

/Flash/optimise your site for Google

Flash websites should work well with Google, the iPhone and the back button. Paul Falgout of Nashville-based digital agency Otterball (otterball.com) explains how to make sure they do

Knowledge needed Flash, PHP, JavaScript

Requires Flash CS3, PHP

Project time One hour

Designers often have misgivings about using Flash in a project because of SEO concerns. But while many Flash websites aren't very SEO friendly, this shouldn't be used as evidence against using Flash – only that the steps necessary to make Flash play well with Google haven't been implemented.

At Otterball, we're keen to make sure that our Flash sites are visible to search engines, as well as the iPhone. In this tutorial, we'll address how to do so, including giving each page of your Flash site a unique URL and making sure it works with the browser's back button.

Before we start, I should point out that this tutorial in no way relies on Google's most recent attempts to index SWF files. In fact, on our sites we typically disallow Google from indexing SWF files. This is because, instead of indexing worthwhile content applicable to the site, Google tends to find all of the test content and lorem ipsum text that we've lazily left in our libraries. Since our Flash sites read content from a database, there's little to no valuable content inside of our SWFs.

To make things worse, Google results that link to SWF files are unlikely to link to the page content that the result references, and you'll lose any of the formatting or flashvars passed to the SWF from the HTML file.

Where's the proof?

For years we've been led to believe that there's no realistic solution to making a Flash website fully SEO-compatible. Consequently, when we built the Otterball portfolio website, we were very concerned with being visible by Google.

We wanted each page of the site to be indexed and searchable within Google, but more importantly we wanted to showcase our interactive Flash development.

We accomplished this by utilising the JavaScript library swfobject. Since the Adobe release of Flash Player 10, Adobe has officially recommended this library for Flash Player detection. (adobe.com/devnet/flashplayer/articles/swfobject.html).

Playing nicely The HTML version of our portfolio. Each thumbnail displayed on the Flash site is visible, with titles beneath. This version is displayed when the browser does not have Flash capabilities, and is specifically formatted for use on the iPhone



This library helps to best embed Flash across all browsers, but most importantly for this tutorial, it enables us to utilise alternate content (meaning the content displayed when Flash is not available).

It's freely distributed under the MIT licence and is available for download at code.google.com/p/swfobject/. It detects if Flash is available, and if it is, replaces the HTML content with the Flash object.

The file `step1.html`, which you can find on this issue's CD, implements a very simplified example of the swfobject library.

```
<script type="text/javascript" src="/javascripts/swfobject.js"></script>
...
var flashvars = false;
var params = { base: "/", scale: "noscale" };
var attributes = { id: "flashMC" }
swfobject.embedSWF("/step1.swf", "flash_div", "100%", "100%", "8.0.0", false,
flashvars, params, attributes);
```

First link to the `swfobject.js` on your server, and set up the swfobject embed code as in the example above. Note the id in the attributes array, and the id passed as the second attribute of the embedSWF method. The following HTML body example contains a single div tag, whose id we pass to swfobject. The contents of this div tag are visible by Google and will be replaced by Flash if Flash is available.

```
<body>
<div id="flash_div">
This is SEO Content.
```



On a roll The portfolio page on the Otterball Flash website (www.otterball.com) includes interactive thumbnails that display the appropriate title on rollover



Be specific Singer-songwriter Dave Barnes' site (davebarnes.com/news) updates fans with news. We've found the techniques detailed here effective in connecting Googlers to specific content in the Flash site. For instance try googling 'dave barnes tour dates'

```
</div>
</body>
```

You've now created a Flash site that also contains content that's indexable by Google. Too easy? Perhaps you've already been using this tool or a similar solution, but your alternate content is instead an error message telling the user they don't have Flash. How much better is it to provide content here for search engines and those users who may not be using Flash? One quickly growing group that doesn't is iPhone users, so why not format your SEO content specifically for the iPhone?

This relatively simple step is reflected in [step2.html](#) on this issue's disc. To some degree this can be adjusted to personal preference, as any HTML site is

You've now created a Flash site that also contains content that's indexable by Google

viewable on the iPhone. At this point, it's your choice how much you choose to cater directly to iPhone users.

We've chosen to allow them to scale the page content, and we've set the viewport width to 480, which is the horizontal resolution in the iPhone Safari browser. These settings are accomplished by a single meta tag only utilised by the iPhone and other browsers that are specifically designed to show a portion of a website – so it shouldn't affect standard browsers not using Flash.

```
<meta name="viewport" content="width=480,user-scalable=yes" />
```

Formatting SEO content

Now that we've defined the iPhone settings, we need to format our SEO content. To do so, we wrap our content in a div tag that's nested inside the div tag that we pass to swfobject.

