifier of a device.
CLIENT_REQUESTED_HOST_NAME	Retrieves the host name that a client requested.
COMPUTER_COS	Computer class of service for the queried device.
CPE_DHCP_CRITERIA	DHCP criteria for the queried device.
CPE_SELECTION_TAGS	CPE DHCP criteria for the queried device.
CUSTOM_COS	Gets custom class of service data about a device.
CUSTOM_CPE_TYPE	Custom CPE type for the queried device.
DEVICE_TYPE	Retrieves the device type.
DHCP_CRITERIA	DHCP criteria for the queried device.
DHCP_ENVIRONMENT	DHCP configuration data - Environment dictionary.
DHCP_INFORM	DHCP configuration data - Inform dictionary.
DHCP_REQUEST	DHCP configuration data - Request dictionary.
DHCP_RESPONSE	DHCP configuration data - Response dictionary.
DOCSIS_COS	Gets the DOCSIS class of service of a device.
DOCSIS_VERSION	Obtains the DOCSIS version of a device.
DSTB_COS	DSTB class of service for the queried device.
DSTB_RELATED_DOCSIS_ MODEM_ID	Related DOCSIS modem ID for the queried DSTB device.
FQDN	FQDN (fully qualified domain name) for the queried device.
IP_ADDRESS	IP address for the queried device.
IS_BEHIND_DEVICE	Device this device is behind as determined by device detection.
IS_PROVISIONED	Provisioned state for this device.

Table A-39 DeviceDetailsKeys (continued)

Key	Description
IS_REGISTERED	Registered state for this device.
LEASE_PROPERTIES	Lease properties
MAC_ADDRESS	MAC address for the queried device.
OWNER_ID	Owner ID for the queried device.
PROPERTIES	Properties for the queried device.
PROV_GROUP	Provisioning group for the queried device.
RELAY_AGENT_CIRCUIT_ID	Relay agent circuit ID for the queried device.
RELAY_AGENT_REMOTE_ID	Relay agent remote ID for the queried device.
XGCP_MTA_PORTS_IN_SERVICE_LIST	List of ports in service for the queried XGCP device.
XGCP_MTA_RELATED_DOCSIS_MODEM_ID	Related DOCSIS modem ID for the queried XGCP MTA device.

DeviceTypeValues Interface

The Device TypeValues interface specifies the allowed values for a device type. These are returned as the values for the DeviceDetailsKeys.DEVICE_TYPE key. Table A-40 identifies these values.

Table A-40 DeviceTypeValues

Value	Description
COMPUTER	Device type value for a computer.
CUSTOM_CPE	Device type value for a custom CPE device.
DOCSIS_MODEM	Device type value for a DOCSIS modem.
DSTB	Device type value for a DSTB.
XGCP_MODEM_MTA	Device type value for a XGCP modem MTA.

DHCPCriteriaKeys Interface

The DHCPCriteriaKeys interface contains constant strings that are used as keys in DHCPCriteria objects. Table A-41 lists the DHCPCriteriaKeys.

Table A-41 DHCPCriteriaKeys

Key	Description
DHCP_CRITERIA_CLIENT_CLASS	DHCP criteria client class.
DHCP_CRITERIA_EXCLUDE_SELECTION_TAGS	DHCP criteria exclude selection tags.
DHCP_CRITERIA_INCLUDE_SELECTION_TAGS	DHCP criteria include selection tags.
DHCP_CRITERIA_NAME	DHCP criteria name.
DHCP_CRITERIA_PROPERTIES	DHCP criteria properties.

DHCPOptionKeys Interface

The DHCPOptionKeys interface specifies constant strings that are used as keys in map objects to set and get properties that apply to BPR DHCP configuration and BPR client-policy support. Table A-42 lists the DHCP option keys.

Table A-42 DHCPOptionKeys

Key	Description
DHCP_CLIENT_POLICY_ALL_SUBNETS_LOCAL	The all-subnets-local client-policy response option.
DHCP_CLIENT_POLICY_ARP_CACHE_TIMEOUT	The arp-cache-timeout client-policy response option.
DHCP_CLIENT_POLICY_BOOT_FILE	The boot-file client-policy response option.
DHCP_CLIENT_POLICY_BOOT_SIZE	The boot-size client-policy response option.
DHCP_CLIENT_POLICY_BROADCAST_ADDRESS	The broadcast-address client-policy response option.
DHCP_CLIENT_POLICY_CHADDR	The chaddr client-policy response option.
DHCP_CLIENT_POLICY_CIADDR	The ciaddr client-policy response option.
DHCP_CLIENT_POLICY_CISCO_CLIENT_LAST_TRANSACTION_TIME	The cisco-client-last-transaction-time client-policy response option.
DHCP_CLIENT_POLICY_CISCO_CLIENT_REQUESTED_HOST_NAME	The cisco-client-requested-host-name client-policy response option.
DHCP_CLIENT_POLICY_CISCO_LEASED_IP	The cisco-leased-ip client-policy response option.
DHCP_CLIENT_POLICY_CLIENT_DOMAIN_NAME	The client-domain-name client-policy response option.
DHCP_CLIENT_POLICY_CLIENT_FQDN	The client-fqdn client-policy response option.
DHCP_CLIENT_POLICY_CLIENT_HOST_NAME	The client-host-name client-policy response option.
DHCP_CLIENT_POLICY_CLIENT_ID	The client-id client-policy response option.
DHCP_CLIENT_POLICY_CLIENT_MAC_ADDRESS	The client-mac-address client-policy response option.
DHCP_CLIENT_POLICY_CLIENT_OS_TYPE	The client-os-type client-policy response option.
DHCP_CLIENT_POLICY_CLIENT_REQUESTED_HOST_NAME	The client-requested-host-name client-policy response option.
DHCP_CLIENT_POLICY_COOKIE_SERVERS	The cookie-servers client-policy response option.
DHCP_CLIENT_POLICY_DEFAULT_IP_TTL	The default-ip-ttl client-policy response option.
DHCP_CLIENT_POLICY_DEFAULT_TCP_TTL	The default-tcp-ttl client-policy response option.
DHCP_CLIENT_POLICY_DHCP_CLASS_IDENTIFIER	The dhcp-class-identifier client-policy response option.

Table A-42 DHCPOptionKeys (continued)

