In the frame

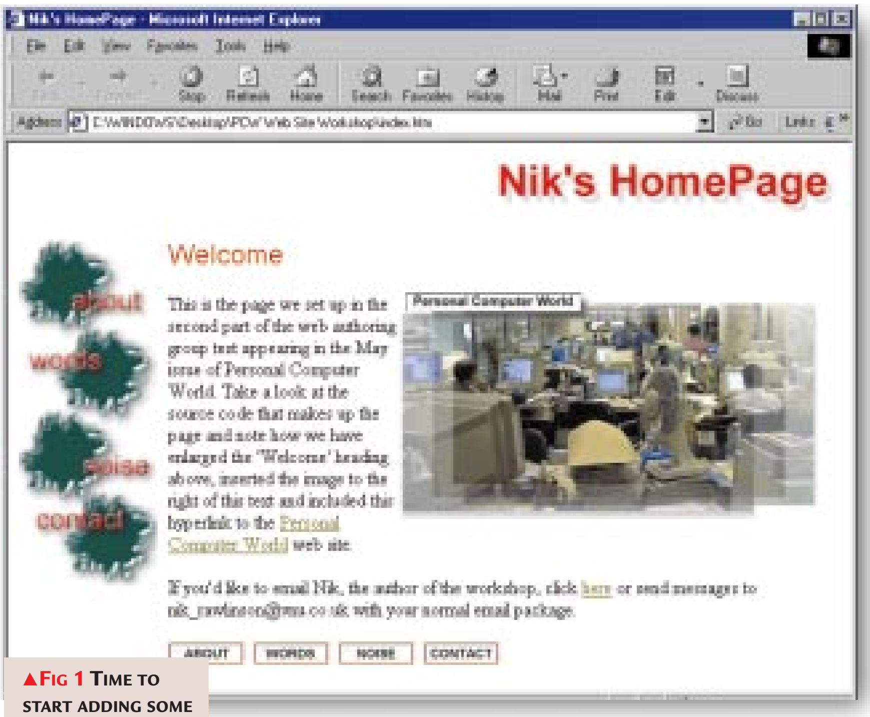
Nik Rawlinson shows you how to fill the content frame of your web page.

In the March *Workshop*, we built the initial structure of our web homepage, defining three frames that would house the individual pages of our site. To recap, a border frame above and a menu frame to the left will contain constant page elements that will not change, giving each of our pages a unified feel. Keeping the menu bar permanently in view will also facilitate easy navigation. In effect, we started with the hardest part first, as frames are notoriously fiddly to set up and it's easy to write code that looks fine in your editor but actually does nothing.

Here, we'll begin to fill the 'contentframe', the large frame that sits below the banner and to the right of the menu. As with the first installation, all the files in the site are contained on this month's cover CD.

First, let's look back at a single line from the menu. This is an imagemap in which certain areas of the picture are defined as hot spots which link to other pages. If you open the source code for the imagemap you'll see that when a user clicks with a rectangle defined by the 'Area Shape=Rect Coords' tag the relevant page is directed into the 'contentframe' with the 'target' command, so we don't need to concern ourselves with pointing any further pages in any particular direction and can concentrate solely on the content.

A sample page, 'welcome.htm' [Fig 1], is on our cover CD. It's not going to win any awards but was written to demonstrate a number of text formatting codes available to the HTML author. It is the first page that will be displayed in the 'contentframe', being specified as such in the 'index.htm' file. Rather than reproducing it here in detail, we'll just look at the most important lines.



Nik's HomePage

Welcome

This is the page we set up in the second part of the web authoring group test appearing in the May issue of Personal Computer World. Take a look at the source code that makes up the page and note how we have enlarged the 'Welcome' heading above, inserted the image to the right of this text and included this hyperlink to the Personal Computer World web site.

If you'd like to email Nik, the author of the workshop, click here or send messages to nik_rawlinson@vnu.co.uk with your normal email package.

