

Downloader Javascript API

3.1.2.0

Generated by Doxygen 1.7.4

Tue Oct 4 2011 22:46:22

Contents

1	Downloader Javascript API	1
1.1	Overview	1
2	Class Index	3
2.1	Class List	3
3	File Index	5
3.1	File List	5
4	Class Documentation	7
4.1	Download Class Reference	7
4.1.1	Detailed Description	8
4.1.2	Member Function Documentation	8
4.1.2.1	checkFile	8
4.1.2.2	checkFileRange	9
4.1.2.3	eraseContent	9
4.1.2.4	expandString	9
4.1.2.5	getAccessible	9
4.1.2.6	getActive	10
4.1.2.7	getBitActive	10
4.1.2.8	getBitDownloadLimit	10
4.1.2.9	getBitUploadLimit	10
4.1.2.10	getBytesLeft	10
4.1.2.11	getConfigUrl	11
4.1.2.12	getConnectionsCount	11
4.1.2.13	getConnectivity	11

4.1.2.14	getDownloadLimit	11
4.1.2.15	getIsRunning	11
4.1.2.16	getLastError	12
4.1.2.17	getLastFailure	12
4.1.2.18	getMaskEnabled	12
4.1.2.19	getMetafileUrl	12
4.1.2.20	getOutput	12
4.1.2.21	getRemainingTime	12
4.1.2.22	getState	13
4.1.2.23	getTotalBytes	13
4.1.2.24	getTransferRates	13
4.1.2.25	getTransferTotal	13
4.1.2.26	getUploadLimit	13
4.1.2.27	getWebActive	14
4.1.2.28	getWebDownloadLimit	14
4.1.2.29	getWebUploadLimit	14
4.1.2.30	maskFile	14
4.1.2.31	maskFileRange	14
4.1.2.32	release	15
4.1.2.33	setActive	15
4.1.2.34	setBitActive	15
4.1.2.35	setBitDownloadLimit	15
4.1.2.36	setBitUploadLimit	15
4.1.2.37	setConfigUrl	16
4.1.2.38	setDownloadLimit	16
4.1.2.39	setMaskEnabled	16
4.1.2.40	setMetafileUrl	16
4.1.2.41	setOutput	16
4.1.2.42	setUploadLimit	17
4.1.2.43	setWebActive	17
4.1.2.44	setWebDownloadLimit	17
4.1.2.45	setWebUploadLimit	17
4.1.2.46	start	17
4.1.2.47	stop	17

4.1.2.48	syncToMask	18
4.1.3	Member Data Documentation	18
4.1.3.1	instanceId	18
4.1.3.2	onComplete	18
4.1.3.3	onConfigComplete	18
4.1.3.4	onProgress	18
4.1.3.5	onStart	18
4.1.3.6	onStateChange	18
4.2	Downloader Class Reference	19
4.2.1	Detailed Description	19
4.2.2	Member Function Documentation	19
4.2.2.1	getActive	19
4.2.2.2	getDownloadLimit	19
4.2.2.3	getUploadLimit	20
4.2.2.4	release	20
4.2.2.5	setActive	20
4.2.2.6	setDownloadLimit	20
4.2.2.7	setUploadLimit	20
4.2.3	Member Data Documentation	20
4.2.3.1	instanceId	20
4.3	DownloadError Class Reference	21
4.3.1	Detailed Description	21
4.3.2	Member Function Documentation	21
4.3.2.1	nameFromId	21
4.3.3	Member Data Documentation	21
4.3.3.1	ALREADYDOWNLOADING	22
4.3.3.2	CONFIGDOWNLOADFAILED	22
4.3.3.3	CONFIGPARSEFAILED	22
4.3.3.4	CONFIGUNZIPFAILED	22
4.3.3.5	DOWNLOADFAILED	22
4.3.3.6	INVALIDMETAFILE	22
4.3.3.7	INVALIDSOURCE	22
4.3.3.8	METAFILEDOWNLOADFAILED	22
4.3.3.9	METAFILEUNZIPFAILED	22

