

Creating a Flash-driven DHTML Menu

A critical issue to decide when developing websites is what navigational menu will be used. First, you need to decide how the menu will fit into the proposed layout; for example, in a horizontal or vertical format. Next, you need to decide what type of navigational menu will be used such as: JavaScript-based rollovers, CSS - styled list menus, or menu creation tools which provide a variety of options, automate creation of the menu and are easily customized.

Often times, one doesn't think of using Flash to create a navigation system. However, if Flash is properly implemented it can provide an interactive option beyond the capability of HTML and CSS. One example is a DHTML menu. In this article, we will walk through creating a DHTML menu and how to show and hide submenus. In addition, we will consider accessibility issues involved with this type of menu.

If you would like to follow the DHTML menu's step-by-step development, or even try your hand at creating a menu as you progress through this article, you will find the project files link helpful.

Finished Page

<http://midwestwebdesign.net/tutorials/flash/dhtmlmenu/dhtmlmenu.html>

Download Project Files

<http://midwestwebdesign.net/flash/dhtmlmenu/dhtmlmenu.zip>

Advantages/Disadvantages to the menus mentioned

It is probably safe to say that no menu presented today is perfect, at least from a code-based perspective. For example, if you use a menu system with CSS and JavaScript you may discover that if:

- JavaScript is disabled in a visitor's browser, the menu will not work
- CSS isn't developed with cross-compatibility in mind, its behavior is likely to be inconsistent between browsers and operating systems

When using Flash to create a menu, the following can also be problematic:

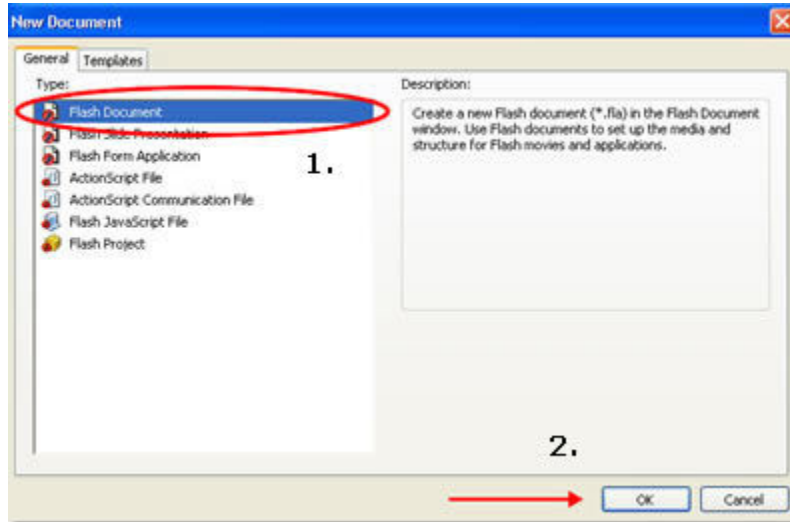
- If the visitor doesn't have the correct Flash player version installed, the navigation will not render or work properly
- You lose the ability to interact with assistive readers. When you use regular HTML anchor tags and CSS you can provide assistive readers with navigation tools such as tabbed key support

From here you can see that it is critically important for you to consider your website's intended audience. Once all factors have been accounted for, choose the navigation system which best suits your needs. For this article, we will choose the Flash-based option. Let us begin.

Create the file

Inside Flash, create a new file named **menuone fla** by following these steps:

- From the main menu, select **File>New**, and in the following window:

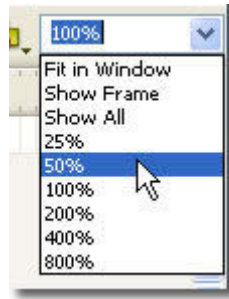


- Leave Flash document selected
- Left click OK

Create the following hierarchy:

Folders	Layers
Actionscript	
	Actionscript
	Stops
	Labels
Button One	Top button
	Sub menu
	Sub menu
	Sub menu
	Sub menu
Button Two	Top button
	Sub menu
	Sub menu
	Sub menu
	Sub menu

