# Downloader Javascript API

3.1.2.0

Generated by Doxygen 1.7.4

Tue Oct 4 2011 22:46:22

# Contents

# Chapter 1

# Downloader Javascript API

## 1.1 Overview

The downloader javascript API allows users to integrate with objects implemented using the download 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 Download Class Reference

Download a file system.

### Public Member Functions

- bool checkFile (string fileName)
- bool checkFileRange (string fileName, string offset, string length)
- void eraseContent ()
- string expandString (string expand)
- bool getAccessible ()
- bool getActive ()
- bool getBitActive ()
- int getBitDownloadLimit ()
- int getBitUploadLimit ()
- string getBytesLeft ()
- string getConfigUrl ()
- string getConnectionsCount ()
- string getConnectivity ()
- int getDownloadLimit ()
- bool getIsRunning ()
- int getLastError ()
- int getLastFailure ()
- bool getMaskEnabled ()
- string getMetafileUrl ()
- string getOutput ()
- double getRemainingTime ()
- int getState ()
- string getTotalBytes ()
- string getTransferRates ()

- string getTransferTotal ()
- int getUploadLimit ()
- bool getWebActive ()
- int getWebDownloadLimit ()
- int getWebUploadLimit ()
- bool maskFile (string fileName)
- bool maskFileRange (string fileName, string offset, string length)
- void release ()
- void setActive (bool value)
- void setBitActive (bool value)
- void setBitDownloadLimit (int value)
- void setBitUploadLimit (int value)
- void setConfigUrl (string value)
- void setDownloadLimit (int value)
- void setMaskEnabled (bool value)
- void setMetafileUrl (string value)
- void setOutput (string value)
- void setUploadLimit (int value)
- void setWebActive (bool value)
- void setWebDownloadLimit (int value)
- void setWebUploadLimit (int value)
- void start ()
- void stop ()
- void syncToMask ()

### Public Attributes

- string instanceId
- event onComplete
- event onConfigComplete
- event onProgress
- event onStart
- event onStateChange

### 4.1.1 Detailed Description

Download a file system.

### 4.1.2 Member Function Documentation

#### 4.1.2.1 bool Download.checkFile ( string *fileName* )

Check download by filename for complete.

**Parameters**

| | |
|---|---|
| *fileName* | filename in metafile to mask |

**Returns**

true if complete, false otherwise.

#### 4.1.2.2 bool Download.checkFileRange ( string *fileName,* string *offset,* string *length* )

Check portion of download by filename and range for complete.

**Parameters**

| | |
|---|---|
| *fileName* | filename in metafile to mask |
| *offset* | offset in the file to start at (64bit int) |
| *length* | length of the bytes to include (64bit int) |

**Returns**

true if complete, false otherwise.

#### 4.1.2.3 void Download.eraseContent ( )

Removes the download from the disk when the download stops.

#### 4.1.2.4 string Download.expandString ( string *expand* )

This method expands the string with the current macros.

**Parameters**

| | |
|---|---|
| *expand* | Name of string to be expanded. |

**Returns**

expanded string.

#### 4.1.2.5 bool Download.getAccessible ( )

gets the download accessibility and configuration has completed.

**Returns**

download accessibility and configuration has completed.

#### 4.1.2.6 bool Download.getActive ( )

gets the active state of the download. (only available after accessible is true).

**Returns**

active state of the download. (only available after accessible is true).

#### 4.1.2.7 bool Download.getBitActive ( )

gets the active state of bittorrent peers in the download. (only available after accessible is true).

**Returns**

active state of bittorrent peers in the download. (only available after accessible is true).

#### 4.1.2.8 int Download.getBitDownloadLimit ( )

gets the bittorrent peer download limit for the download. (In kBps) (only available after accessible is true).

**Returns**

bittorrent peer download limit for the download. (In kBps) (only available after accessible is true).

#### 4.1.2.9 int Download.getBitUploadLimit ( )

gets the bittorrent peer upload limit for the download. (In kBps) (only available after accessible is true).

