

# Launcher Javascript API

## 3.2.1.35

Generated by Doxygen 1.7.4

Fri Mar 16 2012 16:34:33



# Contents

<b>1</b>	<b>Launcher Javascript API</b>	<b>1</b>
1.1	Overview . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Class Documentation</b>	<b>7</b>
4.1	Launcher Class Reference . . . . .	7
4.1.1	Detailed Description . . . . .	7
4.1.2	Member Function Documentation . . . . .	8
4.1.2.1	fileExists . . . . .	8
4.1.2.2	findVisibleWindow . . . . .	8
4.1.2.3	getRegistryInt32 . . . . .	8
4.1.2.4	getRegistryString . . . . .	8
4.1.2.5	getVersion . . . . .	9
4.1.2.6	reboot . . . . .	9
4.1.2.7	registryKeyPathExists . . . . .	9
4.1.2.8	release . . . . .	9
4.1.2.9	setRegistryInt32 . . . . .	9
4.1.2.10	setRegistryString . . . . .	10
4.1.2.11	shellOpen . . . . .	10
4.1.3	Member Data Documentation . . . . .	10
4.1.3.1	instanceId . . . . .	10

4.2	Process Class Reference	11
4.2.1	Detailed Description	11
4.2.2	Member Function Documentation	11
4.2.2.1	getApplicationName	11
4.2.2.2	getArguments	12
4.2.2.3	getElevate	12
4.2.2.4	getSha1	12
4.2.2.5	getShowWindow	12
4.2.2.6	getWorkingDirectory	12
4.2.2.7	launch	12
4.2.2.8	release	13
4.2.2.9	setApplicationName	13
4.2.2.10	setArguments	13
4.2.2.11	setElevate	13
4.2.2.12	setSha1	13
4.2.2.13	setShowWindow	13
4.2.2.14	setWorkingDirectory	14
4.2.3	Member Data Documentation	14
4.2.3.1	instanceId	14
4.2.3.2	onComplete	14
4.2.3.3	onExit	14
4.3	TextFileReader Class Reference	14
4.3.1	Detailed Description	15
4.3.2	Member Function Documentation	15
4.3.2.1	getBufferSize	15
4.3.2.2	getFileName	15
4.3.2.3	release	15
4.3.2.4	setBufferSize	15
4.3.2.5	setFileName	16
4.3.2.6	start	16
4.3.3	Member Data Documentation	16
4.3.3.1	instanceId	16
4.3.3.2	onComplete	16
4.3.3.3	onProgress	16

4.3.3.4	onRead	16
4.3.3.5	onStart	16
4.4	TextFileWriter Class Reference	17
4.4.1	Detailed Description	17
4.4.2	Member Function Documentation	17
4.4.2.1	close	17
4.4.2.2	open	17
4.4.2.3	release	18
4.4.2.4	write	18
4.4.3	Member Data Documentation	18
4.4.3.1	instanceId	18
<b>5</b>	<b>File Documentation</b>	<b>19</b>
5.1	launcher.js File Reference	19
5.1.1	Detailed Description	19
5.1.2	Function Documentation	19
5.1.2.1	createLauncher	19
5.2	process.js File Reference	19
5.2.1	Detailed Description	20
5.2.2	Function Documentation	20
5.2.2.1	createProcess	20
5.3	textfilereader.js File Reference	20
5.3.1	Detailed Description	20
5.3.2	Function Documentation	20
5.3.2.1	createTextFileReader	20
5.4	textfilewriter.js File Reference	21
5.4.1	Detailed Description	21
5.4.2	Function Documentation	21
5.4.2.1	createTextFileWriter	21



## Chapter 1

# Launcher Javascript API

### 1.1 Overview

The launcher javascript API allows users to integrate with objects implemented using the launcher 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:

<a href="#">Launcher</a> (General <a href="#">Launcher</a> functionality ) . . . . .	<a href="#">7</a>
<a href="#">Process</a> ( <a href="#">Process</a> launcher ) . . . . .	<a href="#">11</a>
<a href="#">TextFileReader</a> (Text file reader ) . . . . .	<a href="#">14</a>
<a href="#">TextFileWriter</a> (Text file writer ) . . . . .	<a href="#">17</a>



## Chapter 3

# File Index

### 3.1 File List

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

<a href="#">launcher.js</a> (File containing <a href="#">Launcher</a> class and creation function ) . . . . .	19
<a href="#">process.js</a> (File containing <a href="#">Process</a> class and creation function ) . . . . .	19
<a href="#">textfilereader.js</a> (File containing <a href="#">TextFileReader</a> class and creation function ) .	20
<a href="#">textfilewriter.js</a> (File containing <a href="#">TextFileWriter</a> class and creation function ) . . .	21



## Chapter 4

# Class Documentation

### 4.1 Launcher Class Reference

General [Launcher](#) functionality.

