jQuery Speedo Notify

Product Documentation

Version 1.0

30 June 2013

Table of Contents

1 Introduction

jQuery Speedo Notify is a small, powerful and real customizable jQuery plugin. With Speed Notify you can create complex notifications. It comes with 10 skins, 14 CSS3 transitions and 7 jQuery transitions. Speedo Notify is written as a modular plugin, this means that the plugin is very fast and has a small file. Speedo Notify is built with CSS3 and HTML5, but it is also compatible with older versions of browsers.

1.1 Main Features

jQuery Speedo Notify has many features, but we will only specify the more important ones:

- 14 CSS3 effects.
- 10 CSS3 themes.
- 7 predefined jQuery transitions.
- Run notify 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 notify using a link).
- Can use any kind of browser compatible content:
    - HTML
    - Frame (website, image etc.)
    - Flash file (swf)
- Cross-Browser compatibility
- Multiple callbacks
- Small file size

1.2 Folder Structure

File Name	Description
speedo.notify.min.js	Core plugin file (minified version).
speedo.notify.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 may vary based on the content needed for the skin.

2 Working with jQuery Speedo Notify

In this section you will learn how to use jQuery Speedo Notify. This section will start with basic stuff.

2.1 Getting started

To use Speedo Notify, you need to start by extracting the zip file.

If you want to use Speedo Notify 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, Speedo Notify JavaScript file, and specific CSS.

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

Note: Here we only load the default skin. This skin should always be loaded for fallback.

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

Now, you can start the plugin in three ways, first you can call it using JavaScript:

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

The above, code will show a notify, every time the page finished loading. With the content specified inside content.value and using the default skin.

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

```
<a href="http://www.website.com" class="speedo-notify">Show Notify</a>
```

The above code will show a notify, every time the Show Notify anchor is clicked. With a website as content and using the default skin.

The third way is to use the custom tag:

```
<speedo-notify mode="show"></speedo-notify>
```

The above, code will show a notify, every time the page finished loading. Using default options.

2.2 Using options

To use an option from JavaScript, you need to pass it to the speedoNotify function:

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

Passing options to a direct link is also easy:

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

Passing options to speedo-notify tag is also easy:

```
<speedo-notify mode="link" options='{property: value, }'>Show Notify</speedo-notify>
```

Note: To use the last two you need to follow, valid JSON syntax including quoted property names.

You can add multiple options, 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 notify container. This will be taken into consideration only if the position is 'left' or 'right'.	pixels	'auto'
height	String, Number	Height of the notify container. This will be taken into consideration only if the position is 'top' or 'bottom'.	pixels	'auto'
position	String	The position of the notify container.	'top' 'right' 'bottom' 'left'	'top'
cover	Boolean	Do we want the notify container to cover over the page content?	true, false	true
close	Boolean, String	Show close button. If this is a string, the close button will be visible and have the text/html content from the string.	true, false, caption text	true
theme	String	Select a skin. If the skin don't exists, the default skin will be used.	skin name	'default'
content	Object	Specify the content of the notify bar.	{…}	{…}
content.**alignment**	String	Content alignment.	'left', 'center', 'right'	'left'
content.**type**	String	The content type.	'auto', 'html', 'image', 'flash', 'iframe'	'auto'
content.**link**	Boolean, String	Specify if the content holder should be a link. If this is a string this will be treated as the url.	false, url	false
content.**value**	String	The actual content.	Content passed as a string value.	'<p> Default content

				</p>'
content.**method**	String	The method of the AJAX call. This is only used for AJAX call.	'POST', 'GET'	'POST'
content.**data**	Object	The data to pass to the AJAX call. This is only used for AJAX call.	{...}	{}
cookie	Object	Manage cookie for the Notify.	{...}	{...}
cookie.**enabled**	Boolean	Enable/disable cookies.	true, false	false
cookie.**startCount**	Number	How many times the Notify should be shown?	Number	1
cookie.**interval**	Number	How much time the cookie is valid?	Number in days	30
cookie.**name**	String	The cookie name	Cookie name	'speedo-notify-start-count'
cookie.**activateAfter**	String	When to activate the cookie. After view or after close.	'view', 'close'	'view'
container	Object	Change the container style from JS.	{...}	{...}
container.**zIndex**	Boolean, String, Number	The z-index of the container. If this is false the property will be ignored.	z-index	false
container.**background**	Boolean, String	The background of the container. This will replace the original background. If this is false, the property will be ignored.	Any value that can be added to the CSS background property.	false
container.**opacity**	Boolean, String, Number	The opacity of the container. If this is false, the property will be ignored.	Any number from 0 to 1	false
esc	Boolean	Hide the Notify when pressing escape?	true, false	false
closeMode	String	The mode in which the Notify closes. Hides or is completely removed.	'hide', 'unload'	'hide'
autoClose	Number	Wait an amount of time; after that time passed, the notify will close automatically.	number in ms, false	false
autoShow	Number	Wait an amount of time; after which the Notify will, show.	number in ms, false	false
showOnEvent	Boolean, String	Show the Notify when a certain event occurs on the selected element. If this is false, the Notify will be shown directly.	any jQuery viable event	false
effectIn	String	Effect used while the Notify is showing.	'none', 'fade', 'growBox', 'slideLeft',	'none'