4.3.3.10	NONE	22
4.3.3.11	NOTENOUGHSPACE	23
4.3.3.12	UNACCEPTABLESOURCE	23
4.3.3.13	UNKNOWN	23
4.4	DownloadFailure Class Reference	23
4.4.1	Detailed Description	23
4.4.2	Member Function Documentation	24
4.4.2.1	nameFromId	24
4.4.3	Member Data Documentation	24
4.4.3.1	CONTENTWRITE	24
4.4.3.2	NONE	24
4.4.3.3	PEERLISTEMPTY	24
4.4.3.4	PEERNOGOODONES	24
4.4.3.5	RELIABLESOURCEBADINTEGRITY	24
4.4.3.6	RELIABLESOURCECANTCONTACT	24
4.4.3.7	RELIABLESOURCENOTSPECIFIED	24
4.4.3.8	TRACKERCANTCONTACT	25
4.4.3.9	TRACKERLISTEMPTY	25
4.4.3.10	TRANSPORTHTTP	25
4.4.3.11	TRANSPORTHTTPCLIENT	25
4.4.3.12	TRANSPORTSOCKET	25
4.5	DownloadManager Class Reference	25
4.5.1	Detailed Description	26
4.5.2	Member Function Documentation	26
4.5.2.1	addDownload	26
4.5.2.2	addDownloadAddedEventHandler	26
4.5.2.3	addDownloadCompleteEventHandler	26
4.5.2.4	addDownloadConfigCompleteEventHandler	27
4.5.2.5	addDownloadProgressEventHandler	27
4.5.2.6	addDownloadRemovedEventHandler	27
4.5.2.7	addDownloadStartEventHandler	27
4.5.2.8	addDownloadStateChangeEventHandler	27
4.5.2.9	downloadById	27
4.5.2.10	hasActiveDownload	28

4.5.2.11	init	28
4.5.2.12	release	28
4.5.2.13	removeDownload	28
4.5.2.14	removeDownloadAddedEventHandler	28
4.5.2.15	removeDownloadCompleteEventHandler	28
4.5.2.16	removeDownloadConfigCompleteEventHandler	29
4.5.2.17	removeDownloadProgressEventHandler	29
4.5.2.18	removeDownloadRemovedEventHandler	29
4.5.2.19	removeDownloadStartEventHandler	29
4.5.2.20	removeDownloadStateChangeEventHandler	29
4.5.2.21	setAllDownloadsActive	29
4.6	DownloadState Class Reference	30
4.6.1	Detailed Description	30
4.6.2	Member Function Documentation	30
4.6.2.1	nameFromId	30
4.6.3	Member Data Documentation	31
4.6.3.1	CHECK	31
4.6.3.2	COMPLETE	31
4.6.3.3	CREATE	31
4.6.3.4	DOWNLOAD	31
4.6.3.5	DOWNLOADSTALLED	31
4.6.3.6	DOWNLOADWITHCHECK	31
4.6.3.7	FAILURE	31
4.6.3.8	IDLE	31
4.6.3.9	PAUSE	31
4.6.3.10	RESUME	32
4.6.3.11	SEED	32
4.6.3.12	START	32
4.6.3.13	STOP	32
4.6.3.14	UNKNOWN	32
5	File Documentation	33
5.1	download.js File Reference	33
5.1.1	Detailed Description	33

5.1.2	Function Documentation	33
5.1.2.1	createDownload	33
5.2	downloader.js File Reference	33
5.2.1	Detailed Description	34
5.2.2	Function Documentation	34
5.2.2.1	createDownloader	34
5.3	downloadererror.js File Reference	34
5.3.1	Detailed Description	34
5.3.2	Variable Documentation	34
5.3.2.1	downloadError	34
5.4	downloadfailure.js File Reference	35
5.4.1	Detailed Description	35
5.4.2	Variable Documentation	35
5.4.2.1	downloadFailure	35
5.5	downloadmanager.js File Reference	35
5.5.1	Detailed Description	35
5.5.2	Variable Documentation	36
5.5.2.1	downloadManager	36
5.6	downloadstate.js File Reference	36
5.6.1	Detailed Description	36
5.6.2	Variable Documentation	36
5.6.2.1	downloadState	36

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:

Download (Download a file system)	7
Downloader (Download management class)	19
DownloadError (Download error constants and helper functions)	21
DownloadFailure (Download failure constants and helper functions)	23
DownloadManager (Manages downloads and provides an integrated event structure)	25
DownloadState (Download state constants and helper functions)	30

Chapter 3

File Index

3.1 File List

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

download.js (File containing Download class and creation function)	33
downloader.js (File containing Downloader class and creation function)	33
downloadererror.js (File containing download error constants and helper functions)	34
downloadfailure.js (File containing download failure constants and helper functions)	35
downloadmanager.js (File containing download manager)	35
downloadstate.js (File containing download state constants and helper functions)	36