#### Public Member Functions

- bool [fileExists](#) (string fileName)
- bool [findVisibleWindow](#) (string caption, string className)
- int [getRegistryInt32](#) (string type, string keyPath, string name, int defaultValue)
- string [getRegistryString](#) (string type, string keyPath, string name, string defaultValue)
- string [getVersion](#) (string fileName, string defaultValue)
- void [reboot](#) ()
- bool [registryKeyPathExists](#) (string type, string keyPath)
- void [release](#) ()
- bool [setRegistryInt32](#) (string type, string keyPath, string name, int value)
- bool [setRegistryString](#) (string type, string keyPath, string name, string value)
- bool [shellOpen](#) (string path)

#### Public Attributes

- string [instanceld](#)

#### 4.1.1 Detailed Description

General [Launcher](#) functionality.

### 4.1.2 Member Function Documentation

#### 4.1.2.1 bool Launcher.fileExists ( string *fileName* )

Check if a file exists

##### Parameters

<i>fileName</i>	filename to check
-----------------	-------------------

##### Returns

true if successful, false otherwise

#### 4.1.2.2 bool Launcher.findVisibleWindow ( string *caption*, string *className* )

Searches for a window that is visible

##### Parameters

<i>caption</i>	caption to match (supports wildcards)
<i>className</i>	name of class to match (supports wildcards)

##### Returns

true if window found, false otherwise

#### 4.1.2.3 int Launcher.getRegistryInt32 ( string *type*, string *keyPath*, string *name*, int *defaultValue* )

returns a registry value as an integer

##### Parameters

<i>type</i>	type of hive
<i>keyPath</i>	path of the key
<i>name</i>	name of value
<i>defaultValue</i>	default value returned if unable to read key

##### Returns

value of the key

#### 4.1.2.4 string Launcher.getRegistryString ( string *type*, string *keyPath*, string *name*, string *defaultValue* )

returns a registry value as a string

**Parameters**

<i>type</i>	type of hive
<i>keyPath</i>	path of the key
<i>name</i>	name of value
<i>defaultValue</i>	default value returned if unable to read key

**Returns**

value of the key

**4.1.2.5 string Launcher.getVersion ( string *fileName*, string *defaultValue* )**

returns the version of the filename

**Parameters**

<i>fileName</i>	name of the file to get the version of
<i>defaultValue</i>	default value returned if unable to read version

**Returns**

value of the version

**4.1.2.6 void Launcher.reboot ( )**

Reboots the machine

**4.1.2.7 bool Launcher.registryKeyPathExists ( string *type*, string *keyPath* )**

Check if a registry key path exists

**Parameters**

<i>type</i>	type of hive
<i>keyPath</i>	path of the key

**Returns**

true if exists, false otherwise

**4.1.2.8 void Launcher.release ( )**

releases the object

**4.1.2.9 bool Launcher.setRegistryInt32 ( string *type*, string *keyPath*, string *name*, int *value* )**

sets an integer in the registry

**Parameters**

<i>type</i>	type of hive
<i>keyPath</i>	path of the key
<i>name</i>	name of value
<i>value</i>	value of key

**Returns**

true if successful, false otherwise

4.1.2.10 `bool Launcher.setRegistryString ( string type, string keyPath, string name, string value )`

sets a string in the registry

**Parameters**

<i>type</i>	type of hive
<i>keyPath</i>	path of the key
<i>name</i>	name of value
<i>value</i>	value of key

**Returns**

true if successful, false otherwise

4.1.2.11 `bool Launcher.shellOpen ( string path )`

Launch the process

**Parameters**

<i>path</i>	string to url/filename to open
-------------	--------------------------------

**Returns**

true if successful, false otherwise

**4.1.3 Member Data Documentation**

4.1.3.1 `string Launcher.instanceId`

Instance id used to bind proxy object to native object

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

- [launcher.js](#)



## 4.2 Process Class Reference

[Process](#) launcher.