**Returns**

bittorrent peer upload limit for the download. (In kBps) (only available after accessible is true).

#### 4.1.2.10 string Download.getBytesLeft ( )

gets the bytes left in the download. (64-bit number) (only available after accessible is true).

**Returns**

bytes left in the download. (64-bit number) (only available after accessible is true).

#### 4.1.2.11 string Download.getConfigUrl ( )

gets the url to configuration file.

**Returns**

Url to configuration file.

#### 4.1.2.12 string Download.getConnectionsCount ( )

gets the json object containing the number of active/total incoming/outgoing connections. (only available after accessible is true).

**Returns**

json object containing the number of active/total incoming/outgoing connections. (only available after accessible is true).

#### 4.1.2.13 string Download.getConnectivity ( )

gets the download connectivity. (only available after accessible is true).

**Returns**

download connectivity. (only available after accessible is true).

#### 4.1.2.14 int Download.getDownloadLimit ( )

gets the download limit for the download. (In kBps) (only available after accessible is true).

**Returns**

download limit for the download. (In kBps) (only available after accessible is true).

#### 4.1.2.15 bool Download.getIsRunning ( )

gets the is the download currently running.

**Returns**

is the download currently running.

#### 4.1.2.16 int Download.getLastError ( )

gets the last error encountered by the downloader.

**Returns**

last error encountered by the downloader.

#### 4.1.2.17 int Download.getLastFailure ( )

gets the last failure encountered by the downloader.

**Returns**

last failure encountered by the downloader.

#### 4.1.2.18 bool Download.getMaskEnabled ( )

gets the enable/disable download mask.

**Returns**

Enable/disable download mask.

#### 4.1.2.19 string Download.getMetafileUrl ( )

gets the url to metafile.

**Returns**

Url to metafile.

#### 4.1.2.20 string Download.getOutput ( )

gets the location for downloaded data.

**Returns**

location for downloaded data.

#### 4.1.2.21 double Download.getRemainingTime ( )

gets the estimated time remaining in a download. (only available after accessible is true).

**Returns**

estimated time remaining in a download. (only available after accessible is true).

#### 4.1.2.22 int Download.getState ( )

gets the download state. (only available after accessible is true).

**Returns**

download state. (only available after accessible is true).

#### 4.1.2.23 string Download.getTotalBytes ( )

gets the size of the download. (64-bit number) (only available after accessible is true).

**Returns**

size of the download. (64-bit number) (only available after accessible is true).

#### 4.1.2.24 string Download.getTransferRates ( )

gets the json object containing the rate of the current/average/max throughput for incoming/outgoing connections. (only available after accessible is true).

**Returns**

json object containing the rate of the current/average/max throughput for incoming/outgoing connections. (only available after accessible is true).

#### 4.1.2.25 string Download.getTransferTotal ( )

gets the json object with the number of bytes read/sent/confirmed/corrupt/discarded. (only available after accessible is true).

**Returns**

json object with the number of bytes read/sent/confirmed/corrupt/discarded. (only available after accessible is true).

#### 4.1.2.26 int Download.getUploadLimit ( )

gets the upload limit for the download. (In kBps) (only available after accessible is true).

**Returns**

upload limit for the download. (In kBps) (only available after accessible is true).

#### 4.1.2.27 bool Download.getWebActive ( )

gets the active state of reliable sources in the download. (only available after accessible is true).

**Returns**

active state of reliable sources in the download. (only available after accessible is true).

#### 4.1.2.28 int Download.getWebDownloadLimit ( )

gets the reliable source download limit for the download. (In kBps) (only available after accessible is true).

**Returns**

reliable source download limit for the download. (In kBps) (only available after accessible is true).

#### 4.1.2.29 int Download.getWebUploadLimit ( )

gets the reliable source upload limit for the download. (In kBps) (only available after accessible is true).

**Returns**

reliable source upload limit for the download. (In kBps) (only available after accessible is true).

#### 4.1.2.30 bool Download.maskFile ( string *fileName* )

Mask portion of download by filename.