Key	Description
DHCP_CLIENT_POLICY_DHCP_CLIENT_IDENTIFIER	The dhcp-client-identifier client-policy response option.
DHCP_CLIENT_POLICY_DHCP_LEASE_TIME	The dhcp-lease-time client-policy response option.
DHCP_CLIENT_POLICY_DHCP_MAX_MESSAGE_SIZE	The dhcp-max-message-size client-policy response option.
DHCP_CLIENT_POLICY_DHCP_MESSAGE	The dhcp-message client-policy response option.
DHCP_CLIENT_POLICY_DHCP_MESSAGE_TYPE	The dhcp-message-type client-policy response option.
DHCP_CLIENT_POLICY_DHCP_OPTION_OVERLOAD	The dhcp-option-overload client-policy response option.
DHCP_CLIENT_POLICY_DHCP_PARAMETER_REQUEST_LIST	The dhcp-parameter-request-list client-policy response option.
DHCP_CLIENT_POLICY_DHCP_PARAMETER_REQUEST_LIST_BLOB	The dhcp-parameter-request-list-blob client-policy response option.
DHCP_CLIENT_POLICY_DHCP_REBINDING_TIME	The dhcp-rebinding-time client-policy response option.
DHCP_CLIENT_POLICY_DHCP_RENEWAL_TIME	The dhcp-renewal-time client-policy response option.
DHCP_CLIENT_POLICY_DHCP_REQUESTED_ADDRESS	The dhcp-requested-address client-policy response option.
DHCP_CLIENT_POLICY_DHCP_SERVER_IDENTIFIER	The dhcp-server-identifier client-policy response option.
DHCP_CLIENT_POLICY_DHCP_USER_CLASS_ID	The dhcp-user-class-id client-policy response option.
DHCP_CLIENT_POLICY_DOMAIN_NAME	The domain-name client-policy response option.
DHCP_CLIENT_POLICY_DOMAIN_NAME_CHANGED	The domain-name-changed client-policy response option.
DHCP_CLIENT_POLICY_DOMAIN_NAME_SERVERS	The domain-name-servers client-policy response option.
DHCP_CLIENT_POLICY_DUMP_PACKET	The dump-packet client-policy response option.
DHCP_CLIENT_POLICY_ENV_ACCEPTABLE	The acceptable client-policy environment option.
DHCP_CLIENT_POLICY_ENV_ACTION	The action client-policy environment option.
DHCP_CLIENT_POLICY_ENV_AUTHENTICATE_UNTIL	The authenticate-until client-policy environment option.
DHCP_CLIENT_POLICY_ENV_CLIENT_CLASS_NAME	The client-class-name client-policy environment option.
DHCP_CLIENT_POLICY_ENV_CLIENT_SPECIFIER	The client-specifier client-policy environment option.

Table A-42 DHCPOptionKeys (continued)

Key	Description
DHCP_CLIENT_POLICY_ENV_CNR_ FORWARD_DHCP_REQUEST	The cnr-forward-dhcp-request client-policy environment option.
DHCP_CLIENT_POLICY_ENV_CNR_ REQUEST_FORWARD_ADDRESS_LIST	The cnr-request-forward-address-list client-policy environment option.
DHCP_CLIENT_POLICY_ENV_DEFAULT_ CLIENT_CLASS_NAME	The default-client-class-name client-policy environment option.
DHCP_CLIENT_POLICY_ENV_DOMAIN_ NAME	The domain-name client-policy environment option.
DHCP_CLIENT_POLICY_ENV_DROP	The drop client-policy environment option.
DHCP_CLIENT_POLICY_ENV_ EMBEDDED_POLICY	The embedded-policy client-policy environment option.
DHCP_CLIENT_POLICY_ENV_E XCLUSION_MASK	The exclusion-mask client-policy environment option.
DHCP_CLIENT_POLICY_ENV_HOST_ NAME	The host-name client-policy environment option.
DHCP_CLIENT_POLICY_ENV_I NCLUSION_MASK	The inclusion-mask client-policy environment option.
DHCP_CLIENT_POLICY_ENV_LOG_ DROP_MESSAGE	The log-drop-message client-policy environment option.
DHCP_CLIENT_POLICY_ENV_POLICY_ NAME	The policy-name client-policy environment option.
DHCP_CLIENT_POLICY_ENV_RELEASE_ BY_IP	The release-by-ip client-policy environment option.
DHCP_CLIENT_POLICY_ENV_ SELECTION_CRITERIA	The selection-criteria client-policy environment option.
DHCP_CLIENT_POLICY_ENV_ SELECTION_CRITERIA_EXCLUDED	The selection-criteria-excluded client-policy environment option.
DHCP_CLIENT_POLICY_ENV_SKIP_ CLIENT_LOOKUP	The skip-client-lookup client-policy environment option.
DHCP_CLIENT_POLICY_ENV_ UNAUTHENTICATED_CLIENT_ CLASS_NAME	The unauthenticated-client-class-name client-policy environment option.
DHCP_CLIENT_POLICY_ENVIRONMENT_ PREFIX	The client policy prefix for CNR environment dictionary.
DHCP_CLIENT_POLICY_EXTENSIONS_ PATH	The extensions-path client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ ABSOLUTE_TIME	The failover-absolute-time client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ CLIENT_FLAGS	The failover-client-flags client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ CLIENT_OS_TYPE	The failover-client-os-type client-policy response option.

Table A-42 DHCPOptionKeys (continued)

Key	Description
DHCP_CLIENT_POLICY_FAILOVER_ LAST_TRANSACTION_TIME	The failover-last-transaction-time client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ LEASE_EXPIRATION_TIME	The failover-lease-expiration-time client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ LEASE_STATE	The failover-lease-state client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ MAC_ADDR	The failover-mac-addr client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_MCLT	The failover-mclt client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_NUM BER_ADDRESSES	The failover-number-addresses client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ REJECT_REASON	The failover-reject-reason client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ RESERVED	The failover-reserved client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ START_TIME_OF_STATE	The failover-start-time-of-state client-policy response option.
DHCP_CLIENT_POLICY_FAILOVER_ STATE_SERIAL_NUMBER	The failover-state-serial-number client-response option.
DHCP_CLIENT_POLICY_FILE	The file client-policy response option.
DHCP_CLIENT_POLICY_FINGER_ SERVERS	The finger-servers client-policy response option.
DHCP_CLIENT_POLICY_FLAGS	The flags client-policy response option.
DHCP_CLIENT_POLICY_FONT_SERVERS	The font-servers client-policy response option.
DHCP_CLIENT_POLICY_GIADDR	The giaddr client-policy response option.
DHCP_CLIENT_POLICY_HLEN	The hlen client-policy response option.
DHCP_CLIENT_POLICY_HOPS	The hops client-policy response option.
DHCP_CLIENT_POLICY_HOST_NAME	The host-name client-policy response option.
DHCP_CLIENT_POLICY_HOST_NAME_ CHANGED	The host-name-changed client-policy response option.
DHCP_CLIENT_POLICY_HOST_NAME_ IN_DNS	The host-name-in-dns client-policy response option.
DHCP_CLIENT_POLICY_HTYPE	The htype client-policy response option.
DHCP_CLIENT_POLICY_IEEE8023_ ENCAPSULATION	The ieee802.3-encapsulation client-policy response option.
DHCP_CLIENT_POLICY_IMPRESS_ SERVERS	The impress-servers client-policy response option.
DHCP_CLIENT_POLICY_INTERFACE_MTU	The interface-mtu client-policy response option.
DHCP_CLIENT_POLICY_IP_FORWARDING	The ip-forwarding client-policy response option.
DHCP_CLIENT_POLICY_IRC_SERVERS	The irc-servers client-policy response option.