### Public Member Functions

- string [getApplicationName](#) ()
- string [getArguments](#) ()
- bool [getElevate](#) ()
- string [getSha1](#) ()
- bool [getShowWindow](#) ()
- string [getWorkingDirectory](#) ()
- bool [launch](#) ()
- void [release](#) ()
- void [setApplicationName](#) (string value)
- void [setArguments](#) (string value)
- void [setElevate](#) (bool value)
- void [setSha1](#) (string value)
- void [setShowWindow](#) (bool value)
- void [setWorkingDirectory](#) (string value)

### Public Attributes

- string [instanceId](#)
- event [onComplete](#)
- event [onExit](#)

#### 4.2.1 Detailed Description

[Process](#) launcher.

#### 4.2.2 Member Function Documentation

##### 4.2.2.1 string [Process.getApplicationName](#) ( )

gets the application to launch

#### Returns

application to launch

**4.2.2.2 string Process.getArguments ( )**

gets the arguments for the process

**Returns**

arguments for the process

**4.2.2.3 bool Process.getElevate ( )**

gets the process elevation

**Returns**

process elevation

**4.2.2.4 string Process.getSha1 ( )**

gets the sha1 value of application, blank to skip check

**Returns**

sha1 value of application, blank to skip check

**4.2.2.5 bool Process.getShowWindow ( )**

gets the process window visibility

**Returns**

process window visibility

**4.2.2.6 string Process.getWorkingDirectory ( )**

gets the full path to the current directory for the process

**Returns**

full path to the current directory for the process

**4.2.2.7 bool Process.launch ( )**

Launch the process

**Returns**

true if successful, false otherwise

**4.2.2.8 void Process.release ( )**

releases the object

**4.2.2.9 void Process.setApplicationName ( string *value* )**

sets the application to launch

**Parameters**

<i>value</i>	application to launch
--------------	-----------------------

**4.2.2.10 void Process.setArguments ( string *value* )**

sets the arguments for the process

**Parameters**

<i>value</i>	arguments for the process
--------------	---------------------------

**4.2.2.11 void Process.setElevate ( bool *value* )**

sets the process elevation

**Parameters**

<i>value</i>	process elevation
--------------	-------------------

**4.2.2.12 void Process.setSha1 ( string *value* )**

sets the sha1 value of application, blank to skip check

**Parameters**

<i>value</i>	sha1 value of application, blank to skip check
--------------	--

**4.2.2.13 void Process.setShowWindow ( bool *value* )**

sets the process window visibility

**Parameters**

<i>value</i>	process window visibility
--------------	---------------------------

#### 4.2.2.14 void Process.setWorkingDirectory ( string *value* )

sets the full path to the current directory for the process

##### Parameters

<i>value</i>	full path to the current directory for the process
--------------	--

### 4.2.3 Member Data Documentation

#### 4.2.3.1 string Process.instanceId

Instance id used to bind proxy object to native object

#### 4.2.3.2 event Process.onComplete

Triggered when the process is complete

##### Parameters

<i>successful</i>	true if the process was successful, false otherwise
-------------------	---

#### 4.2.3.3 event Process.onExit

Triggered when the process is exited

##### Parameters

<i>exitCode</i>	Application's exit code
-----------------	-------------------------

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

- [process.js](#)

## 4.3 TextFileReader Class Reference

Text file reader.

### Public Member Functions

- int [getBufferSize](#) ()
- string [getFileName](#) ()
- void [release](#) ()
- void [setBufferSize](#) (int value)
- void [setFileName](#) (string value)
- void [start](#) ()

## Public Attributes

- string [instanceId](#)
- event [onComplete](#)
- event [onProgress](#)
- event [onRead](#)
- event [onStart](#)

### 4.3.1 Detailed Description

Text file reader.