▲Fig 1 Time to start adding some meat to the skeleton of our pages defined in the March issue workshop

■ Fonts

Your site could be viewed on any number of platforms, from PCs and Macs to PDAs. Because of this there is a chance that by specifying a particular font you will make your page inaccessible to visitors using a particular platform. Most likely their browser will revert to its default font, which could destroy your carefully worked out formatting.

For this reason it's good practice to specify font groupings rather than a specific font. When a browser reads a page it will implement the first font of a grouping it has installed and ignore the remainder of the line, so instead of specifying only the Arial font, use the line

```
<font face="Arial, Helvetica,
sans-serif">
```

or, rather than Times New Roman, use

```
<font face="Times New Roman,
Times, serif">
```

As you have specified your font tag there is no reason to define it again before entering text — instead, simply add further attributes.

Let's change the size. There are two ways of doing this. We can either add or subtract incremental values. For example:

```
<font face="Times New Roman,
Times, serif" size=+2>
```

or specify a pre-defined size between 1 and 7;

```
<font face="Times New Roman,
Times, serif" size=7>
```

Now, let's change the colour. If you are using a simple colour like red, blue or black, you can define it by name:

```
<font face="Times New Roman,
Times, serif" size=7 color=
black>
```

Better practice, though, is to use the hexadecimal equivalent to ensure that the tone achieved is exactly what you are after. Another benefit of using a hexadecimal rather than named value is that it allows you to exactly match a page background colour with that of an image. Many graphics packages contain a web-safe palette of colours defined by their hexadecimal value. By using the same values on your pages it is possible to place a non-transparent graphic onto a coloured background without the edges being visible.

The code for black is #000000. Those who know a little about hex code will be able to work out that being at the

[FIG 2] **Specifying font groupings**

```
<p><font face="Arial, Helvetica, sans-serif" size=+2 color="#FF0000">Welcome</font></p>
```

opposite end of the scale, white could be nothing but #FFFFFF. Experiment with the values in between to see the effect that changing a single alphanumeric can have (hint: for red try #FF0000, or try #0000CC for blue), or alternatively have a peek at a site which *Hands On Internet* columnist Nigel Whitfield mentioned in the March issue: www.lynda.com/hexv.html.

So, let's not clash with what we already have and we'll opt for a sans serif red font for our heading. Notice that every opening tag is closed at the end of the line and <p> and </p> define the beginning and end of our paragraph [Fig 2]. By defining it as a paragraph it will have a blank line inserted below. The </font> tag defines the point at which the font attributes selected stop applying to text on the page. By closing the font tag in this way we have instructed the browser to return to the default font style. On a PC this will most likely be 12pt Times New Roman.

The original designers of HTML included a number of font shortcuts, or specific headings tags. Ranging from <H1> the largest, to <H6> the smallest they provide an easy way of changing the size of your text while retaining the current font face and colour.

Images

Images are more or less an essential part of any site. Like text, they can be added 'as is', while supplementary attributes can define the specific way in which a browser will handle them:

```
<img src="../images/picture
.jpg">
```

The above line tells the browser to display the image called picture.jpg which can be found in the images directory. Using the same directory navigation structure employed by DOS, the "../" tells the browser that this directory is found one level up from the HTML file we are currently reading.

If we wanted, we could leave it at that but our picture would simply be dropped onto the page on the next available line. Adding further parameters will allow us to specify exactly where and how it should appear.

The first thing to specify should be the size, measured in pixels. Most graphics packages will tell you this, so it's merely a matter of transposing the dimensions to the image attribute line.

