Chapter 10

Introduction to HTML and CSS

Summary: Creating a small single-purpose website has many uses. This tutorial will introduce you to the basics of building one.

Skills you will learn:

1. The structure of HTML and CSS
2. How they tie together to build websites

The first thing you'll need is a text editor.

This is a program many developers use to edit the raw code that goes into their websites. The two most popular text editors in use today are Sublime Text (link to: http://sublimetext.com/) and Atom (link to: http://atom.io/).

For a longer discussion of code editors, see the tutorial **Choosing a Code Editor**.

Instead of double-clicking an HTML file and watching it open with your favourite web browser, you can open it with Sublime or Atom and watch the raw code appear before your eyes.

This is the blank screen that you'll see in front of you. Save this file as "index.html" and your code editor will begin highlighting any code you write to showcase its structure.

This is one of the main reasons developers like using programs like this: Being able to see colour-coded structures allows you to see errors before they become problematic and better visualize what you're building. To see what that looks like, type or copy this block into your code editor and see what it looks like:

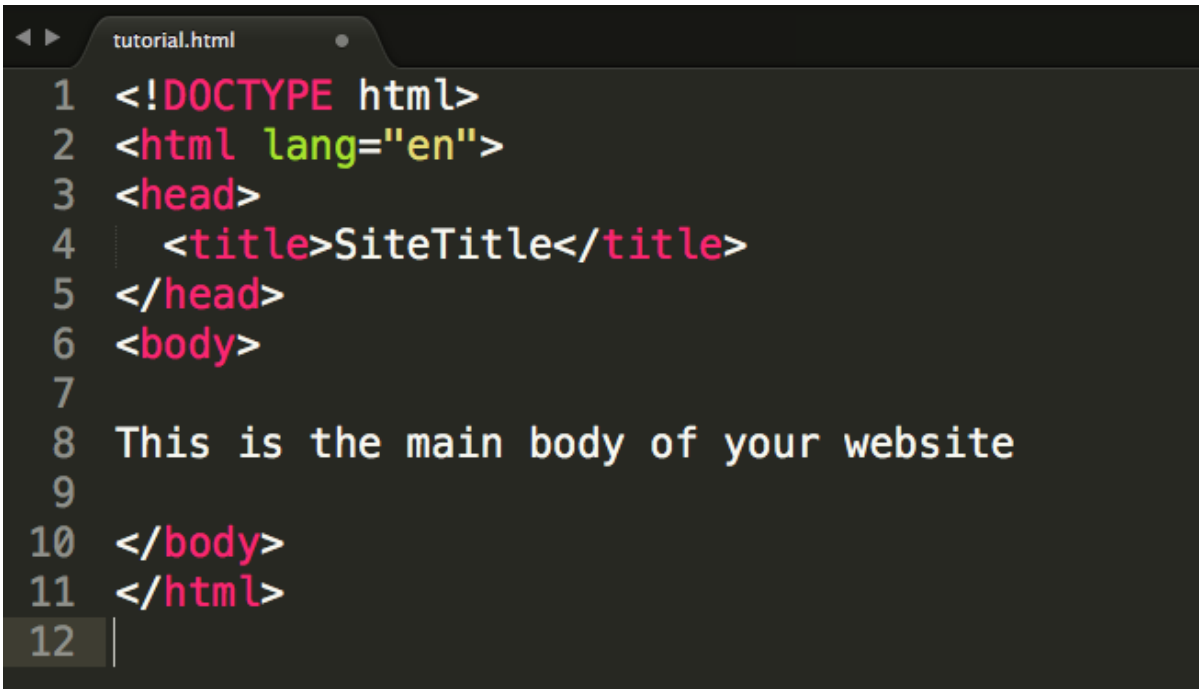
```
<!DOCTYPE html>
<html lang="en">
<head>
 <title>SiteTitle</title>
</head>
<body>
```

This is the main body of your website

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

Hit save. You can also drag this file into an empty tab in your web browser to see what it looks like. This is already a fully functional website.

At this point, your code should look something like this, depending on the colour scheme of your code editor:



So let's walk through it and see what it's doing. The first core concept to understand here is the idea of tags. Anything that appears in angle brackets <> is a tag. So in this example you'll see a <head> tag and a <body> tag and a <title> tag and even an <html> tag.

For every tag, there is also a closing tag (in most cases….we'll get to the exceptions to the rules later). Closing tags include a slash at the front to indicate they are closing tags. So you can see there is also a </title> closing tag and a </body> closing tag and a </html> closing tag, among others.

The text that appears in between the tags is the text that is applicable to that tag. So the Title of the site is <title>SiteTitle</title>. That means the title of the site is SiteTitle and our code knows this because of the tags wrapped around that bit of text.

There are a lot of different HTML tags that you can use, all of which are interpreted by the browser is a slightly different way. You can see the full list of them here.

A note about style: You'll see in the above screen capture that there are indents on some lines of code. Developers use those indents to show which elements (opening and closing tag pairs) are sitting inside of other elements. So you can see that the <title> tag is sitting inside the <head> tag, and is so indented. This helps you to see how your webpage is structured as it gets bigger and more complex.

