jQuery Speedo Popup

Product Documentation

Version 2.1

21 June 2013

Table of Contents

1 Introduction

jQuery Speedo Popup is a small, powerful and real customizable jQuery plugin. With Speedo Popup you can create complex popups very simple. It comes with 14 predefined skins, 15 predefined CSS3 transitions and 7 predefined jQuery transitions. Speedo Popup v2.0 has been completely rewritten, as a modular plugin, this means that the plugin is now faster, consuming lower resource, smaller file size, dynamic etc. Speedo Popup is built with CSS3 and HTML5, but it is also, compatible with older versions of browsers.

1.1 Main Features

jQuery Speedo Popup has many features, but here we will only specify the more important one:

- 15 predefined CSS3 effects.
- 10 predefined CSS3 themes.
- 8 predefined jQuery transitions.
- Run popup only an amount of time through Cookies.
- Multiple instances on one page, with separate settings for each one.
- Automatic Show
- You can play HTML5 audio.
- Close and Show timer.
- Direct Link (Show popup by using a link).
- Can use any kind of browser compatible content:
    - HTML
    - Frame (website, image etc.)
    - Flash file (swf)
    - Embedded video
- Cross-Browser compatibility
- Multiple callbacks
- Small file size

1.2 Folder Structure

File Name	Description
speedo.popup.js	Core plugin file (full size version).
speedo.popup.min.js	Core plugin file (minified version).
speedo.popup.fx.css	Core CSS3 effects file.
skins	All the skins are kept inside this folder.
|-- default	The skin files are kept inside this folder.
| |-- default.css	The core skin file (The skin file is named like the skin name).
| |-- images	All the skin images are kept inside this folder.

Note: The structure of a skin directory mat vary based on the content needed for the skin.

2 Working with jQuery Speedo Popup

In this section you will learn to use jQuery Speedo Popup. This section will start with basic stuff and it will go to advanced stuff.

2.1 Getting started

To use Speedo Popup, you need to start by extracting the files from the zip.

If you want to use Speedo Popup in your website, you need to copy the '**source**' folder into your site root folder. Inside your page in head section you need to include jQuery, if not already, Speedo Popup JavaScript file, and specific CSS.

```
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"></script>
<script type="text/javascript" src="source/speedo.popup.min.js"></script>
<link rel="stylesheet" type="text/css" href="source/skins/default/default.css"></link>
```

Note: Here we only load the default skin, but you can copy the whole last line and change the **default** to any skin you want to use also, you need to specify the skin when you are calling the popup.

Note: The Available skins are listed later in the theme section.

Now, you can start the popup in two ways, first you can call it using JavaScript:

```
<script type="text/javascript">
  $(function () {
    $.fn.speedoPopup({htmlContent: "<p> I'm a simple content </p>"});
  });
</script>
```

The above code will show a popup, every time the page finished loading, with the content specified inside htmlContent and using the default skin.

The second way is to use a direct link, by adding the class **speedo-popup** to an anchor:

```
<a href="image.jpg" class="speedo-popup">Show Popup</a>
```

The above code will show a popup, every time the Show Popup anchor is clicked, with an image as content and using the default skin.

2.2 Using options

To use an option on Speedo Popup from JavaScript, you need to pass it to the speedoPopup function:

```
$.fn.speedoPopup({property: value, ...});
```

Or if you want to use a direct link:

```
<a href="image.jpg" class="speedo-popup" data-speedo-options='{property: value, ...}'>Show Popup</a>
```

Note: To use this you need to follow valid JSON syntax including quoted property names.

You can add multiple contents in both ways using ',' comma to split them.

2.2.1 Available options

Property	Type	Description	Possible Values	Default
width	*String, Number*	Width of the popup container. If this parameter is null, the width will be automatically adjusted from the content width.	pixels	null
height	*String, Number*	Height of the popup container. If this parameter is null, the height will be automatically adjusted from the content height.	pixels	null
left	*String, Number*	Left position, of the popup container. If this value is either null or 'center', the container will automatically be centered.	pixels	'center'
top	*String, Number*	Top position, of the popup container. If this value is either null or 'center', the container will automatically be centered.	pixels	'center'
close	*Boolean*	Show close icon	true, false	true
closeCaption	*String*	Close button caption. If this is either null or an empty string, the close button will have no text/html caption.	caption text	empty string
theme	*String*	Select popup skin. If the skin don't exists, the popup will be loaded with the default skin.	skin name	'default'
htmlContent	*String*	Popup content as an HTML string. If this is null or false, the htmlContent parameter will be ignored and the popup will try to use the href property.	content passed as, string containing html code	' Default content '
caption	*String*	Popup caption. If this is, null or empty string, the popup caption will not be created.	caption text	null
href	*String*	Address to the content to be loaded. If this is either null or false, the popup will try to load the content from htmlContent property.	content address	null
overlayClose	*Boolean*	Close popup when overlay is clicked.	true, false	true
autoClose	*Number*	Wait an amount of time, after that time passed, the popup will close automatically.	number in ms, false	false
autoShow	Number	Wait an amount of time, after which the popup will, show.	number in ms, false	false
effectIn	String	Effect used while the popup is showing.	'none', 'fade', 'growBox', 'incerto', 'slideLeft', 'slideTop', 'slideRight', 'slideBottom', 'slideZoom'	'none'