**Parameters**

| | |
|---|---|
| *fileName* | filename in metafile to mask |

**Returns**

true if successful, false otherwise.

#### 4.1.2.31 bool Download.maskFileRange ( string *fileName,* string *offset,* string *length* )

Mask portion of download by filename and range.

**Parameters**

| | |
|---|---|
| *fileName* | filename in metafile to mask |
| *offset* | offset in the file to start at (64bit int) |
| *length* | length of the bytes to include (64bit int) |

**Returns**

true if successful, false otherwise.

#### 4.1.2.32 void Download.release ( )

releases the object

#### 4.1.2.33 void Download.setActive ( bool *value* )

sets the active state of the download. (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | active state of the download. (only available after accessible is true). |

#### 4.1.2.34 void Download.setBitActive ( bool *value* )

sets the active state of bittorrent peers in the download. (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | active state of bittorrent peers in the download. (only available after accessible is true). |

#### 4.1.2.35 void Download.setBitDownloadLimit ( int *value* )

sets the bittorrent peer download limit for the download. (In kBps) (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | bittorrent peer download limit for the download. (In kBps) (only available after accessible is true). |

#### 4.1.2.36 void Download.setBitUploadLimit ( int *value* )

sets the bittorrent peer upload limit for the download. (In kBps) (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | bittorrent peer upload limit for the download. (In kBps) (only available after accessible is true). |

#### 4.1.2.37 void Download.setConfigUrl ( string *value* )

sets the url to configuration file.

**Parameters**

| | |
|---|---|
| *value* | Url to configuration file. |

#### 4.1.2.38 void Download.setDownloadLimit ( int *value* )

sets the download limit for the download. (In kBps) (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | download limit for the download. (In kBps) (only available after accessible is true). |

#### 4.1.2.39 void Download.setMaskEnabled ( bool *value* )

sets the enable/disable download mask.

**Parameters**

| | |
|---|---|
| *value* | Enable/disable download mask. |

#### 4.1.2.40 void Download.setMetafileUrl ( string *value* )

sets the url to metafile.

**Parameters**

| | |
|---|---|
| *value* | Url to metafile. |

#### 4.1.2.41 void Download.setOutput ( string *value* )

sets the location for downloaded data.

**Parameters**

| | |
|---|---|
| *value* | location for downloaded data. |

#### 4.1.2.42 void Download.setUploadLimit ( int *value* )

sets the upload limit for the download. (In kBps) (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | upload limit for the download. (In kBps) (only available after accessible is true). |

#### 4.1.2.43 void Download.setWebActive ( bool *value* )

sets the active state of reliable sources in the download. (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | active state of reliable sources in the download. (only available after accessible is true). |

#### 4.1.2.44 void Download.setWebDownloadLimit ( int *value* )

sets the reliable source download limit for the download. (In kBps) (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | reliable source download limit for the download. (In kBps) (only available after accessible is true). |

#### 4.1.2.45 void Download.setWebUploadLimit ( int *value* )

sets the reliable source upload limit for the download. (In kBps) (only available after accessible is true).

**Parameters**

| | |
|---|---|
| *value* | reliable source upload limit for the download. (In kBps) (only available after accessible is true). |

#### 4.1.2.46 void Download.start ( )

Starts the download.

#### 4.1.2.47 void Download.stop ( )

Stops the download.

#### 4.1.2.48 void Download.syncToMask ( )

Synchronizes the downloaded content pieces.

### 4.1.3 Member Data Documentation

#### 4.1.3.1 string Download.instanceId

Instance id used to bind proxy object to native object

#### 4.1.3.2 event Download.onComplete

Triggered when the download is complete.

**Parameters**

| | |
|---|---|
| *successful* | true if the download was successful, false otherwise. |

#### 4.1.3.3 event Download.onConfigComplete

Triggered when download is configured, right before the actual download starts.

#### 4.1.3.4 event Download.onProgress

Triggered when the download has made progress.

