/Flash/animated sub navigation

Your essential CD
All the code from this month's tutorial can be found on this issue's CD.

Paul Wyatt explains how to create slick, user-friendly navigation bars for your Flash websites and integrate them into an HTML page with Dreamweaver CS3

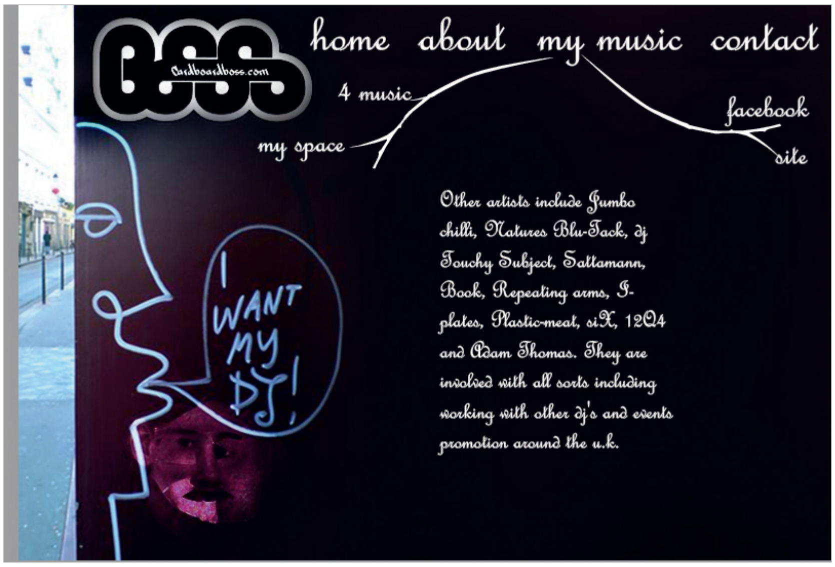
Knowledge needed	Flash CS3
Requires	Flash CS3
Project time	45 minutes

It doesn't matter what kind of content your website provides, if people can't navigate it easily, they'll soon go elsewhere. Some sites score highly, with expertly designed, user-friendly navigation bars that function slickly while seamlessly enhancing the look and feel. Others are just downright annoying, expecting you to know that those unlabelled 3D cubes in the bottom corner are actually the navigation bar.

Most navigation that extends to a sub navigation relies on the same function: clicking a button that prompts an animation to play and load the sub nav bar and content. On clicking another navigation option, this animation then 'rewinds'. This technique is easy to learn and uses a conditional statement in Flash. We'll cover how to do that in this tutorial. It's a technique you can develop to produce your own inspired, creative and, above all, usable navigation bars. We'll also dip our toes into Dreamweaver, looking at the accessibility options it offers in Flash.

Check the final product

Let's start by looking at what we'll create. Open **index.html** from the folder **onthedisc** on your .net CD to run the example. Click on the **my music** link to see the animation play through and reveal a sub navigation of links. If you then click another option from the menu, the previously loaded animation gradually removes itself from the stage while the other content loads. This is much more elegant than the sub navigation suddenly vanishing.


cardboardboss.com
home
about
my music
contact
4 music
my space
facebook
site
Other artists include Jumbo chilli, Natures Blu-Tack, dj Touchy Subject, Saltamann, Book, Repeating arms, I-plates, Plastic-meat, siX, 12Q4 and Adam Thomas. They are involved with all sorts including working with other dj's and events promotion around the u.k.
I WANT MY DJ!

Sub navigation Our final project, the Flash site, contains navigation that uses a conditional statement to determine which actions are taken

Copy the folder **onthedisc** to your hard drive. Open up **partial_tutorial.fla**. On the stage we have the site creative and four buttons – **home**, **about**, **my music** and **contact**. Scrub along the timeline by dragging the small red rectangle above it. You'll see an animation that contains three other buttons. This is the sub navigation, which loads when the **my music** button is pressed. You'll also notice two frame labels: one at frame 2, named **start**, and the other, at frame 13, with the label **play out**. These are important – we'll use ActionScript to refer to these labels, to control when the play head goes to the **start** frame and when it moves to **play out**.

