# Introduction to CSS3 and HTML5: Lesson 1

Alan Simpson

# Lesson 1: Full-Screen View

This view has opened in a new window and will stretch to fit any screen size (large or small). It displays all of this lesson's components. To return to the normal classroom, please click the "close" button or manually close this window.

## Chapter 1

### Introduction

Welcome to *Introduction to CSS3 and HTML5*! My name is Alan Simpson, and I'll be your instructor. For many years, I worked in traditional publishing as a freelance author, and I've had over 100 computer articles published in dozens of languages throughout the world. My interest in developing websites started back when the World Wide Web first became available to the public. Since then, I've devoted most of my time to the online world rather than more traditional print publishing, and I've mastered many Web development tools and technologies, including HTML, XHTML, CSS, JavaScript, and ASP.Net.

I wanted to teach this course for many reasons. For one, I just enjoy creating websites. For another, of all the different Web development tools and technologies out there, CSS and HTML are the most important. There are a few reasons why. First, they're the foundation on which all other tools and technologies stand. Second, you can't really learn or use the other technologies until you've mastered CSS and HTML. And third, CSS and HTML, particularly CSS version 3 and HTML Version 5, which we'll discuss in this course, are cutting edge now and the wave of the future.


So let's talk about the goals of this course. They're pretty straightforward, really. When you complete this course, you should have a strong understanding of what CSS3 and HTML5 are, how to use them, and why you'd want to use them to create all of your Web pages. Most important, this course is about gaining HTML5 and CSS3 coding skills.

To help you achieve these goals, we'll build a working website together. In fact, if you have a place to post your site, you can use what you'll create to publish your own content. That includes pictures, videos, and sound files in your computer that you want to share. You'll also learn sufficient skills to add your own pages or even build entirely new websites from scratch.

Here in Lesson 1, I'd like to begin with some of the most basic, yet most important, tools, techniques, and terminology that you'll need to succeed in this course and in developing websites on your own. For those of you who recently completed the *Creating Web Pages* course, this will be a bit of a review. For those of you who managed to remember everything from that course, I apologize in advance for the overlap. But since few people remember everything they've learned, I think we'd be wise to go over the tools and techniques you really need to have down pat in order to do any kind of web development.

## What You Should Already Know

Before moving on, I just want to make it clear that this isn't a course for beginners. Even though we'll be reviewing *some* material from the more basic *Creating Web Pages* course, if you've never created a Web page in your life or you've never heard of HTML, CSS, or *code*, then you may be in over your head here right from the start. You'd be better off starting with the *Creating Web Pages* course, which is the true beginner's course for all things related to creating websites. In this course, I'll review some of the key topics from that course. But it'll only be a quick review, and we'll only go over the key topics.

Like most online courses, this course assumes you're already computer literate enough to understand terms like *folder*, *file*, *click*, and *code* and are skilled enough to use copy and paste, open folders, save files, and so forth. If you're new to computers and not clear on those things, a beginner's course in a classroom environment would likely be a better starting point.

## Why Learn HTML and CSS?

Many people wonder if it's even necessary to learn HTML nowadays. After all, there are countless programs and online services that allow you to create a site with point-and-click simplicity without even knowing that HTML exists. There's no shortage of programs for creating e-books and other electronic documents without knowing CSS or HTML. So why bother learning these languages when there are programs that will do all the hard work for you?

There are a couple answers to that question. Many of you taking this course may have already "been there, done that" with the fancy authoring systems. Maybe you're here now because you've discovered it's just not enough. Maybe you've learned that, for real creative prowess, you really need to know these languages yourself. Employers all seem to know this. And maybe you're here because your employer sent you here, or you're tired of having to answer "No" at job interviews when you're asked if you're fluent in CSS and HTML.

With CSS and HTML extending their reach into all forms of electronic publishing and application development, the need for people with knowledge to use them will continue to grow.

If you just want to create one website and be done with it, then sure, some kind of authoring system may be just the ticket to help you avoid the learning curve required to truly master CSS and HTML. But if you're looking to make a living in high tech and compete with the pros, gaining knowledge of CSS and HTML is a perfect starting point. So without any further ado, let's mosey over to Chapter 2 and get started.

# Chapter 2

## Getting Your Tools Together

In this chapter, we'll review some key techniques and terminology from *Creating Web Pages*, and I'll help you get geared up for creating some Web pages. Much of this lesson will be review for those of you who have successfully completed the *Creating Web Pages* course. But not everyone will remember everything from that course. And the information we'll discuss in this lesson is fundamental and critical to succeeding in this learning venture. So let's take the time to do it right and make sure everyone is starting off on the right foot.

Whether you create using paints, cameras, textures, or modern languages like HTML5 and CSS, you need tools. In the digital world, that typically involves using an editor for creating your files, and a folder to store them in. So let's start with those.

## Creating a Folder for Your Site

Every website you create should have its own folder where you'll store all of the files that make up the site. This folder is often referred to as the site's *root folder*, or the *site folder*. In this course, I'd like you to start with a new folder. I realize some of you may already have sites you want to work on, and that's fine. But I think you'd be better off keeping this new material separate while learning these new skills, if for no other reason than to have a place to try out new things.

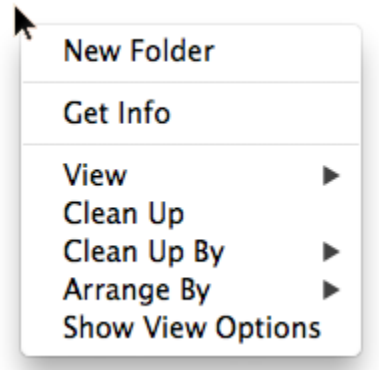
So let's get right into it by creating a new folder named *Intro HTML5* for this course. Feel free to put this folder on your desktop, in your Documents folders, or wherever is most convenient for you to get to when you need it. Just in case you've forgotten how to create a folder, I'll go through the steps below, one set of steps for Mac users, another for Windows users.

**Note**

The term *directory* is synonymous with the term *folder*. There's no difference between a folder and a directory.

## If You're Using a Mac

If you're using a Mac and want to put the new folder on your desktop, then get the mouse pointer to a neutral place on the desktop. If you want to put the new folder in your Documents folder, use **Finder** to open your Documents folder. Then get the mouse pointer to a neutral spot inside that folder. Press CTRL + Click to reveal a context menu (also called a shortcut menu) like the one below, and choose **New Folder**.



