Config.as

Object
|---LoadVars
|---Config

Requires : stringUtils.as

View | Download

The Config object is an ActionScript object that loads name / value pairs from a config / ini file. This is useful for setting and loading default parameters for Flash based apps.

Usage :

```
#include "Config.as"

config = new Config("config.ini");
config.onConfigLoad = function(success)
{
	trace("Accessing data from onConfigLoad : ClassGuid = " + this.get("ClassGuid"));
}
config.loadConfig();
```

The config file loaded must be in a name, value format with names and values separate by an equal sign. Lines are separated by line returns. Lines that begin with '#', ';', '[', '/' are ignored by the Config object. The Config object should be able to work with any standard config file, including windows ini, inf file. Here is an example inf file that could be loaded into Config object.

```
; 1394.INF  -- This file contains descriptions of all the 1394
;              Host controllers supported in Windows NT and Memphis
;
;*** Created 07/09/97  (Creation Date)

[Version]
;Signature="$WINDOWS NT$"       ; JSG - What to do???
Signature="$CHICAGO$"
Class=1394
ClassGuid={6BDD1FC1-810F-11D0-BEC7-08002BE2092F}
Provider=%MSFT%
LayoutFile=layout.inf, layout1.inf
DriverVer=11/14/1999,5.00.2183.1

; ================== Class Sections ==========================

[ClassInstall32]
Addreg=1394ClassReg

[ClassInstall]
Addreg=1394ClassReg
```

Of course, you own config files may be more simple:

```
#config file for flash app

#this is the gateway url
gateWayURL = "http://localhost:8765/flashservices/gateway"

#this is the service name
serviceName = "myService"
```

You can check whether or not the Config.as file is available with the following code snippet:

```
if(Config.configDefined)
{
	//Config.as is included
}
else
{
	//Config.as could not be found
}
```

You can access the version number of the Config.as file like so:

```
var versionNumber = Config.configVersion;
trace(versionNumber)
```

Note that the Config object extends the LoadVars object, and thus any methods available in the LoadVars object will also be available to the Config object.

Useful methods inherited from the LoadVars object : getBytesLoaded(), getBytesTotal()

new Config(string)

This is the constructor for the config object, which can take an optional string parameter that points to the config file to be loaded.

The path to the config file can be a relative or absolute file or URL path. All Flash domain security restrictions apply.

Note that the constructor is placed in the _global scope and can thus be accessed from anywhere within the Flash movie.

loaded

This property is a boolean which indicates whether or not data has loaded into the config file. If loadConfig() or parse() are called, then it will be set to false, and then set to true if the config file is successfully parsed.

Note, that this property can be used with Object.watch().

```
c = new Config("config.ini");

if(!c.loaded)
{
        trace("The config file has not been loaded");
}
```

setFileName(string)

Method that allows you to set the path to the config file.

The path to the config file can be a relative or absolute file or URL path. All Flash domain security restrictions apply.

getFileName()

Returns a string of the path to the current config file.

If a config file has not been specified, then it returns null;

setStripQuotes(Boolean)

Method which takes a Boolean value that indicates whether
or not double quotes surrounding names and values should be removed.

Default is false.

For example, if you had the following config.ini file:

```
foo = "jar jar"
```

Here is with setStripQuotes called with true:

```
c = new Config("config.ini");

c.setStripQuotes(true);

c.onConfigLoad = function(success)
{
        trace(this.get("foo")); //traces jar jar
}
```

And with setStripQuotes passed false (or not called) :

```
c = new Config("config.ini");

c.setStripQuotes(false);

c.onConfigLoad = function(success)
{
        trace(this.get("foo")); //traces "jar jar"
}
```

The difference being that when setStripQuotes is set to true, the value does not contain the quotes.

Note : for performance consideration, you should only call setStripQuotes(true) if there is a possibility that your ini file with have quotes surrounding the data.

setExplodeValues(boolean)

Method which takes a Boolean value that indicates whether or not values that contain commas should be exploded into an array of values.

If set to true, calling get() on the name will return an Array.

If set to false, calling get() on the name will return a string.

The default is false.

For example if you had the following config file:

```
fruit = "apple,lemons,cherries"

c = new Config("config.ini");

c.setExplodeValues(false);

c.onConfigLoad = function(success)
{
        var t = this.get("fruits");
        trace("Value is String : " + (typeof(t) == "string")); //returns true
        trace("Value is Array : " + (t instanceof Array)); //returns false
}
```

	And with pass true to setExplodeValues :
	
	`c = new Config("config.ini");`
	
	`c.setExplodeValues(true);`
	
	`c.onConfigLoad = function(success)`
	`{`
	`  var t = this.get("fruits");`
	`  trace("Value is String : " + (typeof(t) == "string")); //returns false`
	`  trace("Value is Array : " + (t instanceof Array)); //returns true`
	`}`

parse(string)	This is a convenience method that allows you to pass a string representing a config file to be parsed.
	
	It returns true if the parsing succeed, and false if it did not.
	
	Note, since the method returns immediately, you may access the data in the object immediately after it has returned true.
	
	For example:
	
	`var c = new Config();`
	
	`var s = "foo=bar\n";`
	
	`if(c.parse(s))`
	`{`
	`  trace(c.get("foo")); //traces "bar"`
	`}`
	
	If the config object has already parsed data, then the new data will be added to the config object, overwriting any duplicate values that already exist.

get(string)	Takes a string and returns the value for that name in the config file.
	
	For example if you had the following config file:
	
	config.ini
	foo=bar
	name=mike
	
	`c = new Config("config.ini");`
	
	`c.onConfigLoad = function(success)`
	`{`
	`  trace(c.get("foo")); //traces "bar"`
	`  trace(c.get("name")); //traces "mike"`
	`}`

getArray()	Returns an associative array containing the name value pairs contained within the object.

loadConfig()	This method loads the config file specified in the constructor or setConfigFile() method, and then parses it.
	
	Once it has been parsed, the onConfigLoad() method will be called and passed a Boolean value indicating whether the file was able to be parsed.
	
	The onConfigLoad method should be over ridden in order to allow the developer to know when the data has loaded and been parsed.
	
	`#include "Config.as"`
	
	`config = new Config("config.ini");`
	`config.onConfigLoad = function(success)`
	`{`
	`  trace("Accessing data from onConfigLoad : ClassGuid = " + this.get("ClassGuid"));`
	`}`
	`config.loadConfig();`