Signiant Transfer SDK

JavaScript API Guide

Signiant Inc.

Release 4/21/2015

Table of Contents

Overview

This reference documents the Signiant Transfer API. Using a JavaScript API, the Signiant Transfer API uses accelerated transfer technology to allow you to manage file transfers.

Global Methods

Methods

detectPlugin(callbackFuncs:JSON)

Detects the presence of the Signiant Transfer Plug-in. This method ensures the proper version of the plug-in is present, loading a new plug-in if required. The method calls the success callback if the plug-in is successfully detected; the method calls the error callback if it is not.

Parameters

- **callbackFuncs:JSON** - JSON object defining the success and error callbacks for plug-in detection. The format of the JSON object is:

```
{
     "success":successCallback,
     "error":errorCallback
 }
```

Returns

None.

Messages

This object allows you to modify the error/informational messages that are returned by the Transfer API. The messages are stored in this object in the form of a JSON object, with a key-value pair per message. Replacing this JSON object allows the Transfer API to return messages in the language you and the user specify.

Constructor

Signiant.Mst.Messages

Creates a new Messages object to retrieve and modify messages returned by the Transfer API.

Methods

getMessage(messageKey:String)

Returns the message for a specific key.

Parameters

- **messageKey:String** - The key of the message to return.

Returns

- **String** - The message value for the supplied key; returns undefined if there is no value for the key.

setMessages(jsonMap:JSON)

Replaces the current or default JSON map of event messages returned by the Transfer API.

Parameters

- **jsonMap:JSON** - The JSON map of event messages to use by the Transfer API.

Returns

None.