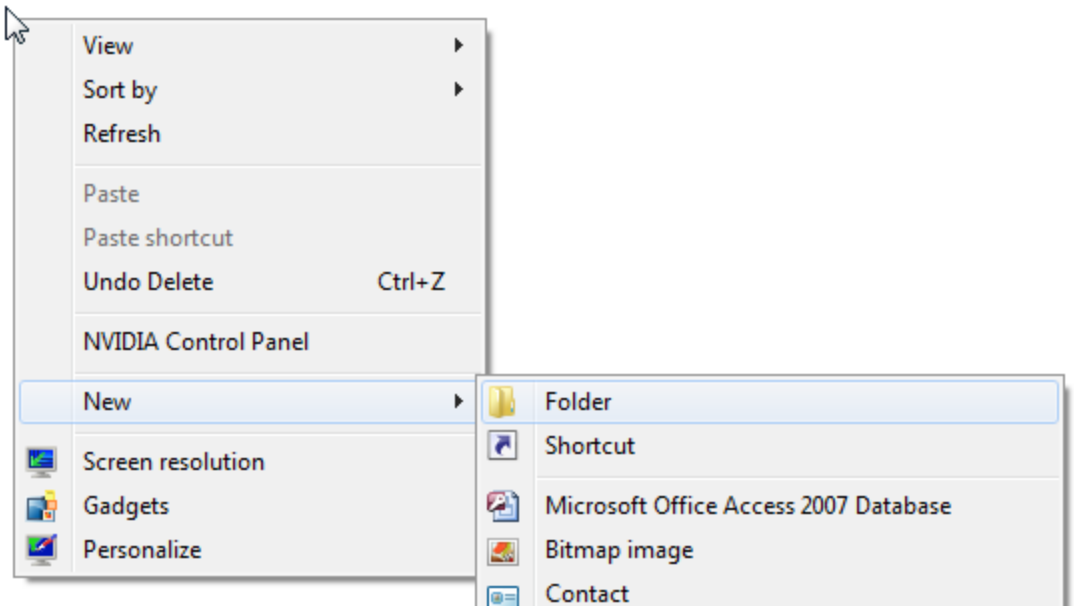

Mac OS context menu

Type the new folder name *Intro HTML5*, and press ENTER. You'll see a folder icon representing the folder. The exact appearance may vary depending on the version of Mac OS you're using and how you're viewing icons at the moment.


New Mac OS folder for course content

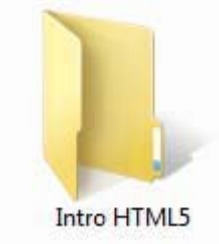
## If You're Using Windows

If you're using Windows and want to put the folder on the desktop, move your mouse pointer to an empty spot on the desktop. If you want to put the folder in your Documents or My Documents folder, open that folder and move your mouse pointer to a neutral area inside the folder. Right-click to reveal a shortcut menu, and choose **New Folder**.




Windows context menu

Type *Intro HTML5*, and press ENTER. You'll see a new folder icon named Intro HTML5. Its exact appearance depends on the version of Windows you're using and how you're viewing icons at the moment. But here's an example.




New Windows folder for course content

## Make Sure Filename Extensions Are Visible

Casual computer users can generally get by without knowing anything about filenames and filename extensions. But for any kind of website or software development, it's important that you can see the filename extensions. As you (hopefully) know, a filename extension is a set of letters at the end of a filename (preceded by a dot) that tells the operating system the file type. For example, in the filename *index.htm*, the *htm* is the filename extension that identifies the file as a Web page.

Many operating systems hide filename extensions by default because they have little or no meaning to beginning and casual computer users. But you'll need them to be visible for this course and any other development work you do. So let's take a moment now to make sure filename extensions are visible on your system:

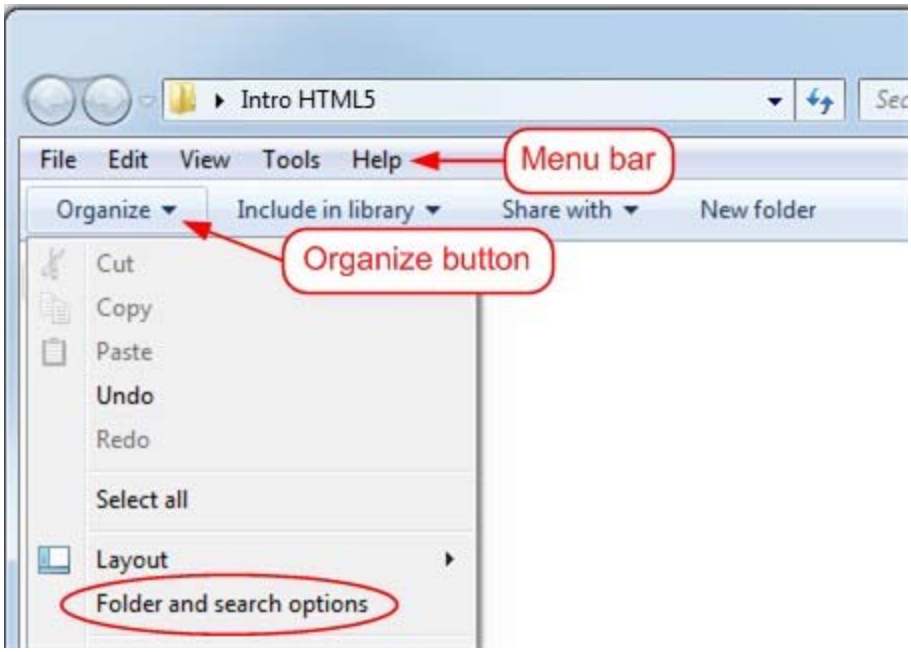
## If You're Using a Mac

If you're using a Mac, click the Finder icon in the dock, and choose **Finder** > **Preferences** from the menu. In the Finder Preferences dialog box, click **Advanced**. Then make sure you have **Show all filename extensions** selected (checked).


Show all filename extensions in Mac OS

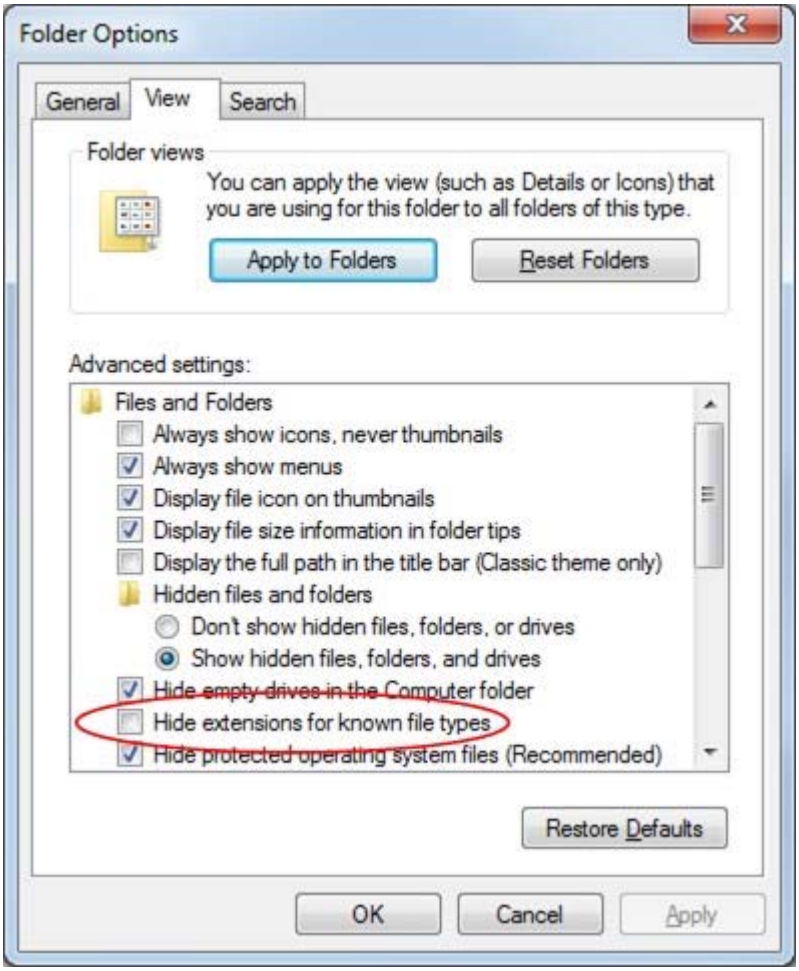
## If You're Using Windows

If you're using Windows, open any folder on your system. If you're using Windows 7, click **Organize** and choose **Folder and search options**. In XP, choose **Tools** > **Folder Options** from the menu bar near the top of the folder window. (If the menu bar is hidden, tap the ALT key on your keyboard.)




