

Playola World of Color

**Fully Customizable
Adobe® Flash® Video Player**

**For Flash Developers
Flash 8 Professional or Later**

**User and Customization Guide
Version 1.0.5a**

From
**Buenacreek Consulting
The "Web Video Made Simple" Guys!**

Table of Contents

Introducing the Playola World of Color Video Player!	3
Easy to Customize.....	3
Where to Go For More Help.....	3
Key Player Features	4
Supported Video Formats	5
Playola World of Color Package Files.....	5
Sample Files.....	5
Main Source Code Files.....	6
Auxiliary Source Code Files.....	6
Running the Sample Files	8
Changing Player Behavior Using Flashvars	8
Using the videoURL Flashvar	9
Using the posterFrame Flashvar	9

Using the autoStart Flashvar	10
Using the preloadVideo Flashvar.....	10
Using the initialVolume Flashvar	11
Setting Main Customizations.....	11
Preparing to Customize the Player.....	12
Basic Customizing and Compiling Steps.....	12
Customizing - As Easy As 1-2-3!	13
Setting Customizing Variables	13
Setting File Location Variables	14
Setting Flashvar Default Behavior Variables	14
Setting Logo Variables	15
Setting Custom Menu Variables	16
Setting Control Panel Variables.....	17
Setting Control Panel Colors	17
Setting Secondary Customizations	20
Secondary General Settings	20
Secondary Control Panel Settings.....	22
Secondary Big Play Button Settings	23
Secondary Throbber (Animated Wait Circle) Settings.....	25
Secondary Advanced Settings.....	25
Control Panel Elements X Offsets	26
Hiding File Assets.....	28
Overriding Build-in Paths	30
Filename Security Notes	30
Adding Characters to the Whitelist.....	31
Bypassing the Security Checks.....	31
Implementation Notes.....	32
Embedding Your Player In an HTML Page	32
Legal Notices	34

Introducing the Playola World of Color Video Player!

At last, match your video player to your Web site color scheme and personality!

Playola World of Color is a flexible and powerful video player designed to make customizing super easy.

Quickly make players that color-coordinate to match any occasion, holiday, or event - Halloween, Christmas, weddings, school colors for graduation, you name it. Precisely match the colors of the player to the colors of your Web site.

Easy to Customize

Playola World of Color is written in Actionscript 2.0, and may be modified using the Flash 8 Professional authoring program, or later.

Your *World of Color* video players can be used by anyone with Flash Player 8 or later. You can use any video format supported by Flash Player. This includes FLV and H.264.

Why is it called "*World of Color*"? Because you can modify the player using an unlimited palette of colors to match your Web site! You're not limited to some basic color scheme someone else thought up.

What's more, the code behind *Playola World of Color* is designed to make customizations super-easy! Over 50 appearance and player option settings can be controlled by editing just one file.

The *World of Color* player is so easy to customize, so you can quickly and effortlessly create new players for any occasion.

Your customized choices are built right into the player. That means you don't have to worry about complicated HTML embedding codes. *World of Color* players are simple to deploy.

Because you don't need to rely on complex scripting, they'll work on any blog, forum, or site where you can embed a Flash object. This includes Facebook and MySpace.

Where to Go For More Help

Be sure to visit us at our Website:

www.buenacreek.com

for helpful FREE tips and tools for getting the most out of Flash video!

Key Player Features

Okay, so here are the main features of the *Playola World of Color* video player. These and other features are fully documented in this Guide.

- **Source code is designed for easy customizing.** Rather than make you hunt through pages of Actionscript code, you *need only modify one file* to change over four dozen customizing features.
- **Resizable, any aspect ratio.** Make your player any height or width - 320x240, 400x300, 512x288, you name it. The player adjusts the scaling of its components to accommodate any video size, including Fullscreen mode.
- **Control panel automatically hides itself.** Choose the hiding effect (fade, slide, or pop) and time delay. Move the mouse, and the control panel immediately pops into view. Or, disable this feature to keep the control panel always in view.
- **Hide or show control panel elements.** You can easily create a "controllerless" player, or one that allows only basic playback with the Play/Pause button and volume control. Users can't skip to another part of the video.
- **Flashvars for setting media and main playback options.** Customize a player for your site, then use it to play an *unlimited number of videos* by using Flashvars.
- **Or, build in all options into the player.** You can alternatively create fully-custom single-video players that contain all the play options, including the video to play. No Flashvars at all!
- **Customizable "Big Play" button.** Choose from among several pre-made *Big Play* buttons that appear when the video is paused. You can set the size, color, and animation effects. For advanced users, add your own Big Play button graphic in your Flash authoring program.
- **The location of your video files may be hidden from site scrapers.** You can store a hidden path or URL inside the player code, and then only provide the filename of the video in HTML. Page scrapers are unable to locate the video file just from reading your page. Save your bandwidth from leechers!
- **Display an optional clickable logo.** Provide a graphic and tell the player where on the screen you wish it to appear. Set up a clickable link, and the player will open a window to the URL you provide. When using a PNG graphic you can even set the transparency of your logo.
- **Display an optional poster image.** You can have your player display a static image (called a poster or "splash") when it first loads. The image dissolves away

when the user clicks the Play button. Used in conjunction with the preload feature, you can save bandwidth by not loading the video file until the user clicks Play. They see the poster image of your choice inside the player, and know to click on the player to start it.

- **Display optional right-click menu link.** Further brand your player with a right-click menu link. The player opens the Web page of your choice.
- **Improved security validation.** URLs are processed through data validation routines to help prevent possible security abuse, such as a rogue user attempting to use a script scheme (javascript: or mailto:) as part of the URL.

Supported Video Formats

Playola World of Color supports the following video formats, given the appropriate minimum Flash Player version.

Video Format	Minimum Flash Player Version
FLV - Sorenson Spark	Flash 6
FLV - VP6	Flash 8
H.264	Flash 9, Update 3 (also referred to as v9r115)

Note! MOV, MP4, F4V, and M4V are essentially container formats for H.264 videos. Adobe Flash Player can generally play videos in these container formats, as long as **1)** the videos use appropriate metadata, and **2)** the Flash player is version 9r115 or later.

Though Flash 6 is the first version of Flash Player to support FLV videos, your users will need Flash 8 or later. You *can* optionally compile your player to target an earlier version of Flash Player, but this is not recommended due to the many security issues with that version. Encourage your users to update their version of Adobe Flash Player.

