

[Adobe Developer Connection](#) / [Fireworks Developer Center](#) /

Rapid interactive prototyping with HTML, CSS, and JavaScript using Fireworks and Dreamweaver CS4

by [Mariano Ferrario](#)



[Fluid, Inc.](#)

by [David Hogue, Ph.D.](#)



Content

[Getting started with the sample project](#)
[Adding HTML to a Fireworks document to display Google maps](#)
[Embedding a SWF slide show](#)
[Incorporating HTML, CSS, and JavaScript in a master page](#)
[Creating a search box in the prototype](#)
[Setting up the search results page to](#)

— Requirements

Prerequisite knowledge

Basic familiarity working with Fireworks is required. Some prior experience working with Dreamweaver is also recommended.

User Level


All

Required products

Fireworks ([Download trial](#))

Dreamweaver ([Download trial](#))

Sample files

 [daves_dogwash_prototype.zip](#)
(2080 KB)

This article describes how to add HTML, CSS, and JavaScript to Fireworks documents to create enhanced functionality in the HTML pages that you export. These HTML pages are then used to facilitate [rapid prototyping](#) and usability testing.

This sample project is presented in several sections. We'll begin by describing how to add simple HTML code directly within a Fireworks document to embed a Google Map interface using an `iframe`. We'll also discuss how to embed a custom Flash SWF slide show into Fireworks pages. These two sections are especially helpful if you do not have prior programming experience but wish to add simple functionality to wireframes and design comps to simulate the intended functionality and interactivity in a product prototype.

The section, "[Incorporating HTML, CSS, and JavaScript in a master page](#)," builds upon these techniques and the following sections extend the project by moving the HTML and JavaScript code outside of the Fireworks document. We'll link the HTML to external CSS style sheets and JavaScript files. We'll also leverage the code included in the jQuery JavaScript Library. The instructions cover how to create an HTML overlay (a "quick view" pop-up window that is displayed when the user rolls their mouse over a product image) on a search results page. The main goal of the remaining sections in this article is to build a richly interactive prototype that is easy to manage and update by using a modular approach to design, manage, and organize files. The emphasis on modularity is essential; if you take the time to set up your prototype files correctly, your workflow will be much more efficient and productive.

This project is based on a fictitious website called Dave's Dog Wash (see Figure 1).

[display the search query](#)

[Customizing a JavaScript toggle function](#)

[Using DIV tags to structure the search results pop-up windows](#)

[Loading external HTML files](#)

[Positioning page elements with CSS](#)

[Adding custom JavaScript to hotspots to create interactivity](#)

Created

6 April 2009

Page tools

[Share on Facebook](#)

[Share on Twitter](#)

[Share on LinkedIn](#)

[Print](#)

[Dreamweaver CS4](#) [Fireworks CS4](#)

[HTML](#) [JavaScript](#) [prototyping](#)

Was this helpful?

Yes No

By clicking Submit, you accept the [Adobe Terms of Use](#).

Thanks for your feedback.

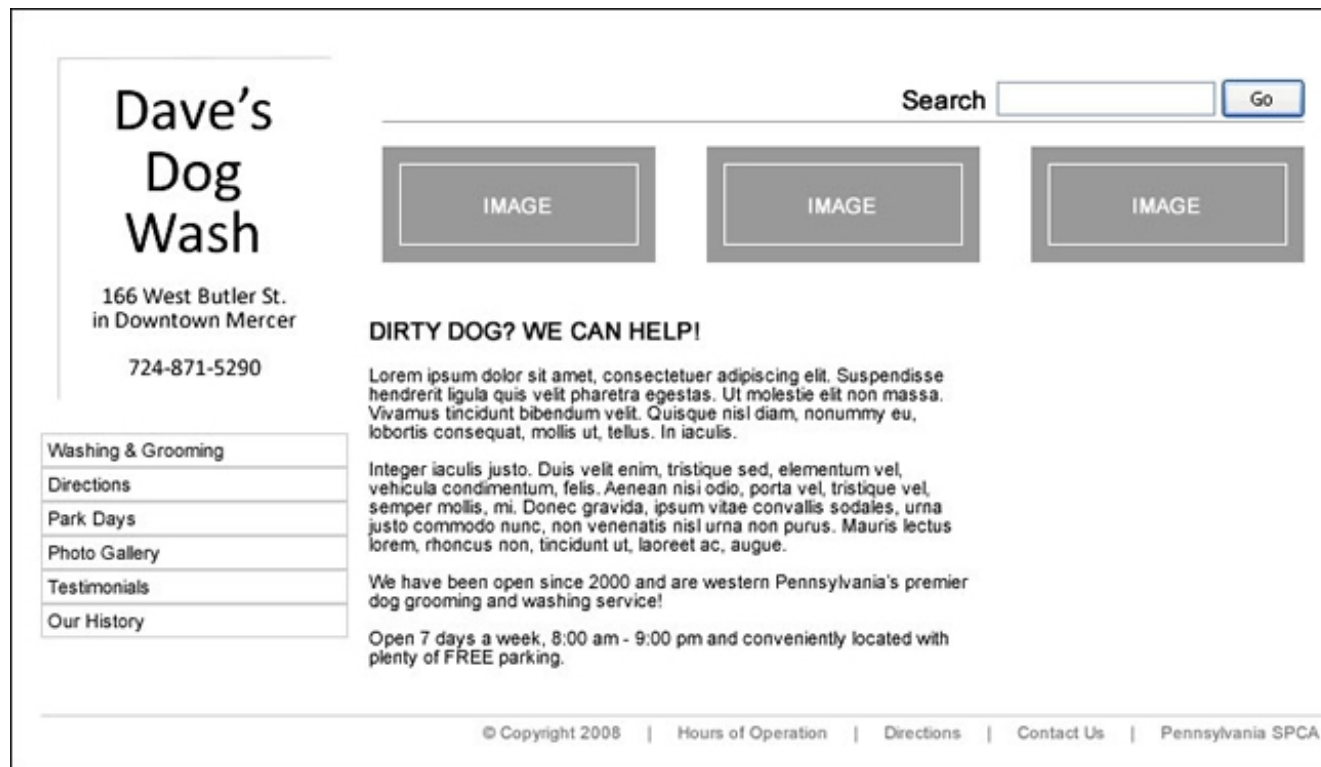


Figure 1. Prototype of the Dave's Dog Wash website

Getting started with the sample project

If you haven't already, be sure to download the sample files provided on the [first page](#) of this article. The following sections include step-by-step instructions to create the interactive prototype. We'll start this project by opening the main prototype template PNG file that contains the existing wireframe for the project, so that we can focus on adding the interactivity and functionality to the prototype.

It is important to have a thorough understanding of the folder structure of the prototype. In this section we'll analyze the contents of the ZIP file to become familiar with the organization of the folders and files, and provide some insight into the strategy of maintaining modular code.

After you unzip the downloaded source file to a location on your hard drive, open the main prototype folder. In the main prototype folder, you'll see six folders and a list of HTML files:

- bubble_S_A
- bubble_S_B

- bubble_S_C
- flash
- images
- scripts

These folders and files will be used to create the sample prototype (see Figure 2).

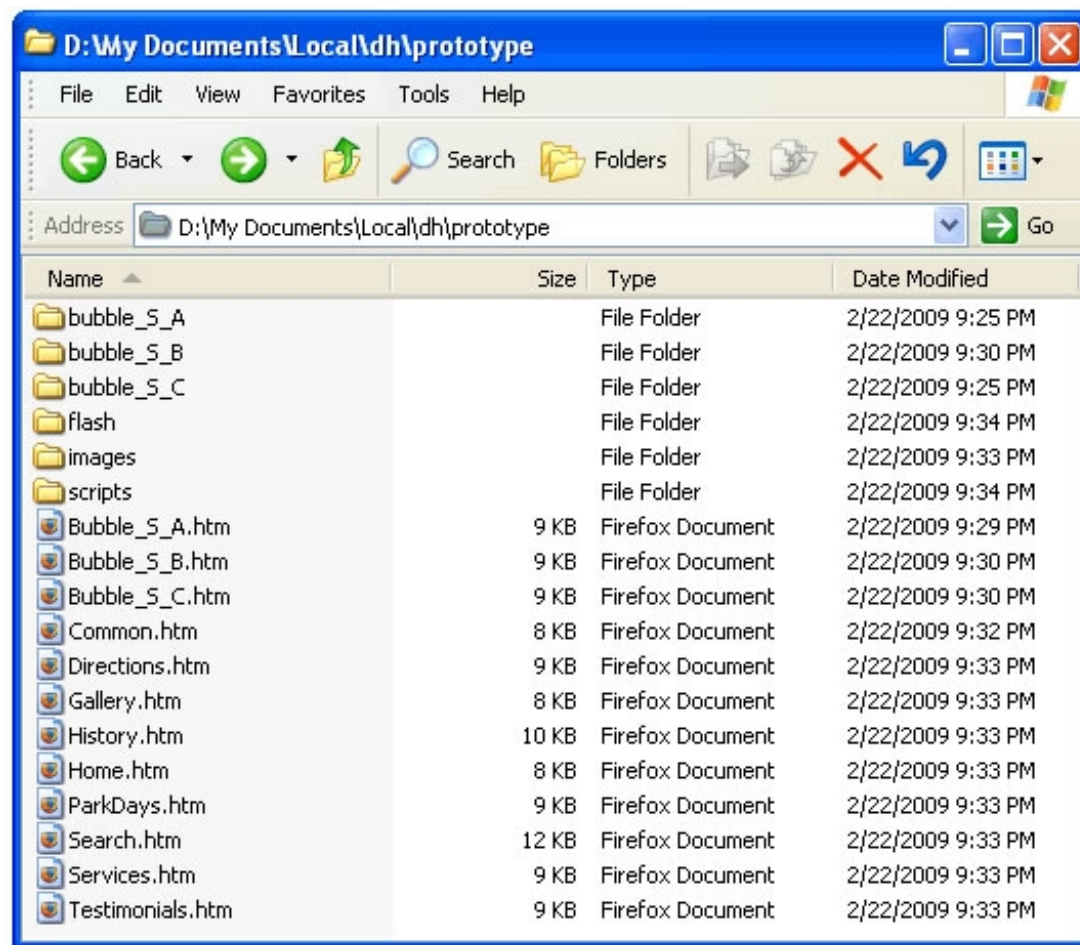


Figure 2. Contents of the prototype folder

The three bubble folders contain the image files for each individual bubble (or pop-up window) overlay that will be created later in this tutorial. This structure will make more sense when we cover that information. For now, just notice that each bubble HTML file has a corresponding images folder. (For example, the bubble_S_A folder contains the images for the Bubble_S_A.htm file.)

The source file for each bubble overlay can be found in the `bubblePrototypeTemplate.png` located in the Template Files folder. Editing the `bubblePrototypeTemplate.png` source file is outside the scope of this article, but the PNG file is provided so that you can review the Fireworks document if you'd like to see how it is set up.

The images folder contains all of the images for the main prototype and the flash folder contains the Flash SWF file that we will use to embed in the Fireworks document.

Finally, the scripts folder contains a series of JavaScript and CSS files that will be used to add functionality to the prototype. As we begin this project, the files in the scripts folder are blank except for the `jQuery.js` file. We'll add custom code to these files in the final sections of this article.

Adding HTML to a Fireworks document to display Google maps

In this section, we'll add HTML code generated by Google Maps into a HTML slice on the Directions page of the website.

Open the `mainPrototypeTemplate.png` source file:

1. In Fireworks, select File > Open.
2. In the Open Document dialog box, navigate to the location where you downloaded the source files and select the `mainPrototypeTemplate.png` file from the `Template_Files` folder.
3. Click OK.

Once the file is open, open the Pages panel. There are multiple pages in the Fireworks document that correspond to each page of the fictitious website:

1. In the Pages panel, select the Directions page (see Figure 3).



Figure 3. Directions page in the Pages panel

The Directions page contains a wireframe-style map of Dave's Dog Wash location in downtown Mercer. This placeholder map will be replaced with an interactive Google map for the working prototype. Notice that we have designed the placeholder map in the original wireframe document so that if the file is exported as a static bitmap image (such as a JPEG, GIF or flattened PNG file), the wireframe will still look complete and display a nice visual representation of the map. However, for the interactive prototype we want to show a live Google Map, so we need to insert some simple code into the Fireworks file that will appear in place of the placeholder map when the Fireworks page is exported as HTML.