Windows folder window components

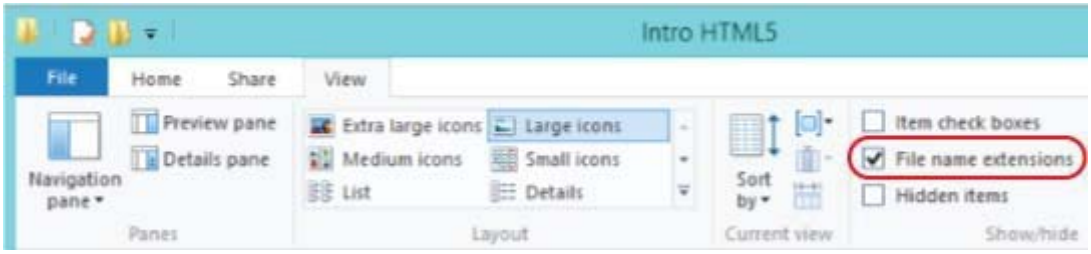
In the Folder Options dialog box, click the **View** tab and make sure that *Hide extensions for known file types* isn't checked (because you don't want to hide filename extensions).




Prevent hiding of filename extensions

Click **OK** to save the current setting and close the dialog box.

If you're using Windows 8, open your Intro HTML5 folder and click the **View** tab. Then just select (check) the **File name extensions** option in the Show/hide section of the Ribbon.




Showing file name extensions in Windows 8

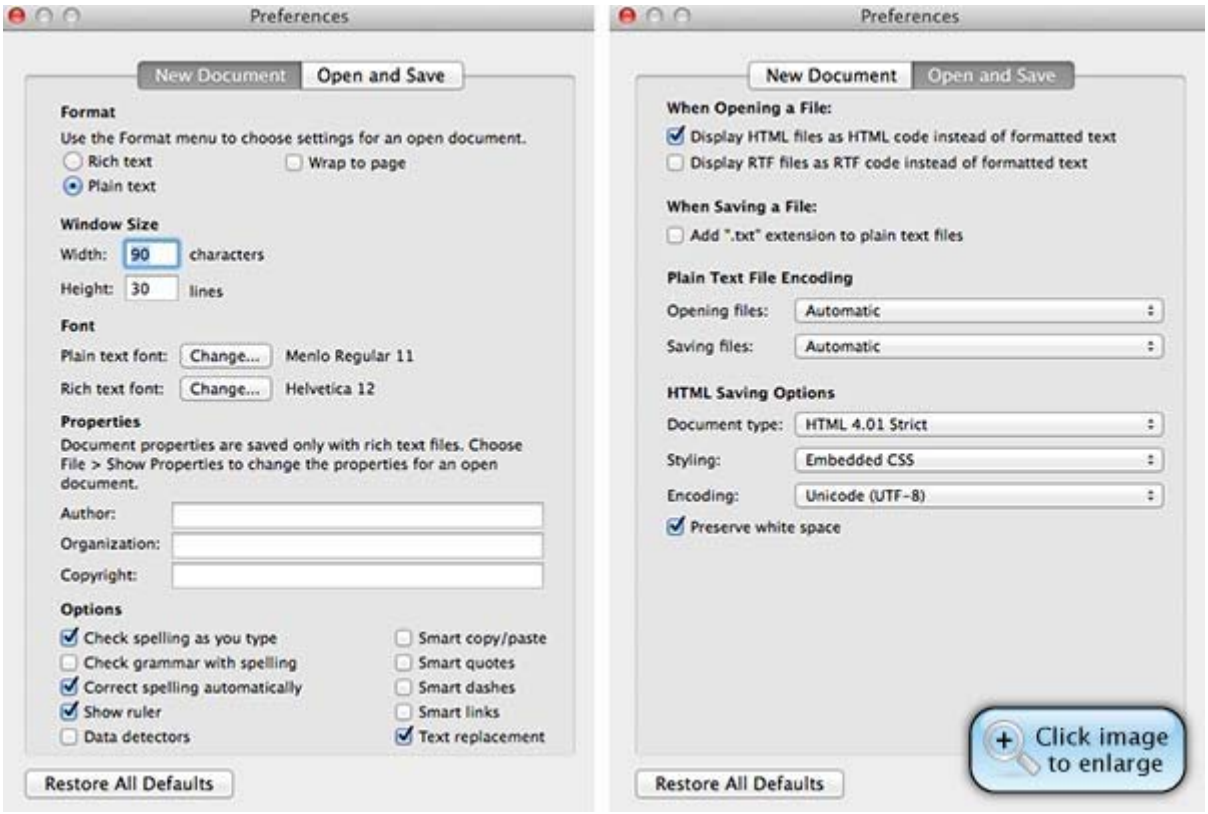
## Choose Your Editor

In this course, I won't require that you use a specific program or editor to create your pages. You can use whatever you've used in the past, assuming you know what you're doing in that program. For example, if you have and use Dreamweaver, Expression Web, or some other fancy editor, and you're used to writing HTML in its Code View or Source View, then you're welcome to continue doing so. Just be aware that this course won't teach you how to use those programs.

If you want to use a simple text editor like Notepad in Windows or TextEdit in Mac OS, that's fine too. Notepad comes free with every version of Windows, and it doesn't require any special configuration. But if you plan to use TextEdit on a Mac, you need to make sure your settings are configured to text files and HTML. To do that, open TextEdit and choose **TextEdit** > **Preferences** from the menu.

Under New Document, make sure you choose **Plain Text** under Format. Under Open and Save choose either **Display HTML files as HTML code instead of formatted text** or **Ignore rich text commands in HTML files**, whichever option is available in your TextEdit version. You might also want to set other preferences as shown in the example below.

Show Text Equivalent




Mac TextEdit preferences

That's all you need to do to get your system ready to start creating HTML documents. You'll get to do some more hands-on activities later in the assignment for this lesson. For now, let's head over to Chapter 3 and make sure we're all speaking the same language in terms of understanding what HTML is all about.

# Chapter 3

## About HTML5

In this chapter, we'll discuss what HTML5 is all about. Hopefully this will be a review or refresher for most of you. But the concepts and terminology are very important, so it all bears repeating to make sure that we're all speaking the same language as we proceed through the course.

*HTML* stands for *Hypertext Markup Language*. Its origins go back to 1991 when developers designed it as the language for authoring Web pages for the Internet's World Wide Web.

Like all software, HTML has evolved over the years to keep up with advances in user agents. The phrase *user agents* refers collectively to all the different gadgets and programs that can access Web pages. That includes Web browsers, tablets, smartphones, assistive devices, and search engine indexing spiders. Here's a look at major versions of HTML released over the years:

**HTML Versions**

| Version | Year |
|---|---|
| HTML 1 | 1991 |
| HTML 2 | 1995 |
| HTML 3 | 1996 |
| HTML 4 | 1997 |
| XHTML 1 | 2000 |
| XHTML 1.1 | 2001 |
| HTML5 | Candidate |

You'll notice the name changed from HTML to XHTML at the turn of the century, and the numbering went back to 1. However, *XHTML* is still *HTML*. They just added the *X* because that version borrowed some syntax conventions from the highly successful XML language. XML is a language for storing and transmitting raw data rather than published pages. You don't need to know, learn, or understand XML to create websites, so you don't have to even give that a second thought. Suffice it to say *XHTML* is just *HTML* with an *X* in front and some minor differences in how you type a few tags.