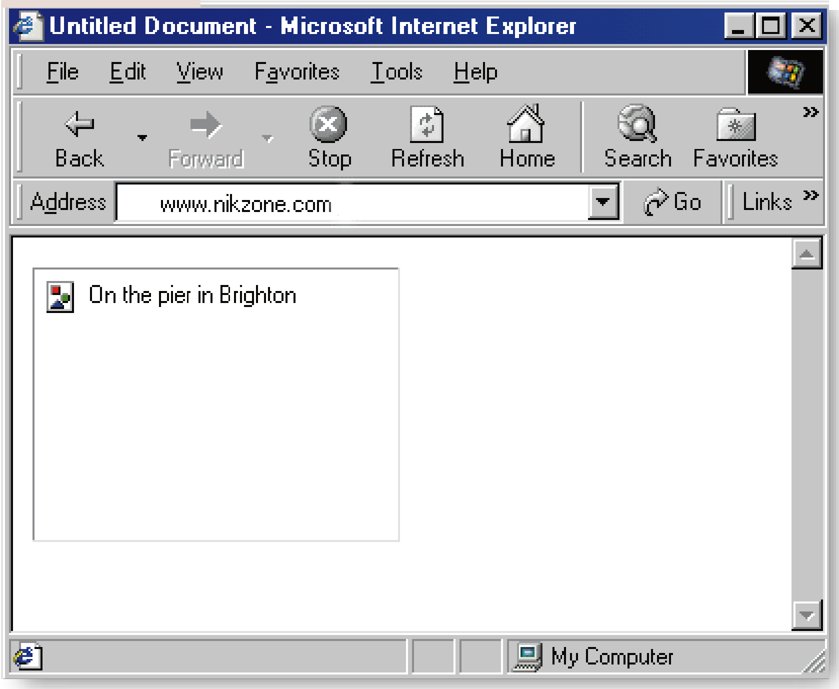
[FIG 4] **Adding a picture**

```
<img src="../images/picture.jpg" height=100 width=150↙
 border=0 align=right alt="Nik and Paul on holiday last↙
 year">
```
(*↙ Code string continues*)

Although this is not strictly necessary since text on a page, which will usually arrive before any images, will move around to accommodate the pictures, it is good practice to specify the image size. This means that should the browser be unable to download the image it will instead display a blank placeholder [Fig 3] of the size defined to retain any page formatting. If no size is specified, a small broken link icon will appear and page formatting will be lost.

Images are often used as links and so it is good practice to get into the habit of specifying whether or not you want a border. Leaving out the border attribute will by default bound the image with a box of whichever colour you have chosen for links on that page. Specifying 'border=0' will prevent this, and images will be presented 'as is'.

▼Fig 3 This browser has been set to ignore images but the <ALT> tag leaves it accessible with this placeholder



It is a good idea to provide a short description of the image subject matter. This is done by using the 'alt' tag [Fig 4] and enables users of text-only browsers, who would otherwise be unable to view the image, to follow what is happening on the page. It is also useful for the sight impaired who may be using a voice recognition system that can read 'alt' tag contents. So, let's put it all together and add a picture to our site. We'll imagine it's going to be dropped into the middle of a long block of text so we'll also specify its alignment. This can be left, center (US spelling) or right. See Fig 4. Note that it is not necessary to place commas between each attribute.

Links

Next, we're going to add some links. We dealt with one of the principle types of

[FIG 5] Text with an image tag

```
<a href="about.htm"><img src="../images/about.jpg" height=20 width=50 border=0 align=↙
center alt="about"></a>
```

(↙ Code string continues)

link in the first part of this workshop (March issue); the direct link to a page. This was hidden within our imagemap menu, so take a second look at the code to identify the hotspots.

Placing a standalone link on a page is far simpler. Technically, we do not need to link this 'welcome' page directly to any of the other pages on the site because the imagemap will take care of it, so we'll add links for those users with older browsers which cannot cope with frames. When we've finished, we'll amend the file 'index.htm' that defines our frameset to point to this page when it is read by such a browser.

A sample page, welcome.htm, is on our cover CD

The tag for a standard link is 'href', as in the example:

```
<a href="about.htm">
```

As we are not straying outside the HTML directory which includes the 'welcome.htm' page we are currently constructing, we are able to specify just the filename and not the location. We will follow this tag with the physical clickable link and close the line with the terminator tag </a>. A clickable link is usually an image or a line of text, as in this example:

```
<a href="about.htm">click
here for the about page</a>
```

Representing the link as an image is simply a matter of combining a link element with the image tags described above.