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

<i>fileName</i>	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

<i>fileName</i>	filename in metafile to mask
<i>offset</i>	offset in the file to start at (64bit int)
<i>length</i>	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

<i>expand</i>	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

<i>fileName</i>	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

<i>fileName</i>	filename in metafile to mask
<i>offset</i>	offset in the file to start at (64bit int)
<i>length</i>	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

<i>value</i>	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

<i>value</i>	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

<i>value</i>	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

<i>value</i>	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

<i>value</i>	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

<i>value</i>	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

<i>value</i>	Enable/disable download mask.
--------------	-------------------------------

4.1.2.40 void Download.setMetafileUrl (string *value*)

sets the url to metafile.

Parameters

<i>value</i>	Url to metafile.
--------------	------------------

4.1.2.41 void Download.setOutput (string *value*)

sets the location for downloaded data.

Parameters

<i>value</i>	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

<i>value</i>	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

<i>value</i>	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

<i>value</i>	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

<i>value</i>	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

<i>successful</i>	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

<i>percent</i>	(-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

<i>state</i>	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

<i>value</i>	active state of all downloads.
--------------	--------------------------------

4.2.2.6 void Downloader.setDownloadLimit (int *value*)

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

Parameters

<i>value</i>	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

<i>value</i>	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

<i>id</i>	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

<i>id</i>	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

<i>configUrl</i>	url to the configuration file
<i>metafileUrl</i>	url to the metafile
<i>outputDirectory</i>	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

<i>handler</i>	callback function for DownloadAdded event
----------------	-------------------------------------------

4.5.2.3 `void DownloadManager.addDownloadCompleteEventHandler (event handler)`

subscribe to the DownloadComplete event

Parameters

<i>handler</i>	callback function for DownloadComplete event
----------------	----------------------------------------------

4.5.2.4 void DownloadManager.addDownloadConfigCompleteEventHandler (event *handler*)

subscribe to the DownloadConfigComplete event

Parameters

<i>handler</i>	callback function for DownloadConfigComplete event
----------------	----------------------------------------------------

4.5.2.5 void DownloadManager.addDownloadProgressEventHandler (event *handler*)

subscribe to the DownloadProgress event

Parameters

<i>handler</i>	callback function for DownloadProgress event
----------------	----------------------------------------------

4.5.2.6 void DownloadManager.addDownloadRemovedEventHandler (event *handler*)

subscribe to the DownloadRemoved event

Parameters

<i>handler</i>	callback function for DownloadRemoved event
----------------	---------------------------------------------

4.5.2.7 void DownloadManager.addDownloadStartEventHandler (event *handler*)

subscribe to the DownloadStart event

Parameters

<i>handler</i>	callback function for DownloadStart event
----------------	-------------------------------------------

4.5.2.8 void DownloadManager.addDownloadStateChangeEventHandler (event *handler*)

subscribe to the DownloadStateChange event

Parameters

<i>handler</i>	callback function for DownloadStateChange event
----------------	-------------------------------------------------

4.5.2.9 void DownloadManager.downloadById (string *id*)

returns the download by the assigned id

Parameters

<i>id</i>	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

<i>id</i>	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

<i>handler</i>	callback function for DownloadAdded event
----------------	-------------------------------------------

4.5.2.15 void DownloadManager.removeDownloadCompleteEventHandler (event *handler*)

unsubscribe from the DownloadComplete event

Parameters

<i>handler</i>	callback function for DownloadComplete event
----------------	----------------------------------------------

4.5.2.16 void DownloadManager.removeDownloadConfigCompleteEventHandler (event *handler*)

unsubscribe from the DownloadConfigComplete event

Parameters

<i>handler</i>	callback function for DownloadAdded event
----------------	-------------------------------------------

4.5.2.17 void DownloadManager.removeDownloadProgressEventHandler (event *handler*)

unsubscribe from the DownloadProgress event

Parameters

<i>handler</i>	callback function for DownloadProgress event
----------------	----------------------------------------------

4.5.2.18 void DownloadManager.removeDownloadRemovedEventHandler (event *handler*)

unsubscribe from the DownloadRemoved event

Parameters

<i>handler</i>	callback function for DownloadAdded event
----------------	-------------------------------------------

