Up and Running with HTML

HTML syntax

Introduction to HTML
HTML syntax
Creating links
HTML reference
Next steps

Because HTML is often the people's first experience with coding, there can be a fair amount of apprehension about learning it. Thankfully, HTML syntax is relatively simple and easy to learn. Most people can learn the basics of HTML and begin coding it within the same day.

A markup language

HTML is a markup language. That means that content on the page is "marked up" by tags, which identify the content inside them. A paragraph, for example, can be identified by placing the " " opening tag prior to the paragraphs content and the " " closing tag at the end of a paragraph. The full paragraph would look like this:

This is a paragraph.

Tags consist of a left-angle bracket (<) followed by character or characters that identify the tag (the "p" for paragraph) and a closing right-angle bracket (>). The closing tags for an element are exactly the same as an opening tag, except that a forward slash (/) will precede the tag's characters.

Although most elements require an opening and a closing tag, the closing tag is optional for some elements and not required at all for others. While there are exceptions to the rule, for the most part any element that contains content inside the opening and closing tags also requires a closing tag.

Basic document structure

The core of all HTML documents revolves around three basic tags. First, an html tag (<html>) is required to identify the document as an HTML file. Directly inside the html tag, you'll find the head element (<head>). The head of a document is where you'll find the document's metadata, the document title, and links to external resources such as style sheets and scripts. A good way to think about the document's head is that it doesn't contain any of the page's *visual* content; rather it contains information about the document and the resources that help make the page work. Directly after the document's head, you'll find its body (<body>). The body is where you'll find all of the page's actual content. Headings, paragraphs, images, lists,

tables, and other content will be located here. At its most basic, an HTML file would look like this:

```
<html>
    <head>
    </head>
    <body>
    </body>
</html>
```

DOCTYPES

If you've looked at HTML pages before, you've probably noticed a long, somewhat intimidating tag just before the opening HTML tag. This is a doctype declaration and it's a very important, but often misunderstood component of HTML pages. Essentially, it tells the user agent parsing your page which version of HTML (or XHTML) to expect, so that it knows which syntax rules to use when rendering your page. The doctype you use should be based on the version of HTML you're using to author the page. While that all sounds good in theory, in reality most of the time a doctype is simply triggering "standards-mode" rather than "quirks mode" (based on older browsers' non-standard way of rendering pages). For that reason alone, all HTML documents should be preceded by a doctype declaration. For more information on doctypes, and the history behind them, check out Mark Pilgrim's excellent section on doctypes from his *Dive Into HTML5* book. (http://diveintohtml5.info/semantics.html#the-doctype)

Here are some of the more common HTML doctypes:

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

HTML 4.0 transitional

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

XHTML 1.0 Transitional

```
<!DOCTYPE HTML>
```

HTML5

Element attributes

Some elements can be enhanced through the use of attributes. Attributes allow you to provide more information or additional functionality to the content. Attributes are added to the opening tag of an element and consist of two parts, the **name** and

value. Although the syntax varies based on the version of HTML you're using, it's standard practice to put values within quotation marks.

<h1 class="headline">Article's main headline</h1>

Replaced elements

Some HTML elements represent content that is replaced by an outside resource such as an image, form control, or a video file. These elements are referred to as **replaced elements** and usually have a predetermined width or height. In some cases the elements will have attributes that tell the browser where to find an external resource like an image or video.
<img src="photo.jpg" alt="my awesome photo">

Code structure

HTML documents create structure by nesting elements inside of one another. You might group a section of page content together by wrapping them within a or <section> element, for example. When nesting one tag within another one, you must first close any child elements prior to closing any parent elements.

This syntax, for example, would be *incorrect*:
You must close all nested tags first!

Rather, the correct syntax would be:
You must close all nested tags first!

HTML also has specific rules about which elements can be nested within other elements. A paragraph ( ), for example, can't be nested inside a heading (<h1>). For the most part, these rules are based around the type of content the tag represents. HTML 4 has two basic types of content: **block** and **inline**. Block-level elements typically occupy their own line within a document, usually stacking one on top of each other, in the order they appear on your web page. Inline-level elements can appear within the flow of block level elements. Elements like strong (), bold (<b>), emphasis (), italic (<i>), and span () are always found inside block-level elements.

HTML5, on the other hand, expands the concept of content types to seven different categories of content, with some elements belonging to multiple categories. This replaces the concept of block-level and inline-level elements and adds some additional nesting syntax rules. For the most part, the rules around nesting elements make sense and are fairly easy to pick up. You should be prepared, however, for it to take some time before you fully understand all of the rules surrounding nesting elements. One way to making sure to your code is structured properly is to validate your code through a service like the W3C's Markup Validation service. (http://validator.w3.org/)

Commenting code

Often it is helpful to leave notes to yourself or other developers within your code. Perhaps you want to remind yourself what a certain script does, or remind a co-worker how to properly structure specific content. To do this, you'll use comments. Comments should appear on their own line, and typically appear directly before or after the code they refer to. Comments begin with a left-angle bracket, an exclamation point, and two hyphens (<!--) and end with two hyphens followed by a right-angle bracket (-->).

Comment syntax:
<!-- This is a comment -->

Using special characters

Certain characters are reserved in HTML, meaning that you shouldn't use them outside of their specific purpose. The angle brackets, for example, are reserved for tags, and using them anywhere outside of a tag could cause parsing errors in browsers. However, you can still use these reserved characters and other special characters or symbols, with what is known as a **named character entity**. These special codes tell the browser to replace the entity code with a specific character. These entities begin with an ampersand (&) followed by the entity name and end with a semicolon (;). To display an ampersand, for example, you would type **&** in place of the actual ampersand. You can find named character entity lists within the HTML specifications, although the format for them can be a bit hard to read. You can also find a useful list of character entities on the HTML Reference page, and a more comprehensive list on Wikipedia. (http://en.wikipedia.org/wiki/List_of_XML_and_HTML_character_entity_references) .