Event Messages

The following is the complete list of event messages used by the Transfer API. To change the messages or localize the messages using the `Messages.setMessages()` method, create a new object with these keys and update the values.

```
{
      'signiant.error.jquery.missing'
: 'Unable to locate jQuery. Aborting.',
      'signiant.error.mst.script.apikey.invalid'
: 'API key does not exist.',
      'signiant.error.mst.script.apikey.missing'
: 'A valid api-key must be specified in the api-key attribute of the <script>
tag',
      'signiant.error.mst.script.apikey.unknown'
: 'Could not validate the API key due to an unexpected error',
      'signiant.error.mst.script.missing'
: 'Could not find a <script> tag with id ms-api',
      'signiant.error.validateKeyUnknown'
: 'An unknown error occurred while validating API key.',
      'signiant.install.plugin.button'
: 'Install Plug-in',
      'signiant.install.plugin.download.prefix'
: 'If your download does not start automatically, please ',
      'signiant.install.plugin.download.click'
: 'click here',
      'signiant.install.plugin.download.suffix'
: '.',
      'signiant.install.plugin.download.step1.label'
: '1',
      'signiant.install.plugin.download.step1.description'
: 'Download',
      'signiant.install.plugin.download.step2.label'
: '2',
      'signiant.install.plugin.download.step2.description'
: 'Run',
      'signiant.install.plugin.download.step3.label'
: '3',
      'signiant.install.plugin.download.step3.description'
: 'Use',
      'signiant.install.plugin.explanation'
: 'This plug-in is required to secure and accelerate your  transfers. Click
Install Plug-in to start the install/update process.',
      'signiant.install.plugin.header'
: 'Transfer Plug-in',
      'signiant.install.plugin.help'
: 'Help',
      'signiant.install.plugin.missing'
: 'A plug-in needed by Signiant could not be found.',
      'signiant.transfer.error.auth'
: 'Transfer authentication error.',
      'signiant.transfer.error.auth.config'
: 'Transfer authentication error due to an incomplete configuration.',
      'signiant.transfer.error.auth.credentials.incomplete'
: 'Transfer authentication error due to missing credentials.',
      'signiant.transfer.error.auth.credentials.invalid'
: 'Transfer authentication error due to invalid credentials.',
      'signiant.transfer.error.auth.ssl.ca'
: 'Transfer authentication error due to an SSL communication error with the
Signiant Flight server.',
      'signiant.transfer.error.auth.ssl.ca.config'
: 'Transfer authentication error due to an SSL configuration error.',
      'signiant.transfer.error.auth.ssl.ca.insufficientMemory'
: 'Transfer authentication error due to an SSL communication error with the
Signiant Flight server.',
      'signiant.transfer.error.auth.ssl.ca.missing'
: 'Transfer authentication error due to an SSL configuration error.',
      'signiant.transfer.error.auth.ssl.certificateRejected'
: "Transfer authentication error because the Signiant Flight server's SSL
certificate was rejected.",
      'signiant.transfer.error.auth.ssl.context'
: 'Transfer authentication error due to an SSL communication error with the
Signiant Flight server.',
      'signiant.transfer.error.autoRetry'
: 'The connection to your Signiant Flight server was lost. Automatically
reconnecting in 1 minute.',
      'signiant.transfer.error.callbackFailedForConnStatus'
: 'Failed to call callback for connection status event.',
      'signiant.transfer.error.callbackFailedForConnStatusCompQueued'
: 'Failed to call callback for connection-queued event.',
      'signiant.transfer.error.callbackFailedForPostXfer'
: 'Failed to call callback for post transfer event.',
      'signiant.transfer.error.callbackFailedForPreXfer'
: 'Failed to call callback for pre transfer event.',
      'signiant.transfer.error.callbackFailedForTooManyFiles'
: 'Failed to send callback for too many files chosen while upload.',
      'signiant.transfer.error.callbackFailedForXferPrefile'
: 'Failed to call callback for pre file event.',
      'signiant.transfer.error.callbackFailedForXferStart'
: 'Failed to call callback for transfer started event.',
      'signiant.transfer.error.connection.dataChannel'
: 'Unable to establish a connection to the Signiant Flight server.',
      'signiant.transfer.error.connection.hostNotFound'
: 'Unable to establish a connection to the Signiant Flight server.',
      'signiant.transfer.error.connection.insufficientMemory'
: 'The connection to the Signiant Flight server was lost.',
      'signiant.transfer.error.connection.lostHeartBeat'
: 'The connection to the Signiant Flight server was lost.',
      'signiant.transfer.error.connection.readChannel'
: 'Unable to read any data from the Signiant Flight server.',
      'signiant.transfer.error.connection.writeChannel'
: 'Unable to write any data to the Signiant Flight server.',
      'signiant.transfer.error.failedToCancelTransfer'
: 'Failed to cancel transfer.',
      'signiant.transfer.error.failedToChooseDownloadFolder'
: 'Failed to choose download folder.',
      'signiant.transfer.error.failedToChooseFolder'
: 'Failed to choose folder.',
      'signiant.transfer.error.failedToChooseUploadFiles'
: 'Failed to choose upload files.',
      'signiant.transfer.error.failedToClearFile'
: 'Failed to clear file.',
      'signiant.transfer.error.failedToClearFileList'
: 'Failed to clear file list.',
      'signiant.transfer.error.failedToClearServerList'
: 'Failed to clear server list.',
      'signiant.transfer.error.failedToDownload'
: 'Failed to start the download.',
      'signiant.transfer.error.failedToDownloadAtPlugin'
: 'Failed to download files.',
      'signiant.transfer.error.failedToHandleCallback'
: 'Failed to handle callback.',
      'signiant.transfer.error.failedToHandleErrors'
: 'Failed to handle errors.',
      'signiant.transfer.error.failedToHandleException'
: 'Failed to handle transfer exception event.',
      'signiant.transfer.error.failedToHandleUploadFilesProcessed'
: 'Failed to handle upload files processed event',
      'signiant.transfer.error.failedToHandleXferProgress'
: 'Failed to handle transfer-progress event.',
      'signiant.transfer.error.failedToSetDownloadFolder'
: 'Failed to set chosen download folder.',
      'signiant.transfer.error.failedToSetLogLevel'
: 'Failed to set plug-in log-level.',
      'signiant.transfer.error.failedToSetUploadFiles'
: 'Failed to set chosen upload files.',
      'signiant.transfer.error.failedToSubscribeBasicEvents'
: 'Failed to subscribe basic events.',
      'signiant.transfer.error.failedToSubscribeForAdvEvent'
: 'Failed to subscribe for event: ',
      'signiant.transfer.error.failedToSubscribeForEvent'
: 'Could not subscribe to event as the transfer-plugin could not be found in
the DOM:',
      'signiant.transfer.error.failedToSubscribeXferErrors'
: 'Failed to subscribe for transfer-errors event.',
      'signiant.transfer.error.failedToSubscribeXferProgress'
: 'Failed to subscribe for transfer-progress events.',
      'signiant.transfer.error.failedToUpload'
: 'Failed to start the upload.',
      'signiant.transfer.error.failedToUploadAtPlugin'
: 'Failed to upload files.',
      'signiant.transfer.error.fallbackToOneStar'
: 'Unable to connect to your Signiant Flight server using 2 Star service.
Connecting using 1 Star service instead.',
      'signiant.transfer.error.fallbackToTwoStar'
: 'Unable to connect to your Signiant Flight server using 3 Star service.
Connecting using 2 Star service instead.',
      'signiant.transfer.error.invalidLogLevel'
: 'Supplied plug-in log-level is invalid: ',
      'signiant.transfer.error.invalidServer'
: 'Please specify a valid server.',
      'signiant.transfer.error.invalidAgentForTrialKey'
: 'With unlicensed API keys, you can only perform transfers to/from our
cloud-hosted servers.',
      'signiant.transfer.error.invalidCallback'
: 'Please supply a callback method.',
      'signiant.transfer.error.invalidDestinationDir'