**Parameters**

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

#### 4.1.3.5 event Download.onStart

Triggered when download is started.

#### 4.1.3.6 event Download.onStateChange

Triggered when the download makes a state change.

**Parameters**

| | |
|---|---|
| *state* | current state of the download. |

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

- download.js

## 4.2 Downloader Class Reference

Download management class.

### Public Member Functions

- bool getActive ()
- int getDownloadLimit ()
- int getUploadLimit ()
- void release ()
- void setActive (bool value)
- void setDownloadLimit (int value)
- void setUploadLimit (int value)

### Public Attributes

- string instanceId

### 4.2.1 Detailed Description

Download management class.

### 4.2.2 Member Function Documentation

#### 4.2.2.1 bool Downloader.getActive ( )

gets the active state of all downloads.

**Returns**

active state of all downloads.

#### 4.2.2.2 int Downloader.getDownloadLimit ( )

gets the download limit for all downloads. (In kBps)

**Returns**

download limit for all downloads. (In kBps)

#### 4.2.2.3 int Downloader.getUploadLimit ( )

gets the upload limit for all downloads. (In kBps)

**Returns**

upload limit for all downloads. (In kBps)

#### 4.2.2.4 void Downloader.release ( )

releases the object

#### 4.2.2.5 void Downloader.setActive ( bool *value* )

sets the active state of all downloads.

**Parameters**

| | |
|---|---|
| *value* | active state of all downloads. |

#### 4.2.2.6 void Downloader.setDownloadLimit ( int *value* )

sets the download limit for all downloads. (In kBps)

**Parameters**

| | |
|---|---|
| *value* | download limit for all downloads. (In kBps) |

#### 4.2.2.7 void Downloader.setUploadLimit ( int *value* )

sets the upload limit for all downloads. (In kBps)

**Parameters**

| | |
|---|---|
| *value* | upload limit for all downloads. (In kBps) |

### 4.2.3 Member Data Documentation

#### 4.2.3.1 string Downloader.instanceId

Instance id used to bind proxy object to native object

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

- downloader.js

## 4.3 DownloadError Class Reference

Download error constants and helper functions.

### Public Member Functions

- string nameFromId (string id)

### Public Attributes

- int ALREADYDOWNLOADING
- int CONFIGDOWNLOADFAILED
- int CONFIGPARSEFAILED
- int CONFIGUNZIPFAILED
- int DOWNLOADFAILED
- int INVALIDMETAFILE
- int INVALIDSOURCE
- int METAFILEDOWNLOADFAILED
- int METAFILEUNZIPFAILED
- int NONE
- int NOTENOUGHSPACE
- int UNACCEPTABLESOURCE
- int UNKNOWN

### 4.3.1 Detailed Description

Download error constants and helper functions.

### 4.3.2 Member Function Documentation

#### 4.3.2.1 string DownloadError.nameFromId ( string *id* )

converts a download error to a string

**Parameters**

| | |
|---|---|
| *id* | download errors |

**Returns**

stringified name of download error

### 4.3.3 Member Data Documentation

#### 4.3.3.1 int DownloadError.ALREADYDOWNLOADING

Download is already been started

#### 4.3.3.2 int DownloadError.CONFIGDOWNLOADFAILED

Unable to download config file

#### 4.3.3.3 int DownloadError.CONFIGPARSEFAILED

Unable to parse solid configuration file

#### 4.3.3.4 int DownloadError.CONFIGUNZIPFAILED

Unable to unzip config file

#### 4.3.3.5 int DownloadError.DOWNLOADFAILED

Download failed for an unknown reason

#### 4.3.3.6 int DownloadError.INVALIDMETAFILE

Invalid metafile

#### 4.3.3.7 int DownloadError.INVALIDSOURCE

Source is invalid

#### 4.3.3.8 int DownloadError.METAFILEDOWNLOADFAILED

Unable to download metafile file

#### 4.3.3.9 int DownloadError.METAFILEUNZIPFAILED

Unable to unzip metafile file

#### 4.3.3.10 int DownloadError.NONE

No error

#### 4.3.3.11 int DownloadError.NOTENOUGHSPACE

Output path does not have enough disk space

#### 4.3.3.12 int DownloadError.UNACCEPTABLESOURCE

Source url is unauthorized

#### 4.3.3.13 int DownloadError.UNKNOWN

Unknown error

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

- downloaderror.js

## 4.4 DownloadFailure Class Reference

Download failure constants and helper functions.