Table A-42 DHCPOptionKeys (continued)

Key	Description
DHCP_CLIENT_POLICY_LEASE_RELAY_AGENT_CIRCUIT_ID	The lease-relay-agent-circuit-id client-policy response option.
DHCP_CLIENT_POLICY_LEASE_RELAY_AGENT_REMOTE_ID	The lease-relay-agent-remote-id client-policy response option.
DHCP_CLIENT_POLICY_LOG_SERVERS	The log-servers client-policy response option.
DHCP_CLIENT_POLICY_LPR_SERVERS	The lpr-servers client-policy response option.
DHCP_CLIENT_POLICY_MAC_ADDRESS	The mac-address client-policy response option.
DHCP_CLIENT_POLICY_MASK_SUPPLIER	The mask-supplier client-policy response option.
DHCP_CLIENT_POLICY_MAX_DGRAM_REASSEMBLY	The max-dgram-reassembly client-policy response option.
DHCP_CLIENT_POLICY_MCNS_SEC_SERVER	The mcns-sec-server client-policy response option.
DHCP_CLIENT_POLICY_MERIT_DUMP	The merit-dump client-policy response option.
DHCP_CLIENT_POLICY_MOBILE_IP_HOME_AGENTS	The mobile-ip-home-agents client-policy response option.
DHCP_CLIENT_POLICY_NAME_SERVERS	The name-servers client-policy response option.
DHCP_CLIENT_POLICY_NETBIOS_DD_SERVER	The netbios-dd-server client-policy response option.
DHCP_CLIENT_POLICY_NETBIOS_NAME_SERVERS	The netbios-name-servers client-policy response option.
DHCP_CLIENT_POLICY_NETBIOS_NODE_TYPE	The netbios-node-type client-policy response option.
DHCP_CLIENT_POLICY_NETBIOS_SCOPE	The netbios-scope client-policy response option.
DHCP_CLIENT_POLICY_NIS_DOMAIN	The nis-domain client-policy response option.
DHCP_CLIENT_POLICY_NIS_SERVERS	The nis-servers client-policy response option.
DHCP_CLIENT_POLICY_NISPLUS_DOMAIN	The nisplus-domain client-policy response option.
DHCP_CLIENT_POLICY_NISPLUS_SERVERS	The nisplus-servers client-policy response option.
DHCP_CLIENT_POLICY_NNTP_SERVERS	The nntp-servers client-policy response option.
DHCP_CLIENT_POLICY_NON_LOCAL_SOURCE_ROUTING	The non-local-source-routing client-policy response option.
DHCP_CLIENT_POLICY_NTP_SERVERS	The ntp-servers client-policy response option.
DHCP_CLIENT_POLICY_OP	The op client-policy response option.
DHCP_CLIENT_POLICY_OS_TYPE	The os-type client-policy response option.
DHCP_CLIENT_POLICY_PAD	The pad client-policy response option.
DHCP_CLIENT_POLICY_PATH_MTU_AGING_TIMEOUT	The path-mtu-aging-timeout client-policy response option.
DHCP_CLIENT_POLICY_PATH_MTU_PLATEAU_TABLE	The path-mtu-plateau-table client-policy response option.

Table A-42 DHCPOptionKeys (continued)

Key	Description
DHCP_CLIENT_POLICY_PERFORM_ MASK_DISCOVERY	The perform-mask-discovery client-policy response option.
DHCP_CLIENT_POLICY_POLICY_FILTER	The policy-filter client-policy response option.
DHCP_CLIENT_POLICY_POP3_SERVERS	The pop3-servers client-policy response option.
DHCP_CLIENT_POLICY_RELAY_AGENT_C IRCUIT_ID	The relay-agent-circuit-id client-policy response option.
DHCP_CLIENT_POLICY_RELAY_AGENT_ INFO	The relay-agent-info client-policy response option.
DHCP_CLIENT_POLICY_RELAY_AGENT_R EMOTE_ID	The relay-agent-remote-id client-policy response option.
DHCP_CLIENT_POLICY_REPLY_ IPADDRESS	The reply-ipaddress client-policy response option.
DHCP_CLIENT_POLICY_REPLY_PORT	The reply-port client-policy response option.
DHCP_CLIENT_POLICY_RESOURCE_ LOCATION_SERVERS	The resource-location-servers client-policy response option.
DHCP_CLIENT_POLICY_RESPONSE_ PREFIX	The client policy prefix for CNR response dictionary.
DHCP_CLIENT_POLICY_REVERSE_ NAME_IN_DNS	The reverse-name-in-dns client-policy response option.
DHCP_CLIENT_POLICY_ROOT_PATH	The root-path client-policy response option.
DHCP_CLIENT_POLICY_ROUTER_ DISCOVERY	The router-discovery client-policy response option.
DHCP_CLIENT_POLICY_ROUTER_ SOLICITATION_ADDRESS	The router-solicitation-address client-policy response option.
DHCP_CLIENT_POLICY_ROUTERS	The routers client-policy response option.
DHCP_CLIENT_POLICY_SECS	The secs client-policy response option.
DHCP_CLIENT_POLICY_SIADDR	The siaddr client-policy response option.
DHCP_CLIENT_POLICY_SMTP_SERVERS	The smtp-servers client-policy response option.
DHCP_CLIENT_POLICY_SNAME	The sname client-policy response option.
DHCP_CLIENT_POLICY_STATIC_ROUTES	The static-routes client-policy response option.
DHCP_CLIENT_POLICY_STREETALK_ DA_SERVERS	The streetalk-da-servers client-policy response option.
DHCP_CLIENT_POLICY_STREETTALK_ SERVERS	The streettalk-servers client-policy response option.
DHCP_CLIENT_POLICY_SUBNET_MASK	The subnet-mask client-policy response option.
DHCP_CLIENT_POLICY_SWAP_SERVER	The swap-server client-policy response option.
DHCP_CLIENT_POLICY_TCP_KEEPALIVE _GARBAGE	The tcp-keepalive-garbage client-policy response option.
DHCP_CLIENT_POLICY_TCP_KEEPALIVE _INTERVAL	The tcp-keepalive-interval client-policy response option.

Table A-42 DHCPOptionKeys (continued)

Key	Description
DHCP_CLIENT_POLICY_TFTP_SERVER	The tftp-server client-policy response option.
DHCP_CLIENT_POLICY_TIME_OFFSET	The time-offset client-policy response option.
DHCP_CLIENT_POLICY_TIME_SERVERS	The time-servers client-policy response option.
DHCP_CLIENT_POLICY_TRAILER_ ENCAPSULATION	The trailer-encapsulation client-policy response option.
DHCP_CLIENT_POLICY_VENDOR_ ENCAPSULATED_OPTIONS	The vendor-encapsulated-options client-policy response option.
DHCP_CLIENT_POLICY_WWW_SERVERS	The www-servers client-policy response option.
DHCP_CLIENT_POLICY_X_DISPLAY_ MANAGER	The x-display-manager client-policy response option.
DHCP_CLIENT_POLICY_XID	The xid client-policy response option.
DHCP_CLIENT_POLICY_YIADDR	The yiaddr client-policy response option.
DHCP_SUPPORTED_OPTIONS_ ENVIRONMENT	Comma-separated list of DHCP environment dictionary option names.
DHCP_SUPPORTED_OPTIONS_INFORM	Comma-separated list of BPR inform dictionary option names.
DHCP_SUPPORTED_OPTIONS_PREDNS_ ENVIRONMENT	Comma-separated list of DHCP pre-dns-add-forward dictionary option names.
DHCP_SUPPORTED_OPTIONS_REQUEST	Comma-separated list of DHCP request dictionary option names.
DHCP_SUPPORTED_OPTIONS_RESPONSE	Comma-separated list of DHCP response dictionary option names.

DocsisDefaultKeys Interface

The DocsisDefaultsKeys interface specifies constant strings that are used as keys in map objects to set and get properties that apply to DOCSIS defaults and RDU defaults. Table A-43 lists the DOCSIS default keys.

Table A-43 DocsisDefaultKeys

Key	Description
CMTS_MIC_ENABLED	The DOCSIS default strings.
CMTS_MIC_SHARED_SECRET	The CMTS MIC shared secret.
DYNAMIC_FILE_EXTENSION	Filename extension for dynamic DOCSIS files.
STATIC_FILE_EXTENSION	Filename extension for static DOCSIS files.
TFTP_MODEM_ADDRESS_OPTION_ ENABLED	Enables the TFTP modem address option.
TFTP_TIMESTAMP_OPTION_ENABLED	The TFTP time stamp option enabled flag.