4.5.2.19 void DownloadManager.removeDownloadStartEventHandler (event *handler*)

unsubscribe from the DownloadStart event

Parameters

<i>handler</i>	callback function for DownloadAdded event
----------------	-------------------------------------------

4.5.2.20 void DownloadManager.removeDownloadStateChangeEventHandler (event *handler*)

unsubscribe from the DownloadStateChange event

Parameters

<i>handler</i>	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

<i>active</i>	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

<i>id</i>	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 downloader.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

- addDownload
 - DownloadManager, [26](#)
- addDownloadAddedEventHandler
 - DownloadManager, [26](#)
- addDownloadCompleteEventHandler
 - DownloadManager, [26](#)
- addDownloadConfigCompleteEventHandler
 - DownloadManager, [27](#)
- addDownloadProgressEventHandler
 - DownloadManager, [27](#)
- addDownloadRemovedEventHandler
 - DownloadManager, [27](#)
- addDownloadStartEventHandler
 - DownloadManager, [27](#)
- addDownloadStateChangeEventHandler
 - DownloadManager, [27](#)
- ALREADYDOWNLOADING
 - DownloadError, [21](#)
- CHECK
 - DownloadState, [31](#)
- checkFile
 - Download, [8](#)
- checkFileRange
 - Download, [9](#)
- COMPLETE
 - DownloadState, [31](#)
- CONFIGDOWNLOADFAILED
 - DownloadError, [22](#)
- CONFIGPARSEFAILED
 - DownloadError, [22](#)
- CONFIGUNZIPFAILED
 - DownloadError, [22](#)
- CONTENTWRITE
 - DownloadFailure, [24](#)
- CREATE
 - DownloadState, [31](#)
- createDownload
 - download.js, [33](#)
- createDownloader
 - downloader.js, [34](#)
- DOWNLOAD
 - DownloadState, [31](#)
 - Download, [7](#)
 - checkFile, [8](#)
 - checkFileRange, [9](#)
 - eraseContent, [9](#)
 - expandString, [9](#)
 - getAccessible, [9](#)
 - getActive, [9](#)
 - getBitActive, [10](#)
 - getBitDownloadLimit, [10](#)
 - getBitUploadLimit, [10](#)
 - getBytesLeft, [10](#)
 - getConfigUrl, [10](#)
 - getConnectionsCount, [11](#)
 - getConnectivity, [11](#)
 - getDownloadLimit, [11](#)
 - getIsRunning, [11](#)
 - getLastError, [11](#)
 - getLastFailure, [12](#)
 - getMaskEnabled, [12](#)
 - getMetafileUrl, [12](#)
 - getOutput, [12](#)
 - getRemainingTime, [12](#)
 - getState, [12](#)
 - getTotalBytes, [13](#)
 - getTransferRates, [13](#)
 - getTransferTotal, [13](#)
 - getUploadLimit, [13](#)
 - getWebActive, [13](#)
 - getWebDownloadLimit, [14](#)
 - getWebUploadLimit, [14](#)
 - instanceId, [18](#)
 - maskFile, [14](#)
 - maskFileRange, [14](#)
 - onComplete, [18](#)
 - onConfigComplete, [18](#)
 - onProgress, [18](#)
 - onStart, [18](#)
 - onStateChange, [18](#)
 - release, [15](#)

