

Chapter 11: Behaviors

Learning Objectives:

- Be able to describe a behavior in Dreamweaver.
- Be able to describe the use of the Behaviors panel.
- Be able to describe the basic procedure in applying a Behavior.
- Be able to describe various examples of Behavior use with the <BODY> tag.
- Be able to describe various examples of Behavior use with user input.
- Be able to describe the process for adding a new behavior and managing the organization of those behaviors.
- Be able to define DHTML and describe how Dreamweaver implements it.
- Be able to create a slide show in Dreamweaver using a timeline.
- Be able to create a simple animation in Dreamweaver using a timeline.

Introduction

Have you ever been to a site where you can move your mouse over an object such as a picture and the picture will change or a message box will be shown alerting the user to some action that is or can be performed? This is usually the result of some complicated programming technique using JavaScript or some other client side scripting language that causes the event to happen. Dreamweaver makes these events relatively easy to introduce to your website. Dreamweaver uses a **behavior** that is further defined by two terms: an **event** and an **action**. An event is simply something that the user does such as moving their mouse over a particular object or clicking button. An action is the result of that event that performs some specific task. Dreamweaver comes prepackaged with about two dozen behaviors or you can create your own behavior to use throughout your website.

In Chapter 10, we introduced the concept of **JavaScript** code to process a form on the client-side. In this chapter, we will examine the use of predefined JavaScript code (behaviors) to perform specific tasks within Dreamweaver. The behavior actions included in Dreamweaver have been written to work in both Netscape and Internet Explorer, however this does not mean that they should not be tested in both as they will sometimes behave differently between the two browsers. The beauty of using behaviors within Dreamweaver, is that the detailed nature of writing and testing JavaScript code can be reduced to a point and click processes to perform very complex tasks.

HTML on the Web

Behaviors on the Web

Dreamweaver behaviors are constantly being updated and added to by third party vendors and interested individuals. Below are several links that have a many different behaviors:

The Dreamweaver Supply Bin
<http://home.att.net/%7EJCB.BEI/Dreamweaver/>

The Dreamweaver Depot
<http://www.andrewwooldridge.com/dreamweaver/>

Andrew Castles Dreamweaver Resources
<http://www.arrakis.es/~andrewc/downloads/dream.htm>

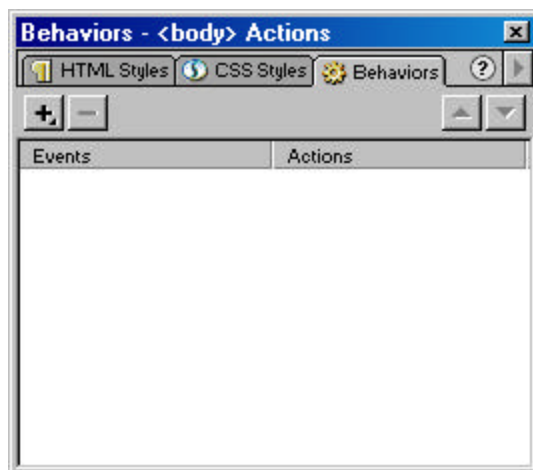
Using Behaviors

Behaviors can make your webpage life much easier. A behavior can validate a form to ensure that when a user enters data into a particular text field that it is the type of data that you want. That is, for example, an e-mail field can be validated to ensure that the ampersand (@) character exists on the screen. The overall objective of any behavior should be to extend the functionality of the website through client-side actions. This will make your website more user-friendly and intuitive to the end-user.

The basic procedure for attaching a behavior to a Dreamweaver element is as follows:

1. Select the element on the page that you wish to attach the behavior to. Note that you cannot attach a behavior to plain text. You can however, attach a behavior to the entire page through the <BODY> tag. We will examine this in more detail in a later example.
2. Choose Window→Behaviors to open the Behaviors panel. (figure 11-1).
3. Click the Plus button and choose an Action from the popup menu. The popup menu will show the behaviors that are available for the current tag that is selected. In figure 11-1, the current tag selected is the <BODY> tag. The behaviors that are shown will only be those that are relevant for this particular tag. In addition, the behaviors that are shown will only work for the selected browsers. You can change the selected browsers by choosing Show Events For submenu on the Plus Button menu.
4. Finally, select the behavior and enter the parameters for the action. These parameters will be different based on the event itself, the browser, and the tag for which you are applying the event.

Figure 11-1



Types of Behaviors

Dreamweaver ships with 25 predefined behaviors. Table 11-1 presents these behaviors and a description of their use.

Behavior	Action	Example Use
Call JavaScript	Executes a JavaScript function	Use for any JavaScript function such as a Message Box.
Change Property	Changes the property of the tag that the behavior is attached to.	Change the background color of a table cell when the mouse passes over it.
Check Browser	Checks the version and name of the browser that the user is viewing your page with.	Can automatically route the user to a webpage that has been designed to take advantage of certain browser features.
Check Plugin	Checks to see whether or not the user has a plugin installed or not.	If the user does not have a plugin needed to view particular components on a page, the user can be alerted.
Control Shockwave or Flash	Controls Shockwave or Flash movies.	Set a rewind, play, pause, or stop button. You could also play or stop a movie based on the result of some other action.
Drag Layer	Allows users to drag and drop layers on the screen.	Create a shopping basket application where the user can drop an image of a product into an image of a shopping basket.
Go to URL	Updates URL addresses.	Can be used to update multiple frames simultaneously.
Jump Menu	Used to edit an existing Jump Menu object.	Essentially implements the jump menu as shown in Chapter 9.
Jump Menu Go	Used to edit the "Go" or "Submit" button on the Jump Menu.	Essentially implements the jump menu go button as shown in Chapter 9.
Open Browser Window	Opens a new browser window.	Can be used to create a new window in which you control the parameters such as status bar, menu bar, size, or scrollbars. Could be used to implement a popup advertisement.
Play Sound	Plays an external sound file.	Inserts an <EMBED> tag with various attributes to automatically play a sound file.
Popup Message	Opens a JavaScript alert box.	Could be used in conjunction with validate form to alert a user that they have not entered a correct value into a form field.
Preload Images	Preloads images into cache before they are displayed on the screen.	When a timeline or slideshow is created, the image will be loaded before it is displayed so that the viewer does not see a time delay in the presentation.
Set Nav Bar Image	Enables editing of a navigation bar image.	Essentially used to implement the navigation bar as shown in Chapter 9.
Set Text of Frame	Change the frame text in a frameset.	A frame could be changed to show user name and other information based on the input into a form.
Set Text of Layer	Change the text in a layer.	Similar to Set Text of Frame
Set Text of Status Bar	Change the text in	Instead of showing the URL in the status bar of