DPEDetailsKeys Interface

The DPEDetails interface specifies constant strings that are used as keys in map objects returned by query methods on a DPE. Table A-44 lists the DPE details keys.

Table A-44 DPEDetailsKeys

Key	Description
DPE_AGENT_HOST_ID	Host identifier of the agent associated with a DPE.
DPE_CONFIGS	Cached configurations of the queried DPE.
DPE_FILES	Cached files of the queried DPE.
DPE_HEALTH	Health of the queried DPE.
DPE_HITS	Cache hits by the queried DPE.
DPE_HOST_ID	Host identifier for the queried DPE.
DPE_MISSES	Cache misses by the queried DPE.
DPE_PRIMARY_PROV_GROUP_LIST	List of primary provisioning group identifiers for the queried DPE.
DPE_PROPERTIES	Properties for the queried DPE.
DPE_REQ	TFTP file requests to the TFTP server on the queried DPE.
DPE_SECONDARY_PROV_GROUP_LIST	List of secondary provisioning group identifiers for the queried DPE.
DPE_SENT	TFTP files successfully sent by the TFTP server on the queried DPE.
DPE_STATE	State of the queried DPE.
DPE_UPTIME	Uptime for the queried DPE.
DPE_VERSION	Software version of the queried DPE.

IPDeviceKeys Interface

The IPDeviceKeys interface specifies constant strings that are used as keys in properties map objects for IP device provisioning interfaces. Table A-45 lists the IP device keys.

Table A-45 IPDeviceKeys

Key	Description
MUST_BE_BEHIND_DEVICE	If a device is not behind the modem or DSTB set in this property, return the device to unprovisioned status.
MUST_BE_IN_PROV_GROUP	If the provisioning group is not one that is specified, insert this property to return the device to the unprovisioned status. **Note** If this is set to the name of a provisioning group, the device will only get provisioned access while in the designated provisioning group.
UNCOMMITED_CONFIG_TIME_TO_LIVE	Specifies the amount of time (in milliseconds) that a DPE should hold on to a configuration file that has not yet been committed for this device.

ModemKeys Interface

The ModemKeys interface specifies constant strings that are used as keys in map objects returned by query methods on DOCSIS interfaces. Table A-46 lists the modem keys.

Table A-46 ModemKeys

Key	Description
CUSTOM_CPE_FIRMWARE_VERSION	Firmware version.
DOCSIS_MODEM_FIRMWARE_VERSION	Firmware version.
PROMISCUOUS_MODE_ENABLED	Tells whether the promiscuous mode is enabled or not.

RDUDetailsKeys Interface

The GetDetailsThisRDU interface specifies constant strings that are used as keys in map objects returned by query methods on the RDU. Table A-47 lists the RDU details keys.

Table A-47 RDUDetailsKeys

Key	Description
RDU_AGENT_HOST_ID	Host identifier of the agent associated with the RDU.
RDU_HEALTH	Health of the RDU.

Table A-47 RDUDetailsKeys (continued)

Key	Description
RDU_HOST_ID	Host identifier for the queried RDU.
RDU_HOST_PORT	Port number for the queried RDU.
RDU_PACE_AVERAGE_BATCH_TIME	PACE average batch time for the queried RDU.
RDU_PACE_AVERAGE_PROCESSING_TIME	PACE average processing time for the queried RDU.
RDU_PACE_BATCHES_DROPPED	PACE batches dropped for the queried RDU.
RDU_PACE_BATCHES_FAILED	PACE batches failed for the queried RDU.
RDU_PACE_BATCHES_PROCESSED	PACE batches processed for the queried RDU.
RDU_PACE_BATCHES_SUCCEEDED	PACE batches succeeded for the queried RDU.
RDU_PACE_UPTIME	PACE uptime for the queried RDU.
RDU_PROPERTIES	Properties for the queried RDU.
RDU_STATE	State of the queried RDU.
RDU_TOTAL_COMPUTERS	Total number of computer in the database for the queried RDU.
RDU_TOTAL_DOCSIS_MODEMS	Total number of DOCSIS modems in the database for the queried RDU.
RDU_TOTAL_DSTB	Total number of DSTB devices in the database for the queried RDU.
RDU_TOTAL_XGCP	Total number of XGCP devices in the database for the queried RDU.
RDU_UPTIME	Uptime for the queried RDU.
RDU_VERSION	Software version of the queried RDU.

SearchType Interface

The SearchType interface provides an enumeration of search types. Table A-48 lists the SearchType keys.

Table A-48 SearchType Keys

Key	Description
ByFQDN	References an FQDN search type.
ByMAC	References a MAC address search type.
ByMACAndFQDN	References a combined MAC address and FQDN search type.

ServerDefaultsKeys Interface

The ServerDefaultsKeys interface contains constants used by servers. Table A-49 lists the server defaults keys.

Table A-49 ServerDefaultsKeys

Key	Description
DPE_CONFIG_POPULATION_BACKOFF	The delay times between device configuration population attempts.
DPE_CONFIG_POPULATION_DELAY	The time to wait between generation requests for device configurations during cache population.
DPE_CONFIG_REQUEST_BUFFERS	The number of buffers for holding configuration generation requests.
DPE_CONFIG_REQUEST_THREADS	The maximum number of concurrent configuration generation requests.
DPE_CONFIG_SYNCHRONIZATION_ BACKOFF	The delay times between getSyncRevNumberForIPDevices retries.
DPE_CONFIG_TIMEOUT	The time to wait for a response to a configuration generation request.
DPE_FILE_POPULATION_BACKOFF	The delay times between external file population attempts.
DPE_FILE_POPULATION_DELAY	The time to wait between requests for external file updates during cache population.
DPE_FILE_REQUEST_BUFFERS	The number of buffers for holding external file requests.
DPE_FILE_REQUEST_THREADS	The maximum number of concurrent external file requests.
DPE_FILE_SYNCHRONIZATION_ BACKOFF	The delay times between getSyncRevNumberForExternalFiles retries.
DPE_FILE_TIMEOUT	The time to wait for a response to an external file request.
DPE_MIXING_IP_ADDRESS_ENABLE_ KEY	Enables mixing the DPE IP address into the DHCP response.
DPE_SYNCHRONIZATION_DELAY	The time to wait between synchronization attempts.
DPE_SYNCHRONIZATION_TIMEOUT	The time to wait for a response to a server synchronization request.
DPE_TIMESERVER_ENABLE_KEY	The enable flag for the time of day server.
RDU_CONFIGURATION_EXTENSION_ POINT	The common RDU technology configuration extension point.
RDU_DEVICE_DETECTION_EXTENSION_ POINT	The RDU device detection extension point.
RDU_MAX_GET_SYNC_REV_NUMBERS_ FOR_IP_DEVICES	The maximum number of concurrent calls to getSyncRevNumbersForIPDevices the RDU will support.
SERVER_CONNECTION_DELAY	The time to wait between connection attempts.

Table A-49 ServerDefaultsKeys (continued)

Key	Description
SERVER_REGISTRATION_DELAY	The time to wait between registration attempts.
SERVER_REGISTRATION_TIMEOUT	The time to wait for a response to a server registration request.
SERVER_REGISTRATION_VERSIONINFO	Implementation-Version of the server being registered.
SERVER_SYSLOG_ENABLE	The log level for the syslog (standard out).
SERVER_TRACE_AGENT	Represents all the tracing properties that can be set via the API. Set the flag to enabled to enable tracing category, and to disabled to disable it. The disable flag is set by default.
UNCOMMITTED_CONFIG_TIME_TO_ LIVE	Specifies the default amount of time, in milliseconds, a DPE should hold on to a configuration file that has not been committed yet.