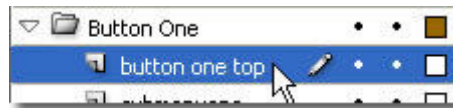
As you can see, folders are on the left and layers are on the right. Assign colors to your folders and layers as you desire. Make sure you reduce the magnification to 50% through the zoom drop down list:



Create main menu buttons – first and second tier

Since each of the main menu buttons has the same characteristics, we will use "topbutton" from our first tier as our example. Follow these steps to create the button:

Make sure you are on the “button one top” layer as shown below:



- Right click frame one
- Select the text tool button (“A”) from the **Tools** menu and drag a text field instance on the stage
- Make sure the text field is static
- Type **Button One** with the following properties:
 - Font: Arial
 - Size: 20
 - Color: Black (#000000)
 - Font-weight: Bold
 - Font-style: Italic
 - Text-alignment: Center

In case the text field doesn’t expand to accommodate the text, move the cursor to the top right corner and drag to the right until the text fits inside:



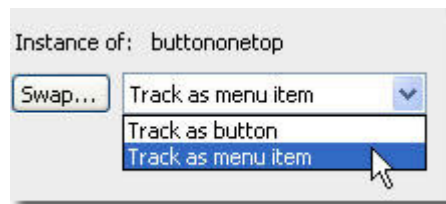
- Use the pointer tool from the tools menu to mark the desired text:



- Then, right-click it and select **Convert to symbol**, in the symbol window, type **buttononetop** and then select **button**
- With the button selected, in the properties window, locate the Instance text field:



- Type **buttononeBtn**
- Position the button at the following coordinates:
 - X : 86.7
 - Y : 23.9
- In the properties window, change the drop down list next to the Instance text field:



- **Track as menu item**

Repeat these steps for the other button, replacing instance and library names where appropriate.

Why Track as menu item?

Often times when placing multiple buttons in a deep hierarchy, such as this one, leaving the default setting for buttons to **Track as buttons** will cause buttons to stop working. Setting the buttons to **Track as menu item** fixes the issue.

Create fade sequence for main menu buttons – first and second tier

Since each of the main menu buttons have the same fade sequence, we will use top button from our first tier as our example. Follow these steps to create the fade sequence:

- Make sure you are on the top button layer
- Right click frame ten and select **Insert Key Frame**

For frame one:

- Select the button
- In the properties window, set **Alpha** as color

For frame ten:

- Select the button
- In the properties window, set **none** as color

To complete the fade sequence, follow these steps:

- Between frames one through ten, right click and select **Create Motion Tween**:

Repeat this process for button two of the second tier from the main menu except position the button at the following coordinate:

- X : 82.0
- Y : 57.9

Creating the submenu buttons – first tier

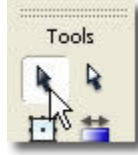
For simplicity, we'll keep our submenu buttons inside the main timeline. Since each of the buttons has the same properties, we'll use submenu one from the first tier submenu as our example. After completing the button, we'll adjust the second tier submenu accordingly. Follow these steps to create the first tier submenu button:

- Make sure you are on the submenu one layer as shown below:



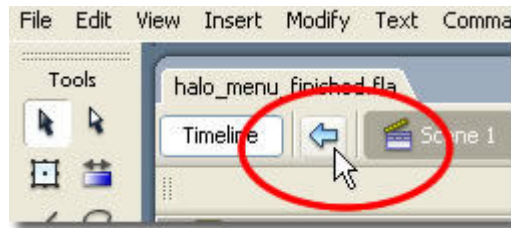
- Right click frame eleven and select **Insert Key Frame**
- Select the **text tool button** (“A”) from the tools menu and drag a text field instance on the stage
- Make sure the text field is static
- In the text field, type **submenuone** with the following properties:
 - Font: Arial
 - Size: 20
 - Color: Black (#000000)
 - Font-weight: Bold
 - Font-style: Italic
 - Text-alignment: Left
- In case the text field doesn't expand to accommodate the text, move the cursor to the top right corner and drag to the right until the text fits inside

- Use the pointer tool from the tools menu to mark the desired text:

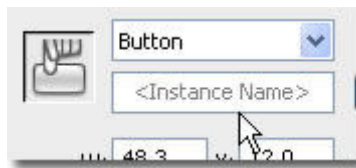


- Then, right-click it, and select **Convert to Symbol**. In the symbol window, type **submenuone** and select **button**
- In the library, double click the button to enter symbol editing mode
- Rename your layer **submenuone**
- Create the other three states for your button; remember the hit state to make your button clickable

