Patcher Javascript API

3.2.1.35

Generated by Doxygen 1.7.4

Fri Mar 16 2012 16:35:14

Contents

Chapter 1

Patcher Javascript API

1.1 Overview

The patcher javascript API allows users to integrate with objects implemented using the patcher API.

The usage of this API is restricted to entities which have signed a license agreement with Solid State Networks, Inc.

Licenses are valid for only one PRODUCT usage. Please see the license agreement for further details.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 Patch Class Reference

Extract a patch.

Public Member Functions

- void fail ()
- string getBytesLeft ()
- string getCurrentItemCompressedSize ()
- string getCurrentItemDiskFileName ()
- string getCurrentItemDiskOffset ()
- string getFileName ()
- string getOutputDirectory ()
- double getRemainingTime ()
- bool getStalled ()
- string getTotalBytes ()
- string getWriteRates ()
- void release ()
- void resume ()
- void setFileName (string value)
- void setOutputDirectory (string value)
- void skip ()
- void start ()

Public Attributes

- string instanceId
- event onComplete
- event onCorruptDestFile
- event onCorruptSourceFile

- event onDidPatchFile
- event onFileError
- event onPrepareComplete
- event onPreparePatchFile
- event onProgress
- event onStart
- event onWillPatchFile

4.1.1 Detailed Description

Extract a patch.

4.1.2 Member Function Documentation

4.1.2.1 void Patch.fail ( )

Fail the patch after an action has stalled

4.1.2.2 string Patch.getBytesLeft ( )

gets the bytes left in the patch (64-bit number)

Returns

bytes left in the patch (64-bit number)

4.1.2.3 string Patch.getCurrentItemCompressedSize ( )

gets the compressed size of the file in the zip

Returns

compressed size of the file in the zip

4.1.2.4 string Patch.getCurrentItemDiskFileName ( )

gets the the zip file currently being used to patched

Returns

the zip file currently being used to patched

4.1.2.5 string Patch.getCurrentItemDiskOffset ( )

gets the offset in the zip where the patch file exists

Returns

offset in the zip where the patch file exists

4.1.2.6 string Patch.getFileName ( )

gets the file name of path

Returns

file name of path

4.1.2.7 string Patch.getOutputDirectory ( )

gets the location where patch is applied

Returns

location where patch is applied

4.1.2.8 double Patch.getRemainingTime ( )

gets the estimated time remaining in a patch

Returns

estimated time remaining in a patch

4.1.2.9 bool Patch.getStalled ( )

gets the patch stalled state

Returns

patch stalled state

4.1.2.10 string Patch.getTotalBytes ( )

gets the size of the patch (64-bit number)

Returns

size of the patch (64-bit number)

4.1.2.11 string Patch.getWriteRates ( )

gets the json object containing the rate of the current/average/max throughput for patch writes

Returns

json object containing the rate of the current/average/max throughput for patch writes

4.1.2.12 void Patch.release ( )

releases the object

4.1.2.13 void Patch.resume ( )

Resume the patch file after an action has stalled

4.1.2.14 void Patch.setFileName (string *value*)

sets the file name of path

Parameters

value	file name of path

4.1.2.15 void Patch.setOutputDirectory (string *value*)

sets the location where patch is applied

Parameters

value	location where patch is applied

4.1.2.16 void Patch.skip ( )

Skip the patch file after an action has stalled

4.1.2.17 void Patch.start ( )

Starts the patch manifest

4.1.3 Member Data Documentation

4.1.3.1 string Patch.instanceId

Instance id used to bind proxy object to native object

4.1.3.2 event Patch.onComplete

Triggered when the patch is complete

Parameters

successful	true if the patch was successful, false otherwise

4.1.3.3 event Patch.onCorruptDestFile

Triggered when the patch has detected a corrupt dest file

Parameters

name	Name of the corrupt file
expected-SHA1	SHA1 the dest file is supposed to be
SHA1	SHA1 of the dest file

4.1.3.4 event Patch.onCorruptSourceFile

Triggered when the patch has detected a corrupt source file

Parameters

name	Name of the corrupt file
expected-SHA1	SHA1 the source file is supposed to be
SHA1	SHA1 of the source file

4.1.3.5 event Patch.onDidPatchFile

Triggered after a file is patched

Parameters

name	Name of the file to patched
successful	was the patch successfully applied

4.1.3.6 event Patch.onFileError

Triggered when the patch encounters a file error

Parameters

name	File where the error was generated
error	Number value representing file error