World of Color players are not intended to play audio-only files, nor are they meant to be used with streaming or pseudo-streaming servers, such as Flash Media Server. They may be used for any *progressive* download video files, which applies to the vast majority of video website applications.

Playola World of Color Package Files

The source code package for *Playola World of Color* contains the following files.

Sample Files

Note: These files are intended to be in the same directory.

- *flv-woc.swf* - Sample pre-compiled player, with *sample logo and menu branding*. This player may be tested locally on your PC, or on a Web site.
- *flv-woc.html* - Sample Web page, embedding *flv-woc.swf*.
- *sample-video.flv* - Sample video for use with *flv-woc.swf*.
- *sample-logo.png* - Sample logo for use with *flv-woc.swf*.
- *sample-poster.jpg* - Sample poster ("splash") image for use with *flv-woc.swf*.

Note! Also included is *flv-woc-unbranded.swf*. This is an unbranded version of the *flv-woc.swf* sample player. It's the same, but doesn't include logo or menu customizations for branding. You can use this player out-of-the-box, if you wish.

Main Source Code Files

The principle player source code is contained in two files, an editable Flash FLA document, and a separate text file containing Actionscript code. You can modify the Actionscript code file (and all the other *.as* files in this project) using a text editor, but we suggest the SEIPY Actionscript editor (www.sephiroth.it/python/sepy.php).

Instructions for customizing are provided in the sections below.

Unless you wish to modify the functionality of the player itself, you will only need to change the *player.as* file. Leave the FLA file alone, unless you wish to change graphics.

- *source\flv-woc fla* - Main Adobe Flash authoring source document. May be opened using Flash 8 or later. **Note!** The rest of the functional Actionscript code for the player is contained in additional **.as* files below.
- *source\player.as* - Contains all the documented customization elements. *Player.as* defines global variables used throughout the player, and provides two marked-off areas for your customizations. These areas are labeled *Customize Area #1* and *Customize Area #2*.

Auxiliary Source Code Files

Playola World of Color uses several other Actionscript files that are referenced (as *#include* files) in *flv-woc fla*. These files are contained within the *source* directory. Multiple files are used as a way to separate the Actionscript functionality.

- *source\com\core.as* - Core player functionality. Includes: import third-party classes; set up Stage; internal member variable declaration; and general player functions.
- *source\com\customizers.as* - Contains customizing functions. These routines take variable input (provided in *player.as*), and apply it to the player elements.
- *source\com\helper.as* - Basic low-level functions for converting strings to other data types; color setting; time formatting; data validation.
- *source\com\keyboard.as* - Creates a keyboard listener so that pressing the Spacebar toggles the player between Play and Pause mode.
- *source\com\logo.as* - Positions and displays an optional clickable logo over the video. You have a choice of placing the logo in any of the four corners of the player. Set up a destination URL associated with the logo and clicking on it opens a window to the Web page you specified.
- *source\com\menu.as* - Modifies the right-click (context) Flash menu. The routine: removes the extraneous Flash-related menu items (except *Settings* and *About Adobe Flash ...*, neither of which may not be removed); adds a Fullscreen toggle menu item; adds a Mute Sound menu item; and adds an optional clickable menu item that opens a URL of your choice.
- *source\com\mouse.as* - Contains mouse listeners and functions for using the mouse to control the player.
- *source\com\netstream.as* - Listeners for the onMetaData and onStatus events, provided by the Flash Player when used with progressive download video media.
- *source\com\poster.as* - Displays (and removes) an optional poster ("splash") image that appears inside the player when it first loads. A poster is intended to be displayed only if the player does not automatically start in Play mode.
- *source\com\resizer.as* - Listeners and handlers for screen resizing, including going to and from Fullscreen mode.
- *source\com\toggleplaypause.as* - Routines for handling the transition between Play and Pause modes. Also includes code for displaying a "Big Play" button, which appears in the center of the video player when in Pause mode. (Note: The *Playola* FLA file contains several Big Play button graphics you may choose from. You can also alter the size, color, alpha transparency, and animation of the button by making customizing changes in the *player.as* file.)

- *source\com\tooltip.as* - Routines for displaying and updating the moving tooltip that displays above the slider thumb (showing the current video playback point). Inside the tooltip is the elapsed time.
- *source\com\volume.as* - Routines for displaying and changing the sound playback level. Users may control audio from mute (0 percent) to full blast (100 percent).
- *source\as2DataValidation.as* and *source\as2ValidationResult.as* - Third party data validation classes. From Adobe; distributed under their free open source code license.
- *source\caurina* - Folder containing third party open source tweener classes. See code.google.com/p/tweener for more general information. Distributed under the MIT open source license.

Running the Sample Files

Unless you have some special Flash security set up on your computer, you can test the sample *World of Color* player directly on your PC. And of course you can test the player by uploading it to your Web site.

Find the *flv-woc.html* file and open it in your favorite browser. Depending on your browser, a security warning may appear. (Specifically, in Internet Explorer the browser may tell you the page contains a reference to an ActiveX component, which is of course Adobe Flash Player. Click to allow the component to run.)

The sample uses media content provided with the source package. You may experiment with other video and graphics files by modifying the Flashvars in the *flv-woc.html* file, and/or customizing the *player.as* file. Both are described below.

Changing Player Behavior Using Flashvars

Though *Playola World of Color* is intended mainly as a platform for creating fully customized Adobe Flash video players, there are a number of Flashvars that can be used to permit one player to run an unlimited number of videos. Flashvars are also provided for several player option settings.

Flashvar	Purpose
<i>videoURL</i>	Filename or full URL of the video to play
<i>posterFrame</i>	Filename or full URL of the poster image
<i>autoStart</i>	Begin playback immediately on player load
<i>preloadVideo</i>	Pre-load video if paused
<i>initialVolume</i>	Initial volume

Remember! Flashvars are case sensitive. This means *autoStart* will work, but *autostart* will **not**!

Flashvars may be applied in a number of different ways. The most straight forward when using OBJECT embedding is to append the Flashvar(s) after the *Playola World of Color* SWF file:

```
flv-woc.swf?Flashvar1=Value1
```

You may string more than one Flashvar by separating them with the & character:

```
flv-woc.swf?Flashvar1=Value1&Flashvar2=Value2
```

See *Embedding Your Player In an HTML Page*, at the end of this document, for a detailed example of how to use your *World of Color* player on your Web site.