			'slideTop', 'slideRight', 'slideBottom', 'slideZoom'	
effectOut	String	Effect used while the Notify is hiding.	'none', 'fade', 'growBox', 'slideLeft', 'slideTop', 'slideRight', 'slideBottom', 'slideZoom'	'none'
css3Effects	String	CSS3 effect used while the Notify is showing or hiding. This effect will override the effectIn or effectOut, in modern browser that supports CSS3 animations.	'none', 'auto', 'random', 'slideTop', 'slideBottom', 'slideLeft', 'slideRight', 'zoomOut', 'flipInHor', 'flipInVer', 'bounceIn', 'pageTop', 'fadeInScale', 'pulse', 'leftSpeedIn', 'rollIn', 'rollOut'	false
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 Notify completely (Remove the html code from page). If this is set to false, the Notify will just be hidden, this can be used so the Notify starts faster.	true, false	true
responsive	Boolean	Enable responsive Notify. If this is enabled, the Notify will automatically resize, to fit inside the browser window.	true, false	true
buttons	Object	Add custom buttons to the Notify.	[{html: 'About', action: function () { alert('Speedo Notify'); }, class: 'button-test'}, ...]	null

onBeforeShow	Function	Called before the Notify is showing.	function () { ... }	function(){}
onShow	Function	Called when the Notify is showing,	function () { ... }	function (){}
onComplete	Function	Called after the content, finished loading.	function () { ... }	function (){}
onHide	Function	Called when the Notify is hiding.	function () { ... }	function (){}
onClose	Function	Called when the Notify 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

Speedo Notify can handle multiple content types. The content is easy to manage.

To set the content using a JavaScript call, you'll need to use the following structure:

```
$(this).speedoNotify({
    content:{
        type: "html",
        alignment: "center",
        value: "<p>Simple HTML content</p>"
    }
});
```

To set the content using a Direct Link, you'll need to use the following structure:

```
<a href="#" class="speedo-notify" data-speedo-options='{"content": {"value": "Simple
Content"}}'>Show Notify</a>
```

Note: You can also pass the content on the href.

To set the content using the speedo-notify tag, you'll need to use the following structure:

```
<speedo-notify mode="link" options='{"content": {"value": "Simple Content"}}'>Show
Notify</speedo-notify>
```

Note: You can also register settings with content and pass them to the Direct Link or speeod-notify tag.

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 Notify content.	Low
SWF 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 Notify comes with 10 predefined themes/skins:

1. default
2. dark
3. facebook
4. lime
5. metro
6. notify
7. blueglass
8. darkglass
9. notify-glass
10. ignito

The list above is composed from the default themes available in jQuery Speedo Notify v1.0. Speedo Notify 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="source/skins/metro/metro.css" />
    ...
</head>
```

Now, when you call the Notify, you need to specify the theme:

```
$(this).speedoNotify({theme: "facebook"})
```

2.5 Effects

Speedo Notify comes with 7 jQuery predefined effects, and 14 CSS3 effects.

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

```
$(this).speedoNotify({
    effectIn: "fade",
    effectOut: "fade"
});
```

The following predefined jQuery effects are available:

1. none
2. fade
3. growBox
4. slideLeft
5. slideRight

6. slideTop
7. slideBottom
8. slideZoom

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

```
$(this).speedoNotify({
    effectIn: "fade",
    effectOut: "fade",
    css3Effects: "slideTop"
});
```

Note: We are still using jQuery effects just for old browsers that doesn't support CSS3.

The following predefined CSS3 effects are available:

1. none
2. auto
3. random
4. slideTop
5. slideBottom
6. slideLeft
7. slideRight
8. zoomOut
9. flipInHor
10. flipInVer
11. bounceIn
12. pageTop
13. fadeInScale
14. pulse
15. leftSpeedIn
16. rollIn
17. rollout

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

2.6 Custom Buttons

Speedo Notify lets you add custom buttons to the Notify. You can add an unlimited number of buttons, each button can be customized to look however you want. You also can add any custom attribute you want on the button.

The buttons can also be added using a Direct Link and a speedo-notify tag.

You can create custom buttons using the following structure:

```
$(this).speedoNotify({
    buttons: [{
        html: "About",
        action: function ()
        {
            alert("This is a button sample, for Speedo Notify.");
        },
        class: "about-button"
    }]
});
```

The 'html' and 'action' property are required for every button, any other property will be added to the button as an html attribute.

2.7 Direct Link

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

```
<a href="image.jpg" class="speedo-notify">Show Notify</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-notify">Show Notify</a>
```

You can also register settings and use them using the **data-speedo-use** attribute:

```
<a href="image.jpg" data-speedo-use="sample-notify" class="speedo-notify">Show Notify</a>
```

2.8 Speedo Notify Tag

You can use the Speedo Notify tag to show the Notify, create a link to show, hide or toggle the Notify.