4.1.3.7 event Patch.onPrepareComplete

Triggered when the patch preperation is complete

Parameters

successful	true if the patch preperation was successful, false otherwise

4.1.3.8 event Patch.onPreparePatchFile

Triggered before patch starts on each file in the patch

Parameters

name	Name of the file to be patched

4.1.3.9 event Patch.onProgress

Triggered when the patch has made progress

Parameters

percent	(-1.0: Still calculating) (0.0 to 1.0: Percent completed)

4.1.3.10 event Patch.onStart

Triggered when patch is started

4.1.3.11 event Patch.onWillPatchFile

Triggered before a file is patched

Parameters

name	Name of the file to be patched

Returns

bool: true to patch file, false to skip

The documentation for this class was generated from the following file:

- patch.js

4.2 PatchManager Class Reference

Manages patches and provides an integrated event structure.

Public Member Functions

- void addManifestAddedEventHandler (event handler)
- void addManifestCompleteEventHandler (event handler)
- void addManifestDependencyEventHandler (event handler)
- void addManifestRemovedEventHandler (event handler)
- void addManifestRequiredCompleteEventHandler (event handler)
- void addManifestStartEventHandler (event handler)
- void addManifestStateChangeEventHandler (event handler)
- void addManifestVerifyCompleteEventHandler (event handler)
- void addManifestVerifyProgressEventHandler (event handler)
- void addManifestVerifyStartEventHandler (event handler)
- void addPatchCompleteEventHandler (event handler)
- void addPatchCorruptDestFileEventHandler (event handler)
- void addPatchCorruptSourceFileEventHandler (event handler)
- void addPatchDidPatchFileEventHandler (event handler)
- void addPatchDownloadErrorEventHandler (event handler)
- void addPatchFileErrorEventHandler (event handler)
- patchManifestItem addPatchManifest (string manifestUrl)
- void addPatchPrepareCompleteEventHandler (event handler)
- void addPatchPreparePatchFileEventHandler (event handler)
- void addPatchProgressEventHandler (event handler)
- void addPatchStartEventHandler (event handler)
- void addPatchWillPatchFileEventHandler (event handler)
- void manifestById (string id)
- void release ()
- void removeManifestAddedEventHandler (event handler)
- void removeManifestCompleteEventHandler (event handler)
- void removeManifestDependencyEventHandler (event handler)
- void removeManifestRemovedEventHandler (event handler)
- void removeManifestRequiredCompleteEventHandler (event handler)
- void removeManifestStartEventHandler (event handler)
- void removeManifestStateChangeEventHandler (event handler)

- void removeManifestVerifyCompleteEventHandler (event handler)
- void removeManifestVerifyProgressEventHandler (event handler)
- void removeManifestVerifyStartEventHandler (event handler)
- void removePatchCompleteEventHandler (event handler)
- void removePatchCorruptDestFileEventHandler (event handler)
- void removePatchCorruptSourceFileEventHandler (event handler)
- void removePatchDidPatchFileEventHandler (event handler)
- void removePatchDownloadErrorEventHandler (event handler)
- void removePatchFileErrorEventHandler (event handler)
- bool removePatchManifest (string id)
- void removePatchPrepareCompleteEventHandler (event handler)
- void removePatchPreparePatchFileEventHandler (event handler)
- void removePatchProgressEventHandler (event handler)
- void removePatchStartEventHandler (event handler)
- void removePatchWillPatchFileEventHandler (event handler)
- patchManifestItem repairPatchManifest (string manifestUrl, array maskedFiles)
- void rootManifestById (string id)
- void rootManifestIdById (string id)
- void verifyManifestById (string manifestId, bool quickScan)

4.2.1 Detailed Description

Manages patches and provides an integrated event structure.