Shortly after the turn of the century, another Internet group called *WHATWG* (*Web Hypertext Application Technology Working Group*) began working on a version of HTML that would extend the language to include features you'd commonly find in application programs (apps), which tend to be more interactive than Web pages. All the different names and acronyms were becoming too confusing for most developers. So the World Wide Web Consortium (W3C), the standards body that creates all the Web languages, decided it was time to merge the WHATWG group with the XHTML group and go back to a single, standard name and numbering system. And from that decision, the current version of HTML, dubbed *HTML5*, was born. HTML5 is simply the version of HTML/XHTML that everyone will be using for the next decade or so. You don't need to worry about choosing a specific version for a specific project. It doesn't work that way. Each new version is a refinement of (and a replacement for) the previous version.


The World Wide Web Consortium is made up of members from high-tech giants like IBM, Microsoft, Apple, Google, and others. HTML5 *is* whatever the W3C says it is. There are no exceptions—no "different flavors" for different browsers, editors, or operating systems. But the W3C doesn't simply create a language in a vacuum and then force it on the rest of the world. Creating a new version of any language is a process that takes many years and many people outside the W3C.

Part of the process of defining a new version of a language involves putting it out to the public as a *candidate* recommendation or *proposed* recommendation when the W3C is reasonably confident that the spec is good and will work for everyone. But it's not an official or final recommendation. It's more of a "let's let everyone take it for a spin and kick the tires" sort of trial run that could last several years. There's still a chance that they might make some changes, based on the feedback from folks who are using the product during that trial period.

Right now, HTML5 is such a candidate spec, not a final spec. But at this stage, any changes are likely to be so minor as to be inconsequential to most folks. So development with HTML5 is well under way in the real world. And if you're going to learn HTML, Version 5 is definitely the way to go. And that's why you'll be learning HTML5 in this course.

## HTML Tags

HTML is a language for creating Web pages and other electronic documents (including some kinds of applications). People who create apps, Web pages, and e-books often refer to what they do as *writing code*. The term *code* is a general term for writing any kind of computer software in any computer language. You'll be seeing the term *code* a lot in this course, because you'll be learning how to write CSS and HTML code throughout this course.

Like all of its predecessors, HTML5 consists primarily of *tags*. Each tag tends to define an element type, such as a heading, paragraph, list item, table, graphic image, or something else that shows in the document. Every tag starts with < and ends with > with one or more lowercase characters in between—usually an abbreviation for the type of element the tag creates.

Many (though not all) tags come in pairs where an *opening tag* marks the start of the element, and a *closing tag* marks the end of the element. The closing tag is the same as the opening tag, except that it contains a slash (/) right after the < symbol.

For example, the  tag marks the start of a paragraph, and  marks the end of a paragraph using the syntax below.

```
<p>...</p>
```

Above, the ellipsis (...) is just a placeholder. In real life, you'll put the words and sentences that make up the paragraph content between the tags. Those words and sentences, of course, can be any words and sentences you like. There's no limit to how much stuff (words, sentences) you can put between any opening and closing tag pair.

```
<p>This is a sample paragraph. A paragraph can contain any number of words and
sentences.</p>
```

Not all tags come in pairs, though. Some tags stand alone. We commonly refer to such tags as *empty tags*. For example, we use the br tag to indicate a line break. In XHTML, all empty tags end with /> usually preceded by a space. For example, here's the proper way to type a br tag in XHTML:

```
<br />
```

Older versions of HTML didn't require the space or slash. In traditional HTML, the br tag is written as:

```
<br>
```

One of the goals of HTML5 is to do away with some of the unnecessary confusion and complication that comes with having too many brands on versions of the HTML language out there. Another is to make sure that bringing forth this new language doesn't break any of the pages that are already out there. So the plan right now is for HTML5 to accept and support either method of writing empty tags. So the space and slash at the end of an empty tag are optional.

**Tip**

XHTML has been around for over 10 years now, and many developers are accustomed to writing code that way. So you'll see plenty of people using a / in empty tags, even though it's optional, and to the beginner might seem like extra unnecessary effort.

## HTML Attributes

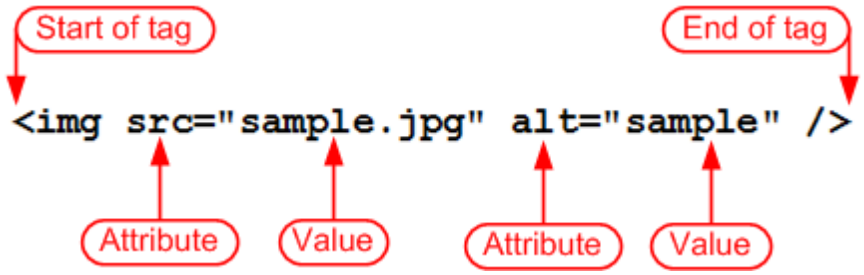
In addition to tags, the HTML language consists of *attributes*. You can use attributes inside some tags to provide additional information about the element that the tag is creating. All attributes follow the syntax:

*name*="*value*"

Where *name* is an attribute name and *value* is some number or other information that's assigned to that attribute. You should always precede each attribute with a space, and you should always enclose the value in quotation marks. For example, here's an img (image) tag with two attributes, a src (source) attribute and an alt (alternate text) attribute. Don't worry about the specifics of the tag and attribute right

now, though. All that's important now is syntax. Notice how there's a space before each attribute name, and the value assigned to each attribute is in quotation marks.

Show Text Equivalent




Sample tag with two attributes

Since img is an empty tag, we can also write it without the space and slash at the end in HTML5. In other words, the tag below is also a valid way to write the one above. It doesn't really matter which you use, since both syntaxes are equal and acceptable in HTML5, and the picture that the tag shows will look exactly the same in the Web page either way.

```
<img src="sample.jpg" alt="sample">
```

Tags that come in pairs can also use attributes. The syntax is the same, and you don't have to do anything to "close" the attributes in the closing tag. For example, the <a> tag below contains two attributes with values. The closing </a> tag is fine the way it is. You don't need to "close" the attributes or mention them at all in the closing tags.

```
<a href="sample.htm" title="sample">
...
</a>
```

In all attributes, spaces around the equal sign are allowed but not required. If you find the code a little easier to read with spaces around the equal signs, that's fine. The code will work with or without the spaces.

```
<a href = "sample.htm" title = "sample">
```

There are about 100 tags and about 100 attributes in HTML5. That may seem like a lot, but there's no need to learn them all at once. You start with the most important and most commonly used ones and work your way up from there. You won't have to memorize them all either, because it's easy to find lists online or printed cheat sheets, as you'll learn later. For now, let's start with a few important tags—the tags that are required in all Web pages.

**Note**

There's no such thing as a *command* in any version of HTML. However, people occasionally misuse that term as a synonym for *tag* or *attribute*.