Click the blue backward arrow button to exit symbol editing mode:



In the properties window, locate the Instance text field:



- Type **oneBtn**
- Position the button at the following coordinates:
 - X: 120.0
 - Y: 55.0

Create the fade sequence – first tier

When a visitor clicks the first main menu button, the first tier submenu will fade in beneath. Since the fade sequence is the same for each submenu button, we will use our current submenu button as our example. Follow these steps to create the fade sequence:

- Right click frame twenty -one and select **Insert Key Frame**

For frame eleven:

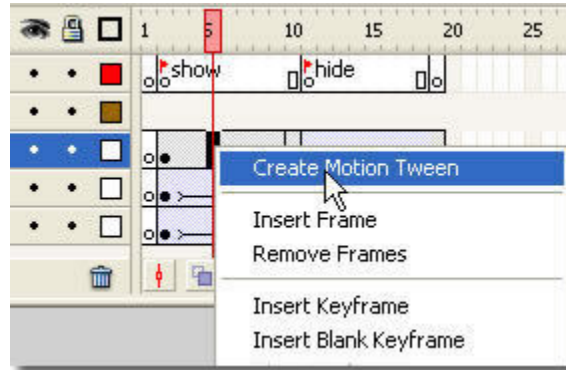
- In the properties panel, set **Alpha** as color

For frame twenty- one:

- In the properties panel, set **none** as color

To complete the fade sequence, complete the following:

- Between frames eleven through twenty- one, right click and select **Create Motion Tween**

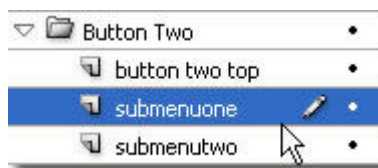


Repeat this process for the remaining first tier submenu buttons replacing instance and library names where appropriate. Change the coordinates as follows:

- Sub menu two
 - X: 120.0
 - Y: 89.0
- Sub menu three
 - X: 120.0
 - Y: 117.0
- Sub menu four
 - X: 120.0
 - Y: 148

Creating the submenu buttons – second tier

Repeat the same button creation process for the second tier as we did for the first tier submenu. Make sure you are on the submenu one layer in the second tier:



Additionally, make sure you start the process on frame twenty- three. Position the second submenu button as follows:

- X: 120.0
- Y: 82.0

Create the fade sequence – second tier

When a visitor clicks the second main menu button, the first tier submenu will be hidden and the second tier menu will show beneath. Since the fade sequence is the same for each submenu button, we will use the first submenu button as our example. Follow these steps to create the fade sequence:

- Right click frame thirty- three and select **Insert Key Frame**

For frame twenty-three:

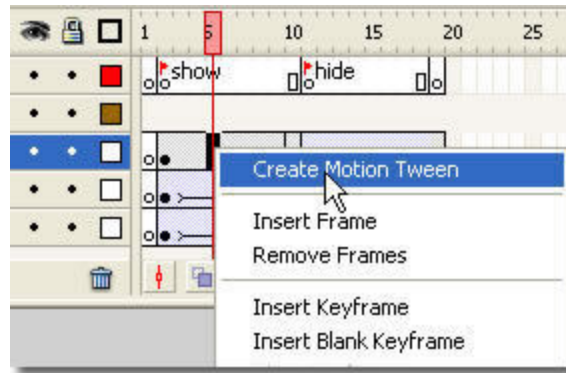
- In the properties panel, set **Alpha** as color

For frame thirty three:

- In the properties panel, set **none** as color

To complete the fade sequence, complete the following:

- Between frames twenty- three through thirty- three, right click and select **Create Motion Tween**:



Repeat this process for the remaining second tier submenu buttons replacing instance and library names where appropriate. Change coordinates as follows:

- Sub menu two
 - X: 120.0
 - Y: 113.0
- Sub menu three
 - X: 120.0
 - Y: 145.0
- Sub menu four
 - X: 120.0

- Y: 177.0

Adding a stop command

Right now, when the movie loads, the two main buttons will fade in and then loop. We add a stop command in the main timeline to solve this issue. Follow these steps:

- Make sure you are on the action script layer
- Right click and select Insert Key Frame
- Press F9 on your keyboard to open the actions panel and type the following:

```
stop();
```

Save your file and use **dhtmlmenu_start.fla** to compare.

