

Introducing ActionScript 3.0

H. Paul Robertson
ActionScript Developer/Writer
Platform Developer Documentation
Adobe Systems



©2006 Adobe Systems Incorporated. All Rights Reserved.



ActionScript: What is it?

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 3.0 – What is it?

Programming language used in the Flash Player runtime

- Based on ECMA-262 editions 3 and 4 standards (ECMAScript)
- ActionScript 3.0 adds:
 - Numerous built-in object types; Flash Player specific (e.g. display objects, drawing, loading content)
 - Event model (based on W3C DOM3 Events specification)

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 3.0 complies fully with ECMA-262, 3rd edition. In terms of the 4th edition standard, which is still under development, it complies with the portions of the standard that were finalized in time, and will be updated to incorporate additional elements of the standard when it is completed.

ActionScript 3.0 – What is it?

Major Milestones in ActionScript history:

- Ability to do interactive programming: Flash 4 (1999)
- ActionScript (1): Flash 5 (2000)
- ActionScript 2: Flash MX 2004 (Flash Player 7): (2003)
- ActionScript 3: Flex 2/ "Blaze" (Flash CS3 Professional): (2006-7)

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 3.0 – What is it?

Design goals for ActionScript 3.0

- Improved, consistent programming model
- Compliance with industry standards
- Performance: 10x greater than ActionScript 2.0
- Easy to learn; easy to quickly develop Rich Internet Applications
 - Type safety – for writing unambiguous, easily maintainable code
 - Simplicity – intuitive syntax and object model
 - Compatibility – short migration path (from ActionScript 2.0 and other languages like JavaScript and Java)

Source: Grossman and Huang (2006) "ActionScript 3.0 overview."

ActionScript 3.0 – What is it?

Where is ActionScript 3.0 used?

- Any development tool that targets Flash Player runtime (9+)
- Currently:
 - Flex 2:
 - Free Flex 2 SDK
 - Flex Builder 2 (nice low academic price)
 - Flash Professional 9 ActionScript 3.0 Preview:
 - “alpha” version of Flash 9 – like Flash 8 with ActionScript 3.0 built in
 - Requires Flash Professional 8 license
 - Flash CS3 Professional (coming soon)

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 2.0 or ActionScript 3.0?

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 2.0 or ActionScript 3.0?

Which one should I use for my project? ActionScript 2.0 or ActionScript 3.0?

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 2.0 or ActionScript 3.0?

Which one should I use for my project? ActionScript 2.0 or ActionScript 3.0?

"it depends"

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 2.0 or ActionScript 3.0?

Which one should I use for my project? ActionScript 2.0 or ActionScript 3.0?

Reasons to use ActionScript 3.0:

- Better performance
- New functionality
- Easier to maintain

ActionScript 2.0 or ActionScript 3.0?

Which one should I use for my project? ActionScript 2.0 or ActionScript 3.0?

Reasons to use ActionScript 2.0:

- Target users only have Flash Player 8 or earlier
- Updating existing code/project
- Reusing existing code libraries
- Too busy to learn ActionScript 3.0
- Good news:
 - Performance improvements (specifically garbage collection improvements) in Flash Player 8+ even with ActionScript 2.0
 - ActionScript 2.0 is still being updated when it's possible (e.g. full screen)

©2007 Adobe Systems Incorporated. All Rights Reserved.



Of course, Flash Player has a lot of mechanisms for updating itself as seamlessly as possible, so the “old versions” argument isn’t as valid as it used to be.

Naturally, in order to use the new features such as full screen, users must have the appropriate Flash Player version installed (e.g. Player 9 update 1 for full screen). In that case you could still use ActionScript 2.0, although you would be targeting Flash Player 9.

ActionScript 2.0 or ActionScript 3.0?

Which one should I learn? ActionScript 2.0 or
ActionScript 3.0?

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 2.0 or ActionScript 3.0?

Which one should I learn? ActionScript 2.0 or
ActionScript 3.0?

ActionScript 3.0, of course!

©2007 Adobe Systems Incorporated. All Rights Reserved.



ActionScript 2.0 or ActionScript 3.0?

Which one should I learn? ActionScript 2.0 or ActionScript 3.0?