## Required Tags

Every HTML5 document must contain some required tags that the various user agents use to identify components of the document. Here are those required tags . . . you must use them in this order.

```
<!DOCTYPE html>
<html>
 <head>
 <title></title>
 </head>
 <body>
 </body>
</html>
```

There are no required attributes, so that's why you don't see any attributes in this list.

Let's quickly summarize the purpose of each required tag:

**<!DOCTYPE html>:** This tag indicates that the page is written in HTML5. It may seem odd that there's no *5* in there. But doctype tags for other versions of HTML are much different and far more complex. So that simple DOCTYPE tag is sufficient to identify an HTML5 document.

**<html>...</html>:** The <html> tag marks the start of the HTML document, and </html> marks the end of the document.

**<head>...</head>:** The <head> tag marks the beginning of the head section (or *metadata* section) of the document. This section contains tags and information about the document rather than content that actually shows on the page. The </head> section marks the end of that section.

**<title>...</title>:** This is the document title that appears in the Web browser's title bar or tabs and in search engine results. You must place the actual text of the title between the <title> and </title> tags and keep it brief (six words or fewer).

**<body>...</body>:** The body section contains the actual content of the page that's visible in the Web browser. The <body> tag marks the start of that section, and </body> marks the end of that section.

Later, in the assignment for this lesson, you'll have a chance to go hands-on and create a page that contains all the required tags. You'll learn to use that page as a template for every new page you create so you don't have to memorize or retype those tags every time you create a new Web page. But first, let's head over to Chapter 4 and talk about CSS, which is the styling component of modern Web page design.

Chapter 4

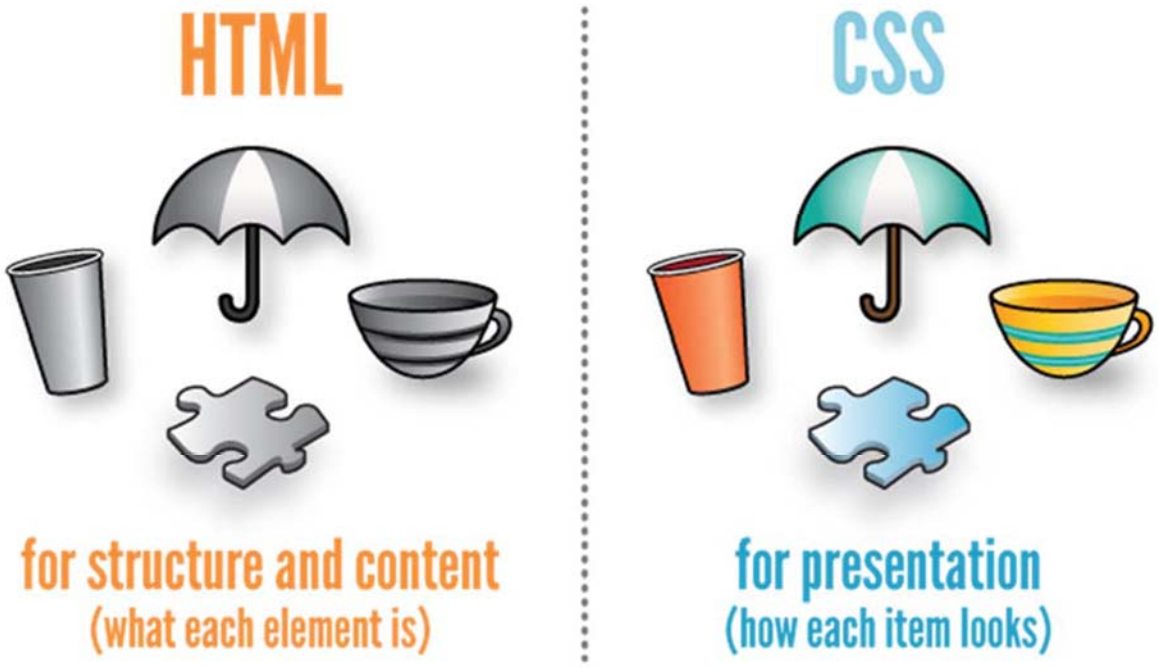
## What Is CSS?

All modern forms of publishing (not just Web development) follow a principle of *separation of structure and presentation* (or *separation of content and presentation*). In this context, *content* and *structure* have to do with the actual words and design elements on the page. For example, these words you're reading right now are *content*. The structure has to do with the various kinds of elements you see on this page, such as headings, paragraphs, lists, and pictures. The content and structure are primarily handled by HTML.

The term *presentation* has more to do with design and aesthetics—things like colors, background colors, fonts, and other things that developers apply to the structure and content to give it its style—its look-and-feel. In modern Web design and electronic publishing, we use CSS (*Cascading Style Sheets*), a separate language, to define those things.


Obviously, when you look at a page like this one, or any other published page in any website, magazine, or book, the content, structure, and design all look right on the page. They're not visually separate for the person viewing the page. But they are separate for the person *creating* the page because there's a different language for each. To the *user* (or *reader*, or *consumer)*, the person who's just looking at the final product, there is no separation. But to you as the *developer* (or *creator*, or *designer*, or whatever you want to call yourself) there is separation in terms of *how* you create; that's why there are two separate languages. In simplest terms:

Show Text Equivalent




To the novice, this might seem frightfully vague, and maybe it even seems like a mistake that overcomplicates things. Maybe if you just wait long enough, the people who define these things will come to their senses and come up with some simple single language that does it all. Well, if you're thinking that, don't hold your breath. As it turns out, just the opposite is true. At first, and for several years, there was only HTML (no CSS), and it proved to be a nightmare for large-scale Web development. CSS came later to solve the many problems of trying to do everything with just one language. Nobody will ever want to un-solve the many problems that CSS solves.

The idea of having two languages was nothing new at the time. People in print publishing, desktop publishing, and word processing have been using *style sheets* successfully for years, which is a method for separating structure from presentation. The invention of CSS was just a matter of taking what was already successful in other forms of publishing and applying it to Web development. And its great success in Web development is what's driving its spread into other forms of electronic publishing and application development.

Like HTML, CSS is a language that's defined by the World Wide Web Consortium. And like all languages and software products, CSS has also evolved through multiple versions over the years:

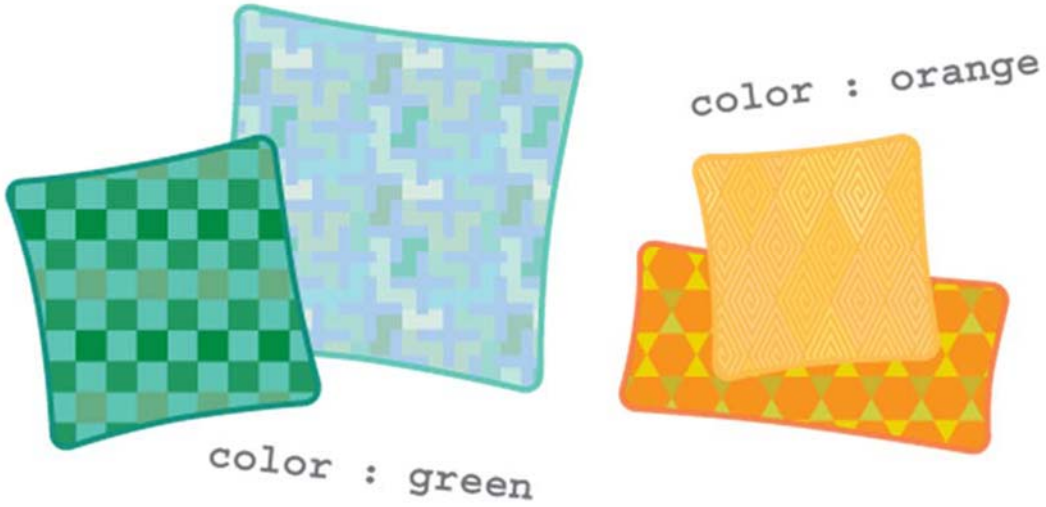
**CSS Versions**

| Version | Year |
|---|---|
| CSS1 | 1996 |
| CSS2 | 1998 |
| CSS2.1 | 2011 |
| CSS3 | Candidate |