### Public Member Functions

- string nameFromId (string id)

### Public Attributes

- int CONTENTWRITE
- int NONE
- int PEERLISTEMPTY
- int PEERNOGOODONES
- int RELIABLESOURCEBADINTEGRITY
- int RELIABLESOURCECANTCONTACT
- int RELIABLESOURCENOTSPECIFIED
- int TRACKERCANTCONTACT
- int TRACKERLISTEMPTY
- int TRANSPORTHTTP
- int TRANSPORTHTTPCLIENT
- int TRANSPORTSOCKET

### 4.4.1 Detailed Description

Download failure constants and helper functions.

### 4.4.2 Member Function Documentation

#### 4.4.2.1 string DownloadFailure.nameFromId ( string *id* )

converts a download failure to a string

**Parameters**

| | |
|---|---|
| *id* | download failures |

**Returns**

stringified name of download failure

### 4.4.3 Member Data Documentation

#### 4.4.3.1 int DownloadFailure.CONTENTWRITE

Content write failed

#### 4.4.3.2 int DownloadFailure.NONE

No error

#### 4.4.3.3 int DownloadFailure.PEERLISTEMPTY

Peer list empty

#### 4.4.3.4 int DownloadFailure.PEERNOGOODONES

No good peers found

#### 4.4.3.5 int DownloadFailure.RELIABLESOURCEBADINTEGRITY

Reliable source integrity error

#### 4.4.3.6 int DownloadFailure.RELIABLESOURCECANTCONTACT

Unable to contact reliable source

#### 4.4.3.7 int DownloadFailure.RELIABLESOURCENOTSPECIFIED

Reliable source not specified

#### 4.4.3.8 int DownloadFailure.TRACKERCANTCONTACT

Unable to contact tracker

#### 4.4.3.9 int DownloadFailure.TRACKERLISTEMPTY

Tracker list empty

#### 4.4.3.10 int DownloadFailure.TRANSPORTHTTP

HTTP transport failed

#### 4.4.3.11 int DownloadFailure.TRANSPORTHTTPCLIENT

HTTP client transport failed

#### 4.4.3.12 int DownloadFailure.TRANSPORTSOCKET

Socket transport failed

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

- downloadfailure.js

## 4.5 DownloadManager Class Reference

Manages downloads and provides an integrated event structure.

### Public Member Functions

- downloadItem addDownload (string configUrl, string metafileUrl, string outputDirectory)
- void addDownloadAddedEventHandler (event handler)
- void addDownloadCompleteEventHandler (event handler)
- void addDownloadConfigCompleteEventHandler (event handler)
- void addDownloadProgressEventHandler (event handler)
- void addDownloadRemovedEventHandler (event handler)
- void addDownloadStartEventHandler (event handler)
- void addDownloadStateChangeEventHandler (event handler)
- void downloadById (string id)
- bool hasActiveDownload ()
- void init ()
- void release ()

- bool removeDownload (string id)
- void removeDownloadAddedEventHandler (event handler)
- void removeDownloadCompleteEventHandler (event handler)
- void removeDownloadConfigCompleteEventHandler (event handler)
- void removeDownloadProgressEventHandler (event handler)
- void removeDownloadRemovedEventHandler (event handler)
- void removeDownloadStartEventHandler (event handler)
- void removeDownloadStateChangeEventHandler (event handler)
- void setAllDownloadsActive (bool active)

### 4.5.1 Detailed Description

Manages downloads and provides an integrated event structure.