4.2.2 Member Function Documentation

4.2.2.1 void PatchManager.addManifestAddedEventHandler (event *handler*)

subscribe to the ManifestAdded event

Parameters

handler	callback function for ManifestAdded event

4.2.2.2 void PatchManager.addManifestCompleteEventHandler (event *handler*)

subscribe to the ManifestComplete event

Parameters

handler	callback function for ManifestComplete event

4.2.2.3 void PatchManager.addManifestDependencyEventHandler (event *handler*)

subscribe to the ManifestDependency event

Parameters

handler	callback function for ManifestDependency event

4.2.2.4 void PatchManager.addManifestRemovedEventHandler (event *handler*)

subscribe to the ManifestRemoved event

Parameters

handler	callback function for ManifestRemoved event

4.2.2.5 void PatchManager.addManifestRequiredCompleteEventHandler (event *handler*)

subscribe to the ManifestRequiredComplete event

Parameters

handler	callback function for ManifestRequiredComplete event

4.2.2.6 void PatchManager.addManifestStartEventHandler (event *handler*)

subscribe to the ManifestStart event

Parameters

handler	callback function for ManifestStart event

4.2.2.7 void PatchManager.addManifestStateChangeEventHandler (event *handler*)

subscribe to the ManifestStateChange event

Parameters

handler	callback function for ManifestStateChange event

4.2.2.8 void PatchManager.addManifestVerifyCompleteEventHandler (event *handler*)

subscribe to the ManifestVerifyComplete event

Parameters

handler	callback function for ManifestVerifyComplete event

4.2.2.9 void PatchManager.addManifestVerifyProgressEventHandler (event *handler*)

subscribe to the ManifestVerifyProgress event

Parameters

handler	callback function for ManifestVerifyProgress event

4.2.2.10 void PatchManager.addManifestVerifyStartEventHandler (event *handler*)

subscribe to the ManifestVerifyStart event

Parameters

handler	callback function for ManifestVerifyStart event

4.2.2.11 void PatchManager.addPatchCompleteEventHandler (event *handler*)

subscribe to the PatchComplete event

Parameters

handler	callback function for PatchComplete event

4.2.2.12 void PatchManager.addPatchCorruptDestFileEventHandler (event *handler*)

subscribe to the PatchCorruptDestFile event

Parameters

handler	callback function for PatchCorruptDestFile event

4.2.2.13 void PatchManager.addPatchCorruptSourceFileEventHandler (event *handler*)

subscribe to the PatchCorruptSourceFile event

Parameters

handler	callback function for PatchCorruptSourceFile event

4.2.2.14 void PatchManager.addPatchDidPatchFileEventHandler (event *handler*)

subscribe to the PatchDidPatchFile event

Parameters

handler	callback function for PatchDidPatchFile event

4.2.2.15 void PatchManager.addPatchDownloadErrorEventHandler (event *handler*)

subscribe to the PatchDownloadError event

Parameters

handler	callback function for PatchDownloadError event

4.2.2.16 void PatchManager.addPatchFileErrorEventHandler (event *handler*)

subscribe to the PatchFileError event

Parameters

handler	callback function for PatchFileError event

4.2.2.17 patchManifestItem PatchManager.addPatchManifest (string *manifestUrl*)

adds patch to the manager

Parameters

manifestUrl	url to the manifest file

Returns

patchManifestItem containing the id and patch manifest

4.2.2.18 void PatchManager.addPatchPrepareCompleteEventHandler (event *handler*)

subscribe to the PatchPrepareComplete event

Parameters

handler	callback function for PatchPrepareComplete event

4.2.2.19 void PatchManager.addPatchPreparePatchFileEventHandler (event *handler*)

subscribe to the PatchPreparePatchFile event

Parameters

handler	callback function for PatchPreparePatchFile event

4.2.2.20 void PatchManager.addPatchProgressEventHandler (event *handler*)

subscribe to the PatchProgress event

Parameters

handler	callback function for PatchProgress event

4.2.2.21 void PatchManager.addPatchStartEventHandler (event *handler*)

subscribe to the PatchStart event

Parameters

handler	callback function for PatchStart event

4.2.2.22 void PatchManager.addPatchWillPatchFileEventHandler (event *handler*)

subscribe to the PatchWillPatchFile event

Parameters

handler	callback function for PatchWillPatchFile event

4.2.2.23 void PatchManager.manifestById (string *id*)

returns the patch manifest by the assigned id

Parameters

id	patch manifest id assigned by manager

4.2.2.24 void PatchManager.release ( )

releases the download manager

4.2.2.25 void PatchManager.removeManifestAddedEventHandler (event *handler*)

unsubscribe from the ManifestAdded event

Parameters

handler	callback function for ManifestAdded event

4.2.2.26 void PatchManager.removeManifestCompleteEventHandler (event *handler*)

unsubscribe from the ManifestComplete event

Parameters

handler	callback function for ManifestComplete event

4.2.2.27 void PatchManager.removeManifestDependencyEventHandler (event *handler*)