In the example, the `flash_div` width is set to 460px, with a left and right padding of 10px. Note that Safari on the iPhone attempts to choose the best readable size of text, so when formatting your CSS you may also choose to use `css-webkit-text-size-adjust:none`; which disables this auto-adjusting of the font size, allowing you full control of the format.

If you don't have an iPhone, you can use any browser without the Flash plug-in to test your formatting, or simply disable the plug-in in your browser.

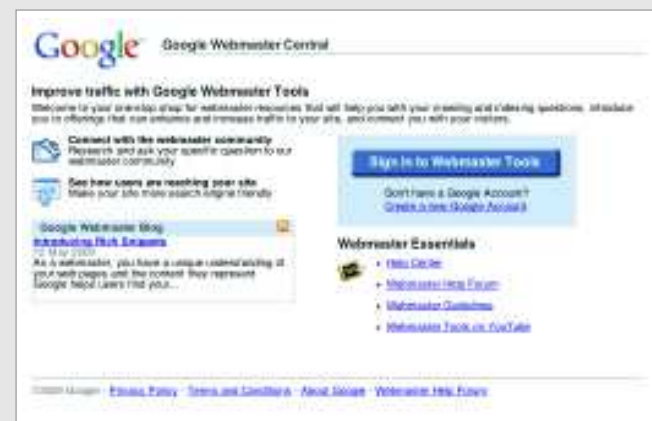
```
<div id="flash_div">
<div id="flash_css">
<div>This is SEO / iPhone Content.</div>
```

Quick fix

The easiest solution is sometimes the best

There's nothing worse than searching for a website on Google and seeing your browser doesn't have the Flash Player within the search results. If you're suffering from this problem, the easiest solution in most cases is to add a meta tag for your description within your head tags. The `<title>` tag and meta tag description `<meta name="description" content="">` are two of the most important elements when dealing with SEO. There's very little reason to not utilise these simple tools. Ideally, your `<title>` tag content is no more than 64 characters, including a keyword-rich, concise and readable summary of the page. Similarly your description should also be readable, include keyword phrases found within your content and be around 200 characters long. An additional easy step would be to add a similar short description before your Flash Player error message.

To help gauge your success and to help you learn how to fine-tune your content to produce better results, Google has provided some useful tools for webmasters at google.com/webmaster. They're free to use and help point out problems with your title or description tags. Additionally, statistics help you determine how users are finding your site on Google, including your conversion rate of clicks vs impressions. You're also able to submit a sitemap directly, which will help Google find and index all of the pages on your website. And if you want to keep Google from indexing a page you can do that here too.



Cool tools You'll find useful tools that reveal how users are finding your site, point out problems with your title tags, and more at google.com/webmaster

```
</div>
</div>
```

We've now made a single Flash page with content visible to Google and the iPhone, but now we need to include navigation code to handle page content and Flash content for different URLs.

To do so we'll utilise a bit of PHP, JavaScript and ActionScript. You'll find these examples in [step3.php](#) on your disc.

```
<?
$page=strtolower($_GET['page']);
if(!$page)$page="home"; //set a default page
?>
```

In this first bit of PHP code we retrieve the variable `page` from the URL query, and if this variable isn't set we give it a default value, in this case `home`. So an example URL that would set our PHP page variable is <http://.../step3.php?page=contact> and in this example <http://.../step3.php> and <http://.../step3.php?page=home> would give the same result. Then, inside the HTML content below, we'll set up a simple menu. Following this menu, we can display page content based on the value of `$page` within PHP.



Friendly URLs

An advanced technique to improve SEO

If your server allows you to use Apache's `mod_rewrite`, included on the CD is a `.htaccess` file, which will help to make your URLs more user-friendly and more SEO compatible. Essentially, this technique removes the query string (ie `?page=`) from your URL, while still passing the variable to PHP.

To really benefit from this technique it's important to make sure your page variables are SEO-friendly. To do this, make sure your page variables are words and separate each work with either an underscore or dash (eg `www.yourwebsite.com/contact-us`).

We've called this an advanced technique, because without a clear understanding of Apache and your server's set-up it can be easy to break your site. However for most users it's likely that simply copying the `.htaccess` along with the tutorial files will work. If you find it causes problems, you can just remove the file. If you intend to pass more than one variable in the URL query, you'll need to modify the file to accommodate multiple variables.