### 4.3.2 Member Function Documentation

#### 4.3.2.1 int TextFileReader.getBufferSize ( )

gets the determines the max size of the buffer being read into

#### Returns

Determines the max size of the buffer being read into

#### 4.3.2.2 string TextFileReader.getFileName ( )

gets the file name where url is saved

#### Returns

File name where url is saved

#### 4.3.2.3 void TextFileReader.release ( )

releases the object

#### 4.3.2.4 void TextFileReader.setBufferSize ( int *value* )

sets the determines the max size of the buffer being read into

#### Parameters

<i>value</i>	Determines the max size of the buffer being read into
--------------	---

#### 4.3.2.5 void `TextFileReader.setFileName ( string value )`

sets the file name where url is saved

##### Parameters

<i>value</i>	File name where url is saved
--------------	------------------------------

#### 4.3.2.6 void `TextFileReader.start ( )`

Start file reading

### 4.3.3 Member Data Documentation

#### 4.3.3.1 string `TextFileReader.instanceId`

Instance id used to bind proxy object to native object

#### 4.3.3.2 event `TextFileReader.onComplete`

Triggered when file reader is complete.

##### Parameters

<i>successful</i>	true if the file reader was successful, false otherwise.
-------------------	--

#### 4.3.3.3 event `TextFileReader.onProgress`

Triggered when file reader has made progress.

##### Parameters

<i>percent</i>	(-1.0: Still calculating) (0.0 to 1.0: Percent completed)
----------------	---

#### 4.3.3.4 event `TextFileReader.onRead`

Triggered when file reader reads new bytes.

##### Parameters

<i>text</i>	Bytes read from file.
-------------	-----------------------

#### 4.3.3.5 event `TextFileReader.onStart`

Triggered when file reader is started.

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

- [textfilereader.js](#)

## 4.4 TextFileWriter Class Reference

Text file writer.

### Public Member Functions

- bool [close](#) ()
- bool [open](#) (string fileName)
- void [release](#) ()
- int [write](#) (string value)

### Public Attributes

- string [instanceId](#)

### 4.4.1 Detailed Description

Text file writer.

### 4.4.2 Member Function Documentation

#### 4.4.2.1 bool TextFileWriter.close ( )

Close file

#### Returns

true if successful, false otherwise.

#### 4.4.2.2 bool TextFileWriter.open ( string *fileName* )

Open file for writing

#### Parameters

<i>fileName</i>	name of the file to close.
-----------------	----------------------------

#### Returns

true if successful, false otherwise.

#### 4.4.2.3 void TextFileWriter.release ( )

releases the object

#### 4.4.2.4 int TextFileWriter.write ( string *value* )

Write string to file

##### Parameters

<i>value</i>	string to write to file.
--------------	--------------------------

##### Returns

number of bytes written to file.

### 4.4.3 Member Data Documentation

#### 4.4.3.1 string TextFileWriter.instanceId

Instance id used to bind proxy object to native object

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

- [textfilewriter.js](#)



## Chapter 5

# File Documentation

### 5.1 launcher.js File Reference

File containing [Launcher](#) class and creation function.

#### Classes

- class [Launcher](#)  
*General [Launcher](#) functionality.*

#### Functions

- void [createLauncher](#) ()

#### 5.1.1 Detailed Description

File containing [Launcher](#) class and creation function.

#### 5.1.2 Function Documentation

##### 5.1.2.1 void createLauncher ( )

Create instance of launcher

### 5.2 process.js File Reference

File containing [Process](#) class and creation function.

## Classes

- class [Process](#)  
*Process launcher.*

## Functions

- void [createProcess](#) ()

### 5.2.1 Detailed Description

File containing [Process](#) class and creation function.

### 5.2.2 Function Documentation

#### 5.2.2.1 void createProcess ( )

Create instance of process

## 5.3 textfilereader.js File Reference

File containing [TextFileReader](#) class and creation function.

## Classes

- class [TextFileReader](#)  
*Text file reader.*

## Functions

- void [createTextFileReader](#) ()

### 5.3.1 Detailed Description

File containing [TextFileReader](#) class and creation function.

### 5.3.2 Function Documentation

#### 5.3.2.1 void createTextFileReader ( )

Create instance of textFileReader

## 5.4 textfilewriter.js File Reference

File containing [TextFileWriter](#) class and creation function.

### Classes

- class [TextFileWriter](#)  
*Text file writer.*

### Functions

- void [createTextFileWriter](#) ()

#### 5.4.1 Detailed Description

File containing [TextFileWriter](#) class and creation function.