In the middle of the two labels is a **stop** action on frame 12. The animation will stop at this point, but will continue once the prompt from the ActionScript causes it to play again from the second frame label. Making sure that the frame **play out** doesn't play after each button selection is made requires the use of an ActionScripted conditional statement. A conditional statement checks to see if a declared condition is true or false. It is shown like this:

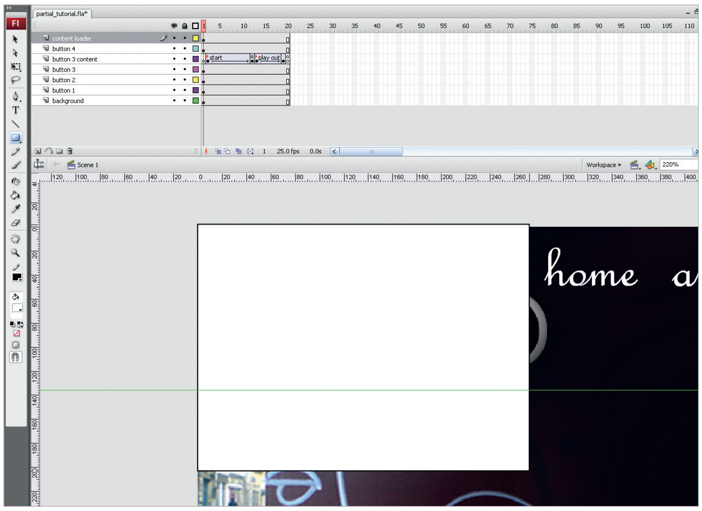
```
if ( condition ){
```

In the next line we tell it to go and do something if the statement is true:

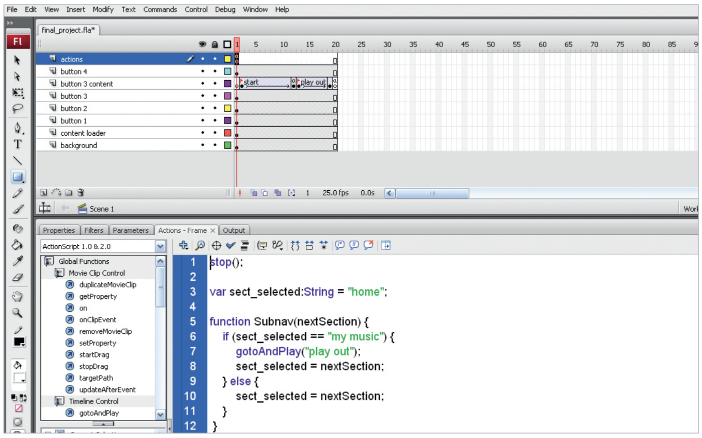
```
gotoAndPlay("play out");
```

If the condition is false, Flash looks at the **else** section of the code, which tells it to carry out an alternate action. **If** and **else** conditional statements operate under very simple logic. If the first statement declared is **true**, then follow this path of action. If it's **false**, then follow this path instead. We'll be using an example of this later to make our sub navigation menu work.

To put this into action, we need to hook up our button instances so they'll work with ActionScript. On the stage of **partial_tutorial.fla**, select the first



Blank clip Create a rectangle to cover the stage. This will become the 'content loader' movie clip – a blank clip into which you can load your content swf files



```
1   stop();
2
3   var sect_selected:String = "home";
4
5   function Subnav(nextSection) {
6       if (sect_selected == "my music") {
7           gotoAndPlay("play out");
8           sect_selected = nextSection;
9       } else {
10          sect_selected = nextSection;
11      }
12  }
```

Creating code Add the code to the **Actions** panel to control when the animation plays, and to define the different buttons that will be used in the sub navigation

Conditional statements If and Else operate under very simple logic

button, **home**. In the **Properties Inspector**, give this the instance name of **btn_1**. Select each remaining button in turn and give them instance names of **btn_2**, **btn_3** and **btn_4**. With the buttons now associated with instance names, we can add the script to make them load the content into the site.