Behavior	Action	Example Use
	the status bar.	the browser, the text could be changed to show a description.
Set Text of Text Field	Changes the text in the Text Field.	Pre-set the entries of a text field based on the feedback of other form entries.
Show-Hide Layers	Shows or hides particular layers on the webpage.	Based on user preferences, a particular layer containing images or text could appear or disappear as needed.
Swap Images	Changes the current image to another.	Could be used to create a slide show in a webpage.
Swap Images Restore	Changes the image back to the original form.	Reset the image based on particular user values or interests. For example, the image could change while the user makes a decision on a selection, then change back to the original image after a selection has been made.
Play Timeline	Plays Dynamic HTML timeline.	Use to control an HTML animation.
Stop Timeline	Stops Dynamic HTML timeline.	Use to control an HTML animation.
Go to Timeline Frame	Goto a specific timeline frame.	Start an animation at any point based on user action.
Validate Form	Validates form input	Allows you to require and implement exact form types such as an e-mail address or a specific number of characters.

Behavior Examples

As you can see, the behaviors that Dreamweaver gives the user are pretty extensive and cover a tremendous amount of webpage applications. In addition, you can combine behaviors to create even more actions and events to implement into your webpage. In the following examples, we will see how to implement a few of the more popular actions. Some of the behaviors are those that are attached to the `<BODY>` tag. These behaviors are triggered by the event of loading the page into the browser. Other behaviors have an action that involves some user action such as clicking on a button, inputting information into a form, or some mouse movement. This is not to mean that the two categories are not overlapping—they are. However, the uses are significantly different.

Examples Attached to the `<BODY>` Tag

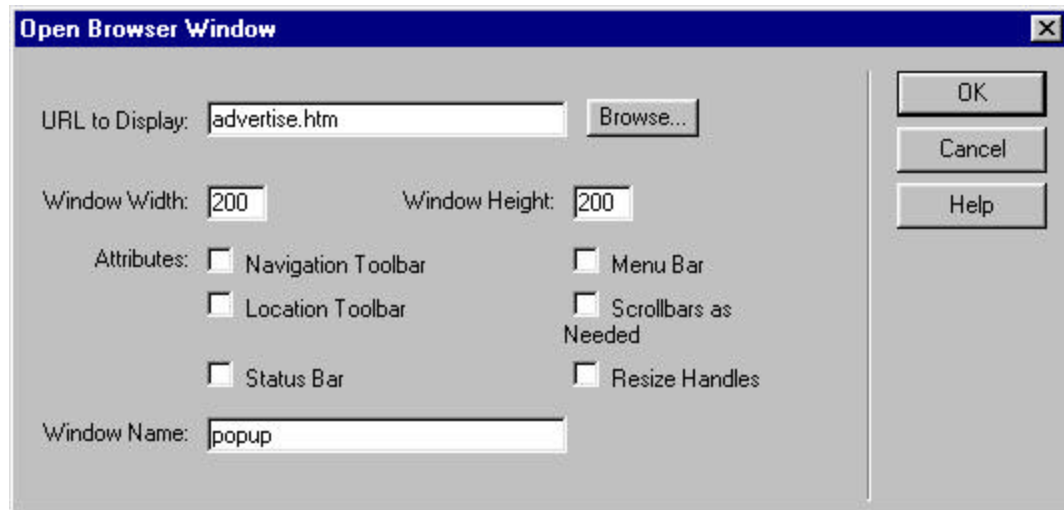
Open a small window

In this example, let's use the behaviors window to open a small window such as a popup advertisement. Ever notice how some popup ads open automatically when you enter a particular webpage? We can implement this functionality by using the Dreamweaver Open Browser Window behavior when it is attached to the `<BODY>` tag. In the exercises folder, you will see an HTML file named `advertise.htm`. This is the file that we will open with the behavior automatically when the user enters our example page.

To implement this function, open a new blank document and follow these steps:

1. Make sure that the <BODY> tag is showing in the tag inspector of the Dreamweaver window (located in the lower left hand side of the document window. This will place the behavior inside the <BODY> tag so that the behavior is triggered upon the loading of the document.
2. Click the "+" button on the Behaviors window and select Open Browser Window. Select the parameters as shown in figure 11-2 and select OK.

Figure 11-2



The parameters selected will open a small window 200 pixels by 200 pixels without a menu bar or toolbars. In addition, the scrollbars will not appear.

3. Now preview the blank document in your browser, the advertisement window will appear on top.

Use the code view and see the following HTML code that has been placed into the <BODY> tag:

1.

```
function MM_openBrWindow(theURL,winName,features) { //v2.0
  window.open(theURL,winName,features);
}
```
2.

```
<bodyonLoad="MM_openBrWindow('file:advertise.htm','popup','width=200,height=200')">
```

Section 1 of the code implements a JavaScript function that resides in the <HEAD> section of the document (the Action), while Section 2 calls the code when the <BODY> of the HTML code is loaded by the browser (the Event).