Server source If you're not so familiar with Apache, head for The Apache HTTP Server Project (<http://httpd.apache.org/docs/>) and start getting acquainted properly

Using the hash enables us to update the URL without refreshing the page

```
>> <div id="flash_css">
  <div><a href="/ step3.php?page=home" class="menu">Home</a><a
href="/ step3.php?page=about" class="menu">About</a><a href="/ step3.
php?page=contact" class="menu">Contact</a></div>
</div>
<?
echo $page; //here you would echo your content specific to the $page
?>
</div>
</div>
```

Set up flashvars

Finally, we need to pass this PHP variable to our Flash movie, so within the JavaScript we'll need to setup our flashvars. With `swfobject` you can build an array of flashvars to pass to Flash: however, I prefer to pass flashvars through the `params` array.

This enables me to build a query string, which is easier to generate in PHP. Either method is valid. Variables passed to Flash via this method will be available as a variable in the `_root` for ActionScript 2.

```
var params = { base: "/", scale: "noscale", flashvars: "page=<?=$page?>";
```



Results revealed By googling 'site:www.otterball.com' we see how Google has indexed the pages of `otterball.com`. Each result leads to a different page within the Flash website

Since we're passing the page variable to Flash, we need to get into the ActionScript and handle it. Open `final fla` from the CD. Inside, you'll find two layers on the stage. The layer labelled `as` contains the ActionScript we'll be reviewing, and the layer labelled `menu` contains a simple Flash menu movieclip. Inside the menu movieclip are three frames, each labelled with the appropriate keyword: `home`, `about` or `contact`. These frames represent the different states the menu can take. Additionally, you'll find three movie clips in the library that are exported for ActionScript. Each movie contains the test content for the applicable content page.

```
import flash.external.ExternalInterface;
ExternalInterface.addCallback("gotoFlashPage", this, gotoFlashPage);
function gotoFlashPage(key){
  _root.themenu.gotoAndStop(key);
  _root.main.attachMovie(key,"pg",1);
}
gotoFlashPage(_root.page);
```

In this ActionScript, we've defined the function `gotoFlashPage`, which handles our page variable and navigates the Flash as necessary. In our example we're changing the frame of our menu movieclip and attaching the content movieclip from the library. After defining the function we call it with the current setting of `_root.page`, which is passed in from the HTML. Additionally, using `ExternalInterface` we've given JavaScript access to the function.

So back in our HTML, we'll want to add the function `gotoPage` that calls our Flash function. We'll come back to this function in a moment to add more functionality for our browser history.

```
function gotoPage(p){
  swfobject.getObjectById("flashMC").gotoFlashPage(p);
}
```

Now that we've created our JavaScript function, we can add ActionScript code to each of our buttons to call it, passing the appropriate page variable. Again, we'll use the `ExternalInterface` library to communicate between Flash and JavaScript. This step will complete our handling navigation with Flash.

```
on(release){
  import flash.external.ExternalInterface;
  ExternalInterface.call("gotoPage","contact");
}
```

Unique URLs

The last step of our tutorial will be using the hash to create unique URLs the user can bookmark, as well as creating a history for the browser's back button. This step is the most complicated one, due to having to deal with browser compatibility.

Using the hash enables us to update the URL without refreshing the page. This is essential for our Flash movie and is the same technique used in Ajax sites such as Gmail. The hash appears appended to the end of the URL after the `#` symbol. These steps are reflected in `index.php`.

First, we'll handle Internet Explorer. This technique utilises an iFrame to keep track of the history.

Each time we update the URL, we'll also update the iframe. Changing the location within the iframe adds to the history in Internet Explorer and so as the iframe is updated, the iframe will in turn update the hash. Add the iframe at the bottom of the PHP file, just inside the closing body tag.

```
<iframe src ="/history.php?page=<?=urlencode($page)?>" id="histID"
width="0" height="0" style="height:0px; visibility:hidden;"><!-- -->/iframe>
```

Now we'll want to update the gotoPage function so that it also updates the iframe. We'll include a check to make sure that it only does so if the user is using Internet Explorer.

```
function gotoPage(p){
swfobject.getObjectById("flashMC").gotoFlashPage(p);
if (navigator.appName.indexOf("Microsoft Internet")!=-1) document.
getElementById("histID").src = "/history.php?page="+p;
}
```

Next we need to create a new function for the history iframe to update on changes. It's also necessary to ensure that the page variable being passed is different than the current frame. In this function we'll update the browser hash and the Flash.

```
function goBack(p){
if(location.hash.slice(1)!="/"+p&&location.hash.slice(1)){
location.hash = "/" +p;
swfobject.getObjectById("flashMC").gotoFlashPage(p);
}
}
```

On the CD you'll find the history.php file we reference in the iframe. There are two things to note in this file.

