WEB EXPERT

Master HTML5

More and more browsers are supporting the new version of HTML, so every web designer should learn the language. **Nik Rawlinson** explores a whole new way of working online

HTML5 may be the future of web design, but it's a long way from being a fully ratified language. So it probably seems a little premature to embrace a new way of working at this point. However, although progress has been a little ponderous, HTML5 is already enjoying widespread adoption. Browser manufacturers know their products are effectively dead if they don't adopt the unratified specification to some degree, so we've seen rapidly growing support in the past two years.

When Microsoft shipped Internet Explorer 9 (IE9) in March 2011, it joined Safari, Firefox and Chrome in rendering properly large portions of the HTML5 specification. Since these browsers account for almost all web browsing on any platform, it's safe to use HTML5 tags in web development and start reaping the benefits.

TAG TEAM

HTML5 introduces a number of new tags. Crucially, however, it makes several others obsolete. That doesn't mean your HTML4.01 site won't display in the latest generation of each browser, as they each have excellent backwards-compatibility. However, you should consider swapping obsolete tags for their HTML5 alternatives.

You're probably familiar with using to define a block on the page, which you can then style up using CSS. This has been retained along with (to pick out inline blocks for individual formatting) and joined by specific block types such as <nav>, which is used to define a navigation element, such as a menu.

Many new tags help to define the content of a page rather than just position it within the flow. For example, <figure> ties an image to its caption, allowing you to both style the pair as a single block and to sub-style the contents – image and caption – individually inside it. Even if you don't intend to revisit existing sites and re-code them using HTML5, you should stop using tags such as <frameset>, <frame> and <noframes> as they have been deprecated along with <strike>, <u> and <font>. Luckily, most of HTML4.01's tag structure remains in place, with HTML5 building on it. Still, you should replace all the tags listed in the box below.

Beyond creating a clear distinction between content and presentation, HTML5 has introduced elements that simplify the integration of non-textual elements. The <canvas> tag provides a container for scripts that draw graphical elements such as shapes and graphs. Meanwhile, <localStorage> lets you save data on your visitors' PCs for use in the same session or future visits, and <audio> lets you embed audio directly on a page; a <video> tag does the same for visual content, with optional attributes for autoplaying, embedded controls, looping and a poster frame that displays before the video kicks in.

The code below would embed a file called shopper.mp4 in your page, with accompanying controls. The video window is 640 by 480 pixels, and the movie file would load at the same time as the page. We've also specified a poster frame (shopper.jpg) to display in the video box like a thumbnail on a DVD menu. At no time is a call made to Flash, Silverlight or other playback tool. The text displayed between the opening and closing <video> tags handles errors, and is displayed only if the visitor doesn't have a compatible browser:

```
<video src="shopper.mp4" controls
 width="640" height="480" poster="shopper.
 jpg" preload="preload">Sorry - no video
 for you. Please upgrade your browser.
 </video>
```

Of these elements, only the video source is required. We could embed the video in a simpler manner with:

```
<video src="shopper.mp4"></video>
```

FIVE ALIVE

To demonstrate how easy it is to develop with HTML5, we'll build the basic structure of a simple web page, using HTML5-native tags where possible. This should render properly in up-to-date versions of Chrome, Safari and Firefox, as well as Internet Explorer 9. Next month, we'll build on this to produce a completed page, and over the next few months we'll look at how CSS3 can be used to style our on-page elements.

One benefit of HTML5 is evident in the first line:

```
<!DOCTYPE HTML>
```

This declaration is much simpler than its HTML4.01 equivalent, which was hard to remember and usually ran along the following lines:

INTRODUCTION

In the two decades since Tim Berners-Lee published the groundbreaking *HTML Tags*, the technological world has changed, and our expectations of what the web should deliver have changed along with it. The modern web user wants photos, podcasts and HD video. HTML5 delivers all this and more, in a more efficient manner than its predecessor ever could.

It's simpler to work with, too. If you've been put off coding a site by hand because you think it's too complicated, now's time to take the plunge. You'll be surprised at how far things have moved on.