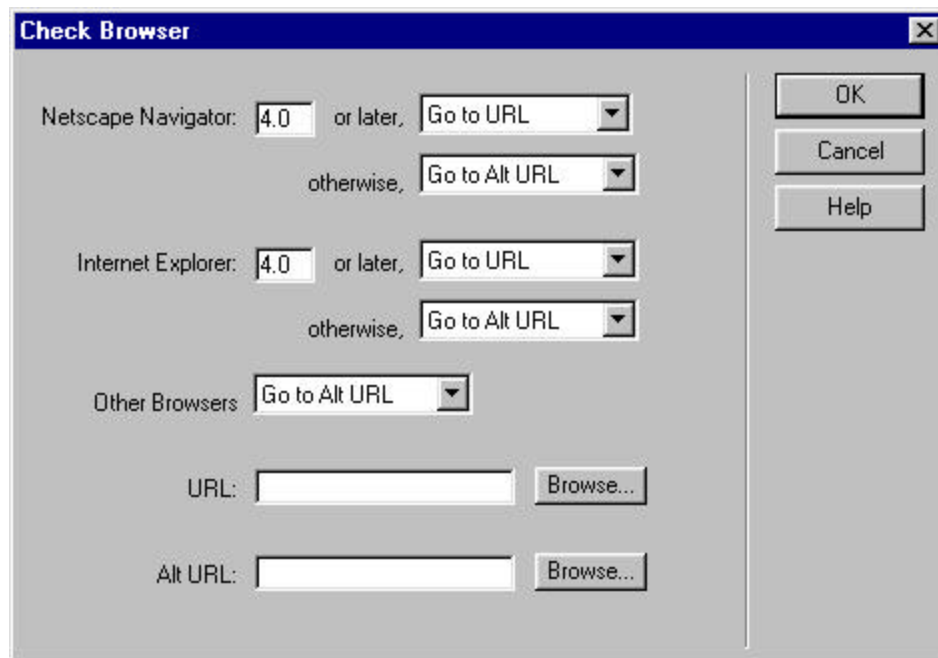
Check Browser

While you will strive to create websites that are compatible with many different browsers, you sometimes will want to use some functions that just are not available in one browser or another. For these instances, you might want to create different versions of the webpage for the different browsers. You can then automatically direct the user to the page based on the browser that they are currently using. For this example, we will need to place the behavior into the <BODY> tag as we did in the previous example. We

will place the behavior into an essentially blank page that the user enters first. This page will be a “director” page to direct the user to the appropriate page based on their browser type. To implement this function, ensure that your that the <BODY> tag is showing in the tag inspector of the Dreamweaver window. Then follow these steps:

1. Click the “+” button on the Behaviors window.
2. Move to Check Browser and click. The dialog window shown in Figure 11-3 will appear.

Figure 11-3



On each dropdown window, the user is given three choices: Goto URL, Goto Alt URL, or Stay on this page. In our example, our “director” page will essentially be the director, that is the user will never see anything on this page. Therefore, we need a URL and an alternative URL. Whichever one you want to designate as the “Netscape page” or the “Internet Explorer page” would be acceptable.

In our example, let’s designate the file named indexnav.htm (located in the exercises folder) as the Netscape page and indexie.htm as the Internet Explorer page. The JavaScript code that is generated essentially looks at the browser and then directs the browser to the appropriate page (the action). This action is activated when the browser loads the document (the event).

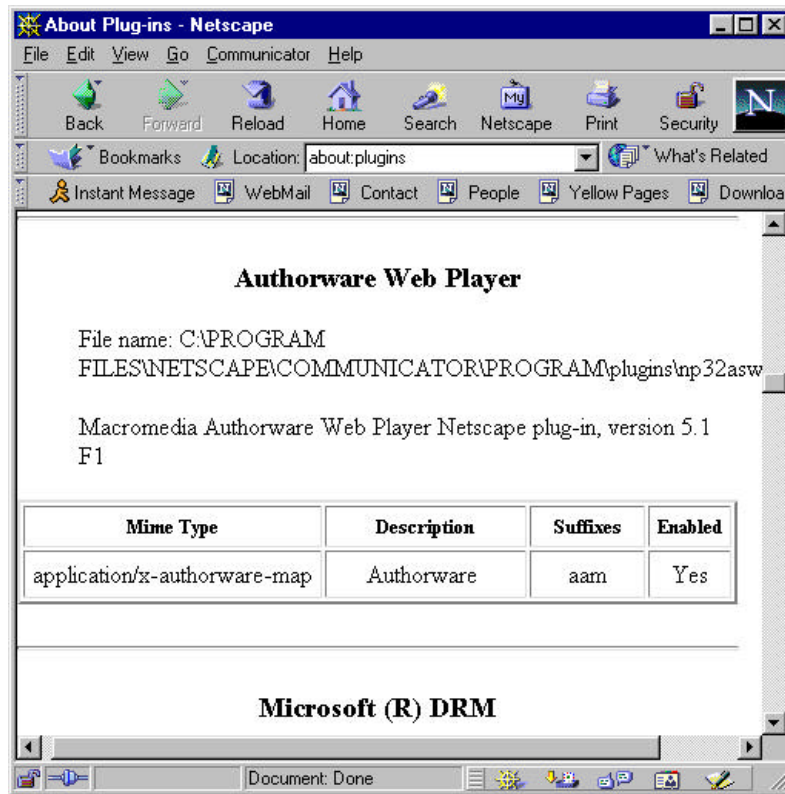
Check For Plugin

Our next example is similar to the previous one in that we might encounter a situation where we require a specific **plugin** to be present. For example for the accompanying online exercises, we require the Authorware Web Player plugin. Before the user tries to activate the pages that require the plugin, we might want to have a function that will check to see if the user has the plugin installed with their browser. This behavior has built in functionality to check for five plugins: Flash, Shockwave, LiveAudio, Netscape Media Player, and QuickTime media plugin. However, as there are literally hundreds of other

plugins available, it is sometimes necessary to check for other types of plugins. This behavior can also route the user to a specific page based on whether or not they have the plugin or not.

To use this behavior, we will have to do a little homework and determine the exact name for the plugin that we wish to look for. In Netscape Navigator, we can find the plugin name by choosing Help→About Plugins from the main menu. This will bring up the window as shown in figure 11-4:

Figure 11-4

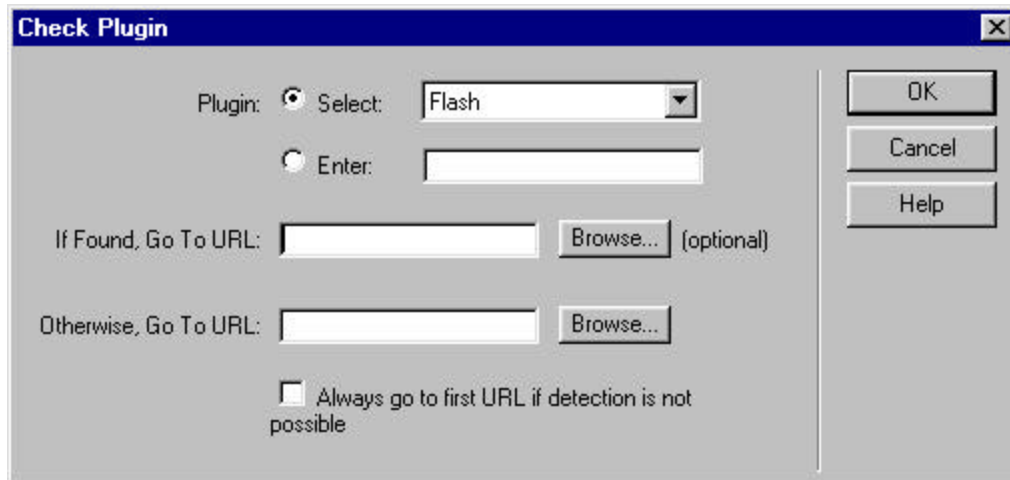


This window will give us the information that we need to search for the Authorware Web Player plugin as we set up our behavior.

In our example, we will search for the plugin and then direct the user to the file named noplugin.htm and alternatively to the file named index.htm if the plugin exists. Let's place this behavior in a blank "director" document. To perform this procedure follow these steps:

1. Make sure that the body tag is selected and then press the "+" button on the Behaviors window.
2. Move to Check Plugin and select it. The dialog window as shown in figure 11-5 will appear.

Figure 11-5



3. Select the Enter radio button and enter the text found in the title of the plugin that you find installed with the Netscape About Plugins function. In our case enter the entire title: Authorware Web Player. You will need to enter the entire title in order for this function to work.
4. In the URL fields enter the address that you wish to have the browser direct the user to based on the absence or presence of the plugin. In our case, enter index.htm for the If Found field and nobrowser.htm for the Otherwise field.
5. You should now check the box next to “Always go to first URL if detection is not possible”. This is a safeguard in case the plugin detection fails. Netscape will usually find most of the plugins. Internet Explorer is a little bit less stable. It will always fail without error in the MacIntosh versions and will not find all plugins in the Windows version. Therefore, this safeguard will insure that the user is not confused if the failure occurs.

User Action Examples

Alert Message and Goto URL

As we said earlier, behaviors can be combined to create complex events and actions. For example, the alert message and the Goto URL can be combined to alert the user that they are about to leave the current website and go to a URL that is outside the current website. You could use this combination to tell the user that the site that they are about to go to is not yours and any other message that you might want to tell the user before they leave.

To implement, let's assume that we have a website that gives your site visitors unbiased consumer information about various products. You want to have a button that will take them to the product manufacturers' website, but you feel compelled to tell the user that the site that they are going to might not be give them objective information. Hence, we will need a form, a button, and two behaviors.

To implement follow these steps:

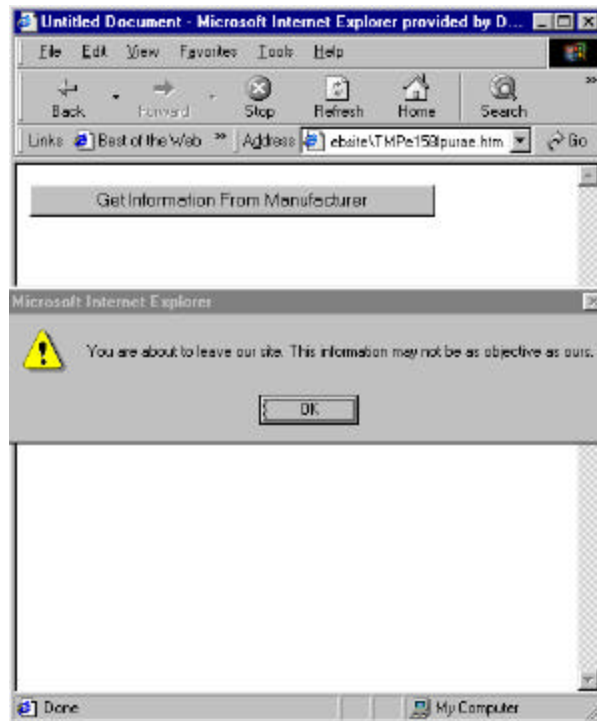
1. First, use the Object window in the forms category and insert a form and a button. Use the Property Inspector to change the form button to something like: “Get Information From Manufacturer”.
2. Now, select the Window→Behaviors to open the Behavior Window.

3. Make sure that the form button is selected and click the "+" button on the behavior window.
4. Go to Popup Message and click.
5. In the dialog box, enter the text that you wish to show in the Alert Box, then click OK.
6. Now, click the "+" button on the behavior window to add our second behavior.
7. Go to Goto Url and click.
8. Enter the URL complete with entire "http" address and click OK. Note that you can also browse for this URL.
9. Now, preview and test the behavior in your browser.

The results in the browser window will look like those in figure 11-6. After the user presses the button to get the product information from the outside website, the alert message will appear. After the user presses OK, they will be directed to the URL that you entered into the behavior.

Use the check code button to see the code that Dreamweaver created to implement. Dreamweaver created the following lines of code:

Figure 11-6



1.

```
function MM_popupMsg(msg) { //v1.0
  alert(msg);
}
```

2.

```
function MM_goToURL() { //v3.0
var i, args=MM_goToURL.arguments; document.MM_returnValue = false;
for (i=0; i<(args.length-1); i+=2) eval(args[i]+".location='"+args[i+1]+'";');
}
```
3.

```
<form name="form1" method="post" action="">

<input type="submit" name="Submit" value="Get Information From Manufacturer"
onClick="MM_popupMsg('You are about to leave our site. This information may not be as
objective as ours. ');MM_goToURL('parent','http://www.ford.com');return
document.MM_returnValue">

</form>
```

Section 1 created a JavaScript function that created the alert message with the variable named “msg” (Action).

Section 2 created a JavaScript function that created the goTo URL function (the second Action).

Section 3 implemented the event into the form button so that when the button was clicked, the popupMsg function and then the goToURL function would be implemented (the Event).