#### 5.4.2 Function Documentation

##### 5.4.2.1 void [createTextFileWriter](#) ( )

Create instance of textFileWriter

# Index

- close
  - TextFileWriter, [17](#)
- createLauncher
  - launcher.js, [19](#)
- createProcess
  - process.js, [20](#)
- createTextFileReader
  - textfilereader.js, [20](#)
- createTextFileWriter
  - textfilewriter.js, [21](#)
- fileExists
  - Launcher, [8](#)
- findVisibleWindow
  - Launcher, [8](#)
- getApplicationName
  - Process, [11](#)
- getArguments
  - Process, [11](#)
- getBufferSize
  - TextFileReader, [15](#)
- getElevate
  - Process, [12](#)
- getFileName
  - TextFileReader, [15](#)
- getRegistryInt32
  - Launcher, [8](#)
- getRegistryString
  - Launcher, [8](#)
- getSha1
  - Process, [12](#)
- getShowWindow
  - Process, [12](#)
- getVersion
  - Launcher, [9](#)
- getWorkingDirectory
  - Process, [12](#)
- instanceId
  - Launcher, [10](#)
- Process, [14](#)
  - TextFileReader, [16](#)
  - TextFileWriter, [18](#)
- launch
  - Process, [12](#)
- Launcher, [7](#)
  - fileExists, [8](#)
  - findVisibleWindow, [8](#)
  - getRegistryInt32, [8](#)
  - getRegistryString, [8](#)
  - getVersion, [9](#)
  - instanceId, [10](#)
  - reboot, [9](#)
  - registryKeyPathExists, [9](#)
  - release, [9](#)
  - setRegistryInt32, [9](#)
  - setRegistryString, [10](#)
  - shellOpen, [10](#)
- launcher.js, [19](#)
  - createLauncher, [19](#)
- onComplete
  - Process, [14](#)
  - TextFileReader, [16](#)
- onExit
  - Process, [14](#)
- onProgress
  - TextFileReader, [16](#)
- onRead
  - TextFileReader, [16](#)
- onStart
  - TextFileReader, [16](#)
- open
  - TextFileWriter, [17](#)
- Process, [11](#)
  - getApplicationName, [11](#)
  - getArguments, [11](#)
  - getElevate, [12](#)
  - getSha1, [12](#)

- getShowWindow, [12](#)
- getWorkingDirectory, [12](#)
- instanceId, [14](#)
- launch, [12](#)
- onComplete, [14](#)
- onExit, [14](#)
- release, [12](#)
- setApplicationName, [13](#)
- setArguments, [13](#)
- setElevate, [13](#)
- setSha1, [13](#)
- setShowWindow, [13](#)
- setWorkingDirectory, [13](#)
- process.js, [19](#)
  - createProcess, [20](#)
- reboot
  - Launcher, [9](#)
- registryKeyPathExists
  - Launcher, [9](#)
- release
  - Launcher, [9](#)
  - Process, [12](#)
  - TextFileReader, [15](#)
  - TextFileWriter, [17](#)
- setApplicationName
  - Process, [13](#)
- setArguments
  - Process, [13](#)
- setBufferSize
  - TextFileReader, [15](#)
- setElevate
  - Process, [13](#)
- setFileName
  - TextFileReader, [15](#)
- setRegistryInt32
  - Launcher, [9](#)
- setRegistryString
  - Launcher, [10](#)
- setSha1
  - Process, [13](#)
- setShowWindow
  - Process, [13](#)
- setWorkingDirectory
  - Process, [13](#)
- shellOpen
  - Launcher, [10](#)
- start
  - TextFileReader, [16](#)
- TextFileReader, [14](#)
  - getBufferSize, [15](#)
  - getFileName, [15](#)
  - instanceId, [16](#)
  - onComplete, [16](#)
  - onProgress, [16](#)
  - onRead, [16](#)
  - onStart, [16](#)
  - release, [15](#)
  - setBufferSize, [15](#)
  - setFileName, [15](#)
  - start, [16](#)
- textfilereader.js, [20](#)
  - createTextFileReader, [20](#)
- TextFileWriter, [17](#)
  - close, [17](#)
  - instanceId, [18](#)
  - open, [17](#)
  - release, [17](#)
  - write, [18](#)
- textfilewriter.js, [21](#)
  - createTextFileWriter, [21](#)
- write
  - TextFileWriter, [18](#)