SNMPPropertyKeys Interface

The SNMP Property Keys interface specifies character strings that are used as keys to store and retrieve values associated with devices that respond to SNMP requests. Table A-50 lists the SNMPPropertyKeys.

Table A-50 SNMPPropertyKeys

Key	Description
READ_COMMUNITY_STRING	Gets the SNMP read community string.
WRITE_COMMUNITY_STRING	Gets the SNMP write community string.

TechnologyDefaultsKeys Interface

The TechnologyDefaultsKeys interface specifies constant strings that are used as keys in map objects to set and get properties that apply to technology defaults. Table A-51 lists the TechnologyDefaultsKeys.

Table A-51 TechnologyDefaultsKeys

Key	Description
COMPUTER_DEFAULT_CLASS_ OF_SERVICE	The default class of service for computers.
COMPUTER_DEFAULT_DHCP_CRITERIA	The default DHCP criteria for computers.
COMPUTER_DEFAULT_PROV_ PROMISCUOUS_DHCP_CRITERIA	The default provisioned promiscuous mode dhcp criteria for the computer technology.
COMPUTER_DISRUPTION_EXTENSION_ POINT	The device disruption extension point for computers.
COMPUTER_EXTENSION_POINT	The extension point for computers.

Table A-51 TechnologyDefaultsKeys (continued)

Key	Description
CUSTOM_CPE_DEFAULT_CLASS_OF_SERVICE	The default class of service for custom CPE devices.
CUSTOM_CPE_DEFAULT_DHCP_CRITERIA	The default DHCP criteria for custom CPE devices.
CUSTOM_CPE_DISRUPTION_EXTENSION_POINT	The device disruption extension point for custom CPE devices.
CUSTOM_CPE_EXTENSION_POINT	The extension point for custom CPE devices.
DOCSIS_DEFAULT_CLASS_OF_SERVICE	The default class of service for DOCSIS devices.
DOCSIS_DEFAULT_DHCP_CRITERIA	The default DHCP criteria for DOCSIS devices.
DOCSIS_DISRUPTION_EXTENSION_POINT	The device disruption extension point for DOCSIS devices.
DOCSIS_EXTENSION_POINT	The extension point for DOCSIS devices.
DSTB_DEFAULT_CLASS_OF_SERVICE	The default class of service for DSTB devices
DSTB_DEFAULT_DHCP_CRITERIA	The default DHCP criteria for DSTB devices.
DSTB_DISRUPTION_EXTENSION_POINT	The device disruption extension point for DSTB devices.
DSTB_EXTENSION_POINT	The extension point for DSTB devices.
XGCP_SIGNALING_TYPE	The XGCP signaling type: S = Simple
XGCP_USE_OLD_STYLE	Specified whether to use the old format for merit-dump string.
XGCP_VERSION_NUMBER	The version number.

UserDetailsKeys Interface

The UserDetailsKeys interface specifies constant strings that are used as keys in map objects returned by query methods on user objects. Table A-52 lists the user details keys.

Table A-52 UserDetailsKeys

Key	Description
USER_DESCRIPTION	User description for the queried user object.
USER_ID	User ID for the queried user object.
USER_ISADMIN	Checks if the user an administrator for the queried user object.
USER_PASSWORD	User password for the queried user object.

VOIPServiceDetailsKeys Interface

The VOIPServiceDetailsKeys interface specifies constant strings that are used as keys in map objects returned by query methods on VoIP services. Table A-53 lists the VoIP service details keys.

Table A-53 VOIPServiceDetailsKeys

Key	Description
CMS_FQDN	CMS FQDN for the queried service.
CMS_PORT_NUMBER	CMS port number for the queried service.
CMS_QUALIFIER	CMS qualifier for the queried service.
PROPERTIES	Properties for the queried service.
TELEPHONE_NUMBER	Telephone number for the queried service.

XGCPCommandStatusCodes Interface

The XGCPCommandStatusCodes interface specifies constants that are used to interpret returned xGCP command and query status information. The CommandStatus.getStatusCode() command returns values to indicate that an xGCP command processing error occurred. Command failure occurs as a result of these causes:

- The xGCP service already exists
- The command is trying to operate on an unknown xGCP service.

XGCPProvisioningKeys Interface

The XGCPProvisioningKeys interface specifies the constants that are used to interpret and set the values associated with xGCP devices, for example, the SGCP version string (SGCP_VERSION). Table A-54 lists the XGCPProvisioningKeys.

Table A-54 XGCPProvisioningKeys

Key	Description
XGCP_SIGNALING_TYPE	Signaling type for an xGCP port.
XGCP_USE_OLD_STYLE	Indicates whether to use the old style merit-dump string.
XGCP_VERSION_NUMBER	xGCP version used for an xGCP port.

com.cisco.provisioning.cpe.events Package

The com.cisco.provisioning.cpe.events package contains the interfaces and methods used to manage events. Table A-55 lists the interfaces in the com.cisco.provisioning.cpe.events package.

Table A-55 The com.cisco.provisioning.cpe.events Package Interfaces

Interface	Description
BatchEvent	Defines the batch events.
BatchListener	The listener interface for BatchEvents.
ProvAPI	The base interface from which individual event category objects inherit.
ProvAPIListener	Tags event listeners as being associated with the BPR event subsystem. All BPR event listeners inherit from this interface.
ProvAPIManager	Outlines the methods for managing events.
COSEvent	The interface that defines the object to be returned as a result of a class of service event notification.
COSListener	The listener interface for COSEvents.
DeviceEvent	Defines the object to be returned as a result of a DeviceEvent notification.
DeviceListener	The listener interface for DeviceEvent.
DHCPCriteriaEvent	The interface that defines the object to be returned as a result of a DHCPCriteriaEvent notification.
DHCPCriteriaListener	The listener interface for DeviceCriteriaEvent.
ExternalFileEvent	Defines the object to be returned as a result of an ExternalFileEvent notification.
ExternalFileListener	The listener interface for ExternalFileEvent.
MessagingEvent	Defines the object to be returned as a result of a MessagingEvent notification.
MessagingListener	The listener interface for MessagingEvent.
ProvGroupEvent	Defines the object to be returned as a result of an event in a provisioning group.
ProvGroupListener	The listener interface for ProvGroupEvent.
SystemConfigEvent	The interface that defines the object to be returned as a result of a SystemConfig event notification.
SystemConfigListener	The listener interface for SystemConfigEvent.

Table A-56 lists the com.cisco.provisioning.cpe.events classes.