Using the videoURL Flashvar

Use *videoURL* to specify the filename or URL of the video to play.

- If the video is in the same directory as the World of Color player, you may specify only the filename.
- If the video is in some other directory, specify a partial or full URL.

Examples:

```
flv-woc.swf?videoURL=myvideo.flv  
flv-woc.swf?videoURL=/videos/myvideo.flv  
flv-woc.swf?videoURL=http://www.mysite.com/assets/myvideo.flv
```

See the section *Hiding File Assets* on how to use the *Playola World of Color* player to help prevent Web page scrapers from finding your bandwidth-intensive video files.

See also the section *Filename Security Notes* for additional information on valid URLs you may use with this Flashvar.

Note! The player will display an error message if the video file cannot be found. The most likely cause of this error is the filename is misspelled, or it is not in the location the player expects it to be.

Using the posterFrame Flashvar

Use *posterFrame* to specify the filename or URL of the poster ("splash") image. Usage is the same as with the *videoURL* Flashvar, described above.

Examples:

```
flv-woc.swf?posterFrame=myposter.jpg  
flv-woc.swf?posterFrame=/images/myposter.jpg  
flv-woc.swf?posterFrame=http://www.mysite.com/img/myposter.flv
```

The poster frame only appears under the following conditions:

- The player is first loaded (this happens when the page first loads).
- The player begins in Pause mode.

See the section *Filename Security Notes* for additional information on valid URLs you may use with this Flashvar.

Using the autoStart Flashvar

The *autoStart* Flashvar controls whether the player stays in Pause mode (the default), or goes immediately into Play mode when the player is loaded.

Valid values for this Flashvar are *true* and *false*.

- *autoStart=true* - Player is put into Play mode on load
- *autostart=false* - Player is put Pause mode on load (default)

Example:

```
flv-woc.swf?autoStart=true
```

Using the preloadVideo Flashvar

The *preloadVideo* Flashvar instructs the player to preload the video if it's initially in Pause mode when the player is first loaded. Preloading is the default.

By preloading the video, it will be buffered into the user's PC memory and available for immediately playback when they click the Play button.

This Flashvar has no influence if the player is set to immediately play upon loading.

Valid values for this Flashvar are *true* and *false*.

- *preloadVideo=true* - Preload the video (default)
- *preloadVideo=false* - Don't preload the video.

Judicious use of the *preloadVideo* Flashvar, combined with *posterFrame*, can save server bandwidth.

If you think most of your site visitors will not view the video (because they are return customers, for example), consider setting *preloadVideo* to *false*. This prevents the player from fetching the video file when it's first loaded with the page. Only if the user clicks the Play button will the video load from your server.

Use *posterFrame* to display an image - it can be from the video, or of anything you wish - to indicate the content of the video.

Using the initialVolume Flashvar

The *initialVolume* Flashvar sets the starting volume of the player. Unless the control bar is fully hidden (which is something you can set; see *Setting Control Panel Variables*, below), the user has the option of turning the sound up or down.

A valid value for this Flashvar is a number (representing a percentage) from 0 to 100. The default setting is 50, for 50 percent.

- Set to 100 to turn the volume all the way up.
- Set to 0 to turn the volume all the way off. The player treats this setting as muted sound, and will display a slash across the speaker icon.
- Set to some other value (50 is the default) depending on the natural sound level of the video clip.

Setting Main Customizations

You can use the unbranded version as-is of *Playola World of Color* player (the file is *flv-woc-unbranded.swf*), without further customizing it. But odds are you'll want to tweak it to make it your own.

Common customizing includes:

- Setting colors of the control panel and its components
- Specifying a logo graphic, and optionally, a URL to follow if the user clicks on it
- Changing the control panel hiding effect

Note! There are even more customizations you can perform. See the section *Setting Secondary Customizations* for more information.

Preparing to Customize the Player

Customizing the player requires that you make one or more changes to a source code file (*player.as*), then compile a new player. You will need the Flash 8 (or later, such as Flash CS4) authoring program.

We'll assume you know the basics of using the Flash program, and jump right to the specifics of customizing the player code.

Note! *Playola World of Color* is engineered so that you DO NOT need to modify the main FLA Flash authoring document. Everything is done in Actionscript code, and all documented customizations involve only changing variable values.

However, you still need to use the Flash authoring program in order to build a new player. If you don't have the authoring program, you can download a trial version from Adobe, and use it for up to 30 days.

Basic Customizing and Compiling Steps

These are the basics steps for customizing and compiling a new player.

Important! These steps assume you are using the same directory structure as provided in the original source code files.

1. Begin by opening the *player.as* source file. This is the main customizing file. Unless you want to get under the hood of the *Playola World of Color* player, it is likely the only one you'll need to edit. You may open the file in a text editor (don't use a word processor), but we recommend the SEIPY Actionscript editor, available at www.sephiroth.it.
2. Locate the *Customize Area #1* section.
3. Find the variable you wish to change. For example, if you wish to modify the auto start behavior, find the *s_autoStart* variable.
4. Make the change, being careful to retain the double quote marks. The contents of the variable must also be in the expected format - a number if the program expects one, for example. The format for each customizing variable is provided in the tables below.

5. Save the *player.as* file.
6. Open the *flv-woc fla* file (located in the source subdirectory) in your Flash authoring program.
7. Recompile the SWF. For example, in Flash 8, choose File->Export->Export Movie. Locate the existing *flv-woc.swf* file, click on it, and choose OK. Answer Yes, you wish to overwrite the file.

Important! Be careful to not disrupt the syntax of any variables, or else your player may not compile.

Be on the lookout for missing double quote marks, and missing trailing semi-colons!

All of the customizing variables in the section of the code begin with *s_*, which is a mnemonic to remind you that the value is a string type.

Customizing - As Easy As 1-2-3!

The *Playola World of Color* source code is designed to make customizing easy. Simply edit one document (*player.as*) and change some variable assignments. There is no need to modify any other underlying Actionscript code, unless of course you want to.

- More than two dozen primary customizing options set colors, player options, media, logo, and control panel settings
- An additional two dozen+ secondary customizing options set advanced options, the Big Play button appearance, size, and color, and animated "throbber."