effectOut	String	Effect used while the popup is hiding.	'none', 'fade', 'growBox', 'incerto', 'slideLeft', 'slideTop', 'slideRight', 'slideBottom', 'slideZoom'	'none'
css3Effects	String	CSS3 effect used while the popup is showing or hiding. This effect will override the effectIn or effectOut, in modern browser that supports CSS3 animations.	'none', 'zoomIn', 'zoomOut', 'flip', 'flipInHor', 'flipInVer', 'bounceIn', 'pageTop', 'flyIn', 'fadeInScale', 'scaleDown', 'fadeSpin', 'pulse', 'leftSpeedIn', 'rollIn', 'rollOut', 'pulseBody', 'fadeSpinBody'	false
useFrame	Boolean	Force the content to load into a frame. The href property will be used as the frame url, the htmlContent property will be ignored.	true, false	false
useAjax	Boolean	Force the content to load with an ajax call. The href property will be used as the url for the ajax call, the htmlContent property will be ignored.	true, false	false
loadingImage	Boolean	Show loading image while the content is being loaded.	true, false	true
mp3Path	String	Play mp3 audio files.	Path to an mp3 audio file.	undefined
oggPath	String	Play ogg audio files.	Path to an ogg audio file.	undefined
volume	Number	Audio volume.	0.0 to 1.0	1.0
unload	Boolean	Destroy the popup completely (Remove the html code from page). If this is set to false, the popup will just be hidden, this can be used so the popup starts faster.	true, false	true
draggable	Boolean	Enable window dragging.	true, false	false
responsive	Boolean	Enable responsive popup. If this is enabled, the popup will automatically resize , to fit inside the browser window.	true, false	true
loop	Boolean	Start from the beginning, when we reach the end of a group.	true, false	true
buttons	Object	Add custom buttons to the popup.	[{html: 'About', action: function () { alert('Speedo	null

			Popup'); }, class: 'button-test'}, ...]	
onBeforeShow	Function	Called before the popup is showing.	function () { ... }	function(){}
onShow	Function	Called when the popup is showing,	function () { ... }	function () {}
onComplete	Function	Called after the content, finished loading.	function () { ... }	function () {}
onHide	Function	Called when the popup is hiding.	function () { ... }	function () {}
onClose	Function	Called when the popup receives a close command.	function () { ... }	function () {}
onAudioStart	Function	Called when the audio starts playing.	function () { ... }	function () {}
onAudioStop	Function	Called when the audio stops playing.	function () { ... }	function () {}

2.3 Content

There are two ways to set the content:

1. Set the popup content using '**htmlContent**' property:

```
$.fn.speedoPopup({htmlContent:: "<p> I'm a simple content </p>"});
```

2. Set the popup content using '**href**' property:

```
$.fn.speedoPopup({href: "image.jpg"});
```

Note: You can use the '**useFrame'** property to force the content to load inside an iframe.

Note: You can use the '**useAjax**' property to force the content to load through an AJAX call.

The type of the content is automatically determined by the assigned content.

Speedo Popup also, supports YouTube, Vimeo, Google, Metacafe, Dailymotion, WordPress, SWF, videos.

The following table specifies the priority based on the content type:

Type	Description	Priority
HTML	Load HTML content from a string.	Lowest
Images	Load images as the popup content.	Low
Videos	Load media clips.	High
Web pages	Load web pages inside frame container or with ajax, directly inside a div.	Highest

2.4 Themes

Speedo Popup comes with 10 predefined themes/skins:

1. default
2. dark
3. light
4. trap
5. metro
6. lightbox
7. blueglass
8. darkglass
9. notify-glass
10. ignite

The list above it's the default themes available in jQuery Speedo Popup v2.1. Speedo Popup allows you to create custom themes.

To use one of the available themes, you first need to include its CSS file in the page:

```
<head>
  ...
  <link rel="stylesheet" type="text/javascript" src="SpeedoPopup/skins/metro/metro.css" />
  ...
</head>
```

Now, when you call the popup you need to specify the theme to use:

```
$.fn.speedoPopup({theme: "dark"});
```

2.5 Effects

Speedo Popup comes with 8 jQuery predefined effects, and 17 CSS3 effects.