- Perception that 3.0 is “more complex” or “only appropriate for advanced programmers”
- The reality – ActionScript 3.0 is:
 - More consistent
 - Better organized
 - Designed to be easy to learn
 - Paul’s experience: They fixed all the issues that were hard to teach

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

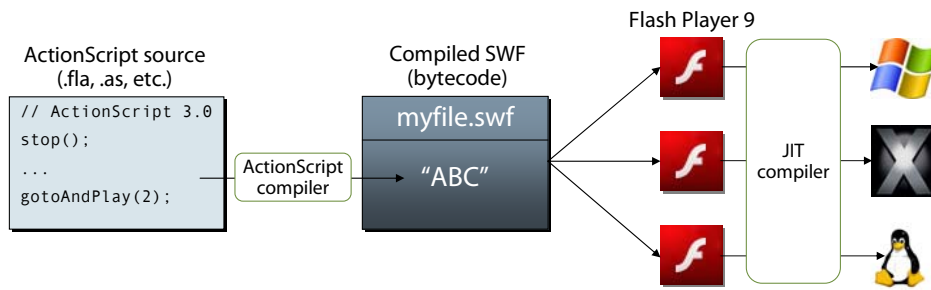
©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

Flash Player 9 features (benefits you get automatically)

- New Virtual Machine ("AVM2")
 - Faster
 - Better memory management/garbage collection
 - JIT compiled to native code



©2007 Adobe Systems Incorporated. All Rights Reserved.



JIT compiling stands for "Just-in-time" compiling.

1. You write your ActionScript code, and compile it using the ActionScript compiler, which turns it into a SWF file that contains the bytecode version of your program.

(Bytecode is like a programming language that's optimized for small size and for being understood by a computer rather than by a human programmer.)

2. When the SWF loads in Flash Player 9 on a particular platform, another compiler built into Flash Player compiles the bytecode, turning it into native operating system instructions (as though the program had been specifically compiled for the particular operating system). Because it's running as native code rather than being interpreted step-by-step by the virtual machine, it runs faster.

What's new in ActionScript 3.0?

Flash Player 9 features (benefits you get automatically)

- New Virtual Machine ("AVM2")
 - Data-type aware
 - Doesn't have to figure out data type at runtime
 - More error messages (this is a good thing)
 - Example: Calculator (subtraction) ActionScript 2.0 versus ActionScript 3.0

©2007 Adobe Systems Incorporated. All Rights Reserved.



In ActionScript 2.0, you could specify data types at compile time, but that information didn't actually persist into the SWF. Flash Player (which actually only uses ActionScript 1.0) would need to determine the data type of each object as the program was running.

In ActionScript 3.0, the runtime (Flash Player) knows the data type of each variable, so it's able to check that the appropriate data types are being passed around.

For a more in-depth discussion of this, see <http://probertson.com/articles/2005/11/08/actionscript-3-unit-testing-recommended/>

What's new in ActionScript 3.0?

Syntax is similar to ActionScript 2.0

- A simple timeline script may be similar or identical

```
stop();  
gotoAndPlay(2);  
// etc.
```

What's new in ActionScript 3.0?

Syntax is similar to ActionScript 2.0

- Functions, variable declarations, and operators are all identical

```
function sayHello(name:String):String
{
    var result:String;
    result = "Hello there, " + name + "!";
    return result;
}
```

What's new in ActionScript 3.0?

Syntax is similar to ActionScript 2.0

- Class syntax is very similar

```
// ActionScript 2.0
public class mypackage.MyClass
{
    public function MyClass()
    {
    }

    public function someMethod():Void
    {
    }

    public var someProperty:String;
}
```

```
// ActionScript 3.0
package mypackage
{
    public class MyClass
    {
        public function MyClass()
        {
        }

        public function someMethod():void
        {
        }

        public var someProperty:String;
    }
}
```

©2007 Adobe Systems Incorporated. All Rights Reserved.



Notice the only difference is the way a package is specified:

- In ActionScript 2.0 the package name is prepended to the class name
- In ActionScript 3.0 the package is declared using a separate package block that surrounds the class block

The only other difference is the capitalization of Void/void

What's new in ActionScript 3.0?

Class hierarchy ("object model") reorganized

- Everything that isn't part of core ECMAScript is in a package
 - MovieClip --> flash.display.MovieClip
 - TextField --> flash.text.TextField
 - TextField.stylesheet --> flash.text.StyleSheet
 - flash.filters.DropShadowFilter --> flash.filters.DropShadowFilter
- Objects are more organized (grouped according to similar functionality or purpose)
- More future-proof
 - e.g. Player class

©2007 Adobe Systems Incorporated. All Rights Reserved.