2. From the Tools panel, select the Slice tool (k).
3. Draw a rectangular slice that covers the entire image of the map (see Figure 4).

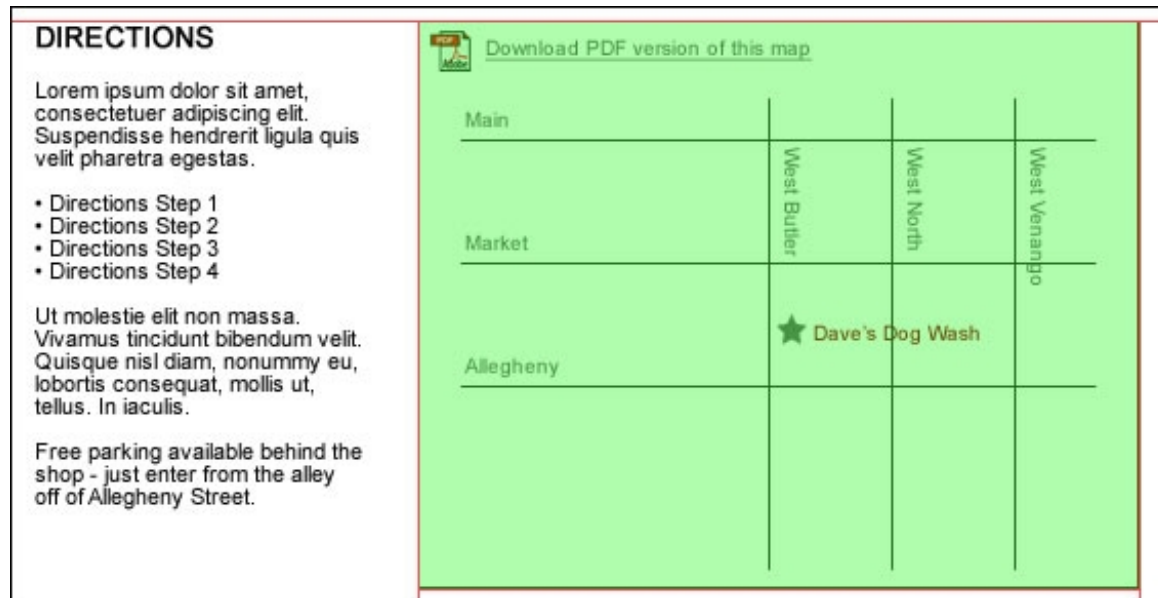


Figure 4. Using the rectangular slice tool to draw a slice that covers the map in the wireframe

Note: To quickly add slices, select the object using the Pointer tool and right-click (or Control-click) on the selected object and choose the option to Insert Rectangular Slice from the context menu that appears.

- After inserting the slice over the map, use the Pointer tool to select the new slice. Check the fields in the Property inspector to ensure that the slice's dimensions are 350 × 275 pixels; this is the size of the placeholder map in the wireframe and the Google Map in the prototype will be the same size. If the slice you inserted is not the same size or does not cover the placeholder map completely, adjust the size and position of the slice until it exactly matches the size and location of the placeholder.
- In the Property inspector, change the slice Type from Foreground Image to HTML using the drop-down menu (see Figure 5).

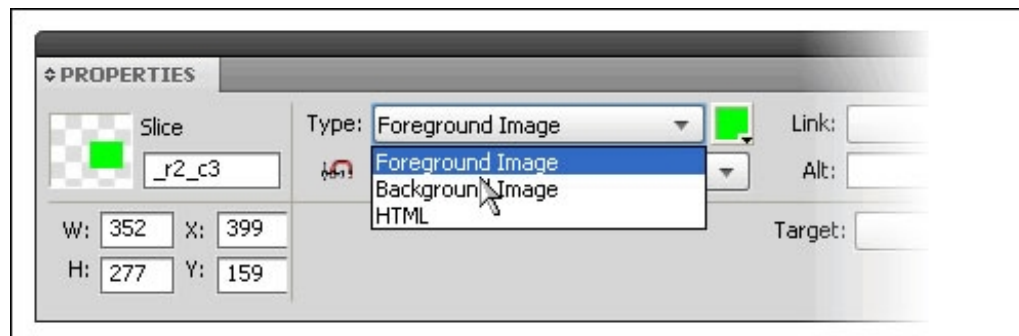


Figure 5. Setting the Type to Foreground Image in the Property inspector

Note: When a Foreground or Background Image slice is changed to a HTML slice and the page is exported as a HTML file, any images beneath the slice will be replaced with the HTML code. This means that any graphics under the HTML slice will not be exported.

In this example, the drawing of the map will be replaced with HTML code.

However, if you export the Fireworks page as a bitmap, the wireframe map will be included and will be displayed in the exported image. This flexibility allows you to create wireframes that contain all of the graphics while also having the ability to export interactive prototypes with working HTML code from the same Fireworks document without making major modifications.

Next, add the HTML code into the new HTML slice on the Directions page:

1. Using the Pointer tool, make sure the HTML slice is selected.
2. In the Property inspector, select the Edit button for the HTML slice (see Figure 6).

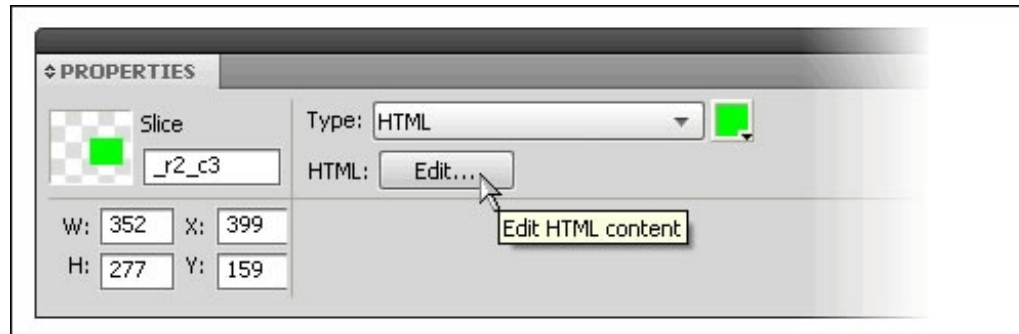


Figure 6. Edit button for editing the HTML slice

3. Copy the following HTML code and paste it into the Edit HTML Slice dialog box:

```
<iframe width="350" height="270" frameborder="0" scrolling="no"
marginheight="0" marginwidth="0" src="http://maps.google.com/maps?f=
q&source=s_q&hl=en&geocode=&q=166+w+butler+,+mercer+pa&
&sll=37.0625,-95.677068&ssp=42.174768,71.455078&ie=UTF8&s=
AARTsJpB36RcGbNvPy4ailBRXRXYQdkwAg&ll=41.234252,-80.234957&spn=
0.017428,0.030041&z=14&iwloc=addr&output=embed"></iframe>
```

After adding the code, the Edit HTML Slice dialog box will look like Figure 7.

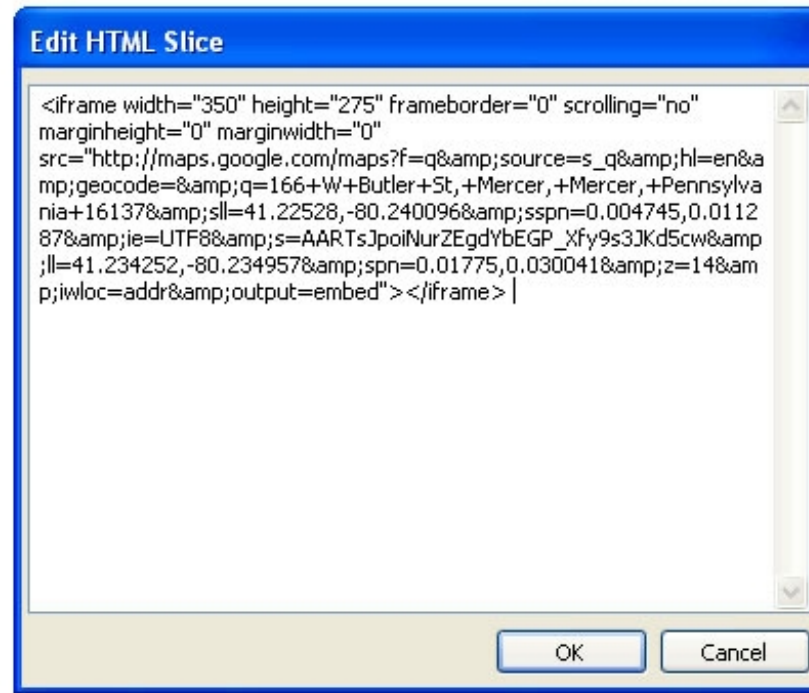


Figure 7. Code is displayed in the Edit HTML Slice dialog box

4. Click OK.

Now export the Directions page as an HTML file to see the results in a browser and verify that the Google Map appears in the correct location and appears correctly:

1. Select File > Export.

Note: The keyboard shortcut for exporting the page is Ctrl+Shift+R (Windows) or Command+Shift+R (Mac).

2. Navigate to the prototype folder, to save the HTML file alongside the other existing HTML files.

3. In the Export dialog box, set the following options:

1. Enter the filename: **Directions.htm**.
2. In the Export menu, select the option: HTML and Images.
3. In the HTML menu, select the option: Export HTML File.

4. In the Slices menu, select the option: Export Slices.
5. Check the check box next to the option: Include areas without slices.
6. Check the check box next to the option: Current page only.

Note: This setting will export only the current page so that you can test or preview it; if you uncheck this option Fireworks will export *all* of the pages in the Fireworks document as HTML.

After making these changes, the Export dialog box should look like Figure 8.

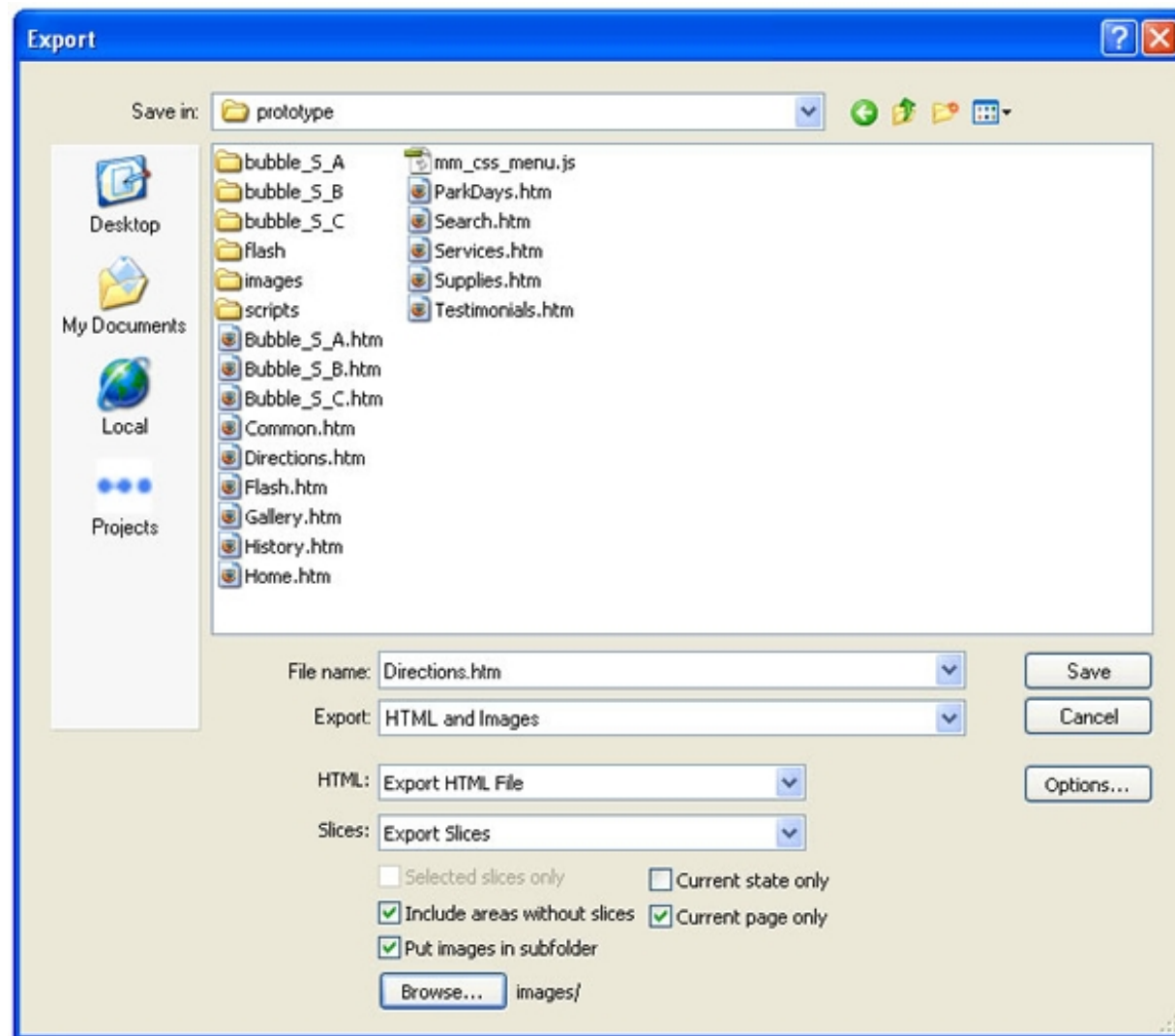


Figure 8. Export options in the Export dialog box

4. Fireworks can separate the image files from the HTML files, which makes it much easier to manage and organize the files exported for your prototype.
 1. Check the check box next to the option: Put images in subfolder
 2. While still in the Export dialog box, select the Browse button to verify the folder for the exported images.
 3. In the Select Folder dialog box, if the selected images folder is not images, then navigate to the images folder in the prototype folder and click the Select "images" button (see Figure 9).