### 4.5.2 Member Function Documentation

#### 4.5.2.1 downloadItem DownloadManager.addDownload ( string *configUrl*, string *metafileUrl*, string *outputDirectory* )

adds download to the manager

**Parameters**

| | |
|---|---|
| *configUrl* | url to the configuration file |
| *metafileUrl* | url to the metafile |
| *outputDirectory* | directory where the download is saved |

**Returns**

downloadItem containing the id and download

#### 4.5.2.2 void DownloadManager.addDownloadAddedEventHandler ( event *handler* )

subscribe to the DownloadAdded event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadAdded event |

#### 4.5.2.3 void DownloadManager.addDownloadCompleteEventHandler ( event *handler* )

subscribe to the DownloadComplete event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadComplete event |

#### 4.5.2.4 void DownloadManager.addDownloadConfigCompleteEventHandler ( event *handler* )

subscribe to the DownloadConfigComplete event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadConfigComplete event |

#### 4.5.2.5 void DownloadManager.addDownloadProgressEventHandler ( event *handler* )

subscribe to the DownloadProgress event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadProgress event |

#### 4.5.2.6 void DownloadManager.addDownloadRemovedEventHandler ( event *handler* )

subscribe to the DownloadRemoved event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadRemoved event |

#### 4.5.2.7 void DownloadManager.addDownloadStartEventHandler ( event *handler* )

subscribe to the DownloadStart event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadStart event |

#### 4.5.2.8 void DownloadManager.addDownloadStateChangeEventHandler ( event *handler* )

subscribe to the DownloadStateChange event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadStateChange event |

#### 4.5.2.9 void DownloadManager.downloadById ( string *id* )

returns the download by the assigned id

**Parameters**

| | |
|---|---|
| *id* | download id assigned by manager |

#### 4.5.2.10 bool DownloadManager.hasActiveDownload ( )

checks if any of the downloads in the manager are currently active

**Returns**

true if a download is active, false otherwise

#### 4.5.2.11 void DownloadManager.init ( )

initializes the download manager

#### 4.5.2.12 void DownloadManager.release ( )

releases the download manager

#### 4.5.2.13 bool DownloadManager.removeDownload ( string *id* )

removes a download from the manager

**Parameters**

| | |
|---|---|
| *id* | id of the download to remove |

**Returns**

true if the item was successfully removed, false otherwise

#### 4.5.2.14 void DownloadManager.removeDownloadAddedEventHandler ( event *handler* )

unsubscribe from the DownloadAdded event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadAdded event |

#### 4.5.2.15 void DownloadManager.removeDownloadCompleteEventHandler ( event *handler* )

unsubscribe from the DownloadComplete event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadComplete event |

#### 4.5.2.16 void DownloadManager.removeDownloadConfigCompleteEventHandler ( event *handler* )

unsubscribe from the DownloadConfigComplete event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadAdded event |

#### 4.5.2.17 void DownloadManager.removeDownloadProgressEventHandler ( event *handler* )

unsubscribe from the DownloadProgress event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadProgress event |

#### 4.5.2.18 void DownloadManager.removeDownloadRemovedEventHandler ( event *handler* )

unsubscribe from the DownloadRemoved event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadAdded event |

#### 4.5.2.19 void DownloadManager.removeDownloadStartEventHandler ( event *handler* )

unsubscribe from the DownloadStart event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadAdded event |

#### 4.5.2.20 void DownloadManager.removeDownloadStateChangeEventHandler ( event *handler* )

unsubscribe from the DownloadStateChange event

**Parameters**

| | |
|---|---|
| *handler* | callback function for DownloadStateChange event |

#### 4.5.2.21 void DownloadManager.setAllDownloadsActive ( bool *active* )

sets all downloads to the active state specified in the parameters

**Parameters**

| | |
|---|---|
| *active* | active state of the download |

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

- downloadmanager.js

## 4.6 DownloadState Class Reference

Download state constants and helper functions.