It used to be that a lot of classes were in the "top-level" package – classes such as MovieClip, TextField, and most of the built-in classes. Now nearly all of them are organized into a package to separate them from potential conflicting names with user-defined classes and to make them more organized.

Notice that all the classes that were added in Flash 8 (e.g. flash.filters.*, flash.display.BitmapData) already conform to this standard.

What's new in ActionScript 3.0?

Class hierarchy ("object model") reorganized

- Good news:
Once you add the (required) import statement, code can be the same

```
// ActionScript 2.0  
var childClip:MovieClip;  
function resize(tf:TextField):Void  
{  
    ...  
}
```

```
// ActionScript 3.0  
import flash.display.MovieClip;  
import flash.text.TextField;  
var childClip:MovieClip;  
function resize(tf:TextField):void  
{  
    ...  
}
```

- (and import statements are automatic in Flash for built-in classes...but better to not get used to that!)

©2007 Adobe Systems Incorporated. All Rights Reserved.



Notice that aside from the import statements, the ActionScript 2.0 code is nearly identical to the ActionScript 3.0 code.

What's new in ActionScript 3.0?

Class elements (methods/properties) renamed

- Consistent naming conventions

No more "Is it '_x'? Or just 'x'?"

```
// ActionScript 2.0  
myMovieClip._x = 250;  
myMovieClip.cacheAsBitmap = true;  
myComboBox.x = 400;
```

```
// ActionScript 3.0  
myMovieClip.x = 250;  
myMovieClip.cacheAsBitmap = true;  
myComboBox.x = 400;
```

©2007 Adobe Systems Incorporated. All Rights Reserved.



It used to be that in order to know whether an object's property had an underscore in the name or not, you had to know the history of when the property was added to ActionScript. Now it's all consistent – underscores are gone.

What's new in ActionScript 3.0?

New "enumeration" classes for sets of (mutually exclusive) values

- No more need to remember all the options

```
// ActionScript 2.0  
Stage.scaleMode = "noScale";  
...  
Stage.scaleMode = "showAll";
```

```
// ActionScript 3.0  
stage.scaleMode = StageScaleMode.NO_SCALE;  
...  
stage.scaleMode = StageScaleMode.SHOW_ALL;
```

©2007 Adobe Systems Incorporated. All Rights Reserved.



In ActionScript 2.0, you would need to remember (or look up) the possible values for several properties like `Stage.scaleMode`.

In ActionScript 3.0, there are many new classes whose sole purpose is to provide the possible values as constants. You may question why this is a benefit, since it means you'll need to remember the class and property names – but with modern code editors (including Flex Builder 2, Flash, and many others) you can get code hints that will tell you the possible options, meaning you don't have to remember.

And, under the hood, those constants that are defined in the enumeration classes really have the same String (or Number or whatever) values as the original values – so if you're migrating code from ActionScript 2.0, you won't have to worry about replacing the String values with constants.

What's new in ActionScript 3.0?

Objects reorganized to remove redundancy

- Duplicate functionality extracted into separate class (sometimes base class)
 - e.g. Loading text/XML/variables

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

Objects reorganized to remove redundancy

- Duplicate functionality extracted into separate class (sometimes base class)
 - ActionScript 2.0: each class has its own methods and properties for identical functionality: adding variables, sending data, loading data

```
// ActionScript 2.0
var result:LoadVars = new LoadVars();
result.load("update.php");
// after loading, "result" is populated with loaded variables
...
var myXML:XML = new XML();
myXML.contentType = "text/xml";
myXML.load("update.xml");
// after loading, myXML is populated with the loaded XML
```

What's new in ActionScript 3.0?

Objects reorganized to remove redundancy

- Duplicate functionality extracted into separate class (sometimes base class)
 - ActionScript 3.0: duplicate functions are separated into other classes: URLRequest, URLVariables, URLLoader

```
// ActionScript 3.0
var request:URLRequest = new URLRequest("update.php");
var loader:URLLoader = new URLLoader(request);
loader.dataFormat = URLLoaderDataFormat.VARIABLES;
loader.load();
// after loading
var result:URLVariables = loader.data;
...
var request:URLRequest = new URLRequest("update.xml");
request.contentType = "text/xml";
var loader:URLLoader = new URLLoader(request);
loader.dataFormat = URLLoaderDataFormat.TEXT;
loader.load();
// after loading
var myXML:XML = new XML(loader.data);
```

©2007 Adobe Systems Incorporated. All Rights Reserved.