For example, to change the control panel color to purple, with a 60% alpha transparency, you merely change one line of code, like this:

```
s_controlbarColor = "0xFF00FF:60";
```

Setting Customizing Variables

As noted, you modify the looks and behavior of the *Playola World of Color* player by changing the values of variables.

You may change the following variables, which are located in the section labeled *Customize Area #1*, inside the *player.as* file.

Setting File Location Variables

Variable	What it Does
s_clipURL	<p>Sets a base path, filename, or URL to the video clip. When setting just a base path, you can combine this path with a filename using the <i>videoURL</i> Flashvar. See the section <i>Hiding File Assets</i> for more information.</p> <p>Valid values: A properly-formed filename/URL string.</p>
s_posterURL	<p>Sets a base path, filename, or URL to the poster image. As with <i>s_clipURL</i>, you can combine a base path with the <i>posterFrame</i> Flashvar to help camouflage the location of your files. See the section <i>Hiding File Assets</i> for more information.</p> <p>Valid values: A properly-formed filename/URL string.</p>
s_logoURL	<p>Sets the filename or URL to the logo image file. By using a PNG file with an alpha channel, you can set the opacity of the logo using the <i>s_logoAlpha</i> variable (see below).</p> <p>Valid values: A properly-formed filename/URL string.</p>

Important! *Playola World of Color* contains additional string validation routines to reject several known security exploits. Specifically, the player will accept only Roman characters, and disallows "double-dot" (..) directory traversal.

We recommend you keep these security safeguards in place, but if required for your application, you can relax or remove them as necessary. See the section ***Filename Security Notes*** for more information.

Setting Flashvar Default Behavior Variables

Variable	What it Does
s_autoStart	<p>Specifies whether to start playback immediately upon player load.</p> <p>Valid values: Boolean string of <i>true</i> or <i>false</i>.</p> <ul style="list-style-type: none">• <i>false</i> - Put player in Pause mode on load• <i>true</i> - Put player in Play mode on load
s_preload	<p>Specifies whether to preload the video. if player is</p>

	<p>loaded in Pause mode.</p> <p>With preload enabled, the player will download the video file in the background, ready for playback when the user clicks Play.</p> <p>With preload disabled, the player starts the download only when the user clicks Play.</p> <p>Valid values: Boolean string of <i>true</i> or <i>false</i>.</p> <ul style="list-style-type: none"> • <i>false</i> - Do not preload video • <i>true</i> - Preload video <p>Note: This setting only has effect when the player loads in Pause mode.</p>
s_initVolume	Specifies the initial volume level of the sound, as a percentage from 0 (mute) to 100 (full)

Setting Logo Variables

Variable	What it Does
s_logoPos	<p>Specifies the screen location for the logo. You can pick any of four corners.</p> <p>Valid values:</p> <ul style="list-style-type: none"> • <i>ul</i> - Position in upper left • <i>ur</i> - Position in upper right • <i>ll</i> - Position in lower left • <i>lr</i> - Position in lower right <p>Note: The position of the logo is restricted to within the active portion of the video picture. If the dimensions of your player do not match the aspect ratio of your video, black bars will appear on the top/bottom or sides of the picture. The logo will appear <i>inside</i> these bars.</p>
s_logoAlpha	<p>Sets the alpha transparency of the logo, as a percentage from 0 to 100.</p> <ul style="list-style-type: none"> • 0 means an invisible logo • 100 means a fully opaque logo

	In use this feature your logo should be in PNG format, and contain an alpha channel.
s_logoLink	<p>Specifies a URL to follow if the logo is clicked. The URL must be fully qualified, such as</p> <p><code>http://www.mysite.com/</code></p> <p>Note: The logo link is disabled when opening the player locally on your PC. This is to avoid an ugly and scary Adobe Flash security warning about accessing a network resource. The link works when your player is hosted on the Internet.</p>
s_logoLinkTarget	<p>Specifies how you wish the link target to open in the browser.</p> <p>Valid values are:</p> <ul style="list-style-type: none"> • <i>_blank</i> - Open in a new window (or tab, depending on browser settings) • <i>_self</i> - Open in the same frame or window/tab as the player • <i>_top</i> - Open in the top-level frame in the current window/tab • <i>_parent</i> - Open in the parent of the current frame

Setting Custom Menu Variables

Variable	What it Does
s_menuLinkText	Specifies the text to show in the right-click context menu. Avoid overly long text (25-30 characters maximum).
s_menuLink	<p>Specifies a URL to follow if the logo is clicked. The URL must be fully qualified.</p> <p>See <i>s_logoLink</i>, above, for usage notes.</p>
s_menuLinkTarget	<p>Specifies how you wish the link target to open in the browser.</p> <p>See <i>s_logoLinkTarget</i>, above, for usage notes.</p>

Setting Control Panel Variables

Variable	What it Does
s_controlbarEffect	<p>Specifies the hiding effect for the control bar after a specified delay.</p> <p>Valid values are:</p> <ul style="list-style-type: none">• <i>slide</i> - Slide the control panel down and out of the way• <i>fade</i> - Fade the control panel out• <i>pop</i> - Quickly pop the control panel off• <i>none</i> - Do not auto-hide the control panel.
s_controlbarDelay	<p>Specify the number of seconds to hide the control panel if there is no mouse or other activity over the player.</p> <p>Valid values: 0 to infinity. (A practical maximum is 10, for 10 seconds).</p> <p>The control panel is redisplayed by moving the mouse anywhere over the player window.</p> <p>Important! Use a value of 0 carefully. When set to 0, the control panel will immediately disappear, and <i>only reappear if the user clicks over the player window</i>. Mouse movement alone will have no affect.</p>
s_showControlBar	<p>Specifies whether to display the control panel at all.</p> <p>Valid values: Boolean string of <i>true</i> or <i>false</i>.</p> <ul style="list-style-type: none">• <i>false</i> - Don't show the control panel• <i>true</i> - Show the control panel

Setting Control Panel Colors

Control panel colors are set using a hexadecimal color, and an alpha transparency value, from 100 (fully opaque) to 0 (fully transparent). The two values are separated by a colon.

For example,

```
0x0000FF:100
```

means the color blue, at 100% opaque. Whereas,

0xFFFFFFFF:50

means the color white, at 50% opaque.