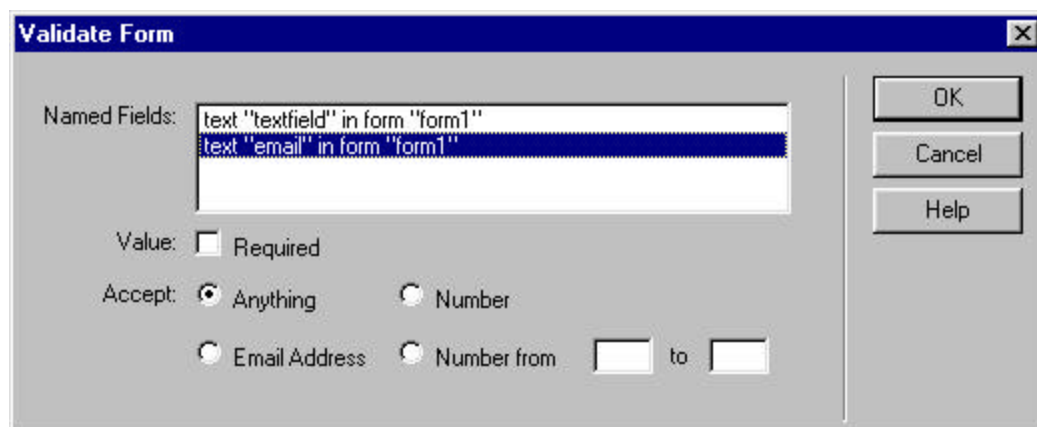
Dreamweaver implemented the JavaScript code without you having to write it!

Validate Form

Our next example, examines another form element: the text field. To control the type of information that is placed into the form, we can use the Dreamweaver behavior Validate Form. For instance, we might want to make sure that an e-mail address contains the ampersand (@) sign. In this example, create a simple form with a text field. Name the text field e-mail address. You can apply the following process to validate an e-mail address in this form.

1. Select the form that contains the e-mail address.
2. Click the “+” sign in the Behaviors panel.
3. Move to and select Validate Form. The dialog window in Figure 11-7 will appear.

Figure 11-7



The first setting for this form is to require the value or not. This setting forces the user to place something in the text field before submission. In our case, select the e-mail address radio button. You can also select that the form will take anything, a number or a range of numbers.

4. Press OK and then test your form.

Dreamweaver produces JavaScript that uses the "onBlur" event to trigger the action of form validation. **OnBlur** is the default event method that is activated when the user leaves this text to go to another object on the form. Therefore, in order for the validation method to work, the form will have to contain more than one form object. If the user omits the ampersand from the e-mail address or if the user omits the field completely, an alert error message will appear as shown in figure 11-8.

Figure 11-8

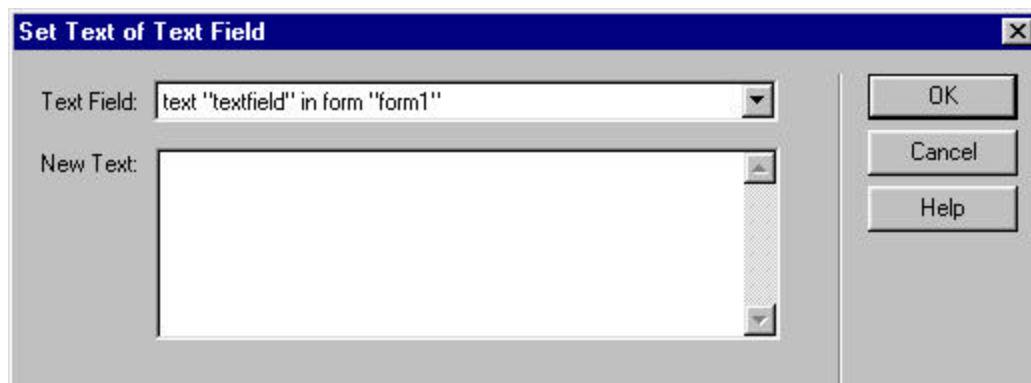


Set Text of Text Field

In our final example, we will show how to set the text field in the text field of a form. Again, we will need some sort of triggering event to cause the action of placing specific text into a field. For example, we might want the text field to give the user instructions when the text field is selected. Therefore, we will need for the trigger event to be **onFocus** as opposed to the default opposite **onBlur**. To accomplish this example, we will use the e-mail address field from the previous example and place an instruction to the user enter their e-mail address into the field. To accomplish this follow these steps:

1. Select the e-mail field. Note that you should already have a behavior attached to this form object. As we will be using the onFocus event, which will trigger the action when the field is selected, the order will not matter.
2. Press the "+" button on the Behavior window and select Set Text→Set Text of Text Field. The dialog window in figure 11-9 will appear.

Figure 11-9



3. Click the drop down menu to select the text field that you wish to attach the event. In our case, we wish to attach the event to the e-mail text field. Then click OK.
4. Click the event method selector on the Behavior window. Move to and select onFocus.
5. Now, test the page in your browser.

When the user selects the e-mail text field (onFocus), the field text will change to "Place your e-mail address here." When the user leaves the e-mail text field (onBlur), the field will be validated to ensure that the ampersand is present in the text string.

There are many other combinations of behaviors that can be used in the development of a website which is more functional and easier for your users. The result will be cleaner data entry, greater engaging content, and less user confusion when using the site. The overall outcome of this is more users that come to the site.

Managing Behaviors

The twenty-five behaviors that come packaged with Dreamweaver are an excellent start, however there are companies and interested individuals who place literally hundreds of other behaviors for free download on the Internet (see HTML on the Web at the beginning of this chapter). Downloading and managing these behaviors is truly a simple task. In addition, you might want to create new submenus to organize your behaviors list into categories that you use often. Both of these tasks are pretty simple to accomplish.

Adding a Pre-Defined Behavior

In this example, I went to a site on the web and downloaded a behavior. In this case, I went to the Dreamweaver Depot and downloaded a behavior that would focus a remote window to the top. That is, this action will bring a window into the user's view when the event triggers it. The file downloaded as a zip file. Hence, to make the behavior usable, I followed these steps:

1. Copy the file from your download directory into the following directory under the Dreamweaver program files:

Configuration/Behaviors/Actions
2. If you had Dreamweaver running you will need to restart it to make the behavior noticeable.
3. That is all there is to it! When I want to use the behavior, I will simply attach it to a triggering event on the website.