Notice that in ActionScript 3.0, the functionality of loading data and communicating with a server is no longer part of the LoadVars, XML, and other classes. Since the functionality isn't related to XML (or variables) per se, it makes sense to just have separate classes that provide that functionality, and can be used by all the other classes.

In addition, the URLLoader provides several events, such as start, progress, and complete, so you can track your data loading as it progresses rather than only knowing when loading starts and ends.

What's new in ActionScript 3.0?

New Event model

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

New Event model

- ActionScript 2.0 uses 4 different approaches

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

New Event model

- ActionScript 2.0 uses 4 different approaches

```
// Event handling code "attached" to objects  
on (release)  
{  
    // do something  
}
```

What's new in ActionScript 3.0?

New Event model

- ActionScript 2.0 uses 4 different approaches

```
// Event handling code "attached" to objects  
(function () {
```

```
    // Event handler properties  
    myButton.onRelease = function ()  
    {  
        // do something  
    }  
};
```

What's new in ActionScript 3.0?

New Event model

- ActionScript 2.0 uses 4 different approaches

```
// Event handling code "attached" to objects
```

```
// Event handler properties
```

```
// Event listeners  
var stageListener:Object = new Object();  
stageListener.onResize = function ()  
{  
    // do something  
};  
Stage.addListener(stageListener);
```


What's new in ActionScript 3.0?

New Event model

- ActionScript 2.0 uses 4 different approaches

```
// Event handling code "attached" to objects
```

```
// Event handler properties
```

```
// Event listeners
```

```
// Component events  
var listener:Object = new Object();  
listener.onChange = function (event:Object)  
{  
    // do something  
};  
myComboBox.addEventListener("change", listener);
```

What's new in ActionScript 3.0?

New Event model

- ActionScript 2.0 uses 4 different approaches

```
// Event handling code "attached" to objects
```

```
// Event handler properties
```

```
// Event listeners
```

```
// Component events  
var listener:Object = new Object();  
listener.onChange = function (event:Object)  
{  
    // do something
```

- And developers often wrote their own!

What's new in ActionScript 3.0?

New Event model

- ActionScript 3.0...

```
// Event handling code "attached" to objects  
// Event handler properties  
// Event listeners  
  
// Component events  
var listener:Object = new Object();  
listener.onChange = function (event:Object)  
{  
    // do something  
};  
myComboBox.addEventListener("change", listener);
```

What's new in ActionScript 3.0?

New Event model

- ActionScript 3.0...
Gets rid of code "attached" to objects

```
// Event handler properties  
// Event listeners  
// Component events  
var listener:Object = new Object();  
listener.onChange = function (event:Object)  
{  
    // do something  
};  
myComboBox.addEventListener("change", listener);
```

What's new in ActionScript 3.0?

New Event model

- ActionScript 3.0...
Replaces the other various approaches with a single, built-in event-handling framework

```
// Event handler properties  
  
// ActionScript 3.0 Event listeners  
function playMovie(event:MouseEvent):void  
{  
    // do something  
}  
myButton.addEventListener(MouseEvent.CLICK, playMovie);  
myComboBox.addEventListener(Change, listener);
```

What's new in ActionScript 3.0?

New Event model

- ActionScript 3.0...
Replaces the other various approaches with a single, consistent, built-in event-handling framework
- (demo: click to play)
- (demo: click to navigate)
- (demo: detecting typing)

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

Display List

- A new way of managing visual content

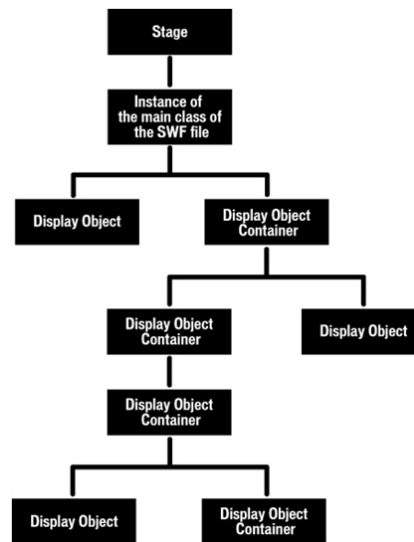
©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

Display list: What is it?