Note! Colors are expressed in the usual six-digit notation used in CSS and HTML. But instead of using the # symbol in front, they use *0x*. The *0x* nomenclature tells Actionscript the numbers that follow are in hexadecimal format.

Variable	What it Does
s_controlbarGradient	<p>When <i>true</i>, specifies that you wish to use a two-color gradient mix. When <i>false</i>, indicates you wish to use a single color, as specified by <i>s_controlbarColor</i>.</p> <p>Valid values: Boolean string of <i>true</i> or <i>false</i>.</p> <ul style="list-style-type: none">• <i>false</i> - Use single color as specified by <i>s_controlbarColor</i>• <i>true</i> - Use gradient mix colors, specified by <i>s_controlbarGradientTop</i> and <i>s_controlbarGradientBottom</i>
s_controlbarGradientTop	<p>When <i>s_controlbarGradient</i> is set to <i>true</i>, specifies the top ("from") color and transparency of the control panel.</p> <p>Note! When <i>s_controlbarGradient</i> is set to <i>false</i>, this value has no effect. Use <i>s_controlbarColor</i> instead.</p>
s_controlbarGradientBottom	<p>When <i>s_controlbarGradient</i> is set to <i>true</i>, specifies the bottom ("blend to") color and transparency of the control panel.</p> <p>Note! When <i>s_controlbarGradient</i> is set to <i>false</i>, this value has no effect. Use <i>s_controlbarColor</i> instead.</p>
s_controlbarColor	<p>Specifies the background color of the control panel.</p> <p>Note! This value only has effect when <i>s_controlbarGradient</i> is set to <i>false</i>.</p>
s_sliderColor	<p>Specifies the slider thumb for both the video seek bar and the volume bar.</p>
s_sliderBackgroundColor	<p>Specifies the slider background for both the video seek bar and the volume bar.</p>

	<p>For the video seek bar, the background is the color of the bar for the portion of video that has been loaded (buffered) but not yet viewed.</p> <p>When a video is buffering from a network resource, this bar will grow left to right, to indicate loading progress.</p>
s_sliderFillColor	Specifies the slider fill color, which represents the portion of the video that has been viewed.
s_buttonColor	Specifies the background color for the Play/Pause and Fullscreen buttons.
s_buttonColorFG	Specifies the foreground (icon graphic) color for the Play/Pause and Fullscreen buttons.
s_speakerColor	Specifies the color of the speaker icon.
s_speakerWavesColor	Specifies the color of the speaker "waves." A different number of waves appear (from 0 to 3), depending on the volume level.
s_tooltipBGColor	<p>Specifies the color and alpha transparency of the tooltip (elapsed time indicator) background.</p> <p>Note: For readability don't set a transparency value less than 30.</p>
s_tooltipTextColor	Specifies the color and alpha transparency of the tooltip text. A bright contrasting color against the background works best.
s_durationBGColor	<p>Specifies the color and alpha transparency of the time duration indicator background.</p> <p>Note: For readability don't set a transparency value less than 30.</p>
s_durationTextColor	Specifies the color and alpha transparency of the time duration indicator text. A bright contrasting color against the background works best.

Here are the default color settings, in case you want to revert to them following experimentation:

```
s_controlbarGradient = "true";
s_controlbarGradientTop = "0x333366:100";
s_controlbarGradientBottom = "0x606BD2:100";
s_controlbarColor = "0xC0C0C0:75"
```

```

//ignored if s_controlbarGradient is true
s_sliderColor = "0x696969:100";
s_sliderBackgroundColor = "0x8B0000:27";
s_sliderFillColor = "0x8B0000:75";
s_buttonColor = "0x696969:25";
s_buttonColorFG = "0xF5F5F5:100";
s_speakerColor = "0x8B0000:76";
s_speakerWavesColor = "0xF5F5F5:100";
s_tooltipBGColor = "0x000000:70";
s_tooltipTextColor = "0xFFFFFFFF:100";
s_durationBGColor = "0x000000:70";
s_durationTextColor = "0xFFFFFFFF:100";

```

Setting Secondary Customizations

Additional, and less used, customizations may be made by altering the values of variables within the *player.as* file marked *Customize Area #2*.

Unlike the string variables noted above, those in *Customize Area #2* are mixed data types. Some are Strings, others are Boolean, and still others are Numbers.

As before, take care when modifying these variables. Using a wrong data type (using a string for a Number type), or entering invalid data, will cause a compilation error.

Secondary General Settings

Variable	Data Type	What it Does
m_bufferTime	Number	<p>Sets the amount of video, in seconds, to load before playback will commence.</p> <p>The lower the number the faster playback will begin, but the more chance the video may pause or even "stutter" if the user's connection to your Web site is slow, and your videos are encoded at a moderate to high bit rate.</p> <p>A value of between 3 to 5 is appropriate for most Web sites. Choose a higher number if your visitors complain of uneven playback.</p> <p>Note: If the player outruns the video that has been loaded, it will pause, and the spinning wait circle throbber</p>

		will re-appear. This is in keeping with the design of most modern Web video players, such as YouTube®.
m_loop	Boolean	<p>Specifies whether you wish the video to automatically replay when it reaches the end.</p> <ul style="list-style-type: none"> • <i>true</i> - Repeat • <i>false</i> - Don't repeat
m_sliderHeight	Number	<p>Specifies the height of the slider thumb, the part that you drag to seek the video to another part.</p> <p>You may specify fractional values. Practical smallest value is 2; largest is 25. A value of 11 is normal.</p>
m_sliderWidth	Number	<p>Specifies the width of the slider thumb.</p> <p>You may specify fractional values. Practical smallest value is 2; largest is 15. A value of 8 is normal.</p>
m_railThickness	Number	<p>Specifies the height of the slider rail behind the slider thumb.</p> <p>You may specify fractional values. Practical smallest value is 1; largest is 25. A value of 5 is normal.</p>
m_showMetadata	: Boolean	<p>Specifies whether you wish to display diagnostic metadata over the video. This metadata is obtained from the currently loaded video file, and includes - among other information - height, width, and duration.</p> <p>Note: The amount of metadata displayed depends on the height of the player. The larger (taller) the player, the more metadata that will be displayed.</p> <ul style="list-style-type: none"> • <i>true</i> - Show metadata • <i>false</i> - Don't show metadata