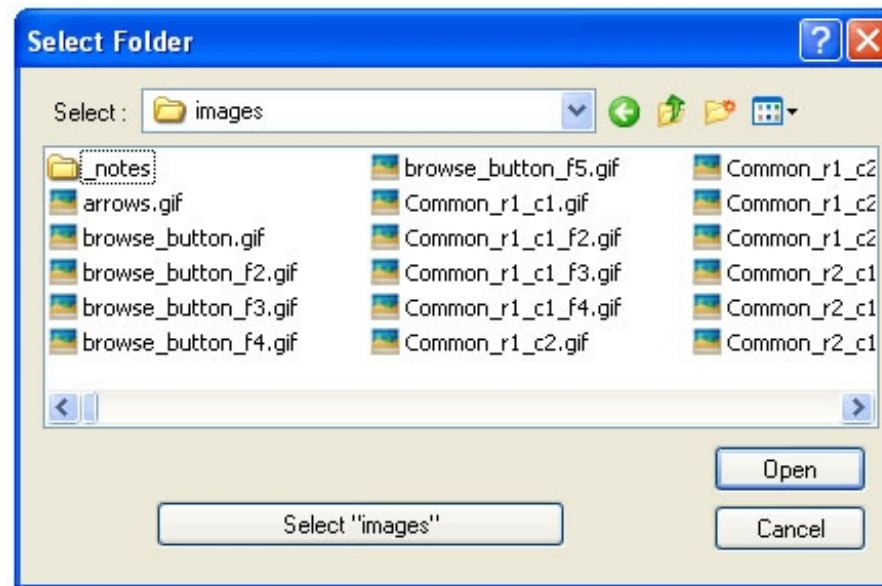


Figure 9. Select "images" button in the Select Folder dialog box

5. Back in the Export dialog box, click Save.
6. In the confirmation box, click OK to replace the existing HTML file.
7. If a second confirmation box appears, click OK to replace the existing image file.

Navigate to the prototype folder and preview the exported Directions.htm file in a browser to see the placeholder map replaced with a live Google Map in the prototype.

Understanding the HTML

The HTML code for the map was copied from the Google Map's website. Custom Google Map code can be easily generated for your own project. After entering the address on Google Maps, look for the "Link" button above the map. Follow the simple directions and Google will generate the map HTML code for you. Simply copy that code from Google and paste it into your Fireworks prototype, but make sure that your Google Map and the HTML slice where it will be

inserted are the same size (see Figure 10).

The screenshot shows the Google Maps website in a Mozilla Firefox browser window. The address bar contains the URL: `http://maps.google.com/maps/empw?url=http:%2F%2Fmaps.google.com%2Fmaps%3Fq%3D166%2Bwest%2Bbutler,%`. The page title is "Google Maps".

1. Customize

Map size

- Small
- Medium
- Large
- Custom

Width Height

2. Preview

The preview shows a map of Mercer, PA with a red location pin. A white information box is overlaid on the map with the following content:

Address:
[166 W Butler St](#)
[Mercer, PA 16137](#)

Get directions: [To here](#) - [From here](#)
[Search nearby](#)

Map Sat Ter

[View Larger Map](#)

3. Copy and paste this HTML to embed in your website

```
<iframe width="350" height="260" frameborder="0"
scrolling="no" marginheight="0" marginwidth="0"
src="http://maps.google.com
/maps?q=166+west+butler,+merc+pa&oe=utf-
8&client=firefox-a&ie=UTF8&hl=en&
```

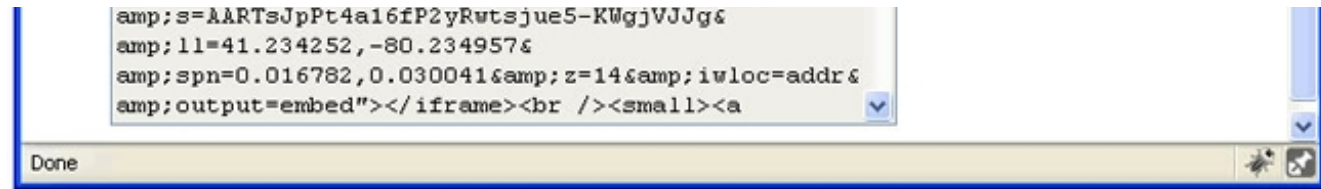


Figure 10. Using Google Map's customize and preview options to generate the map code

The HTML code provided by Google is a snippet that contains an `iframe` that connects to the Google Maps website. An `iframe` allows an external HTML page to be embedded inside another HTML page. In this prototype a Google Map HTML page has been embedded into the Directions page.

Note: In this example, the width and height of the embedded Google Map was customized to match the dimensions of the HTML slice in the Fireworks wireframes document.

To practice using an `iframe`, try changing the `src=` field of the `iframe` code to point to different websites to explore how this functionality can be modified for different types of prototypes. For example, a travel site might display an embedded weather forecast in the prototype.

Embedding a SWF slide show

Another useful way to add custom HTML code into your Fireworks document is to embed a SWF file into a prototype. In this section we'll add a SWF slide show into the photo gallery page of Dave's Dog Wash using the HTML `embed` code for a Flash movie object:

1. In the Pages panel, select the Gallery page. Similar to the Directions page, the Gallery page is fully designed. It includes a placeholder for the slide show in the wireframe. As we saw in the previous section, it is important to include a drawing of the slide show to maintain the integrity of the wireframe when the page is exported as a static bitmap file. The slide show placeholder will be replaced with a custom Flash SWF movie for the interactive prototype.
2. From the Tools panel, select the Slice tool (k).
3. Draw a rectangular slice that covers the entire image of the slide show area (see Figure 11).

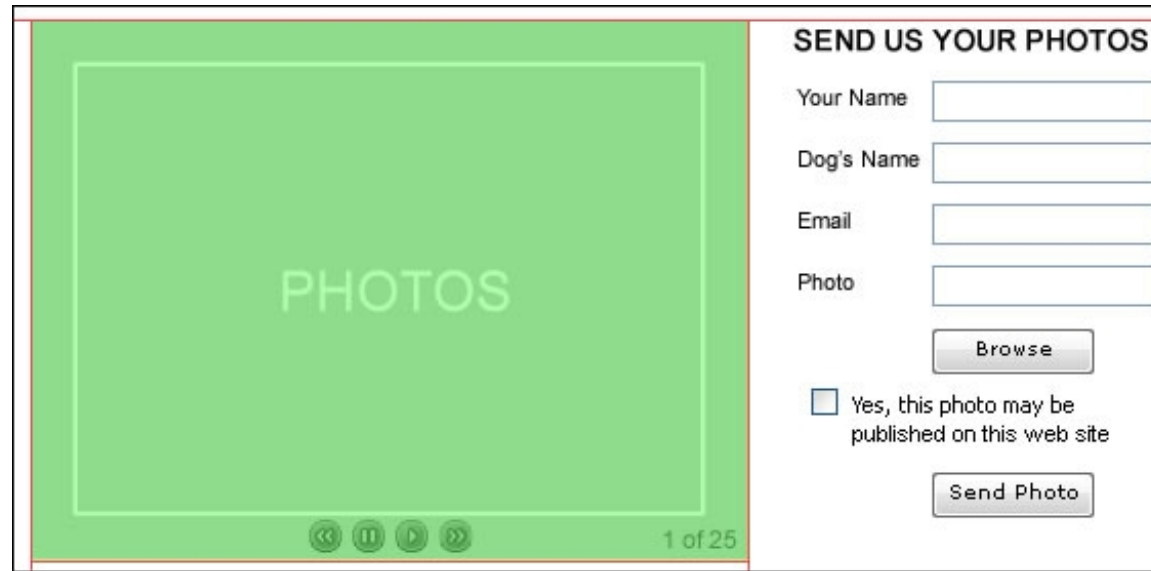


Figure 11. Rectangular slice covering the slide show region in the wireframe

Note: To quickly add a slice, select the placeholder object using the Pointer tool and right-click (or Control-click) on the selected object and choose the option to Insert Rectangular Slice from the context menu that appears.

4. After drawing the slice, use the Pointer tool to select the newly created slice.
5. In the Property inspector, change the slice type from Foreground Image to HTML. Remember that when an image slice is changed to a HTML slice, the image beneath the slice will be removed and replaced with HTML. The HTML slice will contain the background color of main page.
6. Using the Pointer tool, select the slice.
7. In the Property inspector, select the Edit button for the HTML slice.
8. In the Edit HTML Slice dialog box, copy and paste the following HTML code:

```
<object width="350" height="260">
<param name="movie" value="flash/photoGallery.swf">
<embed src="flash/photoGallery.swf" width="350" height="260">
</embed>
</object>
```

After adding the code, the dialog box will resemble Figure 12.

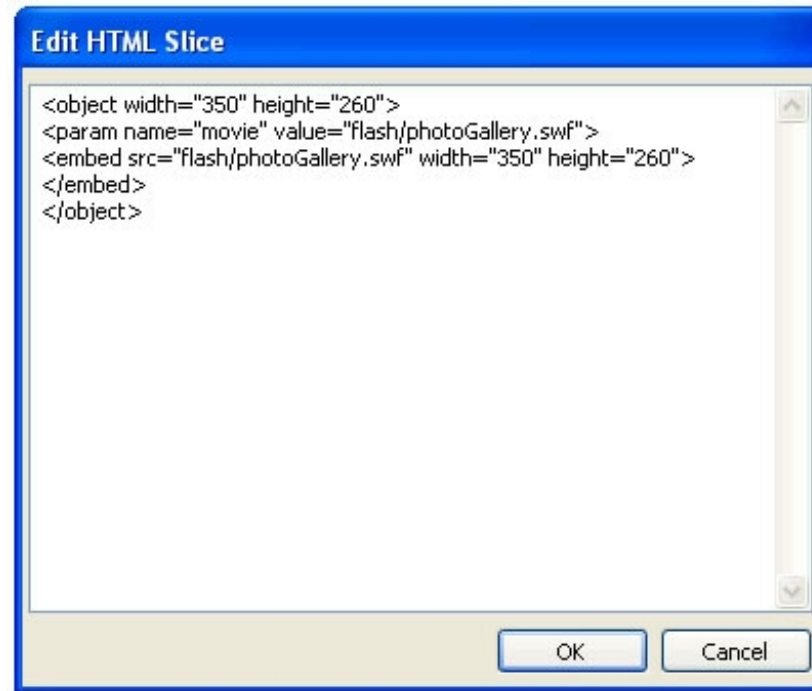


Figure 12. Edit HTML Slice dialog box displaying the pasted code

9. Click OK.

Now export the updated Gallery page as an HTML file:

1. Select File > Export.

Note: The keyboard shortcut for exporting the page is Ctrl+Shift+R (Windows) or Command+Shift+R (Mac).