- Hierarchical tree of content elements (“display objects”)
 - **Stage**
 - **Main class instance** (the “Main Timeline”)
 - **Display objects**
 - **Display object containers**



©2007 Adobe Systems Incorporated. All Rights Reserved.



Hierarchical tree of content elements (“display objects”)

Stage: the outermost element that contains all the others

Main class instance: the display object that is immediately contained by the Stage (the “Main Timeline”). Now known as the “Document class”.

Display objects: objects that represent visual content of some type

Display object containers: display objects that can also hold child content in their own “child list”

What's new in ActionScript 3.0?

Display list advantages

- New classes for better performance:
 - **Sprite**: when you want a display object container like a MovieClip, but don't need a Timeline
 - **Shape**: when you just need a display object (not a container) to use as a "canvas" for drawing

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

Display list advantages

- MUCH better depth management
 - No need to keep track of depth numbers
 - No more "getNextHighestDepth()"
 - No more worrying that you'll overwrite existing content
 - (demo: "add children")

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

Display list advantages

- ActionScript 2.0: factory methods for screen objects
 - `createEmptyMovieClip()` – creates a new, blank movie clip
 - `createTextField()` – creates a new, empty text field
- ActionScript 3.0: construct an instance like any object
 - `new MovieClip()` – creates a new, blank movie clip
 - `new Sprite()` – creates a new, blank sprite
 - `new TextField()` – creates a new, empty text field
- (then call `addChild()` to add to the display list)

©2007 Adobe Systems Incorporated. All Rights Reserved.



In ActionScript 3.0, you can create display objects using standard constructor methods rather than factory methods. When you create an instance then you can use one of the `addChild()` methods to add it to the display list. You don't have to add it immediately – you can have “offlist” display objects; you can remove display objects from the list without discarding them; and you can “reparent” display objects, removing them from one container's child list and moving them to another container.

What's new in ActionScript 3.0?

Display list advantages

- ActionScript 2.0: factory methods for screen objects
 - `attachMovie()` – attaches a predefined movie clip from the library
- ActionScript 3.0: give library symbols a class name, and construct an instance like any object
 - `new MySymbol()` – creates an instance of a predefined movie clip from the library
 - (demo: remove children)

©2007 Adobe Systems Incorporated. All Rights Reserved.



In ActionScript 3.0, you can specify a class name for a library symbol. If you want to specify custom functionality in that class, you can; otherwise it will create an empty class associated with that library symbol, so you can create new instances using `new MySymbolClass()` syntax rather than needing `attachMovie()`

What's new in ActionScript 3.0?

- Better XML handling (E4X)
 - E4X – the way I expected XML handling to work in the first place

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

- Better XML handling

- Example XML:

```
<?xml version="1.0"?>
<playlist>
  <name>Classical</name>
  <owner>Paul</owner>
  <creationDate>2006-09-01</creationDate>
  <songs>
    <song>
      <title>Canon in D</title>
      <composer>Pachelbel</composer>
    </song>
    <song>
      <title>Nessun dorma</title>
      <composer>Puccini</composer>
    </song>
  </songs>
</playlist>
```

©2007 Adobe Systems Incorporated. All Rights Reserved.



Consider this sample XML document

- Better XML handling

<pre><?xml version="1.0"?> <playlist> <name>Classical</name> <owner>Paul</owner> <creationDate>2000</creationDate> <songs> <song> <title>Candide</title> <composer>Poulenc</composer> </song> <song> <title>Nessun Dorma</title> <composer>Puccini</composer> </song> </songs> </playlist></pre>	<pre>// XML DOM approach // (assume our XML data is in an object named "myXML") // get the playlist owner var rootNode:XMLNode = myXML.firstChild; var nameNode:XMLNode = rootNode.firstChild; var ownerNode:XMLNode = nameNode.firstSibling; var ownerName:String = ownerNode.firstChild.nodeValue; // get the first song's title var songsNode:XMLNode = rootNode.childNodes[3]; var song1Node:XMLNode = songsNode.childNodes[0]; var titleNode:XMLNode = song1Node.firstChild; var title:String = titleNode.firstChild.nodeValue;</pre>
--	--



What's new in ActionScript 3.0?