The current recommended CSS version is CSS2.1. CSS3, like HTML5, is newer and hasn't yet reached that full recommended status. But it's far enough along now that few (if any) changes are likely to take place. Web browsers and other user agents are well on their way to supporting its features. Many developers are currently using it for websites and other types of documents, and more will be using it in the future. Now is the time to start learning CSS3, and so that's the version we'll be discussing in this course.

## Properties and Values

While HTML involves using tags and attributes to define what each element *is*, CSS involves using *properties* and *values* to define how each element *looks*. The term *property* comes from the way we use the term in the everyday world. For example, think of any object. Say, a pillow. There are lots of pillows, but what makes one pillow different from other pillows? Its unique properties. For example, every pillow has a height, width, and color. Things like *height*, *width*, and *color* are properties in the real world, and they're also properties in the CSS language. The various properties that make one particular pillow different from others are sometimes referred to as its *style*. We use properties in CSS to define the style of things on the Web pages we create.




You can think of the property as the *what-to-style* part (height, width, color). Think of the value as the *how-to-style* part. For example, *color:red* means "make the color (property) red (value)".

The CSS syntax requires that the property name always comes first, followed by a colon, followed by the value you're assigning to the property, like this:

*property*:*value*

Again, using our simple color example, *color:* is an actual CSS property that refers to the color of text. You can apply a color value to it. For example, *red* is a valid color name value in CSS. So to make the text red, you type the property:value pair like this:

```
color:red
```

You can put a space before the colon, after the colon, or both. It doesn't really matter because spaces there are optional and the code works the same with or without spaces. Sometimes you may see a property:value pair typed like this:

```
color: red
```

Or even like this:

```
color : red
```

Nothing to worry about, they're both fine and work exactly the same.

You can apply multiple *property*:*value* pairs to the things you're styling. But when you use multiple property:value pairs, you must use a semicolon (;) to separate each pair, like this:

*property*:*value*;

*property*:*value*;

*property*:*value*;

The semicolon after the last property:value pair is optional. There's no harm putting it there, and there's no harm leaving it off. When you're looking at other peoples' code, you'll often see that the last property:value pair has a semicolon at the end. The main reason developers do that is so that if later they decide to add another property:value pair to the list, they won't forget to type the necessary semicolon at the end of the property:value pair above the new one they're adding.

The scope of a *property*:*value*, or how far-reaching the style is, can be almost anything from one tiny letter or character on one page in the site to every character on every page in the site. There are many ways to control the scope of a CSS style, as you'll learn throughout this course. But suffice it to say, CSS gives you *very* precise control over styling, which is one of the reasons for its great success as a language.

In CSS3, you have about 250 properties to choose from and countless values (including over 16 million colors). But don't worry about all of that right now, and don't let it intimidate you. You'll learn about the important property:value pairs as we go through this course, and I'll show you easy ways to find whatever you need to know, whenever you need to know it. For now, it's important to just stay focused on the important fundamentals, including the terminology and syntax we're discussing right now.

Let's head over to Chapter 5 and wrap up what you've learned.

# Chapter 5

## Conclusion

Our goal for today's lesson was mostly to lay the groundwork for lessons to come. Much of it is a review of material you may have learned from the *Creating Web Pages* course. So if you've completed that

course recently and happened to remember almost everything, I apologize for the overlap. But since few people remember everything in the course, I thought it was important to review the fundamentals to make sure everyone is starting off with sufficient knowledge to move forward.

We started by getting your system set up to start creating a website. Then in Chapter 3, we reviewed what HTML is about and the key concepts and terminology you'll use when learning and discussing that language. Then in Chapter 4, we discussed CSS, which is the second major component of modern Web development and a major component of this course.

In the Assignment for this lesson, you'll get some more hands-on activity by creating a basic HTML5 template that you can use for starting any new Web page you ever create. In the next lesson, we'll get deeper into CSS, and you'll find out how you can use it to start defining the look and feel of an entire website, not just an individual page. See you there!

**Next Steps**



Okay, you've finished your first lesson. Now what do you do?

You'll want to take the following steps, in any particular order you like:

- **Take the quiz**. Reinforce what you learned in the lesson by testing yourself with a short five-question quiz. You can access the quiz for each lesson by clicking the **Quizzes** link.

- **Do the assignment.** Want some hands-on practice applying what you've just learned? Then roll up your sleeves and dig into the assignment! Just click the **Assignments** link to get to each lesson's assignment.

- **Check out the FAQs**. Since learning something new usually raises questions, every lesson in this course comes with an FAQs section. To get to the FAQs, click the **Resources** link, and then click **FAQs**.

- **Drop by the Discussion Area**. Come talk with me and your fellow students in the Discussion Area! Ask questions about anything that came up in the lesson, and share your insights with everyone. This is where we'll create a learning community.

- **View the index**. If you want to find a topic but can't quite remember where it was, then the index is the place to go. You'll find it by clicking the **Resources** link, and then clicking **Course Index**.

- **Browse resources for further learning**. I've included a list of recommendations for books so you can continue learning more about this topic long after our time together ends. You'll find these by clicking the **Resources** link.

## Supplementary Material

Please click here for current

Supplementary Material

links.

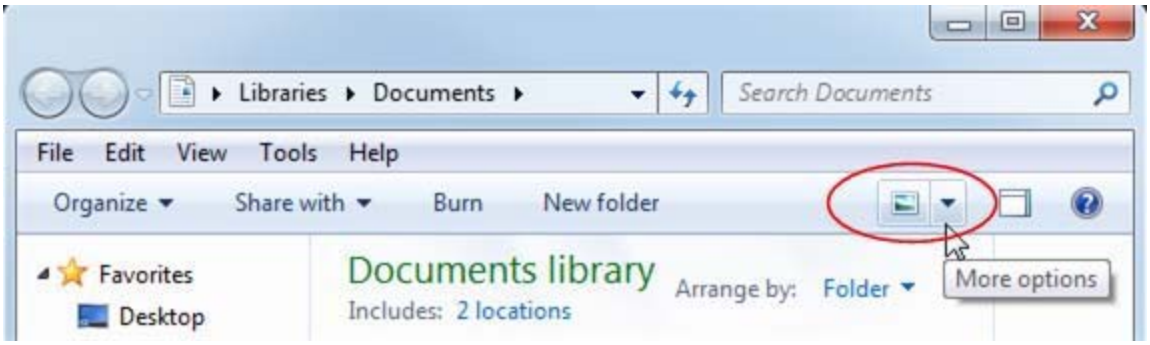
## FAQs

### Q. I created the Intro HTML5 folder as per the instructions, but my icon looks different. What gives?

**A.** Most operating systems let you choose how you want to view icons, so it's just a matter of learning how to use your operating system. Unfortunately, we don't teach all the different operating systems in this course because if we even tried, that wouldn't leave much room for discussing HTML or CSS. This course assumes you're already fairly computer literate and capable of using your operating system fairly well. But we try to help where we can. So if you're using a Mac and you want to change the appearance of icons in a folder, look for the various View icons and click each one until you find a view you like.