: 'Please supply a valid Destination Directory.',
      'signiant.transfer.error.invalidListOfFiles'
```

```
: 'Please supply a list of files as an Array.',
      'signiant.transfer.error.invalidPassword'
: 'Please supply a valid Password.',
      'signiant.transfer.error.invalidPropertyDelete'
: 'Please provide a String value for name of the property you wish to
delete.',
      'signiant.transfer.error.invalidPropertyName'
: 'Please provide a String value for the name of the property you wish to
modify.',
      'signiant.transfer.error.invalidPropertyRetrieval'
: 'Please provide a String value for the name of the property you wish to
retrieve.',
      'signiant.transfer.error.invalidTransferProperties'
: 'Please provide transfer properties in the form of a JSON object.',
      'signiant.transfer.error.invalidTrustedCA'
: 'Please supply a valid certificate for the Certificate Authority',
      'signiant.transfer.error.invalidUsername'
: 'Please supply a valid Username.',
      'signiant.transfer.error.nonExistingEvent'
: 'Please supply a valid event name.',
      'signiant.transfer.error.invalidFilePathHandlingMode'
: 'Please supply a valid File Path Handling Mode. The Mode can be either PATH
or FULL or FLAT',
      'signiant.transfer.error.invalidFileCollisionHandling'
: 'Please supply a valid File Collision Handling Mode. The Mode can be either
overwrite or version ',
      'signiant.transfer.error.noTransferToCancel'
: 'No transfer in progress to cancel.',
      'signiant.transfer.error.property.agent.nameTooLong'
: 'The agent name property is too long.',
      'signiant.transfer.error.property.bandwidthThrottle.invalid'
: 'The bandwidth throttle property setting is invalid.',
      'signiant.transfer.error.property.destinationDir.invalid'
: 'The destination dir property setting is invalid.',
      'signiant.transfer.error.property.destinationDir.notFound'
: 'The destination dir property setting is missing.',
      'signiant.transfer.error.property.encryption.unsupported'
: 'The encryption property setting specifies an unsupported encryption
type.',
      'signiant.transfer.error.property.file.invalid'
: 'The file specification property is invalid.',
      'signiant.transfer.error.property.file.notFound'
: 'The file property specifies a file which does not exist on the agent.',
      'signiant.transfer.error.property.file.nameTooLong'
: 'The file property specifies a file with a name which is too long.',
      'signiant.transfer.error.property.file.open'
: 'The file property specifies a file which can not be opened by the agent.',
      'signiant.transfer.error.property.file.unsupported'
: 'The file property specifies a file which can not be transferred because
its file type is unsupported.',
      'signiant.transfer.error.property.fileCollision.invalid'
: 'The file collision handling property is invalid.',
      'signiant.transfer.error.property.fileList.notFound'
: 'Unable to find any of the files requested from the plug-in.',
      'signiant.transfer.error.property.mode.invalid'
: 'The mode property must be set to send or receive.',
      'signiant.transfer.error.property.sandbox.notFound'
```

```
: 'The sandbox directory property setting is invalid.',
      'signiant.transfer.error.property.transport.unsupported'
: 'The transport property setting specifies an unsupported transport type.',
      'signiant.transfer.error.property.url.invalid'
: 'Encountered a URL with an invalid format.',
      'signiant.transfer.error.specifyConfigForDownload'
: 'Please specify configuration properties to perform an download.',
      'signiant.transfer.error.specifyConfigForUpload'
: 'Please specify configuration properties to perform an upload.',
      'signiant.transfer.error.specifyFileToDownload'
: 'Please specify at least one file to download.',
      'signiant.transfer.error.specifyFileToUpload'
: 'Please specify at least one file to add to the list of files to be
uploaded.',
      'signiant.transfer.error.stream.fileDisposition.unsupported'
: 'Encountered an unsupported file disposition request.',
      'signiant.transfer.error.stream.fileType.unsupported'
: 'Encountered an unsupported file type.',
      'signiant.transfer.error.tooManyFiles'
: 'The number of files chosen for upload exceeds the maximum number of
allowable files.',
      'signiant.transfer.error.undefinedFileSelectCallback'
: 'Callback is not defined for file selector.',
      'signiant.transfer.error.undefinedFolderSelectCallback'
: 'Callback is not defined for folder selector.',
      'signiant.transfer.error.undefinedListOfFiles'
: 'Please supply a valid list of files for the transfer.',
      'signiant.transfer.error.undefinedPropertyDelete'
: 'Please specify a valid name for the property you wish to delete.',
      'signiant.transfer.error.undefinedPropertyName'
: 'Please specify a valid name for the property you wish to set.',
      'signiant.transfer.error.undefinedPropertyRetrieval'
: 'Please specify a valid name for the property you wish to retrieve.',
      'signiant.transfer.error.undefinedPropertyValue'
: 'Please specify a valid value for property ',
      'signiant.transfer.error.undefinedTransferProperties'
: 'Please specify some properties for the transfer.',
      'signiant.transfer.error.unexpectedError'
: 'Unable to perform transfer. Please verify the supplied transfer properties
and selected files.',
      'signiant.transfer.error.unexpectedProperties'
: 'Unable to perform transfer. Please verify the supplied transfer
properties.',
      'signiant.transfer.error.unexpectedResponse'
: 'The Signiant Flight server returned an unexpected response during file
transfer. Your transfer has been canceled.',
      'signiant.transfer.error.unexpectedResponse.fileSequenceNum.notFound'
: 'The Signiant Flight server returned an unexpected response during file
transfer. Your transfer has been canceled.',
      'signiant.transfer.error.unexpectedResponse.server.dataType'
: 'The Signiant Flight server returned an unexpected response during file
transfer. Your transfer has been canceled.',
      'signiant.transfer.error.unexpectedResponse.server.malformedData'
: 'The Signiant Flight server returned an unexpected response during file
transfer. Your transfer has been canceled.',
      'signiant.transfer.error.jobNamePrefix.invalidJobNamePrefix'
: 'The specified Job Name Prefix is invalid. Value must be alpha numeric and
```

```
limited to 32 characters. ',
      'signiant.transfer.error.jobNamePrefix.failure'
: 'Please provide a value for Job Name Prefix.',
      'signiant.transfer.error.serviceData.failure'
: 'Please provide a value for Service Data.',
      'signiant.transfer.status.transferStarted'
: 'Transfer started',
      'signiant.transfer.status.preTransfer'
: 'Pre Transfer',
      'signiant.transfer.status.preFile'
: 'Pre File',
      'signiant.transfer.status.connectionStatusComponentQueued'
: 'Connection Queued',
      'signiant.transfer.transferException'
: 'Exception in transfer',
      'signiant.transfer.uploadFilesProcessed'
: 'Upload Files Processed',
      'signiant.transfer.status.connected'
: 'Connected to server',
      'signiant.transfer.status.connecting'
: 'Connecting to server..',
      'signiant.transfer.status.disconnected'
: 'Disconnected from server',
      'signiant.transfer.status.disconnecting'
: 'Disconnecting from server..',
      'signiant.transfer.status.transferFailed'
: 'Transfer Failed.',
      'signiant.transfer.status.transferring'
: 'Started transferring files..',
      'signiant.transfer.status.transferCanceled'
: 'Transfer Canceled.',
      'signiant.transfer.status.transferSuccessful'
: 'Transfer Successful.',
      'signiant.error.detectPlugin.failureAlert'
: 'Failed to query the latest plug-in version from Signiant'
      'signiant.transfer.error.injected.file.entries.found'
: 'Injected file entries detected.'
      'signiant.transfer.error.injected.dest.dir.found'
            : 'Injected destination directory detected.'
 }
```