Hit **F9** to bring up the **Flash Actions** panel. To prevent our sub navigation animation playing out as soon as the movie is loaded, we need to add an ActionScript **stop();** on line one, before entering any other code. Each button will load an swf file into a blank movie clip on the Flash stage.

Create a blank movie clip

Create a new layer and name it **content loader**. Lock all the other layers by clicking on the small dot underneath the padlock icon for each. In the **Tools** palette, select the **Rectangle** tool. Position the mouse over the state and you'll notice that the cursor has changed to a crosshair. Position the centre of this at the top left-hand corner of the stage. The stage should now be covered by a large rectangle. Select this and, in the **Properties Inspector**, change the width and height parameters to 800x550 pixels. Change the X and Y coordinates to **0**. Select the rectangle again and hit **F8** to convert it into a movie clip. Give it the name **content loader** and then hit the **OK** button.

In the **Properties Inspector**, give the **content loader** movie clip the instance name of **loader**. Now double-click the **content loader** movie clip on the stage to enter editing mode for it. Select the rectangle within the movie clip and hit **Delete**. Return to the stage and you'll notice the rectangle has vanished and has been replaced by a small circular symbol in the top left corner of the stage. Lock this layer and unlock the rest. You've just created a blank movie clip that matches the size of the stage but has no content in it.

We'll use this to load in our content swf files. As they're exactly the same size as the stage of **partial_tutorial.fla**, it's important that when they're loaded in they're positioned accurately. We've taken care of this by ensuring that the **content loader** movie clip loads in the right position on the stage.

Now we have a content loader set up, we can target it and the buttons to load the swf content files. In the **Actions** panel, add the following code:

```
btn_1.onRelease = function() {
loader.loadMovie("btn1_content.swf");
                                        }
```

This code instructs our first button (with the instance name of **btn_1**) to carry out the function of loading our content movie (**btn1_content.swf**) into the blank movie clip with the instance name of **loader**.

Adapt this code for the remaining buttons by changing the button instance name, and the name of the swf file to be loaded into the blank movie clip.

In depth BBC Audio and Music Interactive's **Tom Kershaw** says Dreamweaver is essential

Tom Kershaw

Occupation Client side developer

Areas of expertise Tom has worked in interactive media for the NHS and BBC Audio and Music Interactive

URL www.twotwentytwo.co.uk

.net: What's so hot about Dreamweaver?

TK: Dreamweaver is a great tool for beginners and experts. If you're just starting out, features such as code hinting, complete CSS support and the template library are extremely valuable. When I started out I used the split view (**Design/Code**) to see what code was being written, and that formed the basis of my HTML and CSS education. How many times have you had to work on a site that's either been published out of a CMS, or has been worked on by loads of developers, meaning your HTML is all messed up? A simple **select all** and **apply source code formatting** helps with that.

There's a lot of power in Dreamweaver these days, and a good example of this is the **Check Browser Compatibility** feature. This allows you to check a page for CSS/HTML that might not work correctly in a particular browser. Dreamweaver has had strong support for server-side technologies for some time. Implementing systems such as user authentication, adding, editing, deleting and browsing records now takes minutes rather than hours. All that, plus a decent FTP package and strong site management features, adds up to one fine deal.

.net: But didn't it create loads of extraneous, bloatworthy code?

TK: That's true, but it's got a lot better these days. The extra code wasn't the main issue I had with it. At first, the menus and toolbars can be a bit confusing. There seems to be an option for just about everything and, when you're getting to grips with the app, this can sometimes feel a little overwhelming. Also, the help system can be a little unclear at times.

.net: What project have you used Dreamweaver in recently?

