JSProgressMonitor

Apr 04, 2024 04:03

Supported Clients

SmartClient WebClient NGClient

Methods Summary

Boolean	cancel()	Cancels the transfer process.
Number	getCurrentBytesToTransfer()	Returns the number of bytes to transfer for the current file.
Number	getCurrentFileIndex()	Returns the index of the current file being transferred.
Number	getCurrentTransferredBytes()	Returns the number of bytes already transferred for the current file.
String	getCurrentTransferredFileName()	Returns the name of the current file being transferred.
Number	getTotalBytesToTransfer()	Returns the total bytes to transfer to or from the server (sum of all the files size)
Number	getTotalFilesToTransfer()	Returns the total number of files to transfer.
Number	getTotalTransferredBytes()	Returns the total bytes already transferred (for all files)
Boolean	isCanceled()	Returns true if the process was canceled.
Boolean	isFinished()	Returns true if the process is finished.
JSProgressMonitor	setProgressCallBack(function, interval)	Sets a method to be called repeatedly at the given interval (in seconds), the method will receive an instance of this JSProgressMonitor updated with the latest values.
JSProgressMonitor	setProgressCallBack(function, interval, delay)	Sets a method to be called repeatedly at the given interval (in seconds), the method will receive an instance of this JSProgressMonitor updated with the latest values.

Methods Details

cancel()

Cancels the transfer process.

Returns

Boolean

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
monitor.cancel();
```

getCurrentBytesToTransfer()

Returns the number of bytes to transfer for the current file.

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

getCurrentFileIndex()

Returns the index of the current file being transferred.

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

getCurrentTransferredBytes()

Returns the number of bytes already transferred for the current file.

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

getCurrentTransferredFileName()

Returns the name of the current file being transferred.

Returns

String

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

getTotalBytesToTransfer()

Returns the total bytes to transfer to or from the server (sum of all the files size)

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

getTotalFilesToTransfer()

Returns the total number of files to transfer.

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

getTotalTransferredBytes()

Returns the total bytes already transferred (for all files)

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

isCanceled()

Returns true if the process was canceled.

Returns

Boolean

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

isFinished()

Returns true if the process is finished.

Returns

Boolean

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
application.output('total transferred: ' + monitor.getTotalTransferredBytes() + ' / ' + monitor.
getTotalBytesToTransfer());
application.output('current file: ' + monitor.getCurrentTransferredFileName() + ' ( ' + monitor.
getCurrentFileIndex() + ' / ' + monitor.getTotalFilesToTransfer() + ' )');
application.output('current bytes transferred: '+monitor.getCurrentTransferredBytes() + ' / ' + monitor.
getCurrentBytesToTransfer());
if (monitor.isCanceled()) {
        application.output('canceled!')
}
if (monitor.isFinished()) {
        application.output('finished!')
}
```

setProgressCallBack(function, interval)

Sets a method to be called repeatedly at the given interval (in seconds), the method will receive an instance of this JSProgressMonitor updated with the latest values. Can use an optional delay (for testing purpose in developer).

Parameters

Function function the Function to call back at the specified interval
Number interval the interval (in seconds) to use

Returns

JSProgressMonitor this for chaining

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
// call the progressCallbackFuntion every 2 and a half seconds (with a delay of 200ms in developer):
monitor.setProgressCallBack(progressCallbackFunction, 2.5, (application.isInDeveloper() ? 200 : 0));
```

setProgressCallBack(function, interval, delay)

Sets a method to be called repeatedly at the given interval (in seconds), the method will receive an instance of this JSProgressMonitor updated with the latest values. Can use an optional delay (for testing purpose in developer).

Parameters

Function function the Function to call back at the specified interval
Number interval the interval (in seconds) to use
Number delay adds a delay for testing purpose in Developer

Returns

JSProgressMonitor this for chaining

Supported Clients

SmartClient,WebClient,NGClient

Sample

```
// call the progressCallbackFuntion every 2 and a half seconds (with a delay of 200ms in developer):
monitor.setProgressCallBack(progressCallbackFunction, 2.5, (application.isInDeveloper() ? 200 : 0));
```