Table A-56 The com.cisco.provisioning.cpe.events Classes

Class	Description
BatchAdapter	An abstract class that provides no-op definitions of all methods in the BatchListener interface.
BatchEventQualifier	An implementation of the qualifier class for use with BatchEvents.
COSAdapter	The listener interface for DeviceMod events.
COSEventQualifier	An implementation of the qualifier class for use with COSEvents.
DeviceAdapter	An abstract class that provides no-op definitions of all methods in the DeviceListener interface.
DeviceEventQualifier	An implementation of the qualifier class for use with DeviceEvents.
DHCPCriteriaAdapter	The listener interface for DeviceMod events.
DHCPCriteriaEventQualifier	An implementation of the qualifier class for use with DHCPCriteriaEvents.
ExternalFileAdapter	An abstract class that provides no-op definitions of all methods in the ExternalFileListener interface.
ExternalFileEventQualifier	An implementation of the qualifier class for use with ExternalFileEvents.
MessagingAdapter	An abstract class that provides no-op definitions of all methods in the MessagingListener interface.
MessagingQualifier	An implementation of the qualifier class for use with Messaging events.
ProvGroupAdapter	An abstract class that provides no-op definitions of all methods in the ProvGroupListener interface.
ProvGroupEventQualifier	An implementation of the qualifier class for use with ProvGroupEvents.
Qualifier	The class that determines whether or not to notify the associated listener for a given event.
QualifyAll	An implementation of the Qualifier class that returns true all the time. Therefore, all listener's that register with this qualifier receive all events.
ServerConfigEventType	Provides an enumeration of SystemConfigEventTypes signaled by a SystemConfigEvent of subclass SystemConfigListener.SERVER_DEFAULTS_CHANGE.
SystemConfigAdapter	An abstract class that provides no-op definitions of all methods in the SystemConfigListener interface.
SystemConfigEventQualifier	This implementation of the Qualifier class is to be used with SystemConfigEvents.
SystemConfigEventType	Provides an enumeration of SystemConfigEventTypes signaled by SystemConfigEvent.

Table A-57 lists the com.cisco.provisioning.cpe.events exceptions.

Table A-57 The com.cisco.provisioning.cpe.events Exceptions

Exception	Description
RegistrationFailedException	Thrown when registration of an event listener fails.

The following sections provide a summary of the com.cisco.provisioning.cpe.events interfaces, classes, and exceptions.

BatchEvent Interface

The BatchEvent interface defines batch events. Table A-58 lists the BatchEvent interface methods.

Table A-58 BatchEvent Methods

Method	Description
getBatchStatus()	Returns the status of the batch.

BatchListener Interface

The BatchListener interface is the listener interface for batch events. Table A-59 lists the BatchListener events.

Table A-59 BatchListener Methods

Method	Description
completion()	The method invoked when a batch event arrives.

ProvAPIEvent Interface

The ProvAPI interface is the base interface from which individual event category objects inherit. Table A-60 lists the ProvAPI methods.

Table A-60 ProvAPI Methods

Method	Description
getID()	Returns the identifier of the specific event category member.
getSource()	Returns the source of the specific event.
setID()	Sets the identifier of the specific event category member.
setSource()	Sets the source of the specific event.
toString()	Returns a string representation of this object.

ProvAPIEventListener Interface

The ProvAPIListener interface tags the listening entity as associated with the event subsystem. All event listeners inherit from this interface. Table A-61 lists the ProvAPIListener methods.

Table A-61 ProvAPIListener Methods

Method	Description
getOneShot()	Gets the oneShot mode value, specifying whether or not the listener is registered for a single occurrence of the events.
setOneShot()	Sets the oneShot mode value, specifying whether or not the listener is registered for a single occurrence of the events.

ProvAPIEventManager Interface

The ProvAPIManager contains the methods for managing events. Table A-62 lists the ProvAPIManager methods.

Table A-62 ProvAPIManager Methods

Method	Description
addBatchListener()	Adds a BatchListener interface.
addCOSListener()	Adds a COSListener interface to receive those COSEvents that satisfy the given Qualifier.
addDeviceListener()	Adds a DeviceListener interface.
addDHCPCriteriaListener()	Adds a DHCPCriteriaListener interface to receive those DHCPCriteriaEvents that satisfy the given Qualifier.
addExternalFileListener()	Adds an ExternalFileListener interface.
addMessagingListener()	Adds a MessagingListener interface.
addProvGroupListener()	Adds a ProvGroupListener interface.
addSystemConfigListener()	Adds a SystemConfigListener interface.
fireBatchEvent()	Notifies all registers that a BatchEvent has occurred.
fireCOSEvent()	Notifies all registered listeners that the class of service event has occurred.
fireDeviceEvent()	Notifies all registers that a DeviceEvent has occurred.
fireDHCPCriteriaEvent()	Notifies all registered listeners that the given event has occurred.
fireExternalFileEvent()	Notifies all registered listeners that the given event has occurred.
fireExternalFileEvent()	Notifies all registers that a ExternalFileEvent has occurred.
fireMessagingEvent()	Notifies all registers that a MessagingEvent has occurred.
fireProvGroupEvent()	Notifies all registers that a ProvGroupEvent has occurred.
fireSystemConfigEvent()	Notifies all registers that a SystemConfigEvent has occurred.
removeBatchListener()	Stops the BatchListener from receiving event notifications.

Table A-62 ProvAPIManager Methods (continued)

Method	Description
removeCOSListener()	Stops the given COSListener from receiving event notifications of those events that satisfy the given qualifier.
removeDeviceListener()	Stops the DeviceListener from receiving event notifications.
removeDHCPCriteriaListener()	Stops the given DHCPCriteriaListener from receiving event notifications of events that satisfy the given qualifier.
removeExternalFileListener()	Stops the ExternalFileListener from receiving event notifications.
removeMessagingListener()	Stops the MessagingListener from receiving event notifications.
removeProvGroupListener()	Stops the ProvGroupListener from receiving event notifications.
removeSystemConfigListener()	Stops the SystemConfigListener from receiving event notifications.

COSEvent Interface

The COSEvent interface contains the methods that define the object returned as a result of a class of service event notification. Table A-63 lists the COSEvent interface methods.

Table A-63 COSEvent Interface Methods

Method	Description
getName()	Returns the name of the class of service.
getProperties()	Returns the properties of the class of service.
getType()	Returns the type of the class of service.

COSListener Interface

The COSListener is the listener interface for device modification (G50

) events.

Table A-64 COSListener Methods

Method	Description
deletedCOS()	The method invoked when a COSEvent arrives after a class of service is deleted.
newCOS()	The method invoked when a COSEvent arrives after a class of service is added.

DeviceEvent Interface

The DeviceEvent interface defines the object to be returned after a DeviceMod event notification. Table A-65 lists the DeviceEvent methods.

Table A-65 DeviceEvent Methods

Method	Description
getDeviceFQDN()	Returns the FQDN of the device.
getDeviceID()	Returns the device identifier (MAC address).
getDeviceIP()	Returns the IP address of the device.
getNewData()	Returns the new data for the event including IP address, device identifier, class of service, and provisioning group.
getOldData()	Returns the old data for the event including IP address, device identifier, class of service, and provisioning group.

DeviceListener Interface

The DeviceListener interface is the listener interface for DeviceMod events. Table A-66 lists the DeviceListener methods.

Table A-66 DeviceListener Methods

Method	Description
changedCoS()	Invoked when a DeviceEvent arrives after a new device receives a new class of service.
changedIP()	Invoked when a DeviceEvent arrives after a device receives a new IP address.
deletedDevice()	Invoked when a DeviceEvent arrives after the deletion of a device.
deletedVoiceService()	Invoked when a DeviceEvent arrives as a result of a deletion of voice service.
newProvDevice()	Invoked when a DeviceEvent arrives after the provisioning of a new device.
newUnprovDevice()	Invoked when a DeviceEvent arrives after the provisioning of a new, unprovisioned device.
newVoiceService()	Invoked when a DeviceEvent arrives after the addition of a new voice service.
roaming()	Invoked when a DeviceEvent arrives after relocation of a device.

DHCPCriteriaEvent

The DHCPCriteriaEvent interface defines the object returned as a result of a DHCP criteria event notification. Table A-67 lists the DHCPCriteriaEvent methods.