2. Navigate to the prototype folder, to save the HTML file alongside the other existing HTML files.

3. In the Export dialog box, set the following:

1. Enter the filename: **Gallery.htm**.
2. In the Export menu, choose the option to export: HTML and Images
3. In the HTML menu, choose the option to: Export HTML File
4. In the Slices menu, choose the option to: Export Slices
5. Check the check box next to the option: Include areas without slices.
6. Check the check box next to the option to export: Current page only. **Note:** This setting will cause Fireworks to export only the current page so that you can test or preview it; if you uncheck this option Fireworks will export *all*

of the pages in the Fireworks document as HTML. After making these settings, the Export dialog box should look like Figure 13.

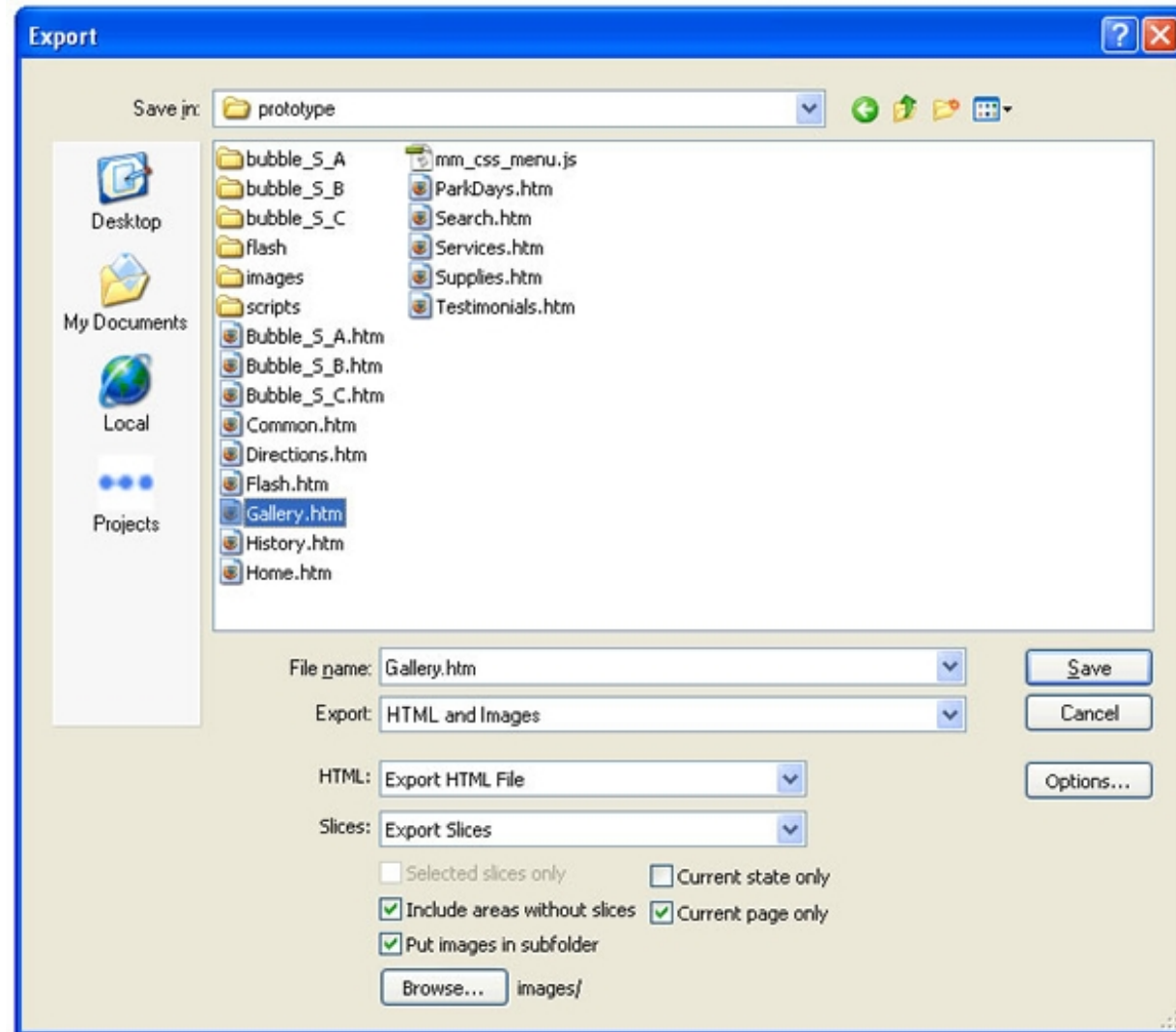


Figure 13. Export options in the Export dialog box

4. Again, it is helpful to ask Fireworks to separate the image files from the HTML files, because it makes it much easier to manage the files exported for your prototype:
 1. Check the check box next to the option: Put images in subfolder.
 2. While still in the Export dialog box, select the Browse button to specify the folder for the exported images.

3. In the Select Folder dialog box, if the selected images folder is not images, then navigate to the images folder inside the prototype folder, select it and click the Select "images" button (see Figure 14).

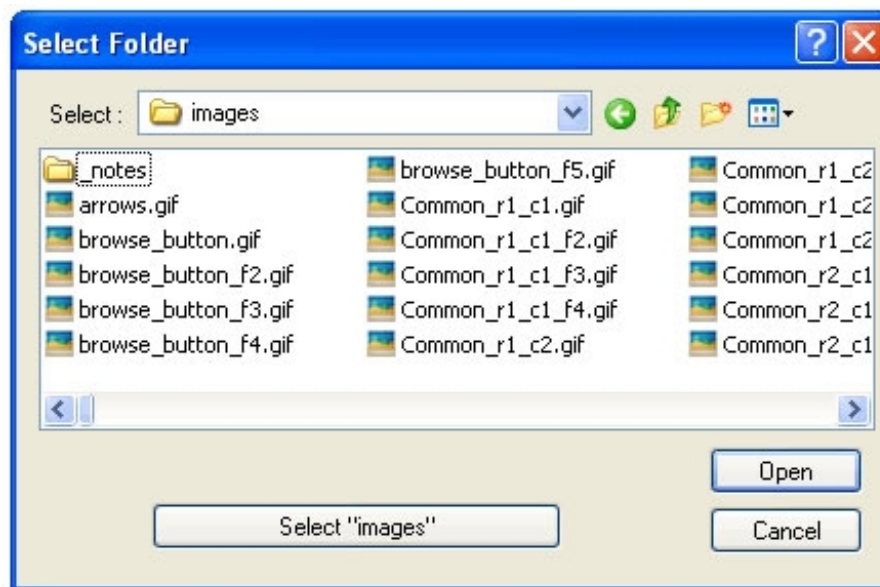


Figure 14. Images folder in the Select Folder dialog box

5. Back in the export dialog box, click Save.
6. In the confirmation box, click OK to replace the existing HTML file.
7. If a second confirmation box appears, click OK to replace the existing image file.

Navigate to the prototype folder and preview the exported Gallery.htm file in a browser to see that the placeholder slideshow has been replaced with a Flash slide show in the prototype.

Understanding the HTML

The code snippet added for the Flash slideshow has both an `<embed>` tag and an `<object>` tag, which is the minimal amount of HTML code needed to embed a SWF movie into a web page. The `<embed>` tag is interpreted by Microsoft Internet Explorer browser, while the `<object>` tag is interpreted by Mozilla browsers (which includes the Firefox browser). Similar to the `iframe` example, the width and height fields of both embed and object tags were changed to match the dimensions of the HTML slice. Finally, the `src` field points directly to the SWF file located in the flash folder in the main prototype folder. If you want to experiment, try placing another SWF file in the flash folder and updating the file name in the links of the code snippet to embed the new SWF file.

Incorporating HTML, CSS, and JavaScript in a master page

The remaining sections of this article describe the process of adding extra functionality to the prototype using JavaScript and the jQuery Library. In this section, we'll create a search results page for the Dave's Dog Wash prototype. The search results page will feature three products in a pop-up window (bubble) that displays additional product information when the mouse is rolled over each product image.

Setting up the master page

First, let's set up the master page for the prototype file. Begin by opening the mainPrototypeTemplate.png file:

1. In Fireworks, select File > Open.
2. In the Open Document dialog box, navigate to the folder containing the prototype source files downloaded from the [first page](#) of this article, and select the mainPrototypeTempate.png file.
3. Click OK. When the file opens, look in the Pages panel to see that there are multiple pages containing the content for each page in the website as well as a single master page (named Common).
4. In the Pages panel, select the Master page named Common (see Figure 15).



Figure 15. Selecting the Master Page named Common in the Pages panel

5. Turn on the visibility of hotspots and slices by either pressing the Number 2 key or clicking the Show slices and hotspots tool located in the Tools panel (see Figure 16).

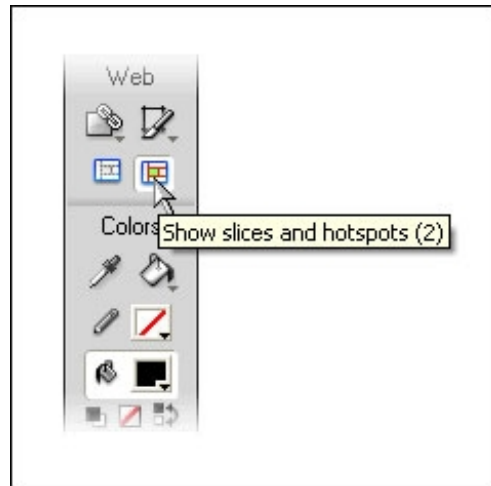


Figure 16. Show slices and hotspots tool in the Tools panel

Adding the global CSS and JavaScript files

Any content or code added to the master page will be displayed across *all* pages in the Fireworks document. Therefore, adding links to the global CSS and JavaScript files on the Master page means that these files will be linked and available from any page within the prototype. The CSS and JavaScript links will be added to HTML slices in the document using the same technique described in the earlier section of this article.

1. Select the Slice tool (k) from the Tools panel.
2. Create a 280 × 30 pixel slice above the image placeholders at the top of the page and position it at x: 190 and y: 25.

Note: It is not always easy to draw an object or slice the exact size and in the exact location desired. You may find it easier to draw a slice in the middle of the page, select it with the Pointer tool, and use the fields in the Property inspector to enter the specific size and location of the object or slice (see Figure 17).

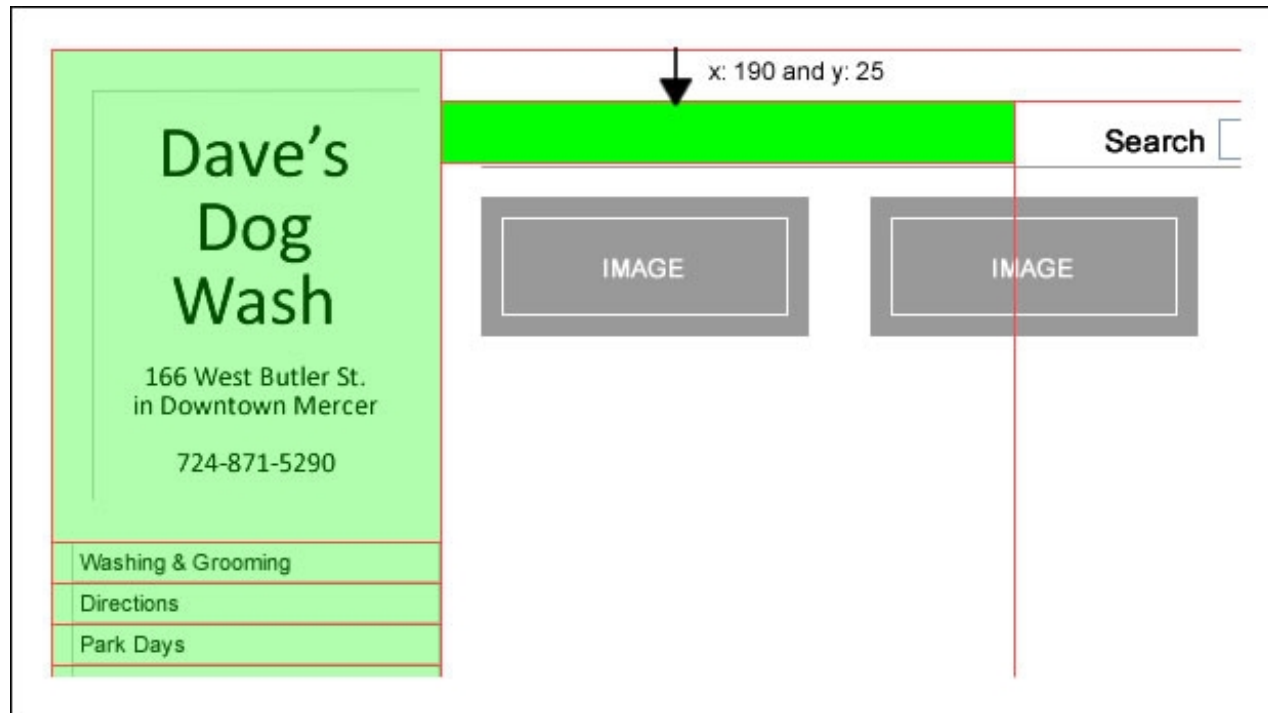


Figure 17. Inserted slice above the image placeholders at the top of the page

3. Use the Pointer tool to select the new slice.
4. In the Property inspector, change the slice type from Foreground Image to HTML using the drop-down menu (see Figure 18).