Transfer

This is the base class for Upload and Download. It contains behavior that's common for uploads and downloads. While your application will use the Upload and Download classes to perform transfers, most of the functionality is contained in this class.

Each Transfer object has a unique transfer ID that you retrieve using the `getTransferId()` method. This ID allows you to identify the transfer object for monitoring and logging purposes.

See the *Signiant Transfer SDK JavaScript Developer Guide* for instructions on setting up and interacting with Transfer objects.

Constructor

This is the base class for `Upload` and `Download` objects. Do no instantiate this class.

Methods

cancel()

This method begins the process of cancelling a transfer.

This method returns as soon as the Signiant Transfer Plug-in has been successfully asked to cancel the transfer. Further notifications related to the state/progress of the transfer can be retrieved by registering event listeners with the `subscribeForBasicEvents()` and `subscribeForTransferProgress()` functions.

This method throws an exception if a transfer is not in progress.

Parameters

None.

Returns

None.

clearServerList()

Removes all servers from the list of servers.

Parameters

None.

Returns

None.

deleteProperty(name:String)

Deletes the transfer configuration property specified by name.

This method throws an exception if name is undefined or if the data type of name is not String.

Parameters

- **name:String** - Key identifying the configuration property to delete.

Returns

None.

getDestinationDir()

Deprecated

Not used with the Signiant Flight service.

getLastError()

Returns the detailed error code for the latest error encountered by the Transfer API during an upload or download.

Parameters

None.

Returns

- **String** - Latest error code returned by the Transfer API.

getProperty(name:String)

Returns the transfer configuration property specified by name.

This method throws an exception if name is undefined or if the data type of name is not String.

Parameters

- **name:String** - Key identifying the configuration property to return.

Returns

- **String** - Value of the configuration property with the key name.

getServer()

Returns the list of servers to use for this parameter.

Parameters

None.

Returns

- **String** - Plain text list of servers to use for this parameter.

getSubFolder()

Returns the sub folder set for the upload or download. If a sub folder was not set undefined will be returned.

Parameters

None.

Returns

- **String** - Plain text sub folder set for the transfer.

getTargetFileName()

Returns the target file name mapping for the upload or download. If a sub folder was not set undefined will be returned.

Parameters

None.

Returns

- **String** – File name specified.

getTransferId()

Returns the unique transfer ID identifying the `Transfer` object. Each `Transfer` object has a unique ID that identifies the object for monitoring, reporting, and debugging. The transfer ID is created by the Transfer API when the Transfer object is created.

Parameters

None.

Returns

- **String** - Transfer ID for the transfer object.

getTrustedCA()

Deprecated

Not used with the Signiant Flight service.

isIEProtectedModeOn()

Returns true if the following is true:

- Current browser is Internet Explorer,
- Protected Mode is turned on in the browser's security settings, and
- Windows User Account Control is enabled.

Otherwise, returns false.

Parameters

None.

Returns

- **Boolean** - True if browser in IE protected mode.

resetTransferProperties()

Clears all the configuration properties for the upload or download.

Parameters

None.

Returns

- **JSON** - Returns undefined since configuration properties is now cleared.

setCredentials(username:String, password:String)

Deprecated

Not used with the Signiant Flight service.

setDestinationDir(destinationDir:String)

Deprecated

Not used with the Signiant Flight service.

Returns

None.

setFileEncryption(properties:JSON)

Enables encryption (uploads) or decryption (downloads) of all files in the transfer object using keys and the provided initialization vector. The keys and initialization vector must be managed by the developer (Signiant does not store or manage these). It’s recommended to use the same key and initialization vector only once per file.

Returns

None.

setJobNamePrefix(value:String)

Deprecated

Not used with the Signiant Flight service.

Returns

None.

setProbeLB(probelb:boolean)

Call this method using a value of true to enable load balancing servers. Since should always be true for the Flight service.

Parameters

- **probelb:boolean** - If true, enable load balancer.

Returns

None.

setProperty(name:String, value:Object)

Sets the value of a specific transfer configuration property.

This method throws an exception if name or value is undefined or if the data type of name or value is not String.

Parameters

- **name:String** - Key identifying the configuration property to set.
- **value:Object** - Value to set for the configuration property.

Returns

None.

setServer(server:String, relays:Array)

Adds servers to the list of servers.

Pass in the host name as the parameter and this method creates a list of three servers with the same host, each containing a different protocol (UDP, TCP, HTTP) and using the default port for each protocol.

Parameters

- **server:String** - Host name for the three servers to add.
- **relays:Array** - *Optional.* Host name for a relay server.