The functionality

Since we have completed the user interface, let's review briefly the menu's intended behavior. When the movie loads, the main menu buttons will fade in until frame ten. At this point, the visitor will be able to left click button one or two and the appropriate submenu will show beneath. We use conditional logic to accomplish simple toggling behavior.

Why does the menu look odd visually?

If you look at **dhtmlmenu_start.fla** and play the movie, you will notice the following:

- The main menu buttons fade in until frame ten
- After frame eleven, both main menu buttons disappear and only the first tier submenu is visible
- After frame twenty -two, both main menu buttons remain hidden and only the second tier submenu is visible

Why is this happening? Because of scope; neither main menu buttons are present in the main timeline at frame eleven or twenty- two. To fix this issue, follow these steps:

For the first main button:

- Right click frame thirty -four and select **Insert Frame**

For the second main button:

- Right click frame eleven, and select **Insert Key Frame**
- Position the button at the following coordinates:
 - X: 82.0
 - Y: 177.0

Continuing, for the second main menu button:

- Right click frame twenty- two and select **Insert Key Frame**
- Position the button at the following coordinates:
 - X: 82.0
 - Y: 57.0

Lastly, for the second main menu button:

- Right click frame thirty -four and select **Insert Frame**

Save your file and use **dhtmlmenu_visible.fla** as a comparison.

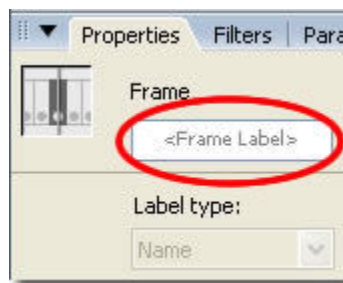
Showing the first-tier and second-tier submenus

In order for the submenus to show or hide, we need to set two frame labels, one set to show and the other set to hide. We use frame labels so that we can move frames containing content without breaking our code. For example, if we instruct a submenu to show by using a frame number...

```
servicesBtn.onRollOver=function() {
servicesmc.gotoAndPlay("10");
}
```

...and then move the show sequence to another frame number our code will not work. Using frame labels solves this issue. To add frame labels to show our submenus, follow these steps:

- Make sure you are on the labels layer
- Right click frame eleven and select **Insert Key Frame**
- In the properties window, locate the frame label text field:



- Type **submenuone**

Repeat this process for the second tier submenu located on frame twenty- three, replacing the frame label with **submenutwo** and extending the submenu to frame thirty-three by right clicking and selecting **Insert Frame**.

Adding Stops

When the movie loads, the main menu buttons will fade in until frame ten and then stop. When the visitor clicks a main menu button, the appropriate submenu will show and loop. Follow these steps to solve the problem:

- Make sure you are on the **stops** layer
- Right click frame ten and select **Insert Key Frame**
- Press **F9** to open the actions panel and type the following:

```
stop();
```

- Right click frame twenty- one and select **Insert Key Frame**
- Press **F9** to open the actions panel and type the following:

```
stop();
```

- Right click frame thirty -three and select **Insert Key Frame**
- Press **F9** to open the actions panel and type the following:

```
stop();
```

Programming the menu

When the movie loads and the visitor selects one of the main menu buttons, we need to determine which submenu to show. We use conditional logic for toggling behavior. From the main timeline follow these steps:

- Make sure you are on the action script layer
- Right click frame one
- Press **F9** on your keyboard to open the actions panel and type the following:

```
var where:String="";  
  
buttonone.onRelease=function() {  
    where="submenuone";  
    gotoAndPlay("submenuone");  
}  
  
buttontwo.onRelease=function() {  
    where="submenutwo";  
    gotoAndPlay("submenutwo");  
}
```

Let's examine the code in greater detail.

