



Search...

Search

- [Explore](#)
- [Gist](#)
- [Blog](#)
- [Help](#)

[klwilles](#)

-
-
-
- [Watch](#) [Unwatch](#)
- [Fork](#)
- - [29](#)
 - [3](#)

[inruntime](#) / [AS3-Global-Object](#)

- [Code](#)
- [Network](#)
- [Pull Requests 0](#)
- [Issues 0](#)
- [Stats & Graphs](#)

ActionScript

1. [ActionScript 100%](#)

AS3 Global Object is a Singleton that lets you store dynamic variables in a globally accessible location within your Actionscript 3 application. This enables developers to accomplish things like self registering visual components, global events and event listeners. — [Read more](#)

<http://www.uza.lt/codex/as3-global-object>

- [🍏 Clone in Mac](#)
- [ZIP](#)
- [HTTP](#)
- [Git Read-Only](#)

<https://github.com/inruntime/AS3-Global-Object>

Read-Only access

- [Tags 0](#)
- [Downloads 0](#)
- [branch: master](#)
Switch Branches/Tags
Filter branches/tags
 - [Branches](#)

- [Tags](#)

[master](#)


- [Files](#)
- [Commits](#)
- [Branches 1](#)

Latest commit to the **master** branch

[Added FlexGlobal -- a version of Global that ...](#)

allows Global variables to be bound to in Flex.

[commit c58e1ff9f3](#)

 Jonathan Dumaine authored a year ago [inruntime](#) committed a year ago

[AS3-Global-Object](#) /

name	age	history
source	a year ago	Added FlexGlobal -- a version of Global that [Jonathan Dumaine]
LICENSE.md	2 years ago	Added LICENSE.md and corrected license text inside source files [inruntime]
README.md	2 years ago	Allows turning weak references off as an option in the Global.getInst... [inruntime]
README.md		

AS3-Global-Object v2.1

Description

AS3 Global Object is a Singleton that lets you store dynamic variables in a globally accessible location within your Actionscript 3 application (also known as the missing `_global` property). This enables developers to accomplish things like self registering visual components, global events and event listeners.

Global Object was created and is maintained by Paulius Uza (<http://www.uza.lt>) and InRuntime Ltd. (<http://www.inruntime.com>). As of November 6, 2009 project is hosted on GitHub.

Examples

Recommended Syntax

```
// We highly recommend using dollar sign as a shorthand for accessing Global Object!
public var $:Global = Global.getInstance();
```

Weak References

AS3 Global Object uses weak references for storing data. This means that if the only pointer to your data is from within Global Object it is eligible for garbage collection. As of v2.1 you can turn this feature off by passing 'false' in the Global Object constructor:

```
public var $:Global = Global.getInstance(false);
```

Instantiating Global Object

Assuming: `Test.as`

```
package {
    import flash.display.*;
    import com.inruntime.utils.*;

    public class Test extends Sprite
    {
```

```

// initialize the global object
// you have to repeat this step in every class that will use the global
private var $:Global = Global.getInstance();

public function Test(){
    // your application code here
}
}
}

```

Wrong way to instantiate Global Object

Assuming: Test.as

```

package {
    import flash.display.*;
    import com.inruntime.utils.*;

    public class Test extends Sprite
    {
        // This is an example how you should NOT instantiate the Global Object
        // Global Object is a singleton and has to be initialized using .getInstance() function
        // Therefore the following will throw an exception:
        private var $:Global = new Global();
    }
}

```

Setting and getting dynamic variables

Assuming: Test.as

```

package {
    import flash.display.*;
    import com.inruntime.utils.*;

    public class Test extends Sprite
    {
        private var $:Global = Global.getInstance();
        private var testSprite:Sprite = new Sprite();

        public function Test(){
            //setting variables is easy, global object accepts any name / value pair, even functions
            $.stage = this.stage;
            $.testA = "a"
            $.testB = testSprite;
            $.testF = test;

            //getting variables is easy too
            trace($.testA);
            this.addChild($.testB);

            //as easy as calling a globally stored function
            $.testF();
        }

        private function test():void {
            trace("this is a global test");
        }
    }
}