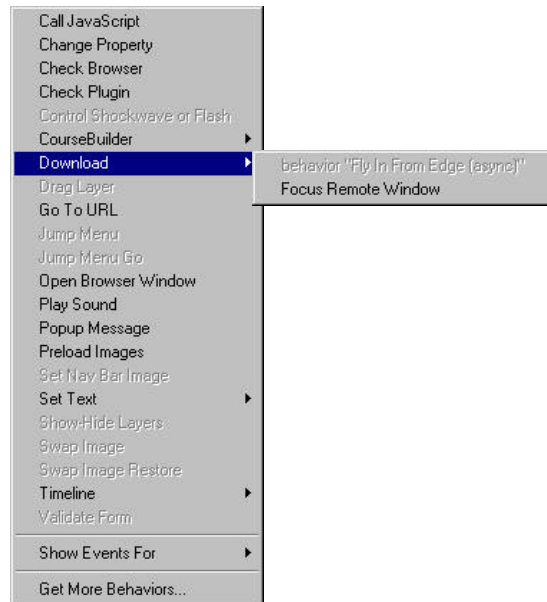
Managing and Organizing Pre-Defined Behaviors

If you find a lot of the behaviors on the web useful and start downloading lot's of them to your hard drive, the behaviors list can become pretty quickly a crowded and messy list. Therefore, you should get

into the habit of creating folders within the Configuration/Behaviors/Actions directory that will manage the Behaviors that you download. It is a pretty simple task to accomplish this:

1. In the Actions folder simply create subfolders and move the files that you wish to have in this category into the newly renamed folder. In my example, I created a folder called "Download" and moved all downloaded behaviors into it. When I click the add behavior button on the behavior window, the menu appears as shown in figure 11-10.

Figure 11-10



There are many different organization configurations that you might wish to create to better manage the behaviors on your website. Experiment and remember that you can always put it back the way you found it if you keep a backup copy of the folder. In summary, Behaviors are an important tool for creating websites that have functionality, are intuitive to the user, and are easier for the website designer to implement.

Dynamic HTML

Basic HTML web pages are static in their display. That is, they appear on the screen and pretty much stay in that one place for the duration. **Dynamic HTML (DHTML)** refers to Web content that changes each time it is viewed. For example, the webpage could change based on time of day, the URL that the page was referenced from, or previous pages that have been viewed by the user.

HTML on the Web

Dynamic HTML

The only limit to DHTML is your own imagination and creativity. However sometimes, it is good to find inspiration with other people's work.

Here is a site that has several examples of DHTML in action.

http://www.webdeveloper.com/html/html_dhtml_1.html

Check out the jigsaw puzzle. It is very cool!

DHTML has made a big splash in the webpage design world. Microsoft.com moved their entire site away from Java to DHTML as many applications that were formerly served by **Java** can be achieved with greater ease with DHTML. In addition, DHTML loads and processes on the client side faster than Java and other webpage technologies (such as ActiveX), thereby making DHTML use more user friendly.

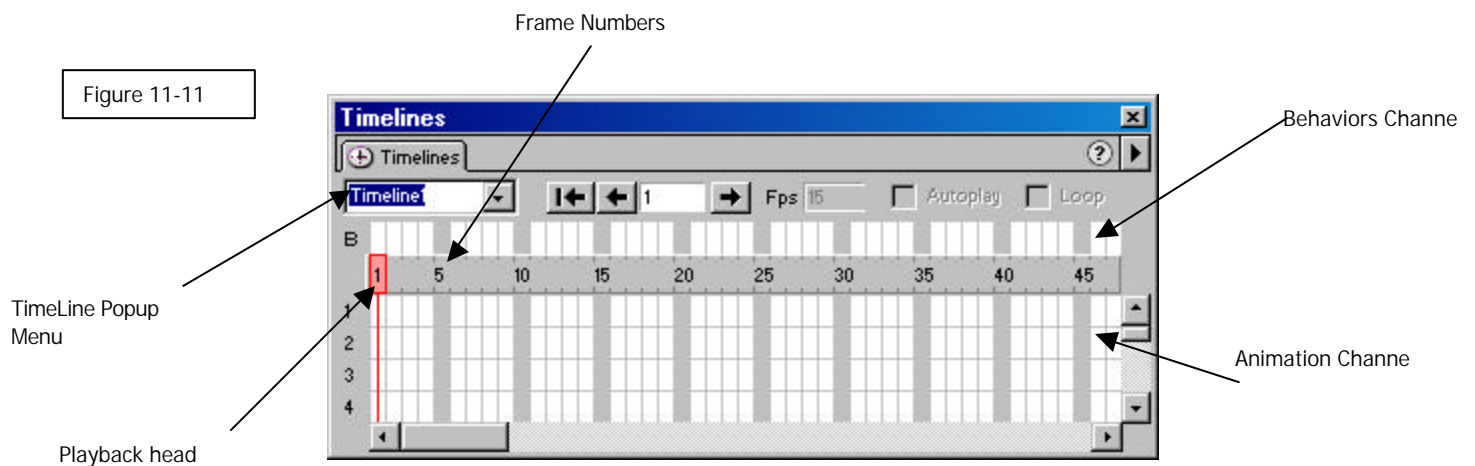
One of the problems with DHTML is cross browser compatibility. However, many webpage designers believe that there is common ground between the browser's implementation of DHTML to make it a feasible alternative. Therefore, the user's browser will become a major factor in the determination of the use of DHTML. Many designers believe that DHTML is perfect solution to a corporate Intranet where you know the browser that will be in use.

An implementation of DHTML in Dreamweaver is the use of Timelines to create animations and slideshows that take full advantage of the DHTML capability. In this section of the text, we will explore the concept of a timeline and show how to create a slideshow that has some pretty interesting effects.

TimeLines with Dreamweaver

Dreamweaver implements a **timeline** animation through Behaviors and Layers. The Timelines Panel pulls these components together to implement both JavaScript and DHTML to create dynamic webpages in which you have control over the position, size, and visibility of a layer over time.

To access the Timelines panel in Dreamweaver, choose Window→Timelines from the main menu. The panel is shown in figure 11-11.