Nik Rawlinson
Web developer

letters@computershopper.co.uk

Gone but not forgotten Tags to avoid with HTML5

Most of the tags used to build HTML4.01-compliant sites have been retained as part of HTML5, but the following should no longer be used:

<acronym> Use <abbr> for abbreviation.

<applet> Use <object> instead.

<basefont>, <big>, , <strike>, <tt>, <u> These have been removed, as they were used for formatting rather than defining content. Formatting should now be conducted entirely using CSS.

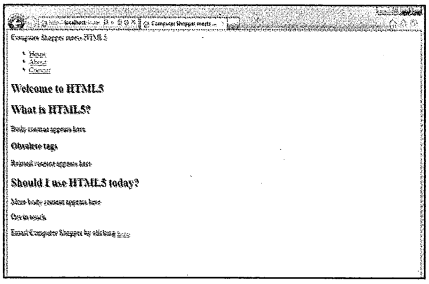
<dir> Use <ul> to create an unordered list.

<font> This was a styling tag, and all styling should now be conducted using CSS.

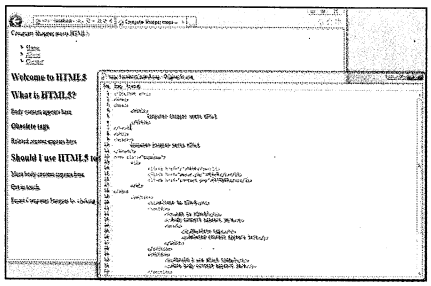
<frame>, <frameset>, <noframes> These tags were formerly used to define spaces into which external files were loaded. However, the same result can be achieved in a neater, accessible way by using layers and blocks.

Several attributes, used to define the way a standard object could be presented on the page, have also been removed. You can no longer specify the body background or height of a horizontal rule, how an iframe should handle scrolling, or how table cells are padded within your tags.

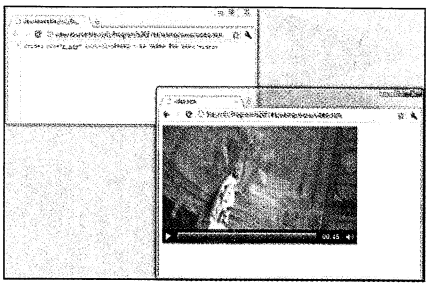
As a general rule, if a tag or attribute in your HTML4.01 code was used to format on-page text or another element, such as an image, table or divider, its function must now be handled by CSS if you intend to develop using HTML5.



▲ The page is basic but well structured with well-defined sections, ready to be styled using CSS



▲ HTML5's semantic markup shows how the various parts of our page relate to one another



▲ New tags such as <video> let us use media content natively with no need for plug-ins

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD
 HTML 4.01//EN" "http://www.w3.org/
 TR/html4/strict.dtd">
```

It's clear how the architects of HTML5 have been working not only to make the language more flexible, but also easier to work with. We'll define our head section, immediately after this doctype declaration, as we would have done previously:

```
<html>
<head>
  <title>
    Computer Shopper meets HTML5
  </title>
</head>
<body>
```

Now we'll look at <header> and <nav>, which are the first of HTML5's semantic markup tags:

```
<header>
  Computer Shopper meets HTML5
</header>
<nav class="topmenu">
  <ul>
    <li><a href="/">Home</a></li>
    <li><a href="about.php">About</
a></li>
    <li><a href="contact.
 php">Contact</a></li>
  </ul>
</nav>
```

The <header> tag didn't appear in HTML4.01, which prompted many developers to assign id="header" to a regular div so they could remember what they'd called it when addressing it in their CSS. By giving it a defined tag of its own, HTML5 lets you address it using CSS while at the same time defining the content. Aggregators and search engines will understand its contents, and can use it in place of the page title in a list of search results.

Likewise, <nav> has been introduced to define a container for navigational elements. It doesn't define the presentation of the navigation, so it won't magic up a properly linked menu for you; again, this should be handled using CSS.

Because both these tags are semantic – defining the content rather than formatting it – they can be used anywhere you like, and several times on the page. You might want to use a <header> as a cross heading before each block of text, and another <nav> further down your page so your visitors can jump to the various parts of a subsection. You can therefore apply specific styles to each tag. We have done this with the <nav> tag, formatting it using a class within our CSS called 'topmenu', to denote the menu that runs across the top of the page.

BODY LANGUAGE

With the menu and header out of the way, we can start to write the bulk of our content – the main body of the page:

```
<section>
  <h1>Welcome to HTML5</h1>
  <section>
    <h2>What is HTML5?</h2>
    <p>Body content appears here</p>
    <aside>
      <h3>Obsolete tags</h3>
      <p>Related content appears
       here</p>
    </aside>
  </section>
  <section>
    <h2>Should I use HTML5 today?
     </h2>
    <p>More body content appears
     here</p>
  </section>
</section>
```

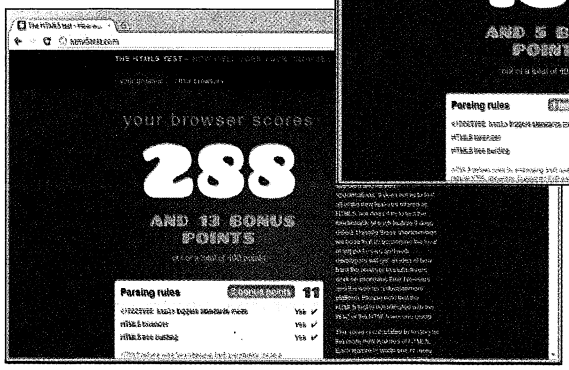
You'll notice a proliferation of <section> tags here, each of which is used to group together various related elements on the page and define the start and end points of a discrete section – like a feature in a magazine that might include headlines, body copy, images, captions and boxes of related text, bundled within a set number of pages. In our example, one overall section holds the body of our page, with smaller subsections to tie together the heading of each part (<h2>) and its body content (<p>). Like the <nav> tag, these are here purely as a semantic hint, allowing human readers to see where one part begins and ends, so we know that the next heading doesn't define another part of the same block of text, but a different one.

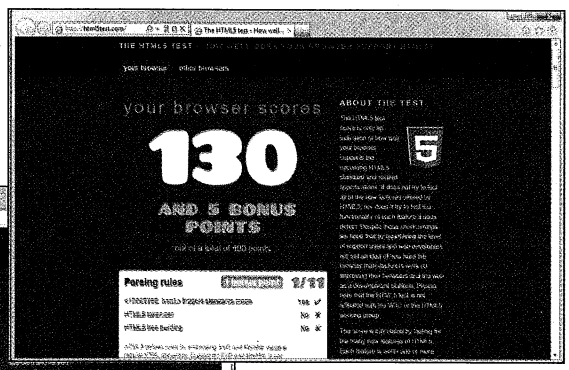
Our first subsection contains an <aside>. This semantic tag ties together a less important header and body content, and tells the reader that the content is related to the contents of the subsection within which its sits, but not directly – it is bonus material. It might be styled to sit to the side of the main page flow. In HTML4.01, we could have achieved the same effect by drawing out a and floating it to the left or right. However, while that would have looked fine in the finished page, it wouldn't have been clear when examining the underlying code that the content related to that which came before it.

We can now close off our page. In doing so, we use the <footer> and <summary> tags:

```
<footer>
  <section>
    <summary>Get in touch</summary>
    <p>Email Computer Shopper by
     clicking <a href="mailto:editor@
     expertreviews.co.uk">here</a>.
  </section>
</footer>
</body>
</html>
```

In HTML4.01, footers were usually defined within a named . Giving footers their own tag in HTML5 is a timesaver, helping to identify them on the page while keeping them CSS-addressable. Our footer contains one section that ties together a summary and a paragraph. The summary defines the content that follows. We can have as many summaries as we want through the page and style them all together or, by directly addressing 'footer section summary' in the CSS, target one specific instance.





◀▲ Not all browsers enjoy the same level of HTML5 support – Chrome scores 288 in html5test.com's test suite, while Internet Explorer 9 scores 130 out of a possible 400

Next month

HTML5 PART II

We show you how you can develop your page outlines further.