unsubscribe from the ManifestDependency event

Parameters

handler	callback function for ManifestDependency event

4.2.2.28 void PatchManager.removeManifestRemovedEventHandler (event *handler*)

unsubscribe from the ManifestRemoved event

Parameters

handler	callback function for ManifestRemoved event

4.2.2.29 void PatchManager.removeManifestRequiredCompleteEventHandler (event *handler*)

unsubscribe from the ManifestRequiredComplete event

Parameters

handler	callback function for ManifestRequiredComplete event

4.2.2.30 void PatchManager.removeManifestStartEventHandler (event *handler*)

unsubscribe from the ManifestStart event

Parameters

handler	callback function for ManifestStart event

4.2.2.31 void PatchManager.removeManifestStateChangeEventHandler (event *handler*)

unsubscribe from the ManifestStateChange event

Parameters

handler	callback function for ManifestStateChange event

4.2.2.32 void PatchManager.removeManifestVerifyCompleteEventHandler (event *handler*)

unsubscribe from the ManifestVerifyComplete event

Parameters

handler	callback function for ManifestVerifyComplete event

4.2.2.33 void PatchManager.removeManifestVerifyProgressEventHandler (event *handler*)

unsubscribe from the ManifestVerifyProgress event

Parameters

handler	callback function for ManifestVerifyProgress event

4.2.2.34 void PatchManager.removeManifestVerifyStartEventHandler (event *handler*)

unsubscribe from the ManifestVerifyStart event

Parameters

handler	callback function for ManifestVerifyStart event

4.2.2.35 void PatchManager.removePatchCompleteEventHandler (event *handler*)

unsubscribe from the PatchComplete event

Parameters

handler	callback function for PatchComplete event

4.2.2.36 void PatchManager.removePatchCorruptDestFileEventHandler (event *handler*)

unsubscribe from the PatchCorruptDestFile event

Parameters

handler	callback function for PatchCorruptDestFile event

4.2.2.37 void PatchManager.removePatchCorruptSourceFileEventHandler (event *handler*)

unsubscribe from the PatchCorruptSourceFile event

Parameters

handler	callback function for PatchCorruptSourceFile event

4.2.2.38 void PatchManager.removePatchDidPatchFileEventHandler (event *handler*)

unsubscribe from the PatchDidPatchFile event

Parameters

handler	callback function for PatchDidPatchFile event

4.2.2.39 void PatchManager.removePatchDownloadErrorEventHandler (event *handler*)

unsubscribe from the PatchDownloadError event

Parameters

handler	callback function for PatchDownloadError event

4.2.2.40 void PatchManager.removePatchFileErrorEventHandler (event *handler*)

unsubscribe from the PatchFileError event

Parameters

handler	callback function for PatchFileError event

4.2.2.41 bool PatchManager.removePatchManifest (string *id*)

removes a patch manifest from the manager

Parameters

id	id of the patch manifest to remove

Returns

true if the item was successfully removed, false otherwise

4.2.2.42 void PatchManager.removePatchPrepareCompleteEventHandler (event *handler*)

unsubscribe from the PatchPrepareComplete event

Parameters

handler	callback function for PatchPrepareComplete event

4.2.2.43 void PatchManager.removePatchPreparePatchFileEventHandler (event *handler*)

unsubscribe from the PatchPreparePatchFile event

Parameters

handler	callback function for PatchPreparePatchFile event

4.2.2.44 void PatchManager.removePatchProgressEventHandler (event *handler*)

unsubscribe from the PatchProgress event

Parameters

handler	callback function for PatchProgress event

4.2.2.45 void PatchManager.removePatchStartEventHandler (event *handler*)

unsubscribe from the PatchStart event

Parameters

handler	callback function for PatchStart event

4.2.2.46 void PatchManager.removePatchWillPatchFileEventHandler (event *handler*)

unsubscribe from the PatchWillPatchFile event

Parameters

handler	callback function for PatchWillPatchFile event

4.2.2.47 patchManifestItem PatchManager.repairPatchManifest (string *manifestUrl,* array *maskedFiles*)

add patch manifest repair to the manager

Parameters

manifestUrl	url to the manifest file
maskedFiles	files to mask for repair

Returns

patchManifestItem containing the id and patch manifest

4.2.2.48 void PatchManager.rootManifestById (string *id*)

returns the root patch manifest by the assigned id or a child id

Parameters

id	root patch manifest id assigned by manager

4.2.2.49 void PatchManager.rootManifestIdById (string *id*)

returns the root patch manifest id by the assigned id or a child id

Parameters

id	root patch manifest id assigned by manager

4.2.2.50 void PatchManager.verifyManifestById (string *manifestId,* bool *quickScan*)

verify manifest by id

Parameters

manifestId	id for the manifest
quickScan	enable quick scanning during verify

The documentation for this class was generated from the following file:

- patchmanager.js

4.3 PatchManifest Class Reference

Download/process a patch manifest.