There are a few key components that we should define here before moving on. First, the **timeline popup menu** allows you to choose the timeline that you wish to work with on the page. The **playback head** shows the current frame of the timeline that is displayed on the screen. The **frame numbers** indicate the sequential numbering of the frames. The **behaviors channel** shows what behavior will be triggered at the particular point in the animation, while the **animation channel** shows what layers and images will be shown at particular points in the animation sequence.

We will refer to these components as we build our two examples in the next sections.

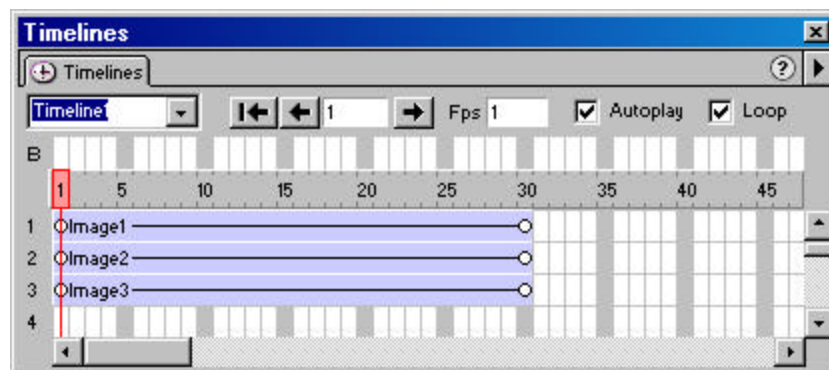
Creating a simple slide show with a timeline

One of the more popular uses of a timeline is the creation of a “slideshow” or a simple sequencing of pictures over a designated time period. The creation of a slideshow uses layers and shows and hides those layers at specific intervals.

In this example, we will create a simple show for a bakery that has an image of cookies, a pastry, and rolls. We want these images to change every 3 seconds on the screen. From your exercises disk, open the file named bakery.htm. You will see the three images on the page. Each image is contained in a separate layer. Our objective is to make the images appear to swap every 10 seconds. To do this, follow these instructions:

1. Make sure that you have two windows open: the Timeline window and the Behaviors window.
2. Click the first image and drag it to the first row of the animation channel.
3. Click the second image and drag it to the second row of the animation channel.
4. Click the third image and drag it to the third row of the animation channel. Click each image bar on the animation channel and drag it to cover 30 seconds. Then change the FPS to 1 and click Autoplay and Loop. Your result should like figure 11-12.

Figure 11-12



5. Now, we want to add a behavior to the first interval. Click the block just above the 1 second marker on the Behavior channel. Then click the “+” sign on the Behavior window to add a behavior.
6. Move to Show Hide Layers and select. In the dialog window, click Show for Layer 1 and Hide for Layers 2 and 3.
7. Click the behavior channel at time interval 10 and add the Show Hide Layer Behavior. Set the settings to show Layer 2 and Hide Layers 1 and 3.
8. Click the behavior channel at time interval 20 and add the Show Hide Layer Behavior. Set the settings to show layer 3 and Hide Layers 1 and 2.

9. Now, move the layers so that they will overlap each other.
10. You can now preview this in your browser.

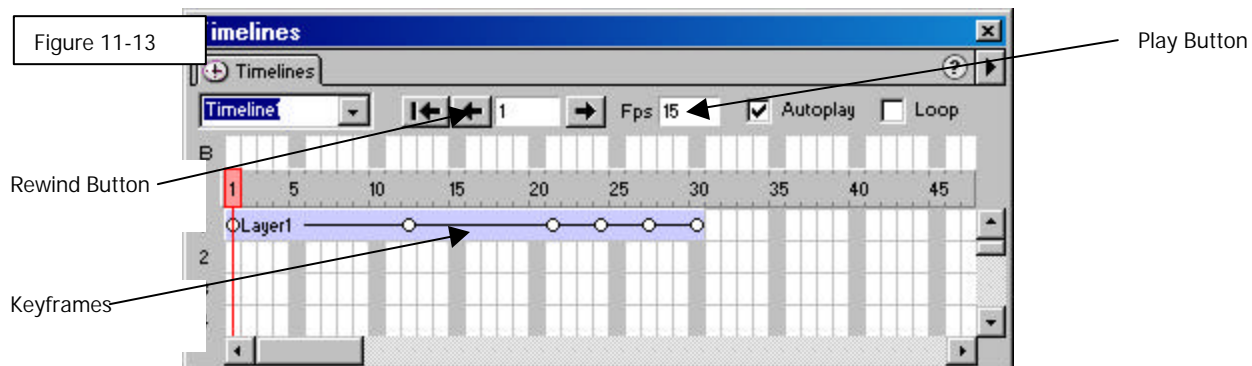
One problem that you might encounter while doing this on your own is how the layers are defined. We found that if the layers are defined as <DIV> tags, they worked properly in both browsers. When the layers were defined as <LAYER> tags, IE would not recognize the tags. This is because the <LAYER> tag is a Netscape supported tag and not one the IE supports. However, the <DIV> is cross compatible. This is an example where the DHTML can be made cross compatible with various browsers if care is taken in its design and implementation.

Creating a Simple Animation with the Timeline

In our next example, we will create a simple animation where a layer moves across the screen to a designated spot on the screen (the action) when the page loads (the event). We will do this by dragging the object across the screen to record the path that we wish the layer to take. This is an easier method than defining the keyframes manually. A **keyframe** is the frame on the timeline where different objects (layers or images) will change to a different property such as a position on the screen.

For this example, open the file named animate.htm on your exercises folder. This file contains a simple layer that holds the title of the document. We wish to make this layer move from an invisible position in the middle of the screen to its final resting place at the top of the screen. To perform this task, follow these steps:

1. Make sure that your Timeline and Behavior window is open.
2. Choose layer object.
3. Select Modify→Timeline→Record Path of Layer
4. Drag the layer to the designated end position. Take any path that you wish to bring it to this point.
5. The path will be drawn as you move the layer to its resting position. Dreamweaver will add the appropriate keyframes for you in the Timeline window.
6. Click the autoplay button on the Timeline window. Your timeline window will resemble that shown in Figure 11-13.