Table A-67 DHCPCriteriaEvent Interface

Method	Description
getClientClass()	Returns the client class.
getExcludeSelectionTags()	Returns the exclude selection tags.
getIncludeSelectionTags()	Returns the include selection tags.
getName()	Returns the DHCP criteria name.
getProperties()	Returns the DHCPcriteria properties.

DHCPCriteriaListener

The DHCPCriteriaListener interface is the listener interface for DeviceMod events. Table A-68 lists the DHCPCriteriaListener methods.

Table A-68 DHCPCriteriaListener Interface

Method	Description
deletedDHCPCriteria()	The method invoked when a DHCPCriteriaEvent arrives after a DHCP criteria is deleted.
newDHCPCriteria()	The method invoked when a DHCPCriteriaEvent arrives after a DHCP criteria is added.

ExternalFileEvent Interface

The ExternalFileEvent interface defines the object to be returned after an ExternalFile event notification. Table A-69 lists the ExternalFileEvent methods.

Table A-69 ExternalFileEvent Methods

Method	Description
getFilename()	Returns the name of the external file.

MessagingEvent Interface

The MessagingEvent interface defines the object to be returned after a MessagingEvent notification. Table A-70 lists the MessagingEvent methods.

Table A-70 MessagingEvent Methods

Method	Description
getAddress()	Returns the internet address of this connection.
getPersistence()	Returns the persistence of the connection.
getPort()	Returns the port number of this connection.

MessagingListener Interface

The MessagingListener interface is the listener interface for MessagingEvents. Table A-71 lists the MessagingListener methods.

Table A-71 MessagingListener Methods

Method	Description
connectionStarted()	Invoked when a MessagingEvent arrives after a connection starts.
connectionStopped()	Invoked when a MessagingEvent arrives after a connection stops.

ProvGroupEvent Interface

The ProvGroupEvent interface defines the object to be returned after an event in a provisioning group. Table A-72 lists the ProvGroupEvent methods.

Table A-72 ProvGroupEvent Methods

Method	Description
getProvGroupID()	Returns the identifier of the provisioning group.
getProvisioningGroup()	Returns the attribute tree containing information about the provisioning group.

ProvGroupListener Interface

The ProvGroupListener interface is the listener for ProvGroup events. Table A-73 lists the ProvGroupListener methods.

Table A-73 ProvGroupListener Methods

Method	Description
modified()	Invoked when a ProvGroup event arrives.

Qualifier Interface

Use the Qualifier interface to determine whether to notify the associated listener for a given event. Table A-74 lists the Qualifier methods.

Table A-74 Qualifier Methods

Method	Description
qualifies()	Returns true after a determination that an event satisfies the requirements for notifying the associated listener.

SystemConfigEvent Interface

The SystemConfigEvent interface defines the object to be returned after a SystemConfig event notification. Table A-75 lists the SystemConfigEvent methods.

Table A-75 SystemConfigEvent Methods

Method	Description
getAddedAndChanged Settings()	Returns the new or changed system defaults or properties.
getRemovedSettings()	Returns the removed system defaults or properties.
getServerHostID()	Returns the server host identifier, if the SystemConfigEvent identifier indicates a change to a particular server.
getType()	Returns the SystemConfigEvent type.

SystemConfigListener Interface

The SystemConfigListener interface is the listener for ExternalFile events. Table A-76 lists the SystemConfigListener methods.

Table A-76 SystemConfigListener Methods

Method	Description
serverDefaultsChanged()	Invoked when a SystemConfigEvent arrives after a change in server defaults.
serverPropertiesChanged()	Invoked when a SystemConfigEvent arrives after a change in a server's properties.
systemConfigChanged()	Invoked when a SystemConfigEvent arrives after a change in system configuration.
systemDefaultsChanged()	Invoked when a SystemConfigEvent arrives after a change in system defaults.

BatchAdapter Class

The BatchAdapter class provides definitions of all methods in the BatchListener interface. Table A-77 lists the BatchAdapter methods.

Table A-77 BatchAdapter Methods

Method	Description
completion()	Invoked when a BatchEvent arrives.
getOneShot()	Gets the oneShot mode value, which specifies whether the listener is registered for just one occurrence of the event.
setOneShot()	Sets the oneShot mode value, which specifies that the registration request is for just one occurrence of the event.

BatchEventQualifier Class

This implementation of the BatchEventQualifier class is used with BatchEvents. Table A-78 lists the BatchEventQualifier methods.

Table A-78 BatchEventQualifier Methods

Method	Description
qualifies()	Returns true if the implementation has determined that this event satisfies the requirements to notify the associated listener.
setBatchID()	Sets the qualifies method to return true to all BatchEvents related to the Batch specified.

COSEventQualifier Class

This implementation of the COSEventQualifier class is used with COSEvents. Table A-79 lists the COSEventQualifier methods.

Table A-79 COSEventQualifier Type Methods

Method	Description
clearAll()	Clears all options that have been set.
qualifies()	Returns true if the implementation has determined that this event satisfies the requirements to notify the associated listener.
setDeletedCOS()	Sets the qualifies method to return true to all COSEvents that are DELETED_COS sub-events.
setNewCOS()	Sets the qualifies met.

DeviceAdapter Class

The DeviceAdapter class provides no-op definitions of all methods in the DeviceListener interface. Table A-80 lists the DeviceAdapter methods.

Table A-80 DeviceAdapter Methods

Method	Description
changedIP()	Invoked when a DeviceEvent arrives after a device receives a new IP address.
deletedDevice()	Invoked when a DeviceEvent arrives after a device has been deleted.
deletedVoiceService()	Invoked when a DeviceEvent arrives after a voice service has been deleted.
getOneShot()	Gets the oneShot mode value, which specifies whether the listener is registered for just one occurrence of the event.
newCos()	Invoked when a DeviceEvent arrives after a new device receives a new class of service.
newProvDevice()	Invoked when a DeviceEvent arrives after a new device receives provisioning.
newUnprovDevice()	Invoked when a DeviceEvent arrives after the detection of a new, unprovisioned device.
newVoiceService()	Invoked when a DeviceEvent arrives after a voice service has been added.
roaming()	Invoked when a DeviceEvent arrives after the relocation of a device.
setOneShot()	Sets the oneShot mode value, which specifies that the registration request is for just one occurrence of the event.

DeviceEventQualifier Class

This implementation of the DeviceEventQualifier class is used with DeviceEvents. Table A-81 lists the SystemConfigEventType methods.

Table A-81 DeviceEventQualifier Type Methods

Method	Description
clearAll()	Clears all options that have been set.
qualifies(ProvAPIEvent event)	Returns true if the implementation has determined that this event satisfies the requirements to notify the associated listener.
setChangedCOS()	Sets the qualifies method to return true to all DeviceEvents that are CHANGED_COS subevents.
setChangedIP()	Sets the qualifies method to return true to all DeviceEvents that are CHANGED_IP subevents.
setDeletedDevice()	Sets the qualifies method to return true to all DeviceEvents that are DELETED_DEVICE subevents.

Table A-81 DeviceEventQualifier Type Methods (continued)

Method	Description
setDeletedVoiceService()	Sets the qualifies method to return true to all DeviceEvents that are DELETED_VOICE_SERVICE subevents.
setDeviceID(java.lang.String deviceMac)	Sets the qualifies method to return true to the DeviceEvents related to the device specified.
setNewDevice()	Sets the qualifies method to return true to all DeviceEvents that corresponds to addition of new devices (both registered and unregistered).
setNewProvDevice()	Sets the qualifies method to return true to all DeviceEvents that are NEW_PROV_DEVICE subevents.
setNewUnprovDevice()	Sets the qualifies method to return true to all DeviceEvents that are NEW_UNPROV_DEVICE subevents.
setNewVoiceService()	Sets the qualifies method to return true to all DeviceEvents that are NEW_VOICE_SERVICE subevents.
setRoamingDevice()	Sets the qualifies method to return true to all DeviceEvents that are ROAMING_DEVICE subevents.