		<p>Also note: Not all videos have the same metadata. The metadata is added by your video encoder, and/or a metadata injection utility, if you use one (<i>and we recommend that you do!</i>).</p> <p>Many injection utilities add additional metadata, such as keyframe positions, audio and video codecs, and more.</p>
--	--	--

Secondary Control Panel Settings

Variable	Data Type	What it Does
m_showSeekBar	Boolean	<p>Specifies whether to show the video seek bar. When not shown, all seek bar elements are not visible. This includes the slider thumb and the slider rail.</p> <ul style="list-style-type: none"> • <i>true</i> - Show seekbar • <i>false</i> - Don't show seekbar <p>One application of this feature is to create a control panel that allows the user to play and pause the video, and adjust the audio, but not seek to a different location.</p>
m_showFullscreenButton	Boolean	<p>Specifies whether to show the Fullscreen button.</p> <ul style="list-style-type: none"> • <i>true</i> - Show button • <i>false</i> - Don't show button <p>Note! Even when not shown, the space for the Fullscreen button remains. If you wish to remove this space entirely you may do so by changing the control panel elements X offsets.</p> <p>Important! Even if you display the</p>

		<p>Fullscreen button you must instruct Adobe Flash Player to allow fullscreen mode. This requires setting the <i>allowFullScreen</i> parameter to <i>true</i>.</p> <p>See the example in the section, <i>Embedding Your Player In an HTML Page</i>, below.</p>
m_showTimeRemaining	Boolean	<p>Specifies whether you wish to display the total clip duration, or the time remaining, in the time indicator to the right of the seek bar.</p> <ul style="list-style-type: none"> • <i>true</i> - Show time remaining • <i>false</i> - Show total time (duration)

Secondary Big Play Button Settings

Variable	Data Type	What it Does
m_bigButton	String	<p>Specifies one of several pre-defined "Big Play" buttons, which appear over the center of the player when it is in Pause mode.</p> <p>Available buttons are:</p> <ul style="list-style-type: none"> • <i>bigplaypause_solidarrow</i> • <i>bigplaypause_circle</i> • <i>bigplaypause_fatarrow</i> • <i>bigplaypause_hollowarrow</i> • <i>bigplaypause_nestedarrow</i> <p>Important! Capitalization and spelling matter. Big Play bButton names are all lower-case, and begin with <i>bigplaypause_</i>.</p> <p>You can add your own button. To do so, in your Flash development program add it as a new MovieClip to the library. Specify a unique Linkage Identifier in the Symbol Properties</p>

		<p>dialog box. Then in the <i>player.as</i> file, use this same name for <i>m_bigButton</i>.</p> <p>Note! Avoid adding lots of extra Big Play button graphics, as this can inflate the size of your compiled SWF file. Delete those you don't need.</p>
s_bigButtonColor	String	<p>Specifies the color and alpha transparency of the Big Play button graphic.</p> <p>As with setting control panel colors, the string is in the form of:</p> <p>0x#####:###</p> <p>where the first value is the color, in hexadecimal, and the second is the alpha transparency, from 0 to 100.</p>
m_bigButtonShowAtStartup	Boolean	<p>Specifies whether you want the Big Play button to display when the player is first loaded.</p> <ul style="list-style-type: none"> • <i>true</i> - Show button • <i>false</i> - Don't show button
m_bigButtonAnimation	Boolean	<p>Specifies whether you wish to display attention-getting animation for the Big Play button when the player is first loaded.</p> <ul style="list-style-type: none"> • <i>true</i> - Animate • <i>false</i> - Don't animate
m_bigButtonSizeStartup	Number	<p>Specifies the scale of the Big Play button when the player is first loaded.</p> <p>A value of 100 means normal size. A size of 50 means half normal size, and 300 means 300% normal size.</p>
m_bigButtonSizeAtPause	Number	<p>Same as <i>m_bigButtonSizeStartup</i>, but when displaying the Big Play button on subsequent pauses.</p>

Secondary Throbber (Animated Wait Circle) Settings

Variable	Data Type	What it Does
m_throbberColor	String	<p>Specifies the color and alpha transparency of the throbber, an animated circle of dots that indicates when more of the video must be loaded before playing can continue.</p> <p>As with setting control panel colors, the string is in the form of:</p> <p>0x#####:###</p> <p>where the first value is the color, in hexadecimal, and the second is the alpha transparency, from 0 to 100.</p>
m_throbberSize	Number	<p>Specifies the scale of the throbber.</p> <p>A value of 100 means normal size. A size of 50 means half normal size, and 250 means 250% normal size.</p>

Secondary Advanced Settings

Variable	Data Type	What it Does
m_pixelErrorComp	Number	<p>Specifies a pixel offset value to compensate for misalignment that can occur with embedded Adobe Flash objects on a Web page. A value of 2 is normal, but may not always be needed.</p>
m_smoothing	Boolean	<p>Specifies whether to apply smoothing and deblocking to the video during playback.</p> <ul style="list-style-type: none"> • <i>true</i> - Smooth and deblock • <i>false</i> - Don't smooth/deblock
m_keyboardEnable	Boolean	<p>Specifies whether to enable keyboard shortcuts. <i>Playola World of Color</i> presently supports one keyboard shortcut: pressing the spacebar when the player has focus in the browser</p>

		<p>toggles between Play and Pause mode.</p> <ul style="list-style-type: none"> • <i>true</i> - Enable keyboard shortcuts • <i>false</i> - Disable keyboard shortcuts
m_picturePauseEnable	Boolean	<p>Specifies whether clicking over the active video portion of the player with the mouse toggles between Play and Pause mode.</p> <ul style="list-style-type: none"> • <i>true</i> - Enable mouse click to toggle Play/Pause • <i>false</i> - Disable mouse click to toggle Play/Pause
m_trimAtEnd	Number	<p>Specifies an amount, in seconds, to trim off the duration of video clips, in case the metadata duration of the clip is slightly longer than the actual duration.</p> <p>This is actually fairly common problem of Flash videos.</p> <p>When the problem occurs, the player will not know when the end of the video has been reached. The problem can be largely avoided by applying a small trim factor.</p> <p>A typical value is 0.25, for a quarter second. You may experiment with longer and shorter values, but remember that longer values may cut off your videos prematurely.</p>

Control Panel Elements X Offsets

Change these offsets to move the control panel elements (Play/Pause button, slider, volume control, and Fullscreen button) to where you want them.