```

Watching a global variable

Assuming: Test.as

```

package {
    import flash.display.Sprite;
    import com.inruntime.utils.Global;
    import com.inruntime.utils.GlobalEvent;
}

```

```

public class Test extends Sprite
{
    private var $:Global = Global.getInstance();
    public function Test()
    {
        // Let's listen for property changes on the global object
        $.addEventListener(GlobalEvent.PROPERTY_CHANGED,onPropChanged);

        // Event fires when you set or update a direct property of Global storage, let's try that
        $.variableA = 1;
        $.variableA = 23;
        $.variableA = 41;
        $.variableB = 20;
        $.variableB = $.variableA;

        var sp:Sprite = new Sprite();
        $.sp = sp;

        // However, event does not fire when you set a property of an object inside Global storage.
        $.sp.x = 10;
        $.sp.y = 23;
        sp.y = 34;
    }

    private function onPropChanged(e:GlobalEvent):void {
        // Property name can be accessed through GlobalEvent object;
        trace ("property " + e.property + " has changed to " + ${e.property});
    }
}

```

Using Global Object to access Stage from anywhere

Assuming: StageExample.as

```

package {
    import flash.display.Sprite;
    import com.inruntime.utils.*;

    public class StageExample extends Sprite
    {
        // Let's instantiate Global Object for the first time
        private var $:Global = Global.getInstance();

        public function StageExample()
        {
            // Let's put the stage reference into our newly instantiated Global Object:
            $.stage = this.stage;
            // Now let's initialize our child class
            var test:TestClass = new TestClass();
        }
    }
}

```

Assuming: TestClass.as

```

package {
    import com.inruntime.utils.*;

    public class TestClass {
        // Let's get the reference to our Global Object inside the test class
        private var $:Global = Global.getInstance();
        public function TestClass() {
            // The following line will trace '[object Stage]'
            trace($.stage);
        }
    }
}

```

GitHub Links

GitHub

- [About](#)
- [Blog](#)
- [Features](#)
- [Contact & Support](#)
- [Training](#)
- [GitHub Enterprise](#)
- [Site Status](#)

Tools

- [Gauges: Analyze web traffic](#)
- [Speaker Deck: Presentations](#)
- [Gist: Code snippets](#)
- [GitHub for Mac](#)
- [Issues for iPhone](#)
- [Job Board](#)

Extras

- [GitHub Shop](#)
- [The Octodex](#)

Documentation

- [GitHub Help](#)
- [Developer API](#)
- [GitHub Flavored Markdown](#)
- [GitHub Pages](#)
- [Terms of Service](#)
- [Privacy](#)
- [Security](#)

© 2012 GitHub Inc. All rights reserved.



Powered by the [Dedicated Servers](#) and [Cloud Computing](#) of Rackspace Hosting®

Markdown Cheat Sheet

Format Text

Headers

```
# This is an <h1> tag
## This is an <h2> tag
##### This is an <h6> tag
```

Text styles

```
*This text will be italic*
_This will also be italic_
**This text will be bold**
```

__This will also be bold__

*You **can** combine them*

Lists

Unordered

- * Item 1
- * Item 2
 - * Item 2a
 - * Item 2b

Ordered

1. Item 1
2. Item 2
3. Item 3
 - * Item 3a
 - * Item 3b

Miscellaneous

Images

![GitHub Logo](/images/logo.png)

Format: ![Alt Text](url)

Links

<http://github.com> - automatic!

[GitHub](http://github.com)

Blockquotes

As Kanye West said:

> We're living the future so
> the present is our past.

Code Examples in Markdown

Syntax highlighting with [GFM](#)

```
```javascript
function fancyAlert(arg) {
 if(arg) {
 $.facebox({div:'#foo'})
 }
}
```
```

Or, indent your code 4 spaces

Here is a Python code example
without syntax highlighting:

```
def foo:
    if not bar:
        return true
```

Inline code for comments

I think you should use an
`<addr>` element here instead.

Something went wrong with that request. Please try again. [Dismiss](#)

Looking for the GitHub logo?

- **GitHub Logo**



[Download](#)

- **The Octocat**



[Download](#)