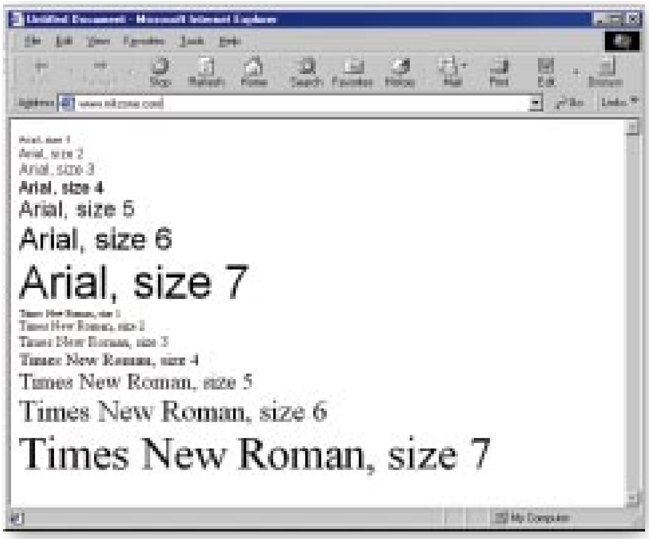
We'll imagine that we have designed a small 'about' button in a graphics package. It is 50 pixels wide and 20 pixels high so it's quite small. To drop it onto the page and make it function as a link we replace the plain text in our example above with an image tag, as in Fig 5.

Note that we have added the word 'about' to the alt tag. When adding an image as a button in this way, an 'alt' tag is even more important than in the instance of the simple illustration on a page (described above) as it ensures that users can still navigate your site even if for some reason they are unable to download the images.

The second most common form of link is the mailto link, enabling users to click and launch an associated email package. The email address specified in the link will be dropped in the 'To' field of the package. It works in much the same way as a standard link in that the physical clickable element can be either text or an image (a picture of an envelope, say) and it must be terminated by the </a> tag. See Fig 6.

[FIG 6] The mailto link

```
<a href="mailto:nik_rawlinson@vnu.co.uk">click here to email me</a>
```



Arial, size 1
Arial, size 2
Arial, size 3
Arial, size 4
Arial, size 5
Arial, size 6
Arial, size 7
Times New Roman, size 1
Times New Roman, size 2
Times New Roman, size 3
Times New Roman, size 4
Times New Roman, size 5
Times New Roman, size 6
Times New Roman, size 7

HTML TAGS INCLUDE SEVEN PRE-SET FONT SIZES FOR QUICK TEXT FORMATTING

Now that we have inserted links into our welcome page, we'll make it accessible to viewers without frames-compliant browsers. Open the file 'index.htm' found in the HTML directory and replace the line that reads: 'This page uses frames and requires a compatible browser which yours, I'm afraid, is not' with;

```
This page uses frames that
your browser cannot display.
To access the site without
frames, click <a href="html/
welcome.htm">here</a>.
```

We are of course only covering the very basics of web authoring here but when it comes to designing your own pages you should have the tools you need to get started. By far the best way of learning is to try it out for yourself and have a look at other sites out there on the net to see how their authors have tackled similar problems to those you will face.

In the meantime, remember this maxim: in web design, less is most often more. Animated graphics and colourful pictures are great but they should be used in moderation. Put too many on your page and your users will probably give up on you as they tire of waiting for them to download. Do not be afraid to leave areas of white space. Far from looking like you have left something out, they can actually draw the reader's attention to the points that matter — the words on your pages.

Next month, in the final part of this workshop, we'll look at tables and forms.

PCW CONTACTS

Nik Rawlinson can be contacted via the PCW editorial office (address, p14) or email nik_rawlinson@vnu.co.uk