None.

setStorageConfig(config:String)

Sets the JSON-formatted string that configures the cloud object storage used by the Signiant Flight service.

See the Signiant Transfer SDK JavaScript Developer Guide for instructions on creating the storage config JSON string.

This method throws an exception if config is undefined or if the data type of config is not String.

Parameters

- **config:String** - Key identifying the configuration property to set.

Returns

None.

setSubFolder(folder:String)

Sets a sub folder set for the upload or download. On upload the sub folder will be prepended to the file and folders specified. The files and folders will be created on the target file system under the sub folder specified. In the download case the sub folder is prepended to the file names specified for download. In this case the files must exist in the specified by the sub folder.

Parameters

- **folder:String** – sub folder.

Returns

None.

setTargetFileName(filename:String)

Set target file name allows for a file being uploaded to be saved with a different name on the target storage system. The current implementation is limited to a single file. If multiple files or folders are specified for upload and a target file name is set the upload will not start.

Parameters

- **filename:String** – File name.

Returns

None.

setTransferProperties(properties:JSON)

Overwrites all the transfer configuration properties with the supplied properties parameter.

This method throws an exception if the properties are undefined or not a JSON object.

Parameters

- **properties:JSON** - JSON object containing the list of transfer configuration properties.

Returns

None.

setTrustedCA(trustedCACertificate:String)

Deprecated

Not used with the Signiant Flight service.

None.

subscribeForBasicEvents(callbackFunc:Function)

Sets the callback function for your application to receive transfer events.

The callback function should take the following parameters:

- Upload or Download transfer object (Object)
- Event Type (String)
- Message (String)
- *Optional:* Event-specific data. On returned for certain event types.

See Events for a list of transfer events.

Parameters

- **callbackFunc:Function** - Callback function for basic events.

Returns

None.

subscribeForTransferErrors(callbackFunc:Function)

Sets the callback function for your application to receive transfer errors.

The callback function should take the following parameters:

- Upload or Download transfer object (Object)
- Event Type (String)
- Error Message (String)
- *Optional:* Name of configuration property that caused the error (String)

See Errors for a list of transfer errors.

Parameters

- **callbackFunc:Function** - Callback function for transfer errors.

Returns

None.

subscribeForTransferProgress(callbackFunc:Function)

Sets the callback function for your application to receive transfer progress.

The callback function should take the following parameters:

- Upload or Download transfer object (Object)
- Bytes Sent/Received (Number)
- Total Bytes to be transferred (Number)
- Estimated Time Remaining, in seconds (Number)

Parameters

- **callbackFunc:Function** - Callback function for transfer progress.

Returns

None.

setLogLevel(logLevel:String)

Sets the detail level for the log function to one of the following:

- INFO - Default. Default logging level.
- DEBUG - Detailed debugging logging level.

Parameters

- **logLevel:String** - Detail level for logging.

Returns

None.

getLogLevel()

Returns the detail level for the log function as one of the following:

- INFO - Default. Default logging level.
- DEBUG - Detailed debugging logging level.

Parameters

None.

Returns

- **String** - Detail level for logging.

Upload

Use the Upload class to create transfers from the client application to the cloud object storage. Most of the functionality of the Upload class is inherited from the Transfer class.

One Signiant.Mst.Upload object must be created per upload.

Constructor

Signiant.Mst.Upload

Creates a new Upload object.

Methods

clearFiles(filesArray:Array)

Removes a list of files from the list of files to transfer. Pass the names of the files to remove, including path, as an array.

This method throws an exception if filesArray is undefined or the data type of filesArray is not Array.

Parameters

- **filesArray:Array** - Array of files to remove from the list of files to transfer.

Returns

None.

clearAllFiles()

Removes all files from the list of files to transfer.

Parameters

None.

Returns

None.

chooseUploadFiles(callbackFunc:Function)

This function launches the platform-specific file chooser so that the user can select the files/folders to be uploaded.

Once the user has selected the desired files/folders, the specified callbackFunc is called.

The callback function should take the following parameters:

- Event Type (String) which will be UPLOAD_FILES_CHOSEN.
- The selected files/folders as a JSON array with a JSON object for each file/folder to upload.

  The format of the JSON objects in the array is:

  ```
  {
        "name":"upload.txt",
        "path":"/tmp/upload.txt",
        "isDirectory":"false",
        "size":"10190"
   }
  ```

If the user does not select any files/folders, the callbackFunc is not invoked.

Parameters

- **callbackFunc:Function** - Callback function to call when the user selects files.

Returns

None.

setFilesToUpload(filePathArray:Array)

Sets the list of files to upload. Pass in an array containing the list of files to transfer, including paths.

This method throws an exception if filePathArray is undefined or empty.

Parameters

- **filePathArray:Array** - Array of files to transfer.

Returns

None.

startUpload()

This method starts the upload process. This method validates all input data such as credentials and file list. It then instructs the Transfer API to transfer the files and immediately returns.

Parameters

None.

Returns

None.

Download

Use the Download class to create transfers from the Signiant Flight service to the client. Most of the functionality of the Download class is inherited from the Transfer class.

One Signiant.Mst.Download object must be created per download.

Constructor

Signiant.Mst.Download

Creates a new Download object.

Methods

clearFiles(filesArray:Array)

Removes a list of files from the list of files to transfer. Pass the names of the files to remove, including path, as an array.

This method throws an exception if filesArray is undefined or the data type of filesArray is not Array.

Parameters

- **filesArray:Array** - Array of files to remove from the list of files to transfer.