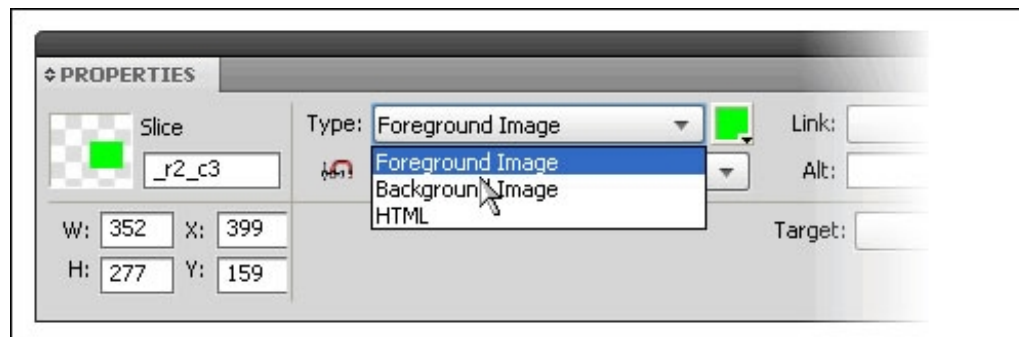


Figure 18. Changing the slice type in the Property inspector

Next, add the HTML to the new HTML slice by following these steps:

1. Select the slice with the Pointer tool.
2. In the Properties panel, select the Edit button for the HTML slice (see Figure 19).

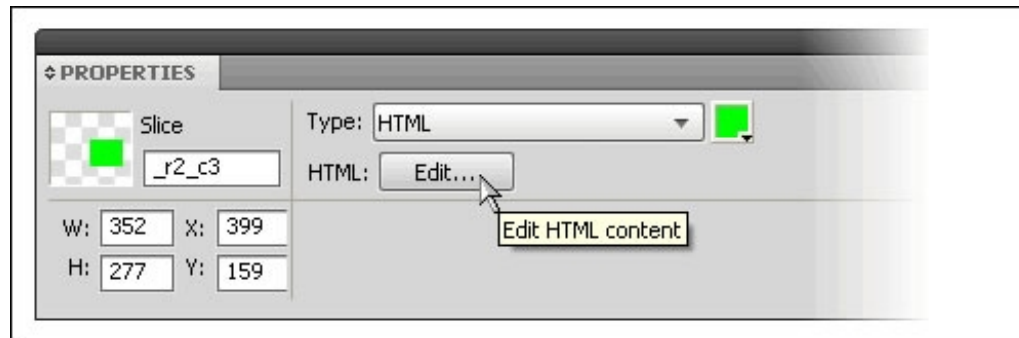


Figure 19. Edit button for editing the HTML slice

3. In the Edit HTML Slice dialog box, add the following HTML code:

```
<script type="text/javascript" src="scripts/jquery.js"></script>  
<script type="text/javascript" src="scripts/myscripts.js"></script>  
<link href="scripts/styles.css" rel="stylesheet" type="text/css">
```

After adding the HTML code, the Edit HTML Slice dialog box will look similar to Figure 20.

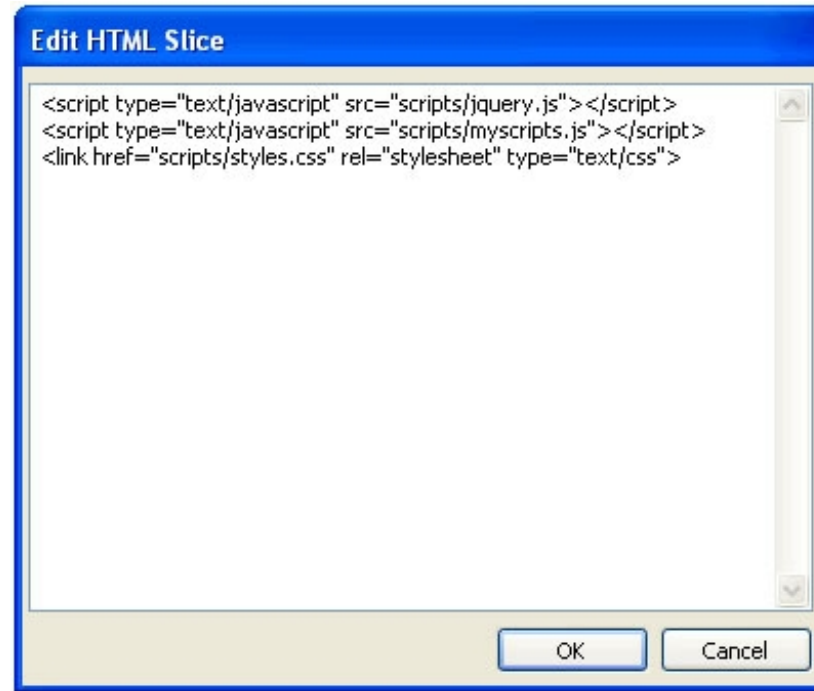


Figure 20. Code snippet displayed in the Edit HTML Slice dialog box

4. Click OK.

Understanding the HTML

The three lines of HTML code are links to external files that will be modified using Dreamweaver later in this tutorial. The following is a brief description of each file that explains why they are important for creating interactive prototypes with Fireworks:

- **jQuery.js:** The first line of HTML code is a link to a JavaScript Library file called jQuery. jQuery has received much attention lately as a very powerful, intuitive, and easy to use framework for adding JavaScript functionality to web applications. jQuery has been included in this prototype to take advantage of the Load feature that makes it easy to import HTML files into the DOM using Ajax. This helps keep the prototype and source files as modular as possible. Editing the jQuery.js file is outside the scope of this article; we've simply added the link to it from the prototype.

Note: The jQuery.js file was modified slightly to include the URL parameter extension. The extension (including full credit and links to the author's website) is included within the jQuery.js file used in this article.

- **myscripts.js:** The myscripts.js file is a JavaScript file containing the global JavaScript functions that can be accessed from any page within the prototype.

- **styles.css:** The styles.css file is a Cascading Style Sheet used to position the HTML elements within the prototype. The advantage of using CSS is that once the prototype is set up correctly, the CSS file can be edited to reposition the HTML elements (such as resetting the location of the pop-up windows, or bubbles) whenever necessary without re-exporting any of the Fireworks pages.

Creating a search box in the prototype

Now that we've added the links to the global JavaScript files in the master page, the interactive search box can also be added to the master page using another HTML slice. By adding it to the master page, you are ensuring that the search box will appear on all pages of the exported prototype. The search box will allow a user to enter a search query in the text field and proceed to the search results page to see their query terms displayed above the search results. Although the search results page will always display the same list of products, the search box interactivity allows the prototype to behave in a more realistic manner.

First, create another HTML slice on the master page:

1. Select the Master Page named Common from the Pages panel.
2. Select the Slice tool (k) from the Tools panel.
3. Create a 300 × 30 pixel HTML slice and position it at x: 470 and y: 25 (see Figure 21).

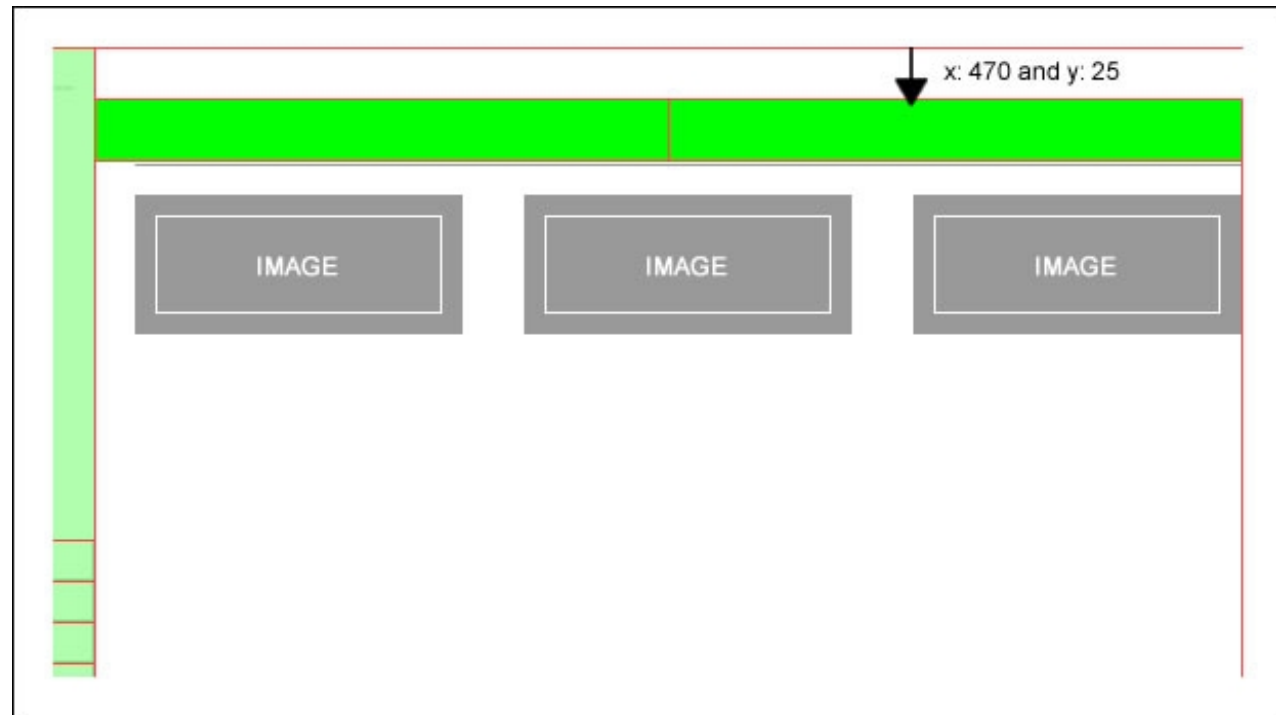


Figure 21. Inserted slice to contain the HTML code for the Search box

Now add the search box HTML code to the new slice:

1. Using the Pointer tool, select the new HTML slice.
2. In the Property inspector, select the Edit button for the HTML slice.
3. Add the following HTML code in the Edit HTML Slice dialog box:

```
<div id="search_box">
  <form id="form1" name="form1" method="get" action="Search.htm">
    <b>Search</b>&nbsp;&nbsp;&nbsp;
    <input type="text" name="search" id="search" size="30" />
    <input type="submit" name="button" id="go" value="Go" />
  </form>
</div>
```

After adding the code, the Edit HTML Slice dialog box will look like Figure 22.