Public Member Functions

- void dependencyFail ()
- string expandString (string expand)
- void fail ()
- int getCurrentReleaseId ()
- string getExtraDataValue (int fromId, int toId, string key)
- int getLastError ()
- bool getLocal ()
- bool getMaintenance ()
- string getName ()
- string getOutput ()

- string getReleaseName (int id)
- int getRequiredReleaseId ()
- bool getRequiresElevation ()
- int getRetryInterval ()
- int getSourceReleaseId ()
- string getSourceReleaseName ()
- string getSourceReleaseSHA1 ()
- string getTargetDirectory ()
- int getTargetReleaseId ()
- string getTargetReleaseName ()
- string getTargetReleaseSHA1 ()
- int getUpcomingReleaseId ()
- string getUrl ()
- void release ()
- void resume ()
- void setLocal (bool value)
- void setOutput (string value)
- void setRetryInterval (int value)
- void setSourceReleaseId (int value)
- void setSourceReleaseName (string value)
- void setSourceReleaseSHA1 (string value)
- void setTargetDirectory (string value)
- void setTargetReleaseId (int value)
- void setTargetReleaseName (string value)
- void setTargetReleaseSHA1 (string value)
- void setUrl (string value)
- void start ()

Public Attributes

- string instanceId
- event onComplete
- event onDependency
- event onLoadComplete
- event onPatch
- event onStart

4.3.1 Detailed Description

Download/process a patch manifest.

4.3.2 Member Function Documentation

4.3.2.1 void PatchManifest.dependencyFail ( )

Special failure state because one of the dependencies failed

4.3.2.2 string PatchManifest.expandString (string *expand*)

This method expands the string with the current macros

Parameters

expand	Name of string to be expanded

Returns

expanded string

4.3.2.3 void PatchManifest.fail ( )

Fail the patch manifest after an action is completed

4.3.2.4 int PatchManifest.getCurrentReleaseId ( )

gets the current release declared in the version file

Returns

current release declared in the version file

4.3.2.5 string PatchManifest.getExtraDataValue (int *fromId,* int *toId,* string *key*)

Gets extra data value from the specified update path

Parameters

fromId	Release id
toId	Release id
key	Key associated with value

Returns

Value of key stored in extra data for a release path (Empty string if not found)

4.3.2.6 int PatchManifest.getLastError ( )

gets the last error encountered by the manifest

Returns

last error encountered by the manifest

4.3.2.7 bool PatchManifest.getLocal ( )

gets the local state for the patch manifest

Returns

local state for the patch manifest

4.3.2.8 bool PatchManifest.getMaintenance ( )

gets the manifest maintenance mode

Returns

manifest maintenance mode

4.3.2.9 string PatchManifest.getName ( )

gets the name of the payload

Returns

name of the payload

4.3.2.10 string PatchManifest.getOutput ( )

gets the location for patch manifest

Returns

location for patch manifest

4.3.2.11 string PatchManifest.getReleaseName (int *id*)

Gets the name for the specified release

Parameters

id	Release id

Returns

Name of the specified release (Empty string if not found)

4.3.2.12 int PatchManifest.getRequiredReleaseId ( )

gets the required release declared in the patch manifest

Returns

required release declared in the patch manifest

4.3.2.13 bool PatchManifest.getRequiresElevation ( )

gets the manifest elevation requirement

Returns

manifest elevation requirement

4.3.2.14 int PatchManifest.getRetryInterval ( )

gets the interval in seconds after completion the patch manifest will restart

Returns

interval in seconds after completion the patch manifest will restart

4.3.2.15 int PatchManifest.getSourceReleaseId ( )

gets the id of the source release

Returns

id of the source release

4.3.2.16 string PatchManifest.getSourceReleaseName ( )