TK: I used Dreamweaver on a recent project for Radio 1's Live Lounge. The file management and CSS tools were the handiest. Checking pages in each browser before you publish is great, and having the ability to check files in and out is crucial for version control if you're working in a team.

Copy the code three more times and change the button instance names to **btn_2**, **btn_3** and **btn_4**. The content files change to **btn2_content.swf**, **btn3_content.swf** and **btn4_content.swf**. For the full code refer back to the final example, **final_project.fla** on the CD.

Hit **Ctrl** and **Enter** to test the movie. If any of the buttons fail to work, the most likely cause is incorrect naming of the instances within the ActionScript. Check that these have been named correctly and preview the movie again.

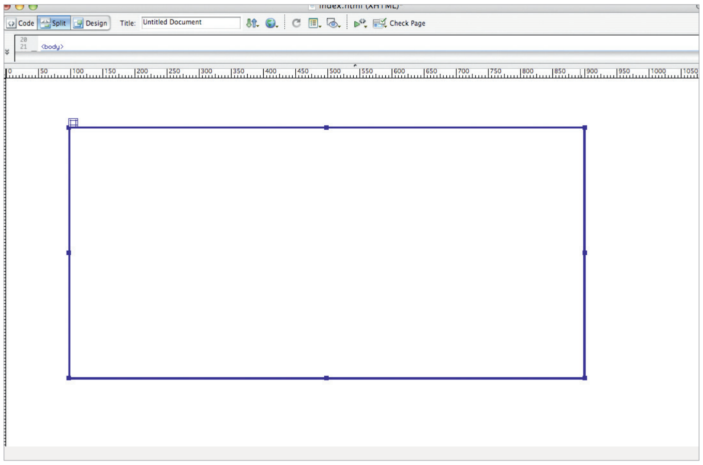
Adding the conditional statement

The **my music** button will cause a content swf and our sub navigation to load. Return to the **Actions** panel and, after the ActionScripted **stop** on line 1, enter the following code:

```
var sect_selected:String = "home";

function Subnav(nextSection) {
 if (sect_selected == "my music") {
 gotoAndPlay("play out");
 sect_selected = nextSection;
 } else {
 sect_selected = nextSection;
 }
}
```

>>



```
<body>
```

Making changes Draw out an AP <div> to contain your Flash movie, and use Dreamweaver to change the movie's properties and combine Flash and HTML elements

>> Let's have a closer look at this code and what makes it tick. We've created a function named **Subnav**. This uses a conditional statement to see if the section currently selected is **my music**. If it is, the play head will move to the frame with the label **play out**. If another button has been selected and this isn't the **my music** one, then the play head will stay where it is.

For this to work, we need to add an extra line of code to each button's ActionScript. First, we need to assign each button a name so that our **Subnav** function knows which has been selected. For **btn_1**, add the code:

```
Subnav("home");
```

under the second line of the button's ActionScript. The full script for the button now reads:

```
btn_1.onRelease = function() {
loader.loadMovie("btn1_content.swf");
 Subnav("home");
}
```

Add the same code to the remaining three buttons. Change the associated name to that of the button itself. So **btn_2** becomes **Subnav("about");**, **btn_3** becomes **Subnav("my music");** and **btn_4**, **Subnav("contact");**. **Btn_3** has two additional lines of code added to it:

```
this._parent.sect_selected = "my music";
gotoAndPlay("start");
```

This ActionScript states that this selection is defined as current and that, when clicked, the play head will move to the frame labelled **start** after loading the content. That's it for buttons. Hit **Ctrl** and **Enter** to preview and test your movie.