Returns

None.

clearAllFiles()

Removes all files from the list of files to transfer.

Parameters

None.

Returns

None.

getFileCollisionHandlingMode()

Returns the value of the configuration property that determines how filename collisions are handled for downloaded content. The following values can be returned:

- version - If the name of a file/folder being downloaded collides with a file/folder that already exists in the destination directory, then a version suffix is appended to the name of the file/folder being downloaded.
- overwrite - If the name of a file/folder being downloaded collides with a file/folder that already exists in the destination directory, then the pre-existing file/folder is overwritten with the file/folder being downloaded.

Parameters

None.

Returns

- **String** - File collision handling mode.

getFilePathHandlingMode()

Returns the value of the configuration property that determines the folder-structure that is recreated for downloaded content. The following values can be returned:

- FLAT - All downloaded content goes directly into the directory specified by the `setDownloadFolder()` method. See `setFileCollisionHandlingMode()` above for more information on using filename collision handling with FLAT file path handling mode.
- PATH - All downloaded content goes into the directory specified by the `setDownloadFolder()` method, but the relative path of the content is duplicated, as it exists in the content repository on the source server.
- FULL - All downloaded content goes into the directory specified by the `setDownloadFolder()` method, but the full path of the content is duplicated, as it exists in the content repository on the source server.

Parameters

None.

Returns

- **String** - File path handling mode.

setDownloadFolder(folder:String)

Sets the path to the folder where files are downloaded to.

Parameters

- **folder:String** - The folder to download files to.

Returns

None.

setFileCollisionHandlingMode(mode:String)

Sets the value of the configuration property that determines how filename collisions are handled for downloaded content.

The mode value is one of the following:

- version - If the name of a file/folder being downloaded collides with a file/folder that already exists in the destination directory, then a version suffix is appended to the name of the file/folder being downloaded.
- overwrite - If the name of a file/folder being downloaded collides with a file/folder that already exists in the destination directory, then the pre-existing file/folder is overwritten with the file/folder being downloaded.

This method throws an exception if mode is undefined, empty, or invalid.

Parameters

- **mode:String** - File collision handling mode.

Returns

None.

setFilePathHandlingMode(mode:String)

Sets the value of the configuration property that determines the folder-structure that is recreated for downloaded content.

The mode value is one of the following:

- FLAT - All downloaded content goes directly into the directory specified by the `setDownloadFolder()` method. See `setFileCollisionHandlingMode()` above for more information on using filename collision handling with FLAT file path handling mode.

- PATH - All downloaded content goes into the directory specified by the `setDownloadFolder()` method, but the relative path of the content is duplicated, as it exists in the content repository on the source server.
- FULL - All downloaded content goes into the directory specified by the `setDownloadFolder()` method, but the full path of the content is duplicated, as it exists in the content repository on the source server.

This method throws an exception if mode is undefined, empty, or invalid.

Parameters

- **mode:String** - File path handling mode.

Returns

None.

setFilesToDownload(filePathArray:Array)

Sets the list of files to download. Pass in an array containing the list of files to transfer, including paths.

This method throws an exception if filePathArray is undefined or empty.

Parameters

- **filePathArray:Array** - Array of files to transfer.

Returns

None.

startDownload()

This method starts the download process. This method validates all input data such as credentials and file list. It then instructs the Transfer API to transfer the files and immediately returns.

Parameters

None.

Returns

None.

Events and Errors

Events

Event	Description	Event Data
`TRANSFER_STARTED`	The upload or download has started. Data is moving between the Signiant Transfer API and the Signiant Flight Server.	None
`PRE_TRANSFER_EVENT`	Upload or download set up event. This event is generated before the transfer started event and the transfer has actually started.	The transfer protocol being used for the upload or download. The value is one of the following: • **mxwan** - Signiant Acceleration protocol • **mxtcp** - Basic transfer (TCP based) • **http** - Signiant protocol tunneled over HTTP
`TRANSFER_CONNECTION_STATUS`	The current status of the transfer. As the transfer transition states the event will be generated with the new status value.	The following status values are generated: • **CONNECTING** - Signiant Transfer API is connecting to the Signiant Flight Server • **CONNECTED** - Signiant Transfer API has connected to the Signiant Flight Server • **TRANSFERRING** - Upload or download is in

Event	Description	Event Data
		progress - **DISCONNECTING** - Signiant Transfer API is disconnecting from the Signiant Flight Server - **DISCONNECTED** - Signiant Transfer API has disconnected from the Signiant Flight Server - **COMPONENT_QUEUED** - Upload or download is queued
`TRANSFER_QUEUED`	The upload or download has been queued for transfer.	The estimated amount of time the transfer will be queued before starting.
`TRANSFER_PRE_FILE_EVENT`	Generated for each file that will be uploaded or downloaded. This event is generated before the upload or download of the file occurs.	The name of the file to be transferred.
`TRANSFER_ERROR_EVENT`	Generated when an error occurs in the transfer.	If the error was generated by an invalid configuration parameter, the name of the parameter is returned.
`INJECTED_FILES_FOUND_EVENT`	Files have been detected for upload that have not been selected by the user with the Transfer API file selector.	None
`INJECTED_DESTINATION_DIR_EVENT`	A destination folder has been detected for download that was not selected by the user with the Transfer API folder	None

Event	Description	Event Data
	selector.	
TRANSFER_CANCEL_EVENT	The upload or download has been cancelled.	None
TRANSFER_COMPLETED	Generated upon completion of the upload or download.	None