gets the name of the source release

Returns

name of the source release

4.3.2.17 string PatchManifest.getSourceReleaseSHA1 ( )

gets the sha1 of the source release

Returns

sha1 of the source release

4.3.2.18 string PatchManifest.getTargetDirectory ( )

gets the directory patch will be installed in

Returns

directory patch will be installed in

4.3.2.19 int PatchManifest.getTargetReleaseId ( )

gets the id of the target release

Returns

id of the target release

4.3.2.20 string PatchManifest.getTargetReleaseName ( )

gets the name of the target release

Returns

name of the target release

4.3.2.21 string PatchManifest.getTargetReleaseSHA1 ( )

gets the sha1 of the target release

Returns

sha1 of the target release

4.3.2.22 int PatchManifest.getUpcomingReleaseId ( )

gets the upcoming release declared in the patch manifest

Returns

upcoming release declared in the patch manifest

4.3.2.23 string PatchManifest.getUrl ( )

gets the url to metafile

Returns

url to metafile

4.3.2.24 void PatchManifest.release ( )

releases the object

4.3.2.25 void PatchManifest.resume ( )

Resume the patch manifest after an action is completed

4.3.2.26 void PatchManifest.setLocal (bool *value*)

sets the local state for the patch manifest

Parameters

value	local state for the patch manifest

4.3.2.27 void PatchManifest.setOutput (string *value*)

sets the location for patch manifest

Parameters

value	location for patch manifest

4.3.2.28 void PatchManifest.setRetryInterval (int *value*)

sets the interval in seconds after completion the patch manifest will restart

Parameters

value	interval in seconds after completion the patch manifest will restart

4.3.2.29 void PatchManifest.setSourceReleaseId (int *value*)

sets the id of the source release

Parameters

value	id of the source release

4.3.2.30 void PatchManifest.setSourceReleaseName (string *value*)

sets the name of the source release

Parameters

value	name of the source release

4.3.2.31 void PatchManifest.setSourceReleaseSHA1 (string *value*)

sets the sha1 of the source release

Parameters

value	sha1 of the source release

4.3.2.32 void PatchManifest.setTargetDirectory (string *value*)

sets the directory patch will be installed in

Parameters

value	directory patch will be installed in

4.3.2.33 void PatchManifest.setTargetReleaseId (int *value*)

sets the id of the target release

Parameters

value	id of the target release

4.3.2.34 void PatchManifest.setTargetReleaseName (string *value*)

sets the name of the target release

Parameters

value	name of the target release

4.3.2.35 void PatchManifest.setTargetReleaseSHA1 (string *value*)

sets the sha1 of the target release

Parameters

value	sha1 of the target release

4.3.2.36 void PatchManifest.setUrl (string *value*)

sets the url to metafile

Parameters

value	url to metafile

4.3.2.37 void PatchManifest.start ( )

Start processing the patch manifest

4.3.3 Member Data Documentation

4.3.3.1 string PatchManifest.instanceId

Instance id used to bind proxy object to native object

4.3.3.2 event PatchManifest.onComplete

Triggered when the manifest is complete

Parameters

successful	true if the manifest was successful, false otherwise

4.3.3.3 event PatchManifest.onDependency

Triggered for each dependency in the patch manifest

Parameters

url	Url of the dependency

4.3.3.4 event PatchManifest.onLoadComplete

Triggered when a patch manifest has loaded

4.3.3.5 event PatchManifest.onPatch

Triggered for each patch required

Parameters

fromId	Release id for the current patch
toId	Release id for the patch destination

4.3.3.6 event PatchManifest.onStart

Triggered when a manifest is started

The documentation for this class was generated from the following file:

- patchmanifest.js

4.4 PatchManifestError Class Reference

Patch manifest error constants and helper functions.

Public Member Functions

- string nameFromId (int id)

Public Attributes

- int DEPENDENCYFAIL
- int DOWNLOADFAIL
- int INVALIDARGUMENT
- int MAINTENANCE
- int NONE
- int REQUIRESELEVATION
- int SOURCERELEASEUNDEFINED
- int TARGETRELEASEUNDEFINED
- int UNZIPFAILED
- int UPDATEPATHUNAVAILABLE
- int USERFAIL
- int VERSIONFAIL
- int XMLPARSEFAIL

4.4.1 Detailed Description

Patch manifest error constants and helper functions.

4.4.2 Member Function Documentation

4.4.2.1 string PatchManifestError.nameFromId (int *id*)

converts a patch manifest error to a string

Parameters

id	patch manifest error

Returns

stringified name of file error.