Switching to Dreamweaver

Select **File > Export > Export movie**. Name your swf file **content_site.swf** and save it into your **onthedisc** working folder. Open up Dreamweaver and, from the splash screen's **Create New** menu, select **HTML**. Save your file into the same folder as your Flash swf. Name it **index.html**. At the top of the interface are a number of panels and options. Select the **Design** view from the available view options. Select the layout category in the list of options under the topmost menu. Here, you'll find options for creating <divs>, tables and collapsible panels. The second icon along is the **Draw AP Div** option. Click this and drag out a container for your Flash file. Click on the handle of the Ap <div> and, in the **Properties** panel, change its size to 800x550 pixels.

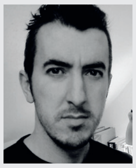
When an AP <div> is created, you'll see the outline of this in the **Design** view of Dreamweaver. If you flip to the **Code** view, you'll notice that Dreamweaver has inserted a <div> tag and given this the value **apDiv1**. You can use the **Properties Inspector** to rename this, or use the **AP Elements** panel.

Select an insertion point by clicking within the AP <div>. Select the **Insert** menu and choose **Media** and then **Flash**. A pop-up box with the heading **Object tag accessibility attributes** will appear. The data entered here allows a screen reader to read the title of the Flash object. Enter a name for the Flash object and a one-letter keyboard equivalent in the **Access key** text box. Working with Dreamweaver, you'll find that it has a number of functions for helping implement the accessibility standards in web pages.

By selecting the Flash movie, you can change a number of properties for it in the **Properties** panel. As well as changing the object's scaling preferences, you can also launch Flash directly from Dreamweaver. Do this by clicking on the **Flash edit** button in the **Properties** panel. If you're combining Flash and HTML elements, Dreamweaver lets you play the Flash movie from the interface by hitting the **Play** button. You can test the Flash button functions directly from the Dreamweaver interface.

Right-click in the **Design** view to select **Page Properties** from the pop-up menu. Select this and change the background colour to a mid-range grey. Click **OK**, save the file and then hit **F12** to preview the HTML page. ●

In depth Steve Alexander, Four Communications' technical lead, casts his eye over Dreamweaver



Steven Alexander

Occupation Full-cycle web developer
Areas of expertise Steve set up creative media company 5cubed.com and is now with Four Communications
URL www.fourcommunications.com

.net: What's not so hot about Dreamweaver?
SA: Dreamweaver is big, dare I say bloated. It packs a lot in there, but in my time I suspect I've only used around 10-20 per cent of its functionality – and I'm willing to bet this applies to the majority of users. The average installation takes up around 200MB of disk space. I can think of many development tools that would fit into that footprint with space left over.

The **Design** view, while useful for quickly prototyping pages and layouts, can all too easily be abused by people who solely use that view and don't have much understanding of the underlying code that's generated. Dreamweaver does its best to create clean, efficient code behind the scenes, but it's no substitute for an experienced developer.

Of course, there's the cost as well. Many people forget that, when their company is paying the licence or they're running their trial version. As a single user, you're looking at around £400 for the most recent version.

.net: What would you prefer to use?
SA: At the risk of sounding like a cliched pretentious developer, give me Notepad and a good FTP client any day! Actually, that's not strictly true. In the real world of tight deadlines, it's useful to have a few time-saving features. Syntax-colouring, auto-formatting and tag completion are often genuinely useful. With this in mind, I'd say my favourite combination is Notepad++ (notepad-plus.sourceforge.net) and Filezilla (filezilla-project.org). Both are feature-rich, flexible and lightweight.

.net: Is there anything on the market that works just as well but is free?
SA: Absolutely, especially for anybody who's comfortable with the Code view of Dreamweaver. There are some old favourites such as HTML-Kit (www.htmlkit.com), which has a free version. Also, I highly recommend checking out Aptana Studio (www.aptana.com). For those who prefer the Design view of Dreamweaver, Nvu (www.nvu.com) may be an alternative.

About the author

Name Paul Wyatt
Site www.paulwyatt.co.uk
Areas of expertise Interactive web design, motion graphics and video
Clients *The X Factor,* Fanta, Smirnoff and Research Studios
Which advert annoys you most? Any Halifax one!