Errors

Error Event Type	Description
INCOMPLETE_USER_CREDENTIALS	The user name and or password have not been specified.
INVALID_CREDENTIALS	The user credentials specified have failed authentication or the Signiant Flight Server has refused the transfer.
SPECIFIED_FILE_DOES_NOT_EXIST	The files specified for upload cannot be located.
DOWNLOAD_FILES_DO_NOT_EXIST	The files specified for download cannot be located by the Signiant Flight Server.
NO_FOLDER_CHOSEN_FOR_DOWNLOAD	A download folder has not been specified.
INVALID_DESTINATION_DIRECTORY	The download folder specified cannot be located or is not accessible.
CONNECTION_FAILURE	A connection failure has occurred between the Signiant Transfer API and the Signiant Flight Server.
TRANSFER_ERROR_EVENT	Generic error event. The event message will detail the nature of the error.

Widget API

This reference documents the Signiant Transfer Widget API. The Widget API provides a set of simple and customizable widgets that can be easily integrated into a web site to enhance accelerated file transfers.

Signiant.Mst.DnDWidget

The Drag and Drop Upload Widget provides an interface that enables users to drag files or folders from the operating system and drop them onto a drop zone in an upload web page or to manually select files for upload. As items are dropped on the drop zone they are displayed as a list of files and folders ready for upload.

Constructor

Signiant.Mst.DnDWidget(targetDivName:String, transferObject:Object)

Creates a new Drag and Drop widget object.

Parameters

- **targetDivName:String** - The id of the `<div>` that will display the widget on the HTML page.
- **transferObject:Object** - The transfer object used to perform the transfer.

Methods

addDivToDOM(divName:String, parentElement:Object)

Embeds the widget in the target `<div>`.

Parameters

- **divName:String** - id of the `<div>` the widget is added to
- **parentElement:Object** - Parent element of the `<div>` the widget is added to

Returns

- **Object** - `<div>` containing the widget added to the DOM

addEvents()

Adds event handlers for each file in the file list and the file selector. The supported events include:

- **Click** to select a single file
- **Ctrl Click** to select multiple files
- **Ctrl Click** to de-select a file
- **Click** to de-select all selected files
- **Delete** (key) to clear all selected files
- **Clear** (x icon) to clear a single non-selected file
- **File Selector** icon to select files using the file selector dialog

Parameters

None.

Returns

None.

addNewPluginToDOM(pluginId:String, parentElement:Object)

Injects a new instance of the Signiant Transfer Plug-in to the drag and drop container.

Parameters

- **pluginId:String** - The id of the `<div>` containing the widget on the HTML page.
- **parentElement:Object** - Parent element of the `<div>` containing the widget.

Returns

- **Object** - `<div>` containing the widget added to the DOM

cancelDragLeaveTimer()

Resets the drag event timer.

Parameters

None.

Returns

None.

clearFile(file:String)

Removes a single file from the file list and updates the file list display.

Parameters

- **file:String** - Name of the file to remove from the list.

Returns

None.

displayFiles()

Creates the artifacts and file references for displaying the file list. This includes the container for the file list, file list items, and file selector.

Parameters

None.

Returns

None.

getDOMObject(domElementId:String)

Returns a reference of the target div DOM object.

Parameters

- **domElementId:String** - id of the object to return

Returns

- **Object** - DOM object identified by the id.

getFiles()

Returns the list of files in the transfer object.

Parameters

None.

Returns

- **Array** - The list of files in the transfer object.

getFilePathArray(files:Array)

Returns the list of files with the absolute path. This function is called from the `uploadFilesChosenCallback()` function.

Parameters

None.

Returns

- **Array** - The list of files in the transfer object with the absolute path.

getPluginDOM()

Returns the DOM object of the instance of the Signiant Transfer Plug-in assigned in the render function.

Parameters

None.

Returns

- **Object** - DOM object of the instance of the Signiant Transfer Plug-in.

hideDivContainer()

Resizes the drop zone container that embeds the instance of the Signiant Transfer Plug-in to hide it.

Parameters

None.

Returns

None.

hideDropZone()

Removes artifacts for the drag over event and restores the file list and the file selector.

Parameters

None.

Returns

None.

hideFullPath()

Displays just the file name for each file in the file list.

Parameters

None.

Returns

None.

onDragEnter()

Displays the drop zone and header. The method detects if the object being dragged is a file type. If the object is a file or folder, the `showDropZone()` function is called.

Parameters

None.

Returns

None.

onDragLeave()

Cancels timers associated to the drag event and calls the `hideDropZone()` function to hide the drop zone and restore the file list display.

Parameters

None.

Returns

None.

render()

Creates and defines the drag and drop container, an instance of the Signiant Transfer Plug-in, widget artifacts, and event handlers.

Parameters

None.

Returns

None.

selectFiles()

Renders the file selector dialog.

Parameters

None.

Returns

None.

showDropZone()

Expands the dimensions of the Signiant Transfer Plug-in instance to fill the drop zone during a drag enter event.

Parameters

None.

Returns

None.

showFullPath()

Displays the absolute path for each file in the file list.

Parameters

None.

Returns

None.

subscribeForEvent(name:String, func:Function)

Subscribes for events. The Drag and Drop Upload Widget supports the following events:

- uploadFilesChosen
- tooManyUploadFilesChosen
- dragEnter
- dragLeave

Parameters

- **name:String** - Name of the event to subscribe to.

- **func:Function** - Callback function to be called when the event occurs.