4.4.3 Member Data Documentation

4.4.3.1 int PatchManifestError.DEPENDENCYFAIL

Dependency failed

4.4.3.2 int PatchManifestError.DOWNLOADFAIL

Download failed

4.4.3.3 int PatchManifestError.INVALIDARGUMENT

Invalid argument

4.4.3.4 int PatchManifestError.MAINTENANCE

Maintenance flag turned on

4.4.3.5 int PatchManifestError.NONE

No error

4.4.3.6 int PatchManifestError.REQUIRESELEVATION

Elevation required

4.4.3.7 int PatchManifestError.SOURCERELEASEUNDEFINED

Source release undefined

4.4.3.8 int PatchManifestError.TARGETRELEASEUNDEFINED

Target release undefined

4.4.3.9 int PatchManifestError.UNZIPFAILED

Unzip failed

4.4.3.10 int PatchManifestError.UPDATEPATHUNAVAILABLE

Update path unavailable

4.4.3.11 int PatchManifestError.USERFAIL

Fail method was called

4.4.3.12 int PatchManifestError.VERSIONFAIL

Version check failed

4.4.3.13 int PatchManifestError.XMLPARSEFAIL

Parsing failed

The documentation for this class was generated from the following file:

- patchmanifesterror.js

4.5 PatchManifestState Class Reference

Patch manifest state constants and helper functions.

Public Member Functions

- string nameFromId (int id)

Public Attributes

- int COMPLETE
- int DEPENDENCY
- int DOWNLOAD
- int GETMANIFEST
- int PATCH

4.5.1 Detailed Description

Patch manifest state constants and helper functions.

4.5.2 Member Function Documentation

4.5.2.1 string PatchManifestState.nameFromId (int *id*)

converts a patch manifest state to a string

Parameters

id	patch manifest state

Returns

stringified name of file error.

4.5.3 Member Data Documentation

4.5.3.1 int PatchManifestState.COMPLETE

Patch Complete

4.5.3.2 int PatchManifestState.DEPENDENCY

Patch dependency required

4.5.3.3 int PatchManifestState.DOWNLOAD

Patch downloading

4.5.3.4 int PatchManifestState.GETMANIFEST

Get manifest state

4.5.3.5 int PatchManifestState.PATCH

Applying patch

The documentation for this class was generated from the following file:

- patchmanifeststate.js

4.6 PatchVersion Class Reference

Enumerate a version file.

Public Member Functions

- string getFileName ()
- string getSha1 ()
- void release ()
- void resume ()
- void setFileName (string value)
- void start ()

Public Attributes

- string instanceId
- event onComplete
- event onFile
- event onStart

4.6.1 Detailed Description

Enumerate a version file.

4.6.2 Member Function Documentation

4.6.2.1 string PatchVersion.getFileName ( )

gets the file name of the version file

Returns

file name of the version file

4.6.2.2 string PatchVersion.getSha1 ( )

gets the sHA1 of the version file

Returns

SHA1 of the version file

4.6.2.3 void PatchVersion.release ( )

releases the object

4.6.2.4 void PatchVersion.resume ( )

Resume the file enumeration

4.6.2.5 void PatchVersion.setFileName (string *value*)

sets the file name of the version file

Parameters

value	file name of the version file

4.6.2.6 void PatchVersion.start ( )

Starts the patch version enumeration

4.6.3 Member Data Documentation

4.6.3.1 string PatchVersion.instanceId

Instance id used to bind proxy object to native object

4.6.3.2 event PatchVersion.onComplete

Triggered when the version file enumeration is complete

Parameters

successful	true if the download was successful, false otherwise

4.6.3.3 event PatchVersion.onFile

Triggered when the version file enumeration has detected a file

Parameters

name	Name of the file
lastModified	Last modified date of the file
size	Size of the file
sha1	SHA1 of the file

4.6.3.4 event PatchVersion.onStart

Triggered when version file enumeration is started

The documentation for this class was generated from the following file:

- patchversion.js

4.7 PatchVersionVerify Class Reference

Verify a version file.

Public Member Functions

- string getFileName ()
- bool getQuickScan ()
- void release ()
- void setFileName (string value)
- void setQuickScan (bool value)
- void start ()

Public Attributes

- string instanceId
- event onComplete
- event onCorruptFile
- event onProgress
- event onStart

4.7.1 Detailed Description

Verify a version file.