- Better XML handling

- E4X approach:

```
<?xml version="1.0"?>
<playlist>
  <name>Classical</name>
  <owner>Paul</owner>
  <creationDate>2000</creationDate>
  <songs>
    <song>
      <title>Candide</title>
      <composer>P. J. Couperin</composer>
    </song>
    <song>
      <title>Nessun Dorma</title>
      <composer>P. J. Couperin</composer>
    </song>
  </songs>
</playlist>
```

```
// E4X approach
// (assume our XML data is in an object named "myXML")

// get the playlist owner
var ownerNode:XMLList = myXML.owner;
var ownerName:String = ownerNode.toString();

// get the first song's title
var songsList:XMLList = myXML.songs.song;
var titleNode:XMLList = songsList[0].title;
var title:String = titleNode.toString();
// OR
var title:String = myXML..title[0].toString();
```

©2007 Adobe Systems Incorporated. All Rights Reserved.



With E4X, you just use node names like property names, so you can “dot down” to a particular node quickly and easily.

There are two objects you'll mainly use: XML represents a whole XML document (in this case it represents the <playlist> node); XMLList represents an XML fragment (e.g. a single child node from an XML document) or a set of nodes.

Rather than treating a plain text value as a separate node, you can just use the toString() method on a node that only contains a text value, and it will return only the value without the XML tag wrapper.

If the property (node) you specify contains multiple sibling tags (e.g. the multiple <song> tags in the <songs> tag), the XMLList object will contain multiple nodes, which can be accessed using the array access operator [].

You can also use the “double dot” descend accessor operator. This operator will scan through the child nodes of an XML object and return a set of all the XML nodes with a name matching the name to the right of the operator. In this case, every <title> node in myXML is returned by the operator. You can treat this result as an XMLList, using [] to access particular nodes etc.

What's new in ActionScript 3.0?

- Low-level and improved data manipulation
 - URLStream
 - ByteArray
 - Binary sockets
 - Dictionary
 - Regular Expressions
 - SoundChannel and SoundMixer
 - TextField: `getCharIndexAtPoint()`, `getFirstCharInParagraph()`, `getLineLength()`, `getLineText()`

©2007 Adobe Systems Incorporated. All Rights Reserved.



What's new in ActionScript 3.0?

More details and descriptions

- [“What's new in ActionScript 3.0”](#) from *Programming ActionScript 3.0*
- [“ActionScript 2.0 Migration”](#) in the *ActionScript 3.0 Language Reference*

Looking forward

©2007 Adobe Systems Incorporated. All Rights Reserved.



Looking forward

“Tamarin” open-source AVM

- Adobe has released source code for AVM2 to the Mozilla Foundation
 - This code will be integrated into a next-generation JavaScript engine in Firefox (probably Firefox 4.0)
 - Firefox gets all the nice benefits of AVM2
 - Adobe gets extra eyes to find bugs and write and review code
 - Adobe gets good “karma” from sharing code
 - ActionScript gets good exposure – “the next-generation JavaScript engine is the current-generation ActionScript engine”
- Nice executive summary by Frank Hecker (Mozilla Foundation):
<http://www.hecker.org/mozilla/adobe-mozilla-and-tamarin>

©2007 Adobe Systems Incorporated. All Rights Reserved.



Looking forward

Final ECMAScript edition 4 specification

- Even more language features still under consideration:
 - Type parameters (similar to generics)
 - Nullable and non-nullable types
 - Block-scoped variables
 - Operator overloading
 - ...and much, much more!
- Committee Wiki (read-only public version):
<http://developer.mozilla.org/es4/>
- Mailing list:
<https://mail.mozilla.org/listinfo/es4-discuss>

©2007 Adobe Systems Incorporated. All Rights Reserved.



You can see the proposed additions to the ECMAScript 4th edition specification, which will tell you what language features will be added into ActionScript once the specification is finalized.

Thanks

H. Paul Robertson

ActionScript Developer/Writer

<http://probertson.com/>

THANKS!

©2007 Adobe Systems Incorporated. All Rights Reserved.



Better by Adobe.™

©2007 Adobe Systems Incorporated. All Rights Reserved.



References/Resources

- Flex 2: <http://www.adobe.com/products/flex/>
- Flash Professional 9 ActionScript 3.0 Preview: <http://labs.adobe.com/technologies/flash9as3preview/>
- ActionScript Technology Center: <http://www.adobe.com/devnet/actionscript/>
- "ActionScript 3.0 overview": http://www.adobe.com/devnet/actionscript/articles/actionscript3_overview.html
- "Programming ActionScript 3.0": <http://www.adobe.com/go/programmingAS3>