You can choose a jQuery effect using the properties '**effectIn**' and '**effectOut**':

```
$.fn.speedoPopup({
        effectIn: "fade",
        effectOut: "fade"
});
```

The following predefined jQuery effects are available:

1. none
2. fade
3. growBox

4. incerto
5. slideLeft
6. slideRight
7. slideTop
8. slideBottom
9. slideZoom

You can choose a CSS3 effect using the '**css3Effects**' property:

```
$.fn.speedoPopup({
        effectIn: "fade",
        effectOut: "fade",
        css3Effects: "flyIn"
});
```

Note: We are still using jQuery effects for old browsers.

The following predefined CSS3 effects are available:

1. zoomIn
2. zoomOut
3. flip
4. flipInHor
5. flipInVer
6. bounceIn
7. pageTop
8. flyIn
9. fadeInScale
10. scaleDown
11. fadeSpin
12. pulse
13. leftSpeedIn
14. rollIn
15. rollOut
16. pulseBody
17. fadeSpinBody

Note: Speedo Popup allows you to create your own jQuery and CSS3 effects.

2.6 Custom Buttons

Speedo Popup lets you add custom buttons to the popup. You can do this very simple:

```
$.fn.speedoPopup({
        buttons: [{
```

```
                html: "About",
                action: function ()
                {
                        alert("This is a simple sample for Speedo Popup Buttons.");
                },
                class: "about-button"
        }]
});
```

You can add an unlimited number of buttons. The '**html**' and '**action**' property are required for every button any other property will be added to the button as an html attribute. The buttons can also, be added through a direct link.

2.7 Direct Link

You can use a simple link to show the popup, just like a light box. The anchor will be found through the '**speedo-popup**' class:

```
<a href="image.jpg" class="speedo-popup">Show Popup</a>
```

You can specify the same options that are available through JavaScript using data-speedo-options attribute:

```
<a href="image.jpg" data-speedo-options='{"width": 640, "height": 360, "theme": "metro"}' class="speedo-popup">Show Popup</a>
```

Note: As of Speedo Popup v2.0 passing options through href has been deprecated, however in v2.0 it is still possible to use the query, but there are no longer available in v2.1.

2.8 Using Cookies

Speedo Popup can also use cookies to show the popup a limited number of times to a user. This feature is very simple to use, you only need to set the '**startCount**' to the number of times you want a user to see the popup, and to set '**interval**' to the number of days the cookie should be available:

```
$.fn.speedoPopup({
        startCount: 1,
        interval: 30                    // The default value for this is 30 days.
});
```

3 Working with advanced features

Speedo Popup has a set of advanced features, for programmers and users who manage to make their way in JavaScript. In this section we will go in more detail about these advanced features.

3.1 Introduction

In order to use the advanced features of the popup, you need to get the instance of the popup, to do that you just need to assign the return of the plugin to a variable:

```
$(function (){
        var myPopup = $.fn.speedoPopup({...});
        window.myPopup = myPopup;          // Make this variable public so it can be accessed everywhere in the page.
});
```

Now, you can use the variable myPopup, to handle popup functionality.

3.2 Show/Hide Popup

You can show/hide the popup by calling showPopup and hidePopup:

```
myPopup.hidePopup();
myPopup.showPopup();
```

These functions have no parameters and no return value.

3.3 Set Content

You can set content programmatically through the function setContent:

```
myPopup.setContent("Simple HTML Content");
```

This function has tree parameters:

- content – The content to set inside popup.
- type – The type of the content this can be any of the following:
    - "html"
    - "image"
    - "ajax"
    - "iframe"
    - "flash"
    - undefined

  If you don't pass anything to this parameter, or you set it to undefined, the function will try to detect the type of the content based on the content passed.

- complete – Callback function called when the content loading is complete.

This function has no return value.

3.4 Set width/height

You can set width and height programmatically using the functions width and height:

```
myPopup.width(320);
myPopup.height(480);
```

These functions have two parameters:

- value – The width or height value based on the function.
- animate – Animate the resize. Default is true.

Return Value:

If you pass a value to the function, the return value is the old width/height, if you don't pass a value to the function the return value is the current width/height.

3.5 Center popup

You can center the popup programmatically using the function centerPopup:

```
myPopup.centerPopup();
```

This function has no parameters and no return value.

3.6 Get box size

You can get the popup box size programmatically using the function get_box_size:

```
var boxSize = myPopup.get_box_size();
```

This function has no parameter.

Return Value:

The function will return a JSON object containing left, top, width, height, with the following structure:

{left: 'center', top: 'center', width: 500, height: 300}

3.7 Using callback

You can use a callback by simply specifying it in the options:

```
$.fn.speedoPopup({onClose: function (){ ... }});
```

For a list of callbacks see the options section.