4.7.2 Member Function Documentation

4.7.2.1 string PatchVersionVerify.getFileName ( )

gets the file name of the version file

Returns

file name of the version file

4.7.2.2 bool PatchVersionVerify.getQuickScan ( )

gets the quickScan state (enables a quicker but less safe verification)

Returns

quickScan state (enables a quicker but less safe verification)

4.7.2.3 void PatchVersionVerify.release ( )

releases the object

4.7.2.4 void PatchVersionVerify.setFileName (string *value*)

sets the file name of the version file

Parameters

value	file name of the version file

4.7.2.5 void PatchVersionVerify.setQuickScan (bool *value*)

sets the quickScan state (enables a quicker but less safe verification)

Parameters

value	quickScan state (enables a quicker but less safe verification)

4.7.2.6 void PatchVersionVerify.start ( )

Starts the patch manifest

4.7.3 Member Data Documentation

4.7.3.1 string PatchVersionVerify.instanceId

Instance id used to bind proxy object to native object

4.7.3.2 event PatchVersionVerify.onComplete

Triggered when the verification is complete

Parameters

successful	true if the download was successful, false otherwise

4.7.3.3 event PatchVersionVerify.onCorruptFile

Triggered when the verification has detected a corrupt file

Parameters

name	Name of the corrupt file

4.7.3.4 event PatchVersionVerify.onProgress

Triggered when the verification has made progress

Parameters

percent	(-1.0: Still calculating) (0.0 to 1.0: Percent completed)

4.7.3.5 event PatchVersionVerify.onStart

Triggered when verification is started

The documentation for this class was generated from the following file:

- patchversionverify.js

Chapter 5

File Documentation

5.1 patch.js File Reference

File containing Patch class and creation function.

Classes

- class Patch

  Extract a patch.

Functions

- void createPatch ()

5.1.1 Detailed Description

File containing Patch class and creation function.

5.1.2 Function Documentation

5.1.2.1 void createPatch ( )

Create instance of patch

5.2 patchmanager.js File Reference

File containing patch manager.

Classes

- class PatchManager

  Manages patches and provides an integrated event structure.

Variables

- PatchManager patchManager

5.2.1 Detailed Description

File containing patch manager.

5.2.2 Variable Documentation

5.2.2.1 PatchManager patchManager

precreated global instance of the patch manager

5.3 patchmanifest.js File Reference

File containing PatchManifest class and creation function.

Classes

- class PatchManifest

  Download/process a patch manifest.

Functions

- void createPatchManifest ()

5.3.1 Detailed Description

File containing PatchManifest class and creation function.

5.3.2 Function Documentation

5.3.2.1 void createPatchManifest ( )

Create instance of patchManifest

5.4 patchmanifesterror.js File Reference

File containing patch manifest error constants and helper functions.

Classes

- class PatchManifestError

  Patch manifest error constants and helper functions.

Variables

- PatchManifestError patchManifestError

5.4.1 Detailed Description

File containing patch manifest error constants and helper functions.

5.4.2 Variable Documentation

5.4.2.1 PatchManifestError patchManifestError

precreated global instance of PatchManifestError

5.5 patchmanifeststate.js File Reference

File containing patch manifest state constants and helper functions.

Classes

- class PatchManifestState

  Patch manifest state constants and helper functions.

Variables

- PatchManifestState patchManifestState

5.5.1 Detailed Description

File containing patch manifest state constants and helper functions.

5.5.2 Variable Documentation

5.5.2.1 PatchManifestState patchManifestState

precreated global instance of PatchManifestState

5.6 patchversion.js File Reference

File containing PatchVersion class and creation function.

Classes

- class PatchVersion

  Enumerate a version file.

Functions

- void createPatchVersion ()

5.6.1 Detailed Description

File containing PatchVersion class and creation function.

5.6.2 Function Documentation

5.6.2.1 void createPatchVersion ( )

Create instance of patchVersion

5.7 patchversionverify.js File Reference

File containing PatchVersionVerify class and creation function.

Classes

- class PatchVersionVerify

  Verify a version file.

Functions

- void createPatchVersionVerify ()

5.7.1 Detailed Description

File containing PatchVersionVerify class and creation function.

5.7.2 Function Documentation

5.7.2.1 void createPatchVersionVerify ( )

Create instance of patchVersionVerify

Index