There are two main sections to your website: The head, and the body.

The head is wrapped in <head> tags and includes all of the little details that your users will normally not see, or only see under special circumstances. The title of your site appears in the browser tab your site is loaded in. You can also control the icon next to it, the information that shows up when you share this page on Facebook or Twitter, all through tags in the <head> field. That's more advanced, though.

The <body> tag is where you build your actual website. All the content that will be loaded when someone goes to your page, all the text and all the images and all the interactives are found in the <body>.

So right now, we have one sentence of raw text in the <body> and that's it. So when you look at your page, it's a blank white space with some text in it. That's not very interesting.

Try wrapping that text in some tags.

```
<h1>This is a headline</h1>
```

Save your document and refresh your browser. See what that did?

```
tutorial.html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title>SiteTitle</title>
5  </head>
6  <body>
7
8      <h1>This is a headline</h1>
9      This is the main body of your website
10
11 </body>
12 </html>
13
```

<h1> stands for Heading 1, and you can include them to make big blocky headlines in your page. You can make a smaller one with <h2> or <h3> or <h4> all the way down to <h6>.

Now drop a <p> tag after them.

```
tutorial.html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title>SiteTitle</title>
5  </head>
6  <body>
7
8      <h1>This is a headline</h1>
9      <h2>Second head</h2>
10     <h3>Third head</h3>
11     <h4>Fourth head</h4>
12     <h5>Fifth head</h5>
13     <h6>Sixth head</h6>
14
15     <p>The main body text in a paragraph tag.</p>
16
17 </body>
18 </html>
19
```

See how that works out?

Now for a final flourish, let's add an image to our page.

Use the <img> tag to make that happen. But there are some core details here: An <img> tag requires secondary attributes inside fo it to make it all work properly.

Attributes are extra pieces of code that sit inside opening HTML tags. They look like this:

```
<img src="toad.jpg" width="200px">
```

Instead of the normal <img> tag, we add the extra two fields src (which stands for "source") and width, which establishes how wide the photo will be on your page, measured in pixels.

The source for your image can either be the name of a file that is sitting in the same folder as your HTML file, or it can be a link to an image on the internet.

In the end, it will look something like this:

tutorial.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>SiteTitle</title>
</head>
<body>

    <h1>This is a headline</h1>
    <h2>Second head</h2>
    <h3>Third head</h3>
    <h4>Fourth head</h4>
    <h5>Fifth head</h5>
    <h6>Sixth head</h6>

    <p>The main body text in a paragraph tag.</p>

    <img src="http://www.tripletsandus.com/disney/Wall-E/characters/Wall-E7.jpg" width="200px">

</body>
</html>

```

But, let's face it, it's all a little ugly right now. We should style it. This is where CSS comes into play.

CSS stands for Cascading Style Sheets and is where everything pretty on the web happens. If HTML is the actor on the stage, CSS are the costumes we wrap them in, the makeup on their faces and the lights above them.

Open a new file in your code editor, and call this one styles.css. You'll notice that they appear in different tabs of your code editor so you can switch easily between them.

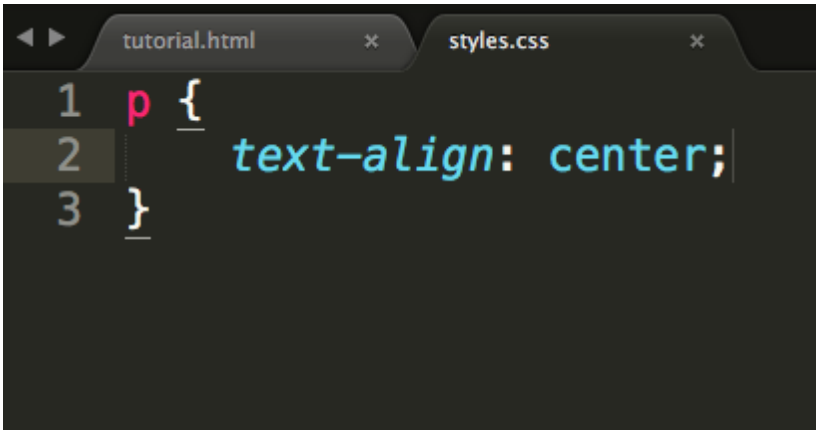
CSS works to enhance HTML pages. Without an HTML page, there is nothing for CSS to do. CSS, therefore, needs to be told two things: What part of the HTML to act upon, and what to do to it.

So it's easiest to think of CSS in those two main components: selectors (what part of the HTML to work on) and rules (how those parts of the HTML need to look). So write this in your CSS:

```
p {
    text-align: center;
}
```

This selects <p> tags in the HTML file and makes them center their text. In this case "text-align" is the CSS property, "center" is the rule being applied to it, and "p" is the HTML element being selected. Collectively, "text-align: center;" is known as a rule. You can add as many rules as you like to each selector.

Now save your file and refresh your page. Right now your window should look like this:



Nothing happened. Your paragraph isn't centered on the page yet. Why not?

HTML pages have to link to their CSS files in order for any styles to take effect. So go back into your HTML file, and this time we're going to add a line of code to the <head> portion of our file.

In the <head>, directly below the <title> add this line of code:

```
<link rel="stylesheet" href="styles.css">
```

This tells your HTML file that, in the same folder as it, sitting directly beside it, is a file called styles.css that is a stylesheet. This stylesheet is filled with rules that the HTML file should be bound by.

tutorial.html styles.css

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>SiteTitle</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>

    <h1>This is a headline</h1>
    <h2>Second head</h2>
    <h3>Third head</h3>
    <h4>Fourth head</h4>
    <h5>Fifth head</h5>
    <h6>Sixth head</h6>

    <p>The main body text in a paragraph tag.</p>

    <img src="http://www.tripletsandus.com/disney/Wall-E/characters/Wall-E7.jpg" width="200px">

</body>
</html>

```

Now save your files and refresh your page. See what happened to the paragraph text.

There are dozens upon dozens of CSS properties that you can select to make new rules for your page. You can see a full list here. Many of them are specific to certain HTML elements: Some will only work on images, while others will only work on text elements, for example. Calling text-align on an image, for example, does nothing at all.

But there are some scenarios where you might not want to treat all paragraphs in your web page identically. In a news site, for example, some blocks of text might be pull-quote, others sidebars, others body text. They should each be styled slightly differently to match their functions on the page. But each are <p> tags, so how do we differentiate them in our styles?

The answer is “classes.”

We can apply classes to any tag in our HTML, and then use those classes to select only that element on the page in our CSS. So let’s add a couple of paragraphs of text to our html to see what that looks like in practice.

tutorial.html styles.css

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>SiteTitle</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>

    <h1>This is a headline</h1>
    <h2>Second head</h2>
    <h3>Third head</h3>
    <h4>Fourth head</h4>
    <h5>Fifth head</h5>
    <h6>Sixth head</h6>

    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Hic, vitae, pariatur, earum, debitis natus velit officiis at voluptatem modi suscipit dolorem laborum minima placeat repellendus quia totam officia! Rem, itaque!</p>

    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Dicta, eligendi, veritatis, quis magnam aliquam consequatur tempora nam sequi minima maxime nesciunt molestias quos nostrum beatae illo dolorem magni autem assumenda?Sunt, nam aliquid natus vero assumenda corporis similique ea blanditiis quos adipisci atque odit fugiat eum impedit nostrum facere at. Dolore, neque quidem excepturi voluptas totam laborum voluptatem qui incidunt.</p>

    <img src="http://www.tripletsandus.com/disney/Wall-E/characters/Wall-E7.jpg" width="200px">

</body>
</html>
```

These are both <p> tags. But if we want the top one to be an introductory paragraph in bold text, but leave the second one in normal unbolded text, we can add a class to the first one. To add a class, we can add a class attribute to our first <p> tag, making it <p class=”leadParagraph”>

```
tutorial.html    styles.css
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title>SiteTitle</title>
5    <link rel="stylesheet" href="styles.css">
6  </head>
7  <body>
8
9      <h1>This is a headline</h1>
10     <h2>Second head</h2>
11     <h3>Third head</h3>
12     <h4>Fourth head</h4>
13     <h5>Fifth head</h5>
14     <h6>Sixth head</h6>
15
16     <p class="leadParagraph">Lorem ipsum dolor sit amet, consectetur adipisicing elit. Hic,
       vitae, pariatur, earum, debitis natus velit officiis at voluptatem modi suscipit dolorem
       laborum minima placeat repellendus quia totam officia! Rem, itaque!</p>
17
18     <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Dicta, eligendi, veritatis, quis
       magnam aliquam consequatur tempora nam sequi minima maxime nesciunt molestias quos nostrum
       beatae illo dolorem magni autem assumenda?Sunt, nam aliquid natus vero assumenda corporis
       similique ea blanditiis quos adipisci atque odit fugiat eum impedit nostrum facere at.
       Dolore, neque quidem excepturi voluptas totam laborum voluptatem qui incidunt.</p>
19
20     <img src="http://www.tripletsandus.com/disney/Wall-E/characters/Wall-E7.jpg" width="200px">
21
22 </body>
23 </html>
24
```

The class name can be anything at all. I've called this one leadParagraph so that it's descriptive of what I'm using it for. The only rule is they must only be one word long.

Back in our CSS file, we would establish one set of rules for all paragraphs to follow, then one set of rules for the leadParagraph that will deviate from the norm. We use the abbreviate .leadParagraph in our CSS to denote the class name. In practice, that looks like this:

```
tutorial.html    styles.css
1  p {
2      text-align: left;
3      font-size: 12px;
4      font-family: arial;
5      text-decoration: none;
6  }
7
8  p.leadParagraph {
9      font-weight: bold;
10 }
```

Collectively, this is how all web pages are built. HTML is the content and then CSS is used to style different portions of it. Experiment with different HTML tags and the CSS properties that apply to them. You can also experiment with the "width" and "float" properties to experiment with more complicated layouts on the page.

For more advanced web development, look at the next tutorial that introduces you to JavaScript, which will add personality to the content and style of your page.