### Public Member Functions

- string nameFromId (string id)

### Public Attributes

- int CHECK
- int COMPLETE
- int CREATE
- int DOWNLOAD
- int DOWNLOADSTALLED
- int DOWNLOADWITHCHECK
- int FAILURE
- int IDLE
- int PAUSE
- int RESUME
- int SEED
- int START
- int STOP
- int UNKNOWN

### 4.6.1 Detailed Description

Download state constants and helper functions.

### 4.6.2 Member Function Documentation

#### 4.6.2.1 string DownloadState.nameFromId ( string *id* )

converts a download state to a string

**Parameters**

| | |
|---|---|
| *id* | download state |

### 4.6.3 Member Data Documentation

#### 4.6.3.1 int DownloadState.CHECK

Check state

#### 4.6.3.2 int DownloadState.COMPLETE

Complete state

#### 4.6.3.3 int DownloadState.CREATE

Create state

#### 4.6.3.4 int DownloadState.DOWNLOAD

Download state

#### 4.6.3.5 int DownloadState.DOWNLOADSTALLED

Download stalled state

#### 4.6.3.6 int DownloadState.DOWNLOADWITHCHECK

Download with check state

#### 4.6.3.7 int DownloadState.FAILURE

Failure state

#### 4.6.3.8 int DownloadState.IDLE

Idle state

#### 4.6.3.9 int DownloadState.PAUSE

Pause state

#### 4.6.3.10 int DownloadState.RESUME

Resume state

#### 4.6.3.11 int DownloadState.SEED

Seed state

#### 4.6.3.12 int DownloadState.START

Start state

#### 4.6.3.13 int DownloadState.STOP

Stop state

#### 4.6.3.14 int DownloadState.UNKNOWN

Unknown state

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

- downloadstate.js

# Chapter 5

# File Documentation

## 5.1 download.js File Reference

File containing Download class and creation function.

### Classes

- class Download

  *Download a file system.*

### Functions

- void createDownload ()

### 5.1.1 Detailed Description

File containing Download class and creation function.

### 5.1.2 Function Documentation

#### 5.1.2.1 void createDownload ( )

Create instance of download

## 5.2 downloader.js File Reference

File containing Downloader class and creation function.

### Classes

- class Downloader

  *Download management class.*

### Functions

- void createDownloader ()

#### 5.2.1 Detailed Description

File containing Downloader class and creation function.

#### 5.2.2 Function Documentation

##### 5.2.2.1 void createDownloader ( )

Create instance of downloader

## 5.3 downloaderror.js File Reference

File containing download error constants and helper functions.

### Classes

- class DownloadError

  *Download error constants and helper functions.*

### Variables

- DownloadError downloadError

#### 5.3.1 Detailed Description

File containing download error constants and helper functions.

#### 5.3.2 Variable Documentation

##### 5.3.2.1 DownloadError downloadError

precreated global instance of DownloadError

## 5.4 downloadfailure.js File Reference

File containing download failure constants and helper functions.

### Classes

- class DownloadFailure

  *Download failure constants and helper functions.*

### Variables

- DownloadFailure downloadFailure

### 5.4.1 Detailed Description

File containing download failure constants and helper functions.

### 5.4.2 Variable Documentation

#### 5.4.2.1 DownloadFailure downloadFailure

precreated global instance of DownloadFailure

## 5.5 downloadmanager.js File Reference

File containing download manager.

### Classes

- class DownloadManager

  *Manages downloads and provides an integrated event structure.*

### Variables

- DownloadManager downloadManager

### 5.5.1 Detailed Description

File containing download manager.

### 5.5.2 Variable Documentation

#### 5.5.2.1 DownloadManager downloadManager

precreated global instance of the download manager

## 5.6 downloadstate.js File Reference

File containing download state constants and helper functions.

### Classes

- class DownloadState

  *Download state constants and helper functions.*

### Variables

- DownloadState downloadState

### 5.6.1 Detailed Description

File containing download state constants and helper functions.

### 5.6.2 Variable Documentation

#### 5.6.2.1 DownloadState downloadState

precreated global instance of DownloadState

# Index