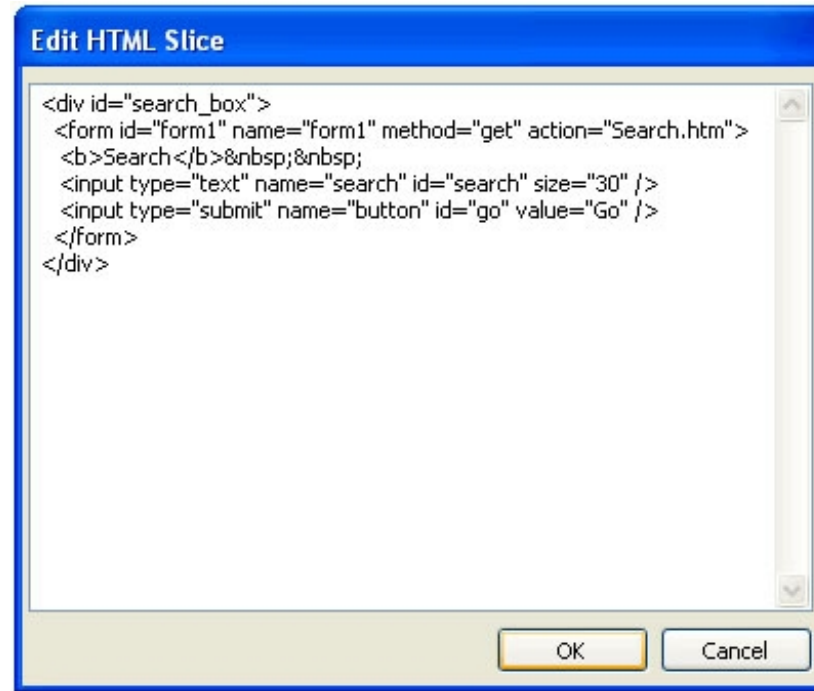


Figure 22. Code snippet is displayed in the Edit HTML Slice dialog box

4. Click OK.

Understanding the HTML

The new snippet of HTML code added includes two features:

- DIV tag wrapper
- HTML form field containing a text field and a submit button labeled Go. The form tag's action parameter links to the Search.htm page.

The DIV tag with an `ID` property is used to position the search box element using CSS. CSS positioning is used in order to make it easy to change an object's location in the browser without requiring you to export every page in the Fireworks document again. You can simply change the code in the CSS style sheet and all of the pages will update.

Using CSS to position the search form field

In this section, we'll use Dreamweaver to edit the CSS file and set the position of the search form field and submit button defined in the inserted HTML code:

1. Launch Dreamweaver.

2. Select File > Open.
3. In the Open Document dialog box, select the styles.css file that exists in the scripts folder inside the prototype folder.
4. Click OK.

Now, in the CSS document:

1. Copy and paste the following code under the search header section:

```
#search_box
{
  position:absolute;
  left:440px;
  top:30px;
  height:28px;
}
```

Note: Pasting the CSS code within the search header section of the styles.css file is not necessary for the code to execute properly in this example, but taking care to place code in the corresponding section results in a well organized file and makes the CSS rules much easier to find.

2. Select File > Save.

The "search_box" DIV tag and all of its contents are now positioned absolutely in the browser window 30 pixels down from the top edge and 440 pixels from the left edge. This will place the search box in the same position where the search form placeholder appears in the wireframe.

Setting up the search results page to display the search query

All of the common external files have now been linked to the prototype and the code is included in the master page. Our next goal is to modify the search results page so that the user's search query is displayed:

1. In Fireworks, open the Pages panel.
2. Select the Search page.

Displaying the search query

When a user enters a search term into the search field at the top of the page and clicks Go, the search term is passed to the search results page using the `GET` method form action. This means that the user's search term is passed to the next page via the browser's URL string, and the search term can be seen appended to the URL in the address bar. JavaScript can then be used to retrieve the search term from the URL and display it on the search results page:

1. Insert an HTML slice that is 140 × 30 pixels using the process described previously. Position the slice at x: 220 and y: 150 (see Figure 23).

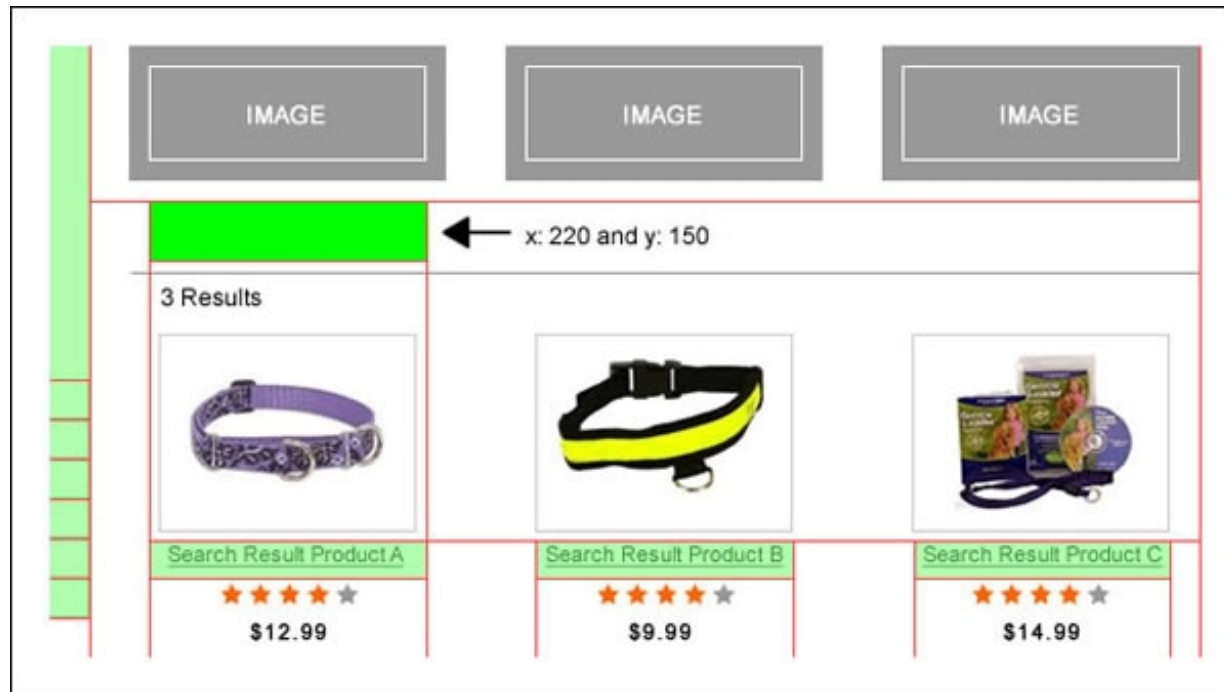


Figure 23. New HTML slice inserted in the search results page

2. Edit the HTML slice as described previously and add the following HTML code:

```
<script language="javascript">
    doSearchText ();
</script>
```

Understanding the HTML

The HTML code pasted into the slice contains a block of JavaScript code that calls the `doSearchText ()` function when

the page is rendered. The `doSearchText()` function will parse the URL using jQuery and display the search term on the browser page in the specified location.

To create the JavaScript function, we'll return to Dreamweaver and edit the global `myscripts.js` file. Follow these steps:

1. Launch Dreamweaver.
2. Select File > Open.
3. In the Open Dialog box, select the `myscripts.js` file located in the `scripts` folder of the prototype directory.
4. Click OK.
5. Add the following JavaScript code to the file:

```
//Parses the search query from the URL and writes it out to the page.
function doSearchText() {

    //Get the search parameter from the URL using a jQuery extension
    var valFinal = $.getUrlParam("search");

    //Clean the returned query term
    valFinal = valFinal.replace(/%2B/g, "&#43;");
    valFinal = decodeURIComponent(valFinal).replace(/\/+/g, " ")

    //Write out the query term to the browser.
    document.write('<div id="searchResults">Search Results for "' + valFinal + '"<
}
```

6. Select File > Save.

The JavaScript function writes out the query term to the browser (as seen in the last line of code in the function) and wraps the text inside the DIV tag named `searchResults`. This snippet of HTML code may be positioned and styled using CSS so that the visitor's search query appears in the correct location when the page is viewed in a browser. In the next section, we'll edit the `style.css` file to position the HTML code:

1. Open Dreamweaver.
2. Select File > Open.
3. In the Open Dialog box, select the `styles.css` file.

4. Click OK.
5. Add the following CSS code under the search header:

```
#searchResults{  
position:absolute;  
top:160px;  
left:225px;  
color:#666;  
font-weight:bold;  
}
```

6. Select File > Save.

The interactive search box is now correctly positioned. Additionally, the code we added completes the functionality of displaying the search term on the search results page. In the next section, we'll take a look at adding a pop-up window (bubble) that is displayed when the user rolls their mouse over a product.

Displaying a quick view bubble (pop-up window) on mouseover

Adding custom JavaScript calls to Foreground Image or Background Image slices enables the display of quick view bubbles (pop-up windows) when a user rolls their mouse over any of the dog collar products on the search results page. There are five steps involved to create the custom JavaScript functions necessary to display the bubbles on the search results page:

1. Create a custom JavaScript function to hide or show the bubble window.
2. Add a DIV tag to the search page to display an external HTML file.
3. Load the external HTML file into the DIV tag using jQuery.
4. Position the DIV tag using CSS rules.
5. Add the custom JavaScript to an image slice via the slice URL (rather than converting the slice to an HTML slice). Note that this technique is different from those previously described in this article.

In the remaining sections, we'll provide the steps to achieve these five steps to display the bubbles (pop-up windows) when the user rolls over the products on the search results page.

Customizing a JavaScript toggle function

In this section, we'll create the function that will toggle the display of the interactive bubbles:

1. Launch Dreamweaver.
2. Select File > Open.
3. In the Open File dialog box, select the `myscripts.js` file.
4. Click OK.
5. Add the following JavaScript code:

```
//Toggle function for bubble pop-ups.  
//Accepts a DIV ID and an action to determine  
//the visibility.  
  
function doToggle (pDiv, pAction){  
    if(pAction == 'on'){  
        $("#" + pDiv).show(); //Show Hop Up  
    }else{  
        $("#" + pDiv).hide(); // Hide Hop Up  
    }  
}
```

6. Select File > Save.

Understanding the HTML

We've just added a simple toggle function to the `myscripts.js` file that, when called, either shows or hides a specified DIV element. The `pAction` parameter determines whether the function should show or hide the associated DIV tag.

Using DIV tags to structure the search results pop-up windows

Each dog collar shown on the search results page will require a unique DIV tag used to display the quick view bubble (pop-up window) for that item. First, we'll need to create a slice for each of the search result products:

1. Insert three HTML slices that have the following dimensions:
 - **A:** 63 × 50 pixels

- **B:** 63 × 50 pixels
- **C:** 64 × 50 pixels

2. Position each of the new slices in the following locations:

- **A:** x: 0 and y: 360
- **B:** x: 63 and y: 360
- **C:** x: 126 and y: 360

3. Using the Pointer tool, change the color of each slice by clicking the color swatch icon in the Property inspector. Assign the following colors to each slice:

- **A:** yellow (#FFFF00)
- **B:** blue (#0000FF)
- **C:** orange (#FF6633)

4. Edit the HTML code for each slice:

Add the following HTML code to slice A:

```
<div id="bubble_S_A" class="iHide" onMouseOver="doToggle('bubble_S_A','on');"
onMouseOut="doToggle('bubble_S_A','off');" ></div>
```

Add the following HTML code to slice B:

```
<div id="bubble_S_B" class="iHide" onMouseOver="doToggle('bubble_S_B','on');"
onMouseOut="doToggle('bubble_S_B','off');" ></div>
```

Add the following HTML code to slice C:

```
<div id="bubble_S_C" class="iHide" onMouseOver="doToggle('bubble_S_C','on');"
onMouseOut="doToggle('bubble_S_C','off');" ></div>
```

The search results page should now resemble Figure 24.