- setActive, [15](#)
- setBitActive, [15](#)
- setBitDownloadLimit, [15](#)
- setBitUploadLimit, [15](#)
- setConfigUrl, [16](#)
- setDownloadLimit, [16](#)
- setMaskEnabled, [16](#)
- setMetafileUrl, [16](#)
- setOutput, [16](#)
- setUploadLimit, [16](#)
- setWebActive, [17](#)
- setWebDownloadLimit, [17](#)
- setWebUploadLimit, [17](#)
- start, [17](#)
- stop, [17](#)
- syncToMask, [17](#)
- download.js, [33](#)
 - createDownload, [33](#)
- downloadById
 - DownloadManager, [27](#)
- Downloader, [19](#)
 - getActive, [19](#)
 - getDownloadLimit, [19](#)
 - getUploadLimit, [19](#)
 - instanceId, [20](#)
 - release, [20](#)
 - setActive, [20](#)
 - setDownloadLimit, [20](#)
 - setUploadLimit, [20](#)
- downloader.js, [33](#)
 - createDownloader, [34](#)
- DownloadError, [21](#)
 - ALREADYDOWNLOADING, [21](#)
 - CONFIGDOWNLOADFAILED, [22](#)
 - CONFIGPARSEFAILED, [22](#)
 - CONFIGUNZIPFAILED, [22](#)
 - DOWNLOADFAILED, [22](#)
 - INVALIDMETAFILE, [22](#)
 - INVALIDSOURCE, [22](#)
 - METAFILEDOWNLOADFAILED, [22](#)
 - METAFILEUNZIPFAILED, [22](#)
 - nameFromId, [21](#)
 - NONE, [22](#)
 - NOTENOUGHSPACE, [22](#)
 - UNACCEPTABLESOURCE, [23](#)
 - UNKNOWN, [23](#)
- downloadError
 - downloaderror.js, [34](#)
- downloaderror.js, [34](#)
 - downloadError, [34](#)
- DOWNLOADFAILED
 - DownloadError, [22](#)
- DownloadFailure, [23](#)
 - CONTENTWRITE, [24](#)
 - nameFromId, [24](#)
 - NONE, [24](#)
 - PEERLISTEMPTY, [24](#)
 - PEERNOGOODONES, [24](#)
 - RELIABLESOURCEBADINTEGRITY, [24](#)
 - RELIABLESOURCECANTCONTACT, [24](#)
 - RELIABLESOURCENOTSPECIFIED, [24](#)
 - TRACKERCANTCONTACT, [24](#)
 - TRACKERLISTEMPTY, [25](#)
 - TRANSPORTHTTP, [25](#)
 - TRANSPORTHTTPCLIENT, [25](#)
 - TRANSPORTSOCKET, [25](#)
- downloadFailure
 - downloadfailure.js, [35](#)
- downloadfailure.js, [35](#)
 - downloadFailure, [35](#)
- DownloadManager, [25](#)
 - addDownload, [26](#)
 - addDownloadAddedEventHandler, [26](#)
 - addDownloadCompleteEventHandler, [26](#)
 - addDownloadConfigCompleteEventHandler, [27](#)
 - addDownloadProgressEventHandler, [27](#)
 - addDownloadRemovedEventHandler, [27](#)
 - addDownloadStartEventHandler, [27](#)
 - addDownloadStateChangeEventHandler, [27](#)
 - downloadById, [27](#)
 - hasActiveDownload, [28](#)
 - init, [28](#)
 - release, [28](#)
 - removeDownload, [28](#)
 - removeDownloadAddedEventHandler, [28](#)
 - removeDownloadCompleteEventHandler, [28](#)
 - removeDownloadConfigCompleteEventHandler, [28](#)
 - removeDownloadProgressEventHandler, [29](#)

- removeDownloadRemovedEventHandler, [29](#)
 - removeDownloadStartEventHandler, [29](#)
 - removeDownloadStateChangeEventHandlers, [10](#)
 - setAllDownloadsActive, [29](#)
- downloadManager
 - downloadmanager.js, [36](#)
- downloadmanager.js, [35](#)
- downloadManager, [36](#)
- DOWNLOADSTALLED
 - DownloadState, [31](#)
- DownloadState, [30](#)
 - CHECK, [31](#)
 - COMPLETE, [31](#)
 - CREATE, [31](#)
 - DOWNLOAD, [31](#)
 - DOWNLOADSTALLED, [31](#)
 - DOWNLOADWITHCHECK, [31](#)
 - FAILURE, [31](#)
 - IDLE, [31](#)
 - nameFromId, [30](#)
 - PAUSE, [31](#)
 - RESUME, [31](#)
 - SEED, [32](#)
 - START, [32](#)
 - STOP, [32](#)
 - UNKNOWN, [32](#)
- downloadState
 - downloadstate.js, [36](#)
- downloadstate.js, [36](#)
- downloadState, [36](#)
- DOWNLOADWITHCHECK
 - DownloadState, [31](#)
- eraseContent
 - Download, [9](#)
- expandString
 - Download, [9](#)
- FAILURE
 - DownloadState, [31](#)
- getAccessible
 - Download, [9](#)
- getActive
 - Download, [9](#)
 - Downloader, [19](#)
- getBitActive
 - Download, [10](#)
- getBitDownloadLimit
 - Download, [10](#)
- getBitUploadLimit
 - Download, [10](#)
- getBytesLeft
 - Download, [10](#)
- getConfigUrl
 - Download, [10](#)
- getConnectionsCount
 - Download, [11](#)
- getConnectivity
 - Download, [11](#)
- getDownloadLimit
 - Download, [11](#)
 - Downloader, [19](#)
- getIsRunning
 - Download, [11](#)
- getLastError
 - Download, [11](#)
- getLastFailure
 - Download, [12](#)
- getMaskEnabled
 - Download, [12](#)
- getMetafileUrl
 - Download, [12](#)
- getOutput
 - Download, [12](#)
- getRemainingTime
 - Download, [12](#)
- getState
 - Download, [12](#)
- getTotalBytes
 - Download, [13](#)
- getTransferRates
 - Download, [13](#)
- getTransferTotal
 - Download, [13](#)
- getUploadLimit
 - Download, [13](#)
 - Downloader, [19](#)
- getWebActive
 - Download, [13](#)
- getWebDownloadLimit
 - Download, [14](#)
- getWebUploadLimit
 - Download, [14](#)
- hasActiveDownload
 - DownloadManager, [28](#)