Variable	Data Type	What it Does
m_fullscreenButtonPos	Number	Position of the Fullscreen button. from

		the right edge of the player. Normal value: 15
m_volPos	Number	Position of the volume control, from the right edge of the player. Normal value: 38
m_sliderL	Number	Position of the video slider, from the left edge of the player. Normal value: 35
m_sliderR	Number	Position offset of the video slider, from the right edge of the player. Normal value: 125
m_playpause	Number	Position of the Play/Pause button, from the left edge of the player. Normal value: 8

Play with these values to manipulate the control panel elements, without having to modify the FLA file.

Important! Because *Playola World of Color* uses Actionscript to position the control panel elements, trying to position them in the FLA file will not work!

For example, to hide the Fullscreen button, and reposition the other elements to make up for the left-over space, change the following variables:

Variable	Revised Setting
m_showFullscreenButton	false
m_fullscreenButtonPos	0
m_sliderL	35
m_sliderR	100
m_volPos	15
m_playpause	8

Or, to display only the seekbar, Fullscreen button, and volume control (disable Play/Pause button and pausing ability), change the following variables:

Variable	Revised Setting
s_autoStart	"true"
m_keyboardEnable	false
m_picturePauseEnable	false
m_showFullscreenButton	true
m_fullscreenButtonPos	15
m_sliderL	12
m_sliderR	100
m_volPos	38
m_playpause	-20

The setting of -20 for *m_playpause* removes the Play/Pause button from sight - it's still there, but it cannot be seen.

The other values widen the slider, and disable pausing using the keyboard or mouse. Finally, the *s_autoStart=true* setting starts playback upon player load.

Your users will not be able to pause the video, which means they are sure to get the message you intend. But there's also a good chance they'll be highly annoyed! So, be sure to use this technique sparingly, and only on short videos.

Hiding File Assets

Playola World of Color is designed to allow you to easily hide video and poster image assets from the prying eyes of Web site scrapers. These automated robots troll the Internet, looking for pages with videos.

When the bot finds a video or image it likes, the location of the file gets added to a database. In worse case, the video is then posted on various tube sites, blogs, and leecher sites. Because the file is still fetched from your server, they're using your bandwidth and you're not getting any benefit from the added views.

And no, you can't control this sort of thing with an *.htaccess* file to prevent hot-linking. The reason: Adobe Flash player does not always properly identify the referring source for content called by a SWF file. This is particularly true when using the Firefox browser, where the referrer information is blank.

While you could write an anti hot-linking script to disallow linking from a site other than yours, as well as empty referrers, this is an inadequate solution at best.

With *Playola World of Color*, you can hide the base directory of your assets, and provide only the filename as a Flashvar in the HTML file. Page scrapers will only see the filename to the media. Lacking a directory path, it will not be able to locate the asset, so it can't as easily be pilfered.

To use this feature,

- Include only the base directory or URL path in the *s_clipURL* variable. (For poster images use *s_posterURL*.)
- In your Web page HTML, provide only the filename to the asset.

Examples:

Embedded In Player	Filename In HTML	Resulting full asset path/URL
<i>videos/</i>	<i>somevideo.flv</i>	<i>videos/somevideo.flv</i>
<i>/myfiles/</i>	<i>myvideo.flv</i>	<i>/myfiles/myvideo.flv</i>
<i>http://mysite.com/vids/</i>	<i>secretvideo.flv</i>	<i>http://mysite.com/vids/secretvideo.flv</i>

Note the use of relative and absolute server paths, and fully qualified URLs.

- Use a relative server path, such as *videos/* (no leading slash) when the containing folder is below the location of the SWF file.
- Use an absolute server path, such as */myfiles/*, when the containing folder is in an arbitrary directory.
- Use a fully qualified URL when you wish to specifically provide the Internet address of the containing folder. Use this approach, for example, when your assets are not located on the same server as your SWF file and Web page.

Example, in code:

```
s_clipURL = "http://mysite.com/vids/";
```

and in your Web page:

```
flv-woc.swf?videoURL=secretvideo.flv
```

Routines in the player automatically combine these two file parts together, and find the video file located here:

http://mysite.com/vids/secretvideo.flv

Important! Be sure to include all the necessary slashes, or else the path/URL may be malformed. For example, if you forget the trailing slash in the above example, you'd end up with

http://mysite.com/vidssecretvideo.flv

and playback will fail, because there is no filename with that name.

Note! There are instances when you want robots to find your videos. For example, you may wish your videos to show up on Google's video search. The Google bot needs to have the full path or URL to your video in order to index it.

Balance the need to protect your bandwidth with the desire to allow potential customers to find your videos using the popular search engines. For Google and other search engines you may also help the robots find your video content with the use of video sitemaps. Naturally, sitemaps are also open to bad robots, some of which may use the information to leech content from your site.

Overriding Build-in Paths

There may be occasion where you use a player that has a path hard-coded into the *s_clipURL* or *s_posterURL* variables, but you wish to use a video or poster image in a completely different location.

Merely add a colon character in front of the filename. This tells the player to ignore the value stored inside it, and use *only* the path and filename provided in the Flashvar.

For example,

```
flv-woc.swf?videoURL=:http://www.anothersite.com/videos/myvideo.flv
```

tells the player to ignore any value already stored in *s_clipURL*, and use only the URL provided in the *videoURL* Flashvar.

Filename Security Notes

Playola World of Color incorporates several security features to make it harder for would-be hackers to abuse your players. These features restrict the type of characters you may use in filenames provided for video files, poster images, and logo files.

- The player checks for common known script schemes, such as using *javascript:* or *mailto:* strings as part of the URL. If the URL contains these or similar strings, the data is ignored completely.
- For video files and poster image URLs, the player validates against a "white list" of known good characters. These characters are limited to 0-9, a-z, and the following symbols: - _ . : / ? & % # = + ~ ().
- All paths are checked for so-called *double-dot* directory traversal characters. Any time a URL contains a sequence of two dots, the entire string is ignored. This means you cannot provide a URL like *../myvideos/video.flv*. You must instead use an absolute path from the document root of your Web site.