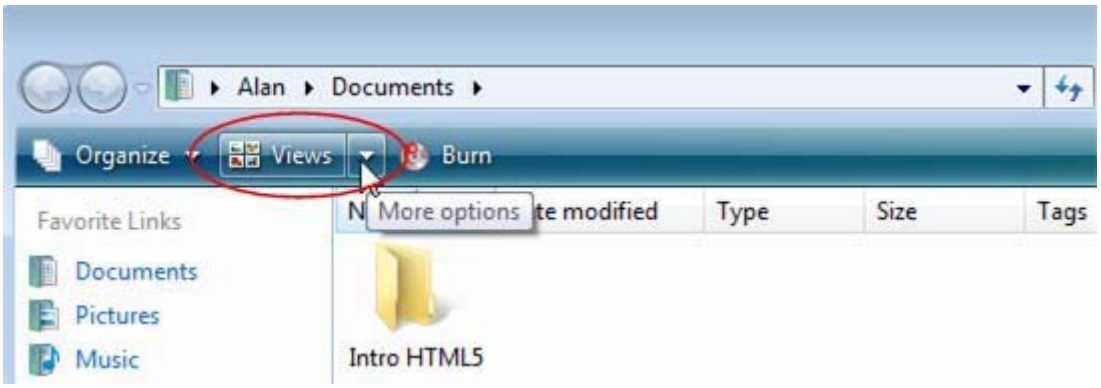

View options in Mac OS Lion

If you're using Windows 7, you should see a More Views button near the top-right side of the folder window (when you're in a folder window). The icon depends on the view you're using at the moment. But when you point to the correct button, the tooltip at the mouse pointer will read *More Options*. You can click that button and try different views until you find one you like.



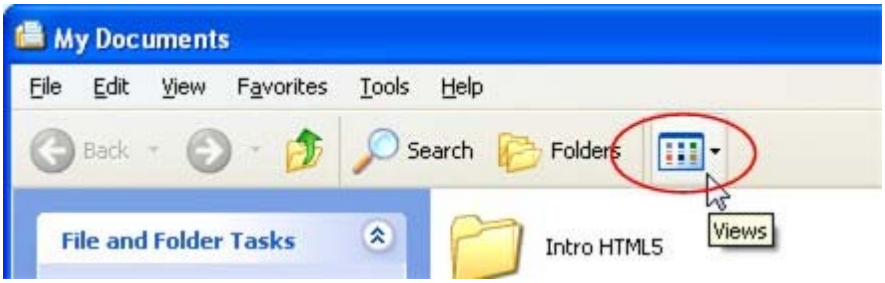

View options in Windows 7

If you're in a folder window in Windows Vista and want to change the appearance of icons in that folder, use the Views button.




View options in Windows Vista

If you're using Windows XP, you can use the Views button on the Standard Buttons toolbar, circled in red in the image below, to change the view. If you don't see that toolbar, click **View** on the menu bar, and choose Toolbars and then Standard Buttons to make that toolbar visible.




View options in Windows XP

## Assignment

### Creating a Page Template

Since every HTML5 page must contain the required tags, you can save yourself some time and effort by creating a file that already contains all those tags. Then you can use the file as a kind of template for each new page you create, saving you from having to retype those same tags from scratch every time you

create a new document. It also avoids the possible errors you might make when trying to retype those tags from scratch.

In the sections below, I'll go through the steps required to create this template in Mac OS and Windows 7. As I mentioned earlier, I'm going to use TextEdit and Notepad because all Mac users have TextEdit and all Windows users have Notepad. But again, if you already know how to create Web pages using some other program or editor, you can use that instead. Here are the required HTML5 tags you'll need to type (or copy and paste) into the page.

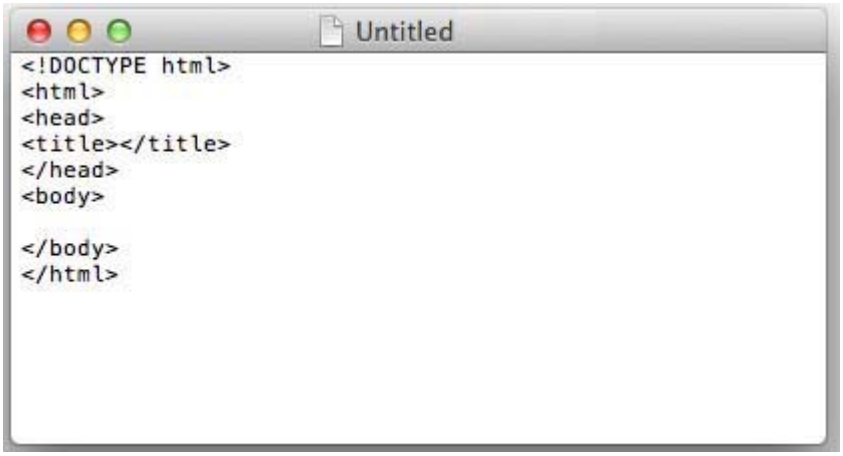
```
<!DOCTYPE html>
<html>
<head>
<title></title>
</head>
<body>
</body>
</html>
```

## If You're Using a Mac

If you're using a Mac, follow these steps to create your HTML5 template page:

1. Open TextEdit (or your preferred text editor)

2. Type (or copy and paste) all the required HTML5 tags shown above into the new, empty TextEdit. There's no margin for error when typing code, so make sure you get it right.