- IDLE
 - DownloadState, 31
- init
 - DownloadManager, 28
- instanceld
 - Download, 18
 - Downloader, 20
- INVALIDMETAFILE
 - DownloadError, 22
- INVALIDSOURCE
 - DownloadError, 22
- maskFile
 - Download, 14
- maskFileRange
 - Download, 14
- METAFILEDOWNLOADFAILED
 - DownloadError, 22
- METAFILEUNZIPFAILED
 - DownloadError, 22
- nameFromId
 - DownloadError, 21
 - DownloadFailure, 24
 - DownloadState, 30
- NONE
 - DownloadError, 22
 - DownloadFailure, 24
- NOTENOUGHSPACE
 - DownloadError, 22
- onComplete
 - Download, 18
- onConfigComplete
 - Download, 18
- onProgress
 - Download, 18
- onStart
 - Download, 18
- onStateChange
 - Download, 18
- PAUSE
 - DownloadState, 31
- PEERLISTEMPTY
 - DownloadFailure, 24
- PEERNOGOODONES
 - DownloadFailure, 24
- release
 - Download, 15
- Downloader, 20
- DownloadManager, 28
- RELIABLESOURCEBADINTEGRITY
 - DownloadFailure, 24
- RELIABLESOURCECANTCONTACT
 - DownloadFailure, 24
- RELIABLESOURCENOTSPECIFIED
 - DownloadFailure, 24
- removeDownload
 - DownloadManager, 28
- removeDownloadAddedEventHandler
 - DownloadManager, 28
- removeDownloadCompleteEventHandler
 - DownloadManager, 28
- removeDownloadConfigCompleteEventHandler
 - DownloadManager, 28
- removeDownloadProgressEventHandler
 - DownloadManager, 29
- removeDownloadRemovedEventHandler
 - DownloadManager, 29
- removeDownloadStartEventHandler
 - DownloadManager, 29
- removeDownloadStateChangeEventHandler
 - DownloadManager, 29
- RESUME
 - DownloadState, 31
- SEED
 - DownloadState, 32
- setActive
 - Download, 15
 - Downloader, 20
- setAllDownloadsActive
 - DownloadManager, 29
- setBitActive
 - Download, 15
- setBitDownloadLimit
 - Download, 15
- setBitUploadLimit
 - Download, 15
- setConfigUrl
 - Download, 16
- setDownloadLimit
 - Download, 16
 - Downloader, 20
- setMaskEnabled
 - Download, 16
- setMetafileUrl
 - Download, 16
- setOutput

- Download, [16](#)
- setUpUploadLimit
 - Download, [16](#)
 - Downloader, [20](#)
- setWebActive
 - Download, [17](#)
- setWebDownloadLimit
 - Download, [17](#)
- setWebUploadLimit
 - Download, [17](#)
- START
 - DownloadState, [32](#)
- start
 - Download, [17](#)
- STOP
 - DownloadState, [32](#)
- stop
 - Download, [17](#)
- syncToMask
 - Download, [17](#)
- TRACKERCANTCONTACT
 - DownloadFailure, [24](#)
- TRACKERLISTEMPTY
 - DownloadFailure, [25](#)
- TRANSPORTHTTP
 - DownloadFailure, [25](#)
- TRANSPORTHTTPCLIENT
 - DownloadFailure, [25](#)
- TRANSPORTSOCKET
 - DownloadFailure, [25](#)
- UNACCEPTABLESOURCE
 - DownloadError, [23](#)
- UNKNOWN
 - DownloadError, [23](#)
 - DownloadState, [32](#)