The first is the JavaScript function pageLoaded. In this function, if a page is passed to the history iframe, it calls the parent window's goBack function.

Note that the function was added to the body tag's onload event. Now your Flash site should work with the back button in Internet Explorer.

```
function pageLoaded() {
<? if($_GET['page']){ ?> window.parent.goBack('<?=$_GET['page']?>'); <? } ?>
}
...
<body onload="pageLoaded();">
```

The method we'll use to create history for other browsers involves polling the hash to see if it's changed. If it has, we'll then update the Flash site. We'll need to create a JavaScript variable to keep track of the last hash value. We'll call this variable recentHash.

Then we need to update our gotoPage function to update the recentHash variable as it's changing pages.



Sharing sounds musiccityunsigned.com connects fans with Nashville's best unsigned musicians, and benefits greatly from our SEO techniques. Since each page has a unique URL, we've included additional share links, connecting Digg and Delicious

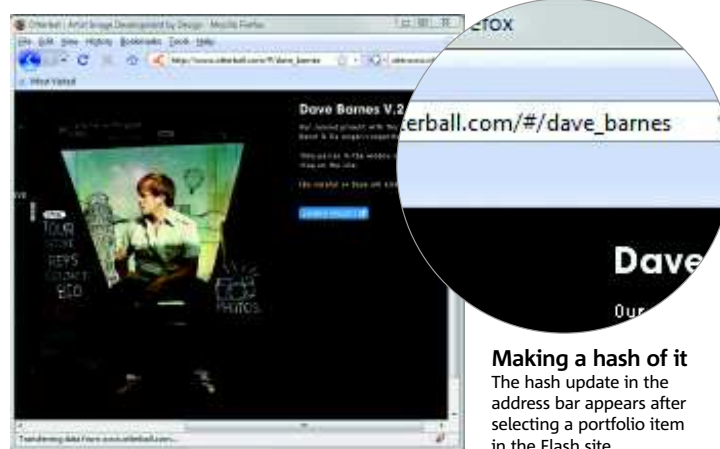
```
var recentHash="";
function gotoPage(p){
recentHash = "#/"+p;
location.hash = "/" +p;
swfobject.getObjectById("flashMC").gotoFlashPage(p);
if (navigator.appName.indexOf("Microsoft Internet")!=-1) document.
getElementById("histID").src = "/history.php?page="+p;
}
```

In this next portion of JavaScript we create the poll using window.setInterval. The interval is set for every 500 milliseconds and each time checks to see if the hash in the address bar differs from the recentHash variable. If it does we'll update the recentHash and the Flash site. You'll notice that we also set a variable called initHash.

This variable is set to the initial page value, so that if the user goes back to a URL that didn't include a hash, we still know which page to change to. Then one additional check redirects the user if the hash is set initially in the address bar. This is an optional step, but it tends to make for more user-friendly URLs. At the completion of this step, all browsers should now have an accurate history from within Flash.

```
var initHash="<?=urlencode($page)?>";
if(location.hash)window.location.href='/?page='+location.hash.slice(2); //
makes incoming hash URLs pretty
if (navigator.appName.indexOf("Microsoft Internet")===-1)
{
window.setInterval(function pollHash() {
if (window.location.hash==recentHash) return;
recentHash = window.location.hash;
if(recentHash)swfobject.getObjectById("flashMC").gotoFlashPage (recentHash.
slice(2));
else swfobject.getObjectById("flashMC").gotoFlashPage(initHash);
}, 500);
}
```

With each step complete, the once-limited Flash site is more accessible and more functional. Soon each of the pages within your website will be indexed by Google, viewed on the iPhone, and bookmarked by users. Successfully implementing these practices will not only improve your own projects, but it will help to lessen the client's apprehension concerning the use of Flash. ●



Making a hash of it
The hash update in the address bar appears after selecting a portfolio item in the Flash site



About the author

Name Paul Falgout
Site otterball.com
Areas of expertise Design, Flash, user experience
Client Warner Bros, EMI, Centricity Records
Favourite film of 2009 so far Haven't seen any yet this year, but I'm looking forward to the new Terminator