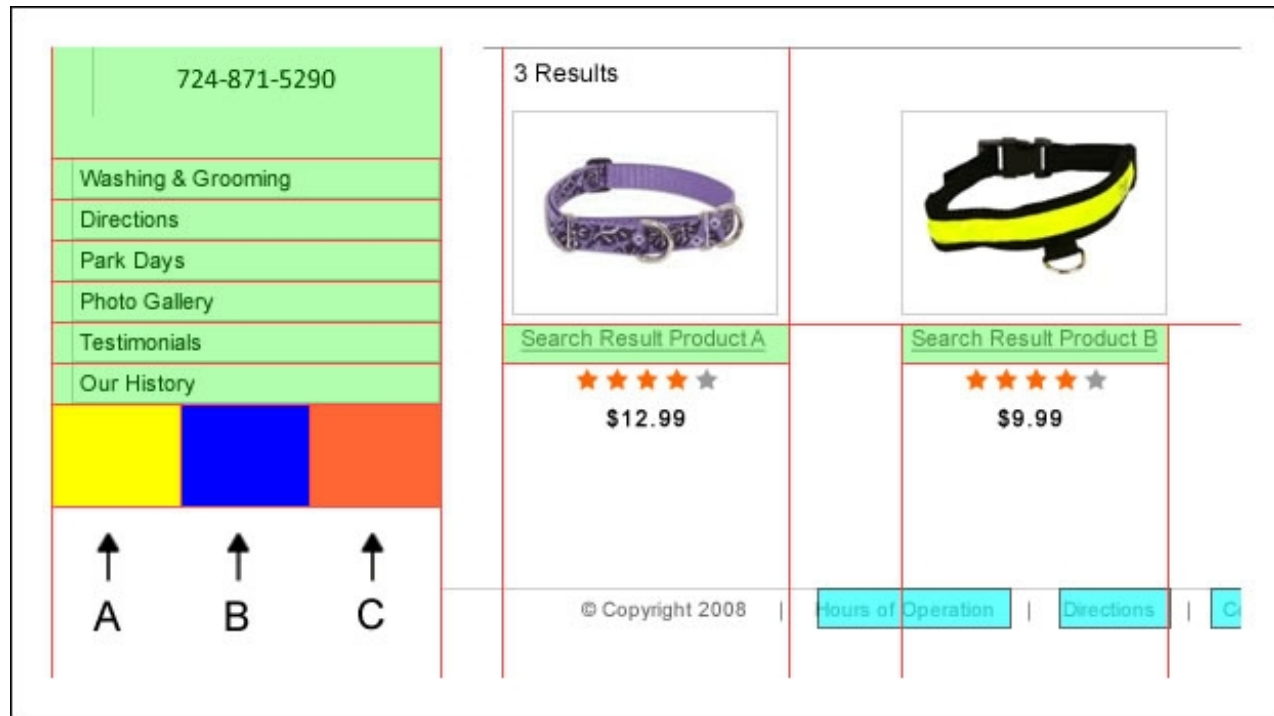


Figure 24. Bubble HTML slices inserted in the left column

Understanding the HTML

Each HTML snippet contains an empty DIV tag with a unique ID. In addition, a rollover and rollout event is added to the DIV tag that calls the doToggle JavaScript function. Finally, the DIV tag should not be displayed when the page is rendered, so we added an `.iHide` class tag to the DIV tag that will turn off the visibility of the quick view bubble window when the page loads. This way, it will only display the bubble when the user rolls their mouse over the product image.

Now add the `.iHide` class to the global CSS file:

1. In Dreamweaver, select File > Open.
2. In the Open Dialog box, select the styles.css file.
3. Click OK.
4. Add the following CSS code under the visibility header:


```
.iHide{
  display:none;
}
```

5. Select File > Save.

Loading external HTML files

In this section, we'll use jQuery to load each HTML bubble file into its corresponding DIV tag for each product when the page is loaded:

1. Launch Dreamweaver.
2. Select File > Open.
3. In the Open File dialog box, select the myscripts.js file.
4. Click OK.
5. Add the following JavaScript code:

```
//Using jQuery we load the external
//html pages into our hidden bubble div tags

$(document).ready(function() {

// Load EXTERNAL Search Pages.

$("#bubble_S_A").load("Bubble_S_A.htm");

$("#bubble_S_B").load("Bubble_S_B.htm");

$("#bubble_S_C").load("Bubble_S_C.htm");

}); // jQuery
```

6. Select File > Save.

Understanding the HTML

jQuery's built-in `Load` function loads the bubble HTML file into each bubble DIV tag, but the `.iHide` property in the CSS file prevents it from being shown until the user's mouse moves over the product image.

Positioning page elements with CSS

With each DIV tag properly loaded, the next task is to position the DIV tags using CSS:

1. Select File > Open.
2. In the Open Dialog box, select the `styles.css` file.
3. Click OK.
4. Add the following CSS code under the search bubbles header:

```
#bubble_S_A
{
position:absolute;
height:267px;

top:80px;
left:340px;
}

#bubble_S_B{
position:absolute;
height:287px;

top:80px;
left:0px;
}

#bubble_S_C{
position:absolute;
height:267px;
top:80px;
left:195px;
}
```

5. Select File > Save.

Understanding the HTML

Earlier, the bubble DIV tags were added to HTML hotspot slices within Fireworks and placed underneath the main navigation. The CSS repositions the DIV tags from underneath the main navigation and places each DIV element next to the corresponding search result product.

Positioning is achieved by applying the `position: absolute;` property. Absolute positioning uses two required value pairs of `top:` and `left:` to position the div tag anywhere on the page relative to the top left edge of the browser. For example, an absolutely positioned element with a left value of 100px and a top value of 50px will be displayed 100 pixels from the left edge of the browser window and 50 pixels down from the top edge of the browser window.

There are various positioning strategies within CSS but going into detail on this subject is outside the scope of this article. However, for additional CSS articles and tutorials, check out the [CSS resources](#) on the Adobe Developer Connection.

Adding custom JavaScript to hotspots to create interactivity

In this section, we'll add hotspots to all three product images and then add custom rollover events to each hotspot using the Link field in the Property inspector.

Follow the steps below to add hotspots:

1. Select the Rectangular Hotspot tool (j) from the Tools panel.
2. Draw a hotspot over each product image.

Note: To quickly add hotspots, select the object using the Pointer tool and right-click (or Control-click) on the selected object and choose the option to Insert Hotspot from the context menu that appears.

Repeat this process three times until you have a total of three hotspots, with one covering each of the three search result page products (see Figure 25).

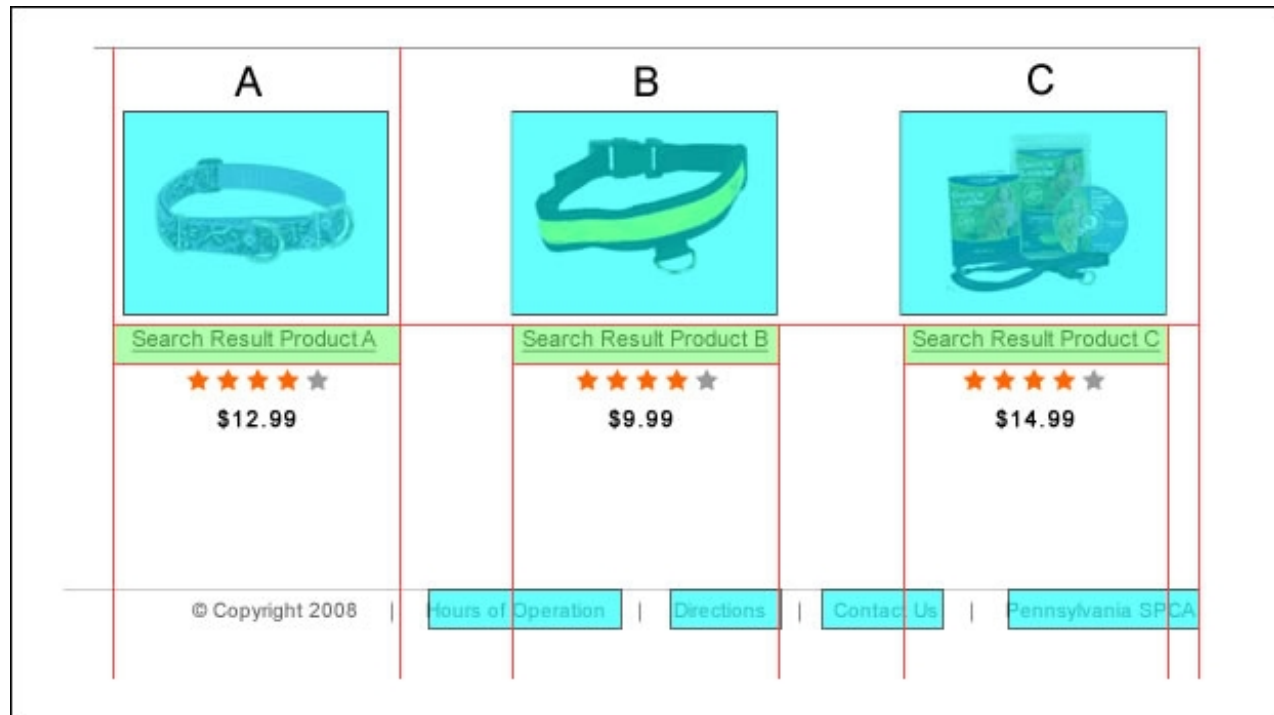


Figure 25. Hotspots inserted over the products displayed in the search results page

Next, add custom rollover events using a common hacking technique and inject HTML and JavaScript code directly into the Link field for each hotspot:

1. Select the Pointer tool from the Tools panel.
2. Select the first product's hotspot (product A) with the Pointer tool. While the object is selected, type the following code in the Link field of the Property inspector:

```
#" onMouseOver="doToggle('bubble_S_A','on');"  
onMouseOut="doToggle('bubble_S_A','off');" "
```

After adding the code, the Property inspector will look like Figure 26.

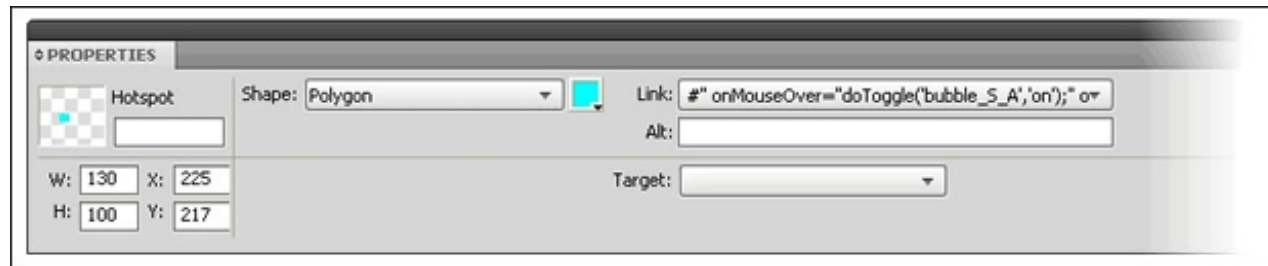


Figure 26. Custom rollover events added to the hotspot covering bubble_A

- Use the Pointer tool to select the middle product's hotspot (product B). While the object is selected, paste the following code into the Link field in the Property inspector:

```
#\" onmouseover=\"doToggle('bubble_S_B','on');\"
 onmouseout=\"doToggle('bubble_S_B','off');\" "
```

After adding the code, the Property inspector will look like Figure 27.



Figure 27. Custom rollover events added to the hotspot covering bubble_B

- Use the Pointer tool to select the third product's hotspot (product C). While the object is selected, paste the following code into the Link field in the Property inspector:

```
#\" onmouseover=\"doToggle('bubble_S_C','on');\"
 onmouseout=\"doToggle('bubble_S_C','off');\" "
```

After adding the code, the Property inspector will look like Figure 28.

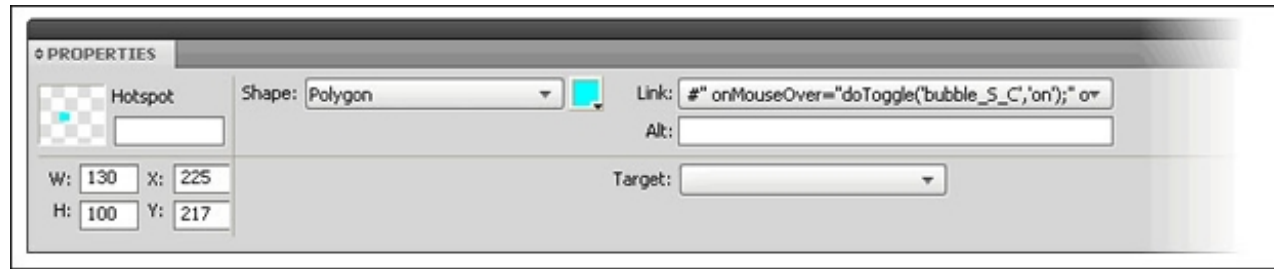


Figure 28. Custom rollover events added to the hotspot covering bubble_C

5. Change the color of each hotspot to correspond to the color-coded slices created earlier for the DIV tags. Use the following colors to update the hotspots:
 1. Change hotspot A to yellow (#FFFF00)
 2. Change hotspot B to blue (#0000FF)
 3. Change hotspot C to orange (#FF6633)

Note: Changing the colors of the hotspots does not affect the functionality of the prototype. However, color-coding improves organization and makes it easier to visually relate each hotspot to the corresponding slice that contains the code relevant to that hotspot.