```
<speedo-notify mode="show" options='{"width": 640, "height": 360, "theme": "metro"}' />
```

To create a link to show, hide or toggle the Notify, use the following structure:

```
<speedo-notify mode="link" action="show" options='{"width": 640, "height": 360, "theme":
"metro"}'>Show Notify</speedo-notify>
```

There are 3 actions that can be used using a link:

Action	Description
show	Show the Notify.
hide	Hide the Notify.
toggle	Show/hide the Notify.

You can specify a Notify using the attribute use and select a registered notify or an instance:

```
<speedo-notify mode="link" action="show" use="sample-notify">Show Notify</speedo-notify>
```

To register a Notify see the next section.

You can also pass custom classes using the class attribute just like on any element:

```
<speedo-notify mode="link" action="show" class="sample-class" use="sample-notify">Show
Notify</speedo-notify>
```

2.9 Register Notify Options

You can register Notify options to use them later. To register you can use the following structure:

```
speedo().notifyOptions.register('sample-notify');
```

You can unregister a registered Notify using the following code:

```
speedo().notifyOptions.unregister('sample-notify');
```

You can update a registered Notify using the following code:

```
speedo().notifyOptions.update('sample-notify', {...});
```

You can get a registered Notify using the following code:

```
speedo().notifyOptions.get('sample-notify');
```

2.10 Using Cookies

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

```
$(this).speedoNotify({
    cookie:
    {
        enabled: true,
        startCount: 5,
        interval: 1,
        activateAfter: "close"
    }
});
```

3 Working with advanced features

Speedo Notify 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 Notify, you need to get the instance of the Notify, to do that you just need to assign the return of the plugin to a variable:

```
$(function (){
    var myNotify = $.fn.speedoNotify({…});
    window.myNotify = myNotify; // Make this variable is public.
});
```

Now, you can use the variable myNotify to handle the Notify functionality.

3.2 Show/Hide Notify

You can show/hide the Notify by calling showNotify, hideNotify or toggleNotify:

```
myNotify.hideNotify();
myNotify.showNotify();
myNotify.toggleNotify();
```

These functions have no parameters and no return value.

3.3 Set Content

You can set content programmatically through the function setContent:

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

This function has tree parameters:

- content – The content
- type – The type of the content this can be any of the following:
    - "auto"
    - "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:

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

These functions has 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 Set Position

You can set a new position programmatically using the function setPosition:

```
myNotify.setPosition("bottom");
```

This function has one parameter:

- position- The position of the notify. This can have one of the following values:
  'top', 'right', 'bottom', 'left'

The function has no return value.

3.6 Slide Page

You can slide the page so the Notify doesn't cover the page content. To do this you can use the function slidePage:

```
myNotify.slidePage();
```

This function has one parameter:

- hide – If this is true the page will be restored to its original position, otherwise it will slide it so the notify will fit.

This function has no return value.

3.7 Set/Get Options Programmatically

You can set or get options through the function options.

To set a specific option, you can use the following structure:

```
myNotify.options("responsive", true);
```

To set multiple options, you can pass an object to the function:

```
myNotify.options({responsive: true, closeCaption: "close"});
```

To get a specific option, you can pass the name of that option and it will return its value:

```
var is_responsive = myNotify.options("responsive");
```

To get all options as an object, you can call the function without any arguments:

```
var options = myNotify.options();
```

This function has two parameters:

- name – The name of a single option to get or set. If this parameter is an object, the function will set all the options from within that object. If this parameter is a string and value is undefined, the function will return the value from the specified option.
- value – The value to set to a specific option. If this parameter is undefined and the name is a string the function will return the current value of that option. If name is an object, this parameter will be ignored.

Return value:

If the name is a string and the value is undefined, the function will return the current value of that option, otherwise the function will return an object with all the options.

3.8 Get box size

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

```
var boxSize = myNotify.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}

3.9 Check if the Notify is Visible

You can check if the Notify is visible by using isVisible:

```
if (myNotify.isVisible())
{
    ...
}
```

This function has no parameter.

Return Value:

The function returns true when the Notify is visible and false otherwise.

3.10 Using callback

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

```
$(this).speedoNotify({onClose: function (){  }});
```

For a list of callbacks see the options section.