Returns

None.

uploadFilesChosenCallback(filesJSONString:JSON)

Accepts a JSON file list array provided from the Signiant Transfer Plug-in instance when files are added for upload.

The method parses and compares files as they are added and removes duplicate entries prior to setting and displaying the file list.

The format of the JSON array is:

```
[{
     "name":"file_0000.txt",
     "path":"U:\/files\/tree\/file_0000.txt",
     "isDirectory":"false","size":"1000"
 }]
```

Parameters

- **filesJSONString:JSON** - JSON object containing the chosen files.

Returns

None.

Signiant.Mst.DownloadWidget

Constructor

Signiant.Mst.DownloadWidget(targetDivName:String, transferObject:Object)

Creates a new Download widget object.

Parameters

- **targetDivName:String** - The id of the `<div>` that will display the widget on the HTML page.
- **transferObject:Object** - The transfer object used to perform the transfer.

Methods

addEvents()

Adds event handlers for each file in the file list and the folder selector. The events supported include:

- **Click** to select a single file
- **Ctrl Click** to select multiple files
- **Ctrl Click** to de-select a file
- **Click** to de-select all selected files
- **Spacebar** to clear a single selected file or all selected files
- **Checkbox** to clear a single non-selected file
- **Folder Selector** icon to select the download folder location using the folder selector dialog

Parameters

None.

Returns

None.

addFileToFileList(filePath:String)

Adds a file to the file list and calls the `updateFilesInTransferObject()` method to update the file list.

Parameters

- **filePath:String** - File to add to file list.

Returns

None.

addSpaceBarEventToFileList(file:String)

Evaluates each selected file and calls a method to add or remove the file from the file list (based on the class attribute) when the spacebar is pressed.

Parameters

- **file:String** - File to evaluate.

Returns

None.

displayFiles()

Creates a `<div>` container (file_list_containerDiv) to display the list of downloadable files. Attributes are applied to the `<div>` element and the method `parseAndDisplayFileList()` is called to parse the file list and apply event handlers.

Parameters

None.

Returns

None.

displayFolderSelector()

Creates the select folder `<div>` container and event handler.

Parameters

None.

Returns

None.

getSelectedFiles()

Returns the download file list array.

Parameters

None.

Returns

- **Array** - List of selected files.

folderChosenCallback(message:String, folder:String)

Callback function that sets the destination folder selected in the transfer object.

Parameters

- **message:String** - Error or success message.
- **folder:String** - Chosen folder.

Returns

None.

parseAndDisplayFileList()

Creates an inner `<div>` (filesDiv) within the file list container. The method then parses the file list, creates corresponding check boxes, and appends the files.

Parameters

None.

Returns

None.

removeFileInFileList(filePath:String)

Removes a file from the file list and calls the `updateFilesInTransferObject()` method to update the file list. The file name is passed to the function as a parameter value.

Parameters

- **filePath:String** - File to remove from the file list.

Returns

None.

render()

Applies a class attribute to the target div and calls the `displayFolderSelector()` method to display the widget artifacts.

Parameters

None.

Returns

None.

selectFolder()

Renders the folder selector dialog.

Parameters

None.

Returns

None.

setFiles(fileList:Array)

Evaluates a file list array passed to the function which it then sets in a global variable. The method clears the file list container and calls the function to display the files and update the transfer object.

Parameters

- **fileList:Array** - New file list to set.

Returns

None.

updateFilesInTransferObject()

Parses the file list to replace backslashes with forward slashes, clears and updates the file list in the transfer object.

Parameters

None.

Returns

None.

Signiant.Mst.TransferProgressWidget

Constructor

Signiant.Mst.TransferProgressWidget(targetDivName:String, transferObject:String)

Creates a new Transfer Progress widget object.

Parameters

- **targetDivName:String** - The id of the `<div>` that will display the widget on the HTML page.
- **transferObject:Object** - The transfer object used to perform the transfer.

Methods

displayProgressBar()

Displays the transfer progress bar.

Parameters

None.

Returns

None.

render()

Creates the widget artifacts and progress bar `<div>` container to render and display the transfer progress for an upload or download.

Parameters

None.

Returns

None.

showEstimatedCompletion(time:Boolean)

Determines whether to display the estimated time remaining for the transfer:

- **true** - the estimated time remaining for the transfer is displayed below the transfer progress bar.
- **false** - the estimated time remaining for the transfer is not displayed.

Parameters

- **time:Boolean** - Determines whether to display the estimated time remaining.

Returns

None.

showPercent(percent:Boolean)

Determines whether to display the percent complete for the transfer:

- **true** - the percent complete for the transfer is displayed in the transfer progress bar.
- **false** - the percent complete of the transfer is not displayed.

Parameters

- **time:Boolean** - Determines whether to display the percent complete.

Returns

None.

setProgress(progress:Number)

Calculates and renders the transfer progress within the progress bar element.

Parameters

- **progress:Number** - Percent complete of the transfer to display.

Returns

None.

transferProgressCallback(transferRef:Number, bytesSent:Number, totalBytes:Number, estimatedTimeRemaining:String)

Calculates the estimated time remaining based on the values provided by the Signiant Transfer Plug-in instance.

Parameters

- **transferRef:Number** - The transfer ID of the transfer.
- **bytesSent:Number** - The total bytes sent so far for the transfer.

- **totalBytes:Number** - The total bytes of all files in the transfer.
- **estimatedTimeRemaining:String** - The estimated time remaining for the transfer.

Returns

None.