DHCPCriteriaEventQualifier Class

This implementation of the DHCPCriteriaEventQualifier class is used with DHCPCriteriaEvents. Table A-82 lists the DHCPCriteriaEventQualifier methods.

Table A-82 DHCPCriteriaEventQualifier Methods

Method	Description
clearAll()	Clears all options that have been set.
qualifies(ProvAPIEvent event)	Returns true if the implementation has determined that this event satisfies the requirements to notify the associated listener.
setDeletedDHCPCriteria()	Sets the qualifies method to return true to all DHCPCriteriaEvents that are DELETED_DHCP_CRITERIA subevents.
setNewDHCPCriteria()	Sets the qualifies method to return true to all DHCPCriteriaEvents that corresponds to addition of new DHCP criteria (NEW_DHCP_CRITERIA subevents).

ExternalFileAdapter Class

The ExternalFileAdapter class provides no-op definitions of all methods in the ExternalFileListener interface. Table A-83 lists the ExternalFileAdapter methods.

Table A-83 ExternalFileAdapter Methods

Method	Description
added()	Invoked when an ExternalFile event arrives after the creation of a new external file.
deleted()	Invoked when an ExternalFile event arrives after the deletion of an external file.
getOneShot()	Gets the oneShot mode value, which specifies whether the listener is registered for just one occurrence of the event.
replaced()	Invoked when an ExternalFile event arrives after the replacement of an external file.
setOneShot()	Sets the oneShot mode value, which specifies that the registration request is for just one occurrence of the event.

ExternalFileEventQualifier Class

This implementation of the ExternalFileEventQualifier class is used with ExternalFileEvents. Table A-84 lists the e ExternalFileEventQualifier type methods.

Table A-84 ExternalFileEventQualifier Methods

Method	Description
clearAll()	Clears all options that have been set.
qualifies(ProvAPIEvent event)	Returns true if the implementation has determined that this event satisfies the requirements to notify the associated listener.
setDeletedExternalFile()	Sets the qualifies method to return true to all ExternalFileEvents that are DELETE_FILE subevents.
setNewExternalFile()	Sets the qualifies method to return true to all ExternalFileEvents that corresponds to addition of a new External File (ADD_FILE subevents).
setReplaceExternalFile()	Sets the qualifies method to return true to all ExternalFileEvents that are REPLACE_FILE subevents.

MessagingAdapter Class

The MessagingAdapter class provides definitions of all methods in the MessagingListener interface. Table A-85 lists the MessagingAdapter methods.

Table A-85 MessagingAdapter Methods

Method	Description
connectionStarted()	Invoked when a MessagingEvent arrives after a connection starts.
connectionStopped()	Invoked when a MessagingEvent arrives after a connection stops.
getOneShot()	Gets the oneShot mode value that specifies whether the listener is registered for just one occurrence of the event.
setOneShot()	Sets the oneShot mode value that specifies a registration request for just one occurrence of the event.

MessagingQualifier Class

The MessagingQualifier class is the implementation of the Qualifier class to be used with the MessagingEvents interface. Table A-86 lists the MessagingQualifier methods.

Table A-86 MessagingQualifier Methods

Method	Description
clearAll()	Clears all options.
qualifies()	Returns true if the implementation has determined that this event satisfies the requirements to notify the associated listener.
setAllConnectionsDown()	Sets the qualifies method to return true to all MessagingEvents that are CONNECTION_DOWN subevents.
setAllPersistentConnectionsDown()	Sets the qualifies method to return true to all MessagingEvents that are CONNECTION_DOWN subevents and are persistent connections.

ProvGroupAdapter Class

The ProvGroupAdapter class provides definitions of all methods in the ProvGroupListener interface. Table A-87 lists the ProvGroupAdapter methods.

Table A-87 ProvGroupAdapter Methods

Method	Description
getOneShot()	Gets the oneShot mode value that specifies whether the listener is registered for just one occurrence of the event.

Table A-87 ProvGroupAdapter Methods (continued)

Method	Description
modified()	Invoked when a ProvGroup event arrives and the event identifier is PROV_GROUP_MODIFIED.
setOneShot()	Sets the oneShot mode value that specifies a registration request for just one occurrence of the event.

QualifyAll Class

The QualifyAll class is an implementation of the Qualifier class. All listeners that register with this qualifier receive all events. Table A-88 lists the QualifyAll methods.

Table A-88 QualifyAll Methods

Method	Description
qualifies()	Returns true if the implementation has determined that this event satisfies the requirements to notify the associated listener.

SystemConfigAdapter Class

The SystemConfigAdapter class provides definitions of all methods in the SystemConfigListener interface. Table A-89 lists the SystemConfigAdapter methods.

Table A-89 SystemConfigAdapter Methods

Method	Description
getOneShot()	Gets the oneShot mode value that specifies whether the listener is registered for just one occurrence of the event.
serverDefaultsChanged()	Invoked when a SystemConfigEvent arrives after a change in server defaults.
serverPropertiesChanged()	Invoked when a SystemConfigEvent arrives after a change in server properties.
setOneShot()	Sets the oneShot mode value that specifies a registration request for just one occurrence of the event.
systemConfigChanged()	Invoked when a SystemConfigEvent arrives after a change in system configuration.
systemDefaultsChanged()	Invoked when a SystemConfigEvent arrives after a change in system defaults.

SystemConfigEventQualifier Class

This implementation of the SystemConfigEventQualifier class is used with SystemConfigEvents. Table A-90 lists the SystemConfigEventQualifier methods.

Table A-90 SystemConfigEventQualifier Methods

Method	Description
clearAll()	Clears all options that have been set.
qualifies(ProvAPIEvent event)	Returns true if the implementation has determined that this event satisfies the requirements to notify the associated listener.
setServerDefaultsChange()	Sets the qualifies method to return true to all SystemConfigEvents that are SERVER_DEFAULTS_CHANGE sub-events.
setServerPropertiesChange()	Sets the qualifies method to return true to all SystemConfigEvents that are SERVER_PROPERTIES_CHANGE sub-events.
setSystemConfigChange()	Sets the qualifies method to return true to all SystemConfigEvents that are SYSTEM_CONFIG_CHANGE sub-events.
setSystemDefaultsChange()	Sets the qualifies method to return true to all SystemConfigEvents that are SYSTEM_DEFAULTS_CHANGE sub-events.

ServerConfigEventType Class

The ServerConfigEventType class provides an enumeration of the SystemConfigEventTypes that can be signalled by a SystemConfigEvent of subclass SystemConfigListener.SERVER_DEFAULTS_CHANGE. Table A-91 lists the SystemConfigEventType methods.

Table A-91 ServerConfigEventType Methods

Method	Description
getNamed()	Returns the ServerConfigEventType object with the selected name (or null if it does not exist).
getValued()	Returns the ServerConfigEventType object with the selected value (or null if it does not exist).

Exceptions

The com.cisco.provisioning.cpe.events package implements this exception to handle processing of errors and other abnormal conditions:

- RegistrationFailedException—Thrown when registration of an event listener fails.