We recommend that you retain these security checks if at all possible, but when necessary - like to allow non-Roman characters in URLs - you may either expand on the white list of characters, or bypass one or more security checks.

Adding Characters to the Whitelist

You may add additional characters to the whitelist by editing the *as2DataValidation.as* file. Locate the line

```
private static var LC_ROMAN_LETTERS
```

and add additional characters to the string value.

Bypassing the Security Checks

You may modify the *as2DataValidation.as* file to bypass any of the security checks you do not wish to use. To do so, add a

```
return true;
```

as the first statement in any of the security routines you don't want.

- The *whiteString* function validates a filename, path, or URL against a list of "approved" characters - namely numbers, the Roman alphabet, and a small collection of standard punctuation used in URLs.
- *containsScriptSchemes* checks the string against a list of common exploits using server or browser scripting.
- The *containsDoubleDots* function checks for a series of two dots, which could indicate a possible exploit attempt at traversing the directory structure of your Web server.

Implementation Notes

Here's a bit of background on why certain things were done in the *Playola World of Color* player.

- *Separate source code files.* We separated out the source code files to make things easier to organize. Functionally, the files are reconstituted, using `#include` statements, inside the FLA file. To Flash, it's all one long file, but to you, having the source code broken out makes things easier to find.
- *Procedural code versus classes.* We wrote the *Playola World of Color* player as an aid to help others build a customized Adobe Flash video player. Rather than go with classes, we decided for this go-around to use flat "procedural" programming, and write it in Flash 8. While we prefer using classes, this approach will hopefully put the player in more people's hands.
- *Why two types of customizing variables.* We separated the collection of customizing variables into two groups. Those in *Customize Area #1* are all string variables, in case you (as a self-exercise) want to add the ability to set player options using an XML file. We chose not to use an external file, as the concept of *World of Color* is to customize it internally, creating unique, independent players.

Embedding Your Player In an HTML Page

Here's a run-down of the code in the *flv-woc.html* sample file, included with the *Playola World of Color* package. Let's look at it one piece at a time.

Important! Do not copy and paste the text from the example below to make your HTML file. Use the *flv-woc.html* sample file instead. The code in the example has been reformatted to fit this page, and the line breaks will cause your script to fail.

- The *html*, *head*, and *body* sections are standard HTML markup. The example sets the background color of the page to white merely for demonstration purposes.
- The player is embedded into the page using the *object* tag. This tag and its contents were created by the Flash 8 authoring program (File->Publish), and then hand-tweaked. So, you can do the same with your Flash authoring program, and it'll give you the framework you need for using the player on your pages.
- The *width* and *height* parameters set the size of the player. The sample video is 512 by 288, so the same dimensions are used for the player. This is good practice; keeping the player the same size as the video dimensions can improve playback looks and performance.

- *Flashvars* have been added to the movie name to specify the video clip and poster image to use. See the discussion in ***Changing Player Behavior Using Flashvars*** for more information on using the *videoURL* and *posterFrame* Flashvars.
- The *allowFullScreen="true"* setting has been added in order to permit use of the Fullscreen button. ***This is very important!*** Adobe Flash defaults to a setting of false for this parameter, meaning that *without this explicit setting, the Fullscreen button will not work!*
- Note the *param* tags and separate *embed* tag. In this variation of Flash object embedding, both are required, in order to support the widest possible number of browsers. Basically speaking, the *param* tags are for Internet Explorer (which uses the ActiveX version of the Adobe Flash Player); the *embed* tag is for Firefox, Google Chrome, and other browsers (which use the plugin version of Flash).
- The *bgcolor* attribute used within the *object* tag (in both the *param* and *embed* sections) set the background color of the player. In the example it's been set to black, but you can change it to any other color. The background player color is irrelevant (you can't see it) unless there is no video loaded, or the player is sized so that letterboxing or pillarboxing bars appear.

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Playola World of Color Example</title>
</head>
<body bgcolor="#ffffff">
<object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000"
        codebase="http://fpdownload.macromedia.com/pub/shockwave/cabs/flash/
        swflash.cab#version=8,0,0,0"
        width="512"
        height="288"
        id="flv-woc"
        align="middle">
<param name="movie" value="flv-woc.swf?videoURL=sample-video.flv&
        posterFrame=sample-poster.jpg" />
<param name="allowFullScreen" value="true" />
<param name="quality" value="high" />
<param name="bgcolor" value="#000000" />
<embed src="flv-woc.swf?videoURL=sample-video.flv&
        posterFrame=sample-poster.jpg"
        quality="high"
        bgcolor="#000000"
        width="512"
        height="288"
        name="flv-woc"
        allowFullScreen="true"
        align="middle"
        type="application/x-shockwave-flash"
        pluginspage="http://www.macromedia.com/go/getflashplayer" />
</object>
```

</body>
</html>

Legal Notices

Unless otherwise specified, this document, and the accompanying *Playola World of Color* player files, are Copyright © 2010, Buenacreek Consulting.

Except for files as noted below, this application is distributed under the Creative Commons 3.0 Attribution-ShareAlike 3.0 Unported license. You are free to use, share, mix, and/or adapt this work in any way or manner, as long as you attribute the work as specified below, and that if you re-distribute the work, you do so under the same or similar license to this one.

For more information on the Creative Commons license see <http://creativecommons.org/licenses/by-sa/3.0/>

Adobe® and Flash® are registered trademarks of Adobe Systems Incorporated.

YouTube® is a registered trademark of Google Inc.

Buenacreek Consulting is not affiliated with the above named companies.

All other trademarks are owned by their respective holders.

Playola is a play-on-words based on a certain brand of motion pictures viewers, and no association, express or implied, shall be made to this or similar marks.

Playola World of Color uses open third-party add-on tools, as follows:

- Caurina Tweeners Class - code.google.com/p/tweener/
- Adobe Systems Flash validators sample code for AS2 and AS3 - code.google.com/p/flash-validators

Certain source code files distributed in *Playola World of Color* are provided under a Creative Commons BSD open source license. Their use and redistribution is encouraged. These files are so indicated in the comments section of the source code.

Attribution

The following attribution shall be retained on all copies of player source code and documentation:

Copyright 2010, Buenacreek Consulting - buenacreek.com. Distributed under the Creative Commons 3.0 Attribution-ShareAlike 3.0 Unported license.