7. Now, let's add a behavior to the timeline to make the heading appear invisible at first and then become visible as it starts to move. Click Frame 1 in the Behaviors channel.
8. Now, click the add behavior button on the Behavior window and select Show/Hide Layer.
9. Click hide layer
10. Now, click frame 2 on the behavior channel and click the add behavior button again.
11. Again, move to Show/Hide layer and click Show the layer. This will make the layer appear at the beginning of the animation.
12. You can now click the rewind button on the TimeLine window to see the path and then preview your creation in both browsers.

Now, you should view the code that you just created for these two interactions. Can you imagine coding this yourself?! Dreamweaver makes creating very interesting and imaginative interactions quite easy with these tools. You should apply your own creativity to these examples. The automatic features that Dreamweaver incorporates clear the way so that the only limitation that you have is the limits of your own imagination.

Case Update

Dean sent an e-mail to Brian and Kelley telling them of the various deals that he had run across with the vendors of their product offerings. He knew that they could increase their margins by running specials on products that they could get quantity discounts for. He asked them both to think of ways that their website could showcase these specials.

Brian suggested that they create a popup window that would appear when the main page loaded that would include pictures of the specials. The pictures could be linked to the product information page so that the customer could complete the ordering process. He began to work on that project by using the various tools in the Behaviors Panel. He wrote the code that would automatically load the popup window and would refresh itself when other pages were clicked on. For example, if the women's page was clicked on the popup window would change to product specials for the women's line.

Kelly began creating the code to create the slide show for the various product offerings.

Summary

- Be able to describe a behavior in Dreamweaver.

A **behavior** is defined by two terms: an **event** and an **action**. An event is simply something that the user does such as moving their mouse over a particular object or clicking button. An action is the result of that event that performs some specific task. Dreamweaver comes prepackaged with about two dozen behaviors or you can create your own behavior to use throughout your website.

- Be able to describe the use of the Behaviors panel.

The behaviors panel is viewed through choosing Window→Behaviors on the menubar. The panel consists of an add and delete button as well as an area which describes the event and the action of the behavior that is attached to the object. A behavior can be attached to most any object with the exception of plain text. In addition, a behavior can be attached to the <BODY> tag so that the action can be triggered by the loading of the page in the browser.

- Be able to describe the basic procedure in applying a Behavior.

First, select the object that the behavior will be attached to. Then, click the Add Behavior button (+) and choose an Action from the popup menu. The popup menu will show the behaviors that are available for the current tag that is selected. In addition, the behaviors that are shown will only work for the selected browsers. You can change the selected browsers by choosing Show Events For submenu on the Plus Button menu.

- Be able to describe various examples of Behavior use with the <BODY> tag.

In this text, we described how to Open a Small Window, Check the version and type of browser that the user currently using, and to check for a particular plugin on the users machine.

- Be able to describe various examples of Behavior use with user input.

The examples that were given in this text described the process for validating form data, popup an alert message and redirect the user to a URL, and to set the text of a text field when the field has been selected.

- Be able to describe the process for adding a new behavior and managing the organization of those behaviors.

There are companies and interested individuals who place literally hundreds of other behaviors for free download on the Internet (see HTML on the Web at the beginning of this chapter). Downloading and managing these behaviors is truly a simple task. Placing a new behavior from an outside party is accomplished by copying the behavior html file into the Configuration/Behaviors/Actions directory under your Dreamweaver program. Managing the behavior categories simply involves creating new folders in your Configuration/Behaviors/Actions directory.

- Be able to define DHTML and describe how Dreamweaver implements it.

Dynamic HTML (DHTML) refers to Web content that changes each time it is viewed. For example, the webpage could change based on time of day, the URL that the page was referenced from, or previous pages that have been viewed by the user.

DHTML has made a big splash in the webpage design world. Microsoft.com moved their entire site away from Java to DHTML as many applications that were formerly served by Java can be achieved with greater ease with DHTML. In addition, DHTML loads and processes on the client side faster than Java and other webpage technologies (such as ActiveX), thereby making DHTML use more user friendly.

- Be able to create a slide show in Dreamweaver using a timeline.

One of the more popular uses of a timeline is the creation of a “slideshow” or a simple sequencing of pictures over a designated time period. The creation of a slideshow uses layers and shows and hides those layers at specific intervals. This process is implemented through the timeline.

- Be able to create a simple animation in Dreamweaver using a timeline.

A simple animation can be created where a layer moves across the screen to a designated spot on the screen (the action) when the page loads (the event). This process is implemented through the timeline.

Key Words

<BODY>
action
animation channel
behavior
behaviors channel
Dynamic HTML (DHTML)
event
frame numbers
Java
JavaScript
keyframe
onBlur
onFocus
plugin
timeline

Exercises and Projects

For the CAO case project implement the following:

1. Use the Behaviors panel to automatically open a “Specials” popup window that tells the user of current special prices on selected items. This window should be 200 x 200 pixels and have no menu or scroll bars.
2. Create a customer feedback form which would include the customer name and e-mail address. Use the Behaviors panel to validate the e-mail address. In addition, use Behaviors to prompt the user to add various correct data to the various fields. All fields should be required.
3. Use the Timeline to create a slide show in the Specials popup window that will rotate various pictures of the products at 3 second intervals.
4. Use the Timeline function to animate a picture on the opening screen. This picture should move around the document window until it goes to its final resting place of your choosing.

Other exercises and projects

5. Surf the web and find examples of DHTML in action. If you were to create these in Dreamweaver, what tools would you use?

6. Surf the web and find third party behaviors that were created for Dreamweaver. What examples would be useful for you and describe their uses on a webpage.

Review Questions

1. Describe the two basic characteristics of a behavior.
2. What is the difference between attaching a behavior to an object on the webpage and attaching it to the <BODY> tag?
3. Can the Check Plugin behavior work with every browser? What steps would you take to alleviate problems of this sort?
4. What is an alert message?
5. What is the difference between onFocus and onBlur?
6. What is the procedure for downloading and implementing a third party behavior?
7. Why do you think DHTML is growing in popularity?
8. Why is cross-compatibility a problem in DHTML?
9. How does DHTML compare with Java?
10. What is the difference between the animation channel and the behavior channel on the TimeLine window?