The search page should now look like Figure 29.



Figure 29. After color-coding the hotspots, the search results page displays hotspots that correspond with the slices in the left column

Now we're ready to export the updated search results page as an HTML file:

1. Select File > Export

Note: The keyboard shortcut for exporting the page is Ctrl+Shift+R (Windows) or Command+Shift+R (Mac).

2. In the Export dialog box, set the following options:

1. Enter the filename: **Search.htm**
2. In the Export menu, select the option: HTML and Images
3. In the HTML menu, select the option to export the: HTML File
4. In the Slices menu, select the option to: Export Slices
5. Check the check box next to the option: Include areas without slices

6. Check the check box next to the option: Current page only

After making these settings, the Export dialog box should look like Figure 30.

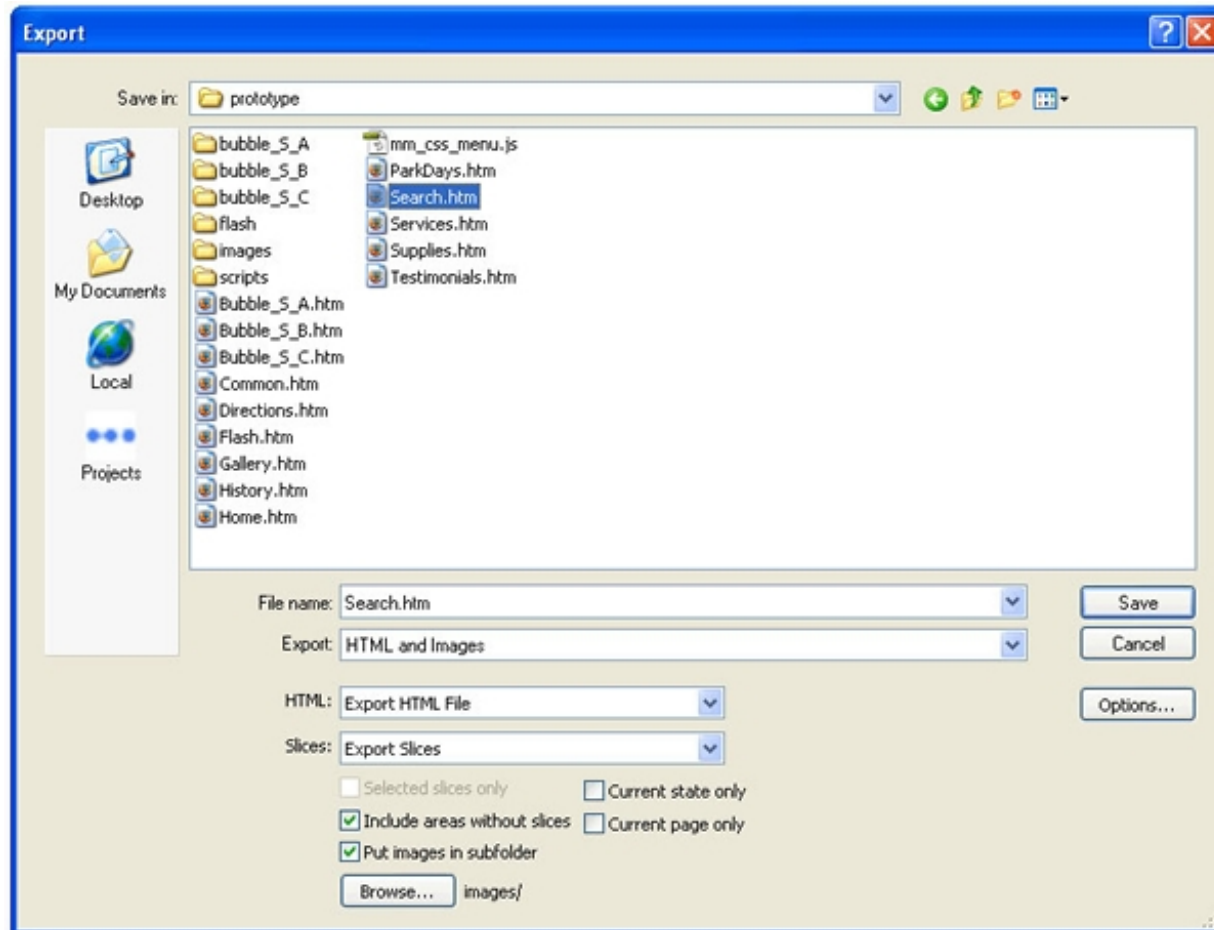


Figure 30. Export options in the Export dialog box

Note: Deselecting the Current page only check box will cause Fireworks to export *all* of the pages in the Fireworks document as HTML.

3. Fireworks can separate the image files from the HTML files, which makes it much easier to manage the exported prototype files:

1. Check the check box next to the option: Put images in subfolder
2. While still in the Export dialog box, select the Browse button to verify that the images folder is the destination for

the exported image files.

3. In the Select Folder dialog box, if the selected images folder is not images, navigate to the images folder located inside the prototype folder. Select the images folder and click the Select "images" button (see Figure 31).

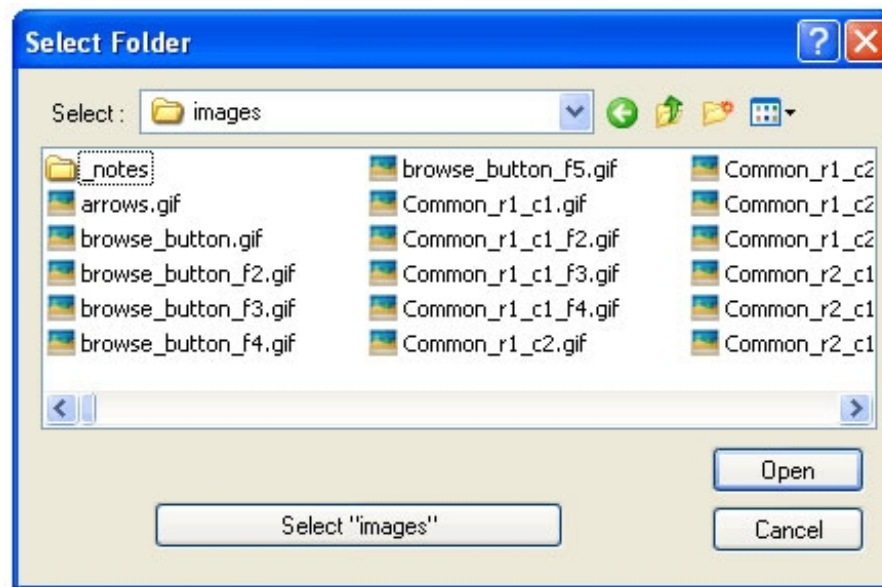


Figure 31. Navigating to the images folder and clicking the Select "images" button

4. Back in the Export dialog box, click Save.
5. In the confirmation box, click OK to replace the existing HTML file.
6. In the subsequent confirmation boxes, click OK to replace all of the existing image files.

Navigate to the prototype folder, open any page, preview it in a browser, and type some keywords into the search field. Click Go to see the search results page. The search results page will always display the same three products, but the search terms you entered will be displayed on the search results page, replicating the behavior of the finished website.

Understanding the HTML

We used a hacking technique that takes advantage of entering code into the Fireworks Link property to insert custom rollover events within the `<a href>` tag in the exported HTML file.

In normal usage, a link defined in the Link property of a hotspot in Fireworks would generate the following HTML code:

```
<a href="page_1.html"></a>
```

The code above would be the result of entering the HTML file name in the Link field of the Property inspector (see Figure 32).

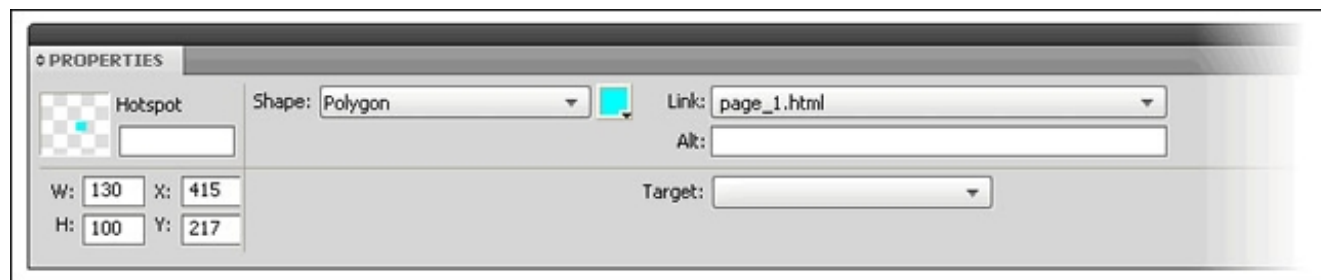


Figure 32. Example of a regular link added to the Link field of a hotspot

Notice that Fireworks takes the URL that is defined in the Link property and wraps the URL with quotes and then places it inside an HTML `<a href>` tag.

In this project, an extra double quote is added at the beginning of the link code to prematurely close the `<a href>` tag. Following the inserted double quote are the custom rollover events. Thus, when Fireworks exports the HTML file, the following hybridized link is created:

```
<a href="#" onmouseover="doToggle('bubble_S_A','on');" onmouseout="doToggle('bubble_S_A','off');" " " > </a>
""
```

The extra two double quotes in the link above is certainly *not* valid or well-written HTML code, but it will suffice for a prototype example, and it will not affect the way the page and the prototype works. If you'd like to add a link—so that the user can also click on the hotspot to access another page—replace the hash (#) symbol with the file name of the HTML file or enter the destination URL.

Where to go from here

Adding HTML, CSS, and JavaScript code can greatly enhance the functionality of Fireworks documents to create interactive prototypes. The technique of separating the HTML, CSS, and JavaScript code from the Fireworks document makes it possible to quickly edit and change a prototype while keeping the source files manageable. This is especially important when you are performing a large usability study where the prototype script and page content may evolve significantly during the testing phase. Setting up the prototype files in a modular way may take an initial investment of

time and effort, but following the strategies outlined in this article will result in increased efficiency and better prototypes for the end product.

To learn more about prototyping, see the following online resources:

- [Industry trends in prototyping](#)
- [Creating simple interactive content using Fireworks, Flash, and Dreamweaver](#)
- [Creating interactive prototypes with Fireworks](#)
- [Rapid prototyping with Fireworks](#)
- [Creating an interactive PDF file from a multipage document in Fireworks CS4](#)

Also, be sure to visit the [Fireworks Developer Center](#) to find more articles and sample projects to help you get up to speed quickly with the new features in the Creative Suite 4 toolkit.



This work is licensed under a [Creative Commons Attribution-NonCommercial-Share Alike 3.0 Unported License](#)

More Like This

[Creating jQuery Mobile website themes in Fireworks](#)

[Creating jQuery Mobile website themes in Fireworks using the CSS3 Mobile Pack](#)

[Prototyping for the Apple iPhone using Fireworks](#)

[Export CSS painlessly from website comps in Fireworks](#)

[Interaction design and rapid prototyping with Fireworks](#)

[Three demos of exporting CSS and images from Fireworks](#)

[Mobile workflows using Fireworks CS5 and Device Central CSS](#)

[Designing interactive products with Fireworks](#)

[Setting up a Fireworks web design mock-up for CSS and images export](#)

[Prototyping AIR applications with Fireworks](#)

Tutorials & Samples

Tutorials

[Creating jQuery Mobile website themes in Fireworks](#)

[Extracting CSS properties from Fireworks design objects](#)

[Working with CSS sprites in Fireworks](#)

[CS6](#)

Samples

[Twitter Trends](#)

[Flex 4.5 reference applications](#)

[Mobile Trader Flex app on Android](#)

[Market](#)