- We use the **var** keyword to declare a variable, **where**, set it as a string data type and set its value to an empty (null) value.
- We use the **onRelease** method of each main menu button to set our variable to a value:
 - If the value of the variable is equal to **submenuone**, we instruct the movie to play the first submenu
 - If the value of the variable is equal to **submenutwo**, we instruct the movie to play the second submenu

Save your file and preview the results.

Submenus replaying

While this does show either of the submenus, if a visitor clicks one of the main menu buttons again, the same submenu plays again. To fix this issue, modify the code as follows:

```
var where:String=" ";

buttonone.onRelease=function() {

if (where!="submenuone") {

where="submenuone";

gotoAndPlay("submenuone");

}

}

buttontwo.onRelease=function() {

if (where!="submenutwo") {

where="submenutwo";

gotoAndPlay("submenutwo");

}

}
```

Let's examine the code in greater detail.

- We can check the value of the variable using a Boolean check, != (not equal to)
- In the both conditions, we check the value of the variable to see if it's **NOT** equal to the appropriate submenu:
 - If the condition is true, which it always will be, then we set the variable to the appropriate value and instruct the movie to play the appropriate submenu

Save your file and preview the results.

Another way to program the menu

While the code from before works fine, there's a better approach that is a bit more logical and structured because it involves a function. The reason for the function is that it performs all the heavy weight lifting in this scenario. It checks the value of the variable and then plays the appropriate submenus. Inside the buttons, we keep the replay check, but remove **gotoAndPlay**. Follow these steps to implement this approach:

Inside the action script panel, remove the following line of code for both buttons:

```
buttonone.onRelease=function() {  
    gotoAndPlay("submenuone");  
}  
}
```

```
buttontwo.onRelease=function() {  
    gotoAndPlay("submenuone");  
}  
}
```

Replace with:

```
buttonone.onRelease=function()  
  
    play();  
  
}
```

```
buttonone.onRelease=function() {  
    play();  
  
}
```

Below the variable declaration (where), add the following:

```
function go() {  
  
  if (where=="submenuone") {  
  
    gotoAndPlay("submenuone");  
  
  }  
  
  else if (where=="submenutwo") {  
  
    gotoAndPlay("submenutwo");  
  
  }  
  
}
```

Let's examine the code in greater detail.

- We create a function named **go**.
- We determine the value of the variable where using a Boolean check, == (double equal sign):
 - In the first condition, we check the value of our variable:
 - If it's equal to **submenuone**, then we play the first submenu
 - If it's equal to **submenutwo**, then we play the second submenu

Since we are using a function to instruct which submenus to play, we need to add a call to the function where these submenus exist in the timeline. Complete the following steps:

Make sure you are on the action script layer

For frame eleven:

- Right click and select **Insert Key Frame**
- Press **F9** to open the actions panel and type the following:

```
go();
```

For frame twenty -three:

- Right click and select **Insert Key Frame**
- Press **F9** to open the actions panel and type the following:

```
go();
```

For frame thirty- four:

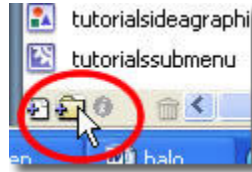
- Right click and select **Insert Key Frame**
- Press **F9** to open the actions panel and type the following:

```
go();
```

One might wonder how this alternation is necessarily more efficient. In this case, it's not. However, when you have multiple main menu buttons which allow multiple submenus to show, using a function to add, modify or remove submenus can make debugging your menu easier. Save your file and preview the results.

Organize the library

Since our library at the moment is disorganized, let's move our work into a logical folder scheme. Use the New Folder icon:



Create the following folders and place the corresponding symbols as follows:

- Sub Menu One – All related sub menu one symbols
- Sub Menu Two – All related sub menu two symbols
- Top Level Buttons- All related top level buttons symbols

Summary

In this article we learned how to use Flash to create a simple DHTML menu. We also learned a bit about:

- Type of menus available
- Advantages and disadvantages for each type of menu
- Creating a DHTML menu similar to HTML and JavaScript with Flash
- Creating buttons using the text tool
- Creating a variable, function and assign different values to the variable
- Creating simple conditional logic using if statements to determine the value of the variable and perform an appropriate action

Take what you learned here and customize your DHTML menu to your heart's content!

If you have questions, please follow the link below.

<http://midwestwebdesign.net/tutorials/contact.aspx>