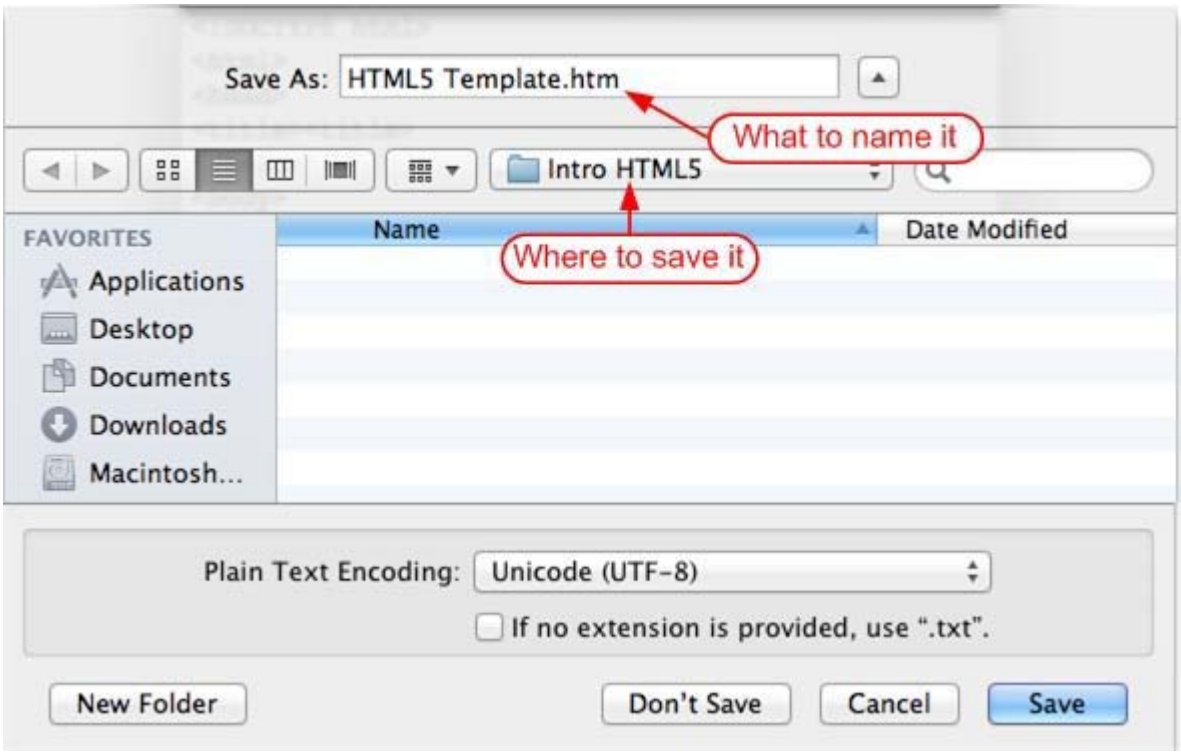

Required HTML5 tags in TextEdit

3. Click the red Close button in the upper-left corner of the TextEdit window.

4. In the Save As box, type *HTML5 Template.htm* as the filename.

5. Choose **Intro HTML5** as the folder in which you want to save the file. If you can't choose that folder name directly from the drop-down and you don't see the navigation tools shown in the

figure, click the **Expand** button to the right of Save As, double-click the parent folder name (Desktop or Documents), and then double-click **Intro HTML5** to ensure that the file saves in that folder.

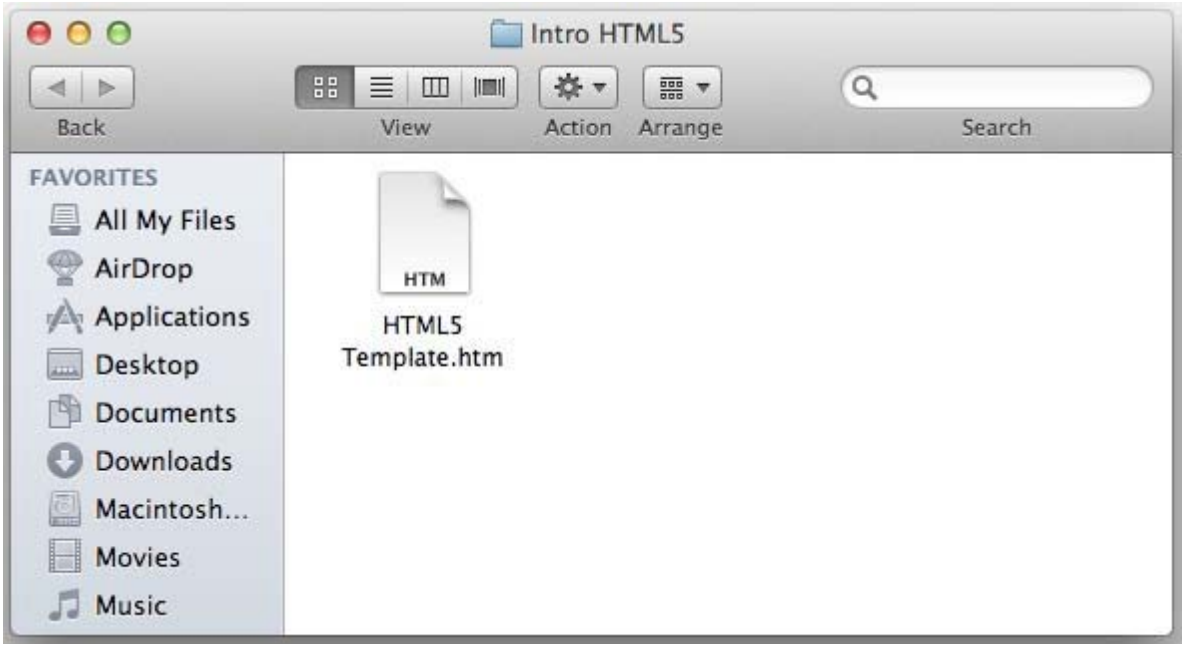



Save page as HTML5 Template.htm in your Intro HTML5 folder

6. Choose **Unicode (UTF-8)** as the Encoding. (For the most part, this just indicates that it's typed using the English alphabet as opposed to Japanese, Chinese, or some other language that uses a different alphabet.)

7. Click **Save**.

8. The document saves in your Intro HTML5 folder, and TextEdit closes. To verify that you saved the file in the right location, open your Intro HTML5 folder (get to the desktop, or open your Documents folder if that's where you put your Intro HTML5 folder, double-click the folder icon to open the folder). You should see an icon representing that file with the .htm extension. If your icon looks radically different, you may just need to click a different view option to change the appearance of the icons.



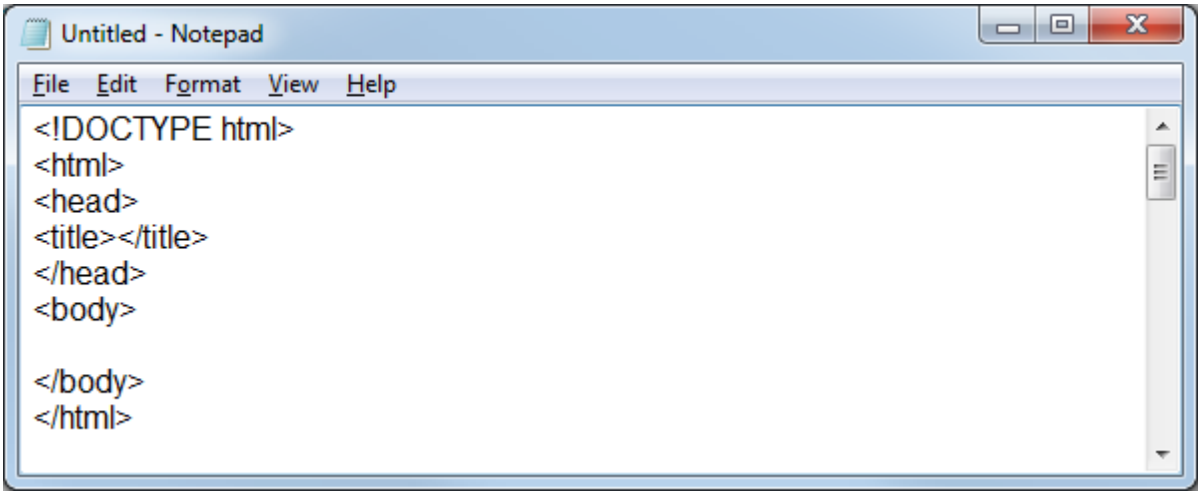

HTML5 Template.htm in the Intro HTML5 folder

## If You're Using Windows

If you're using Windows, you can use these next steps to create and save your document. I'll use Windows 7 as a working example. But other versions of Windows are similar enough that you should be able to follow along, even though what you see on your screen may not look exactly the same as what I'm showing here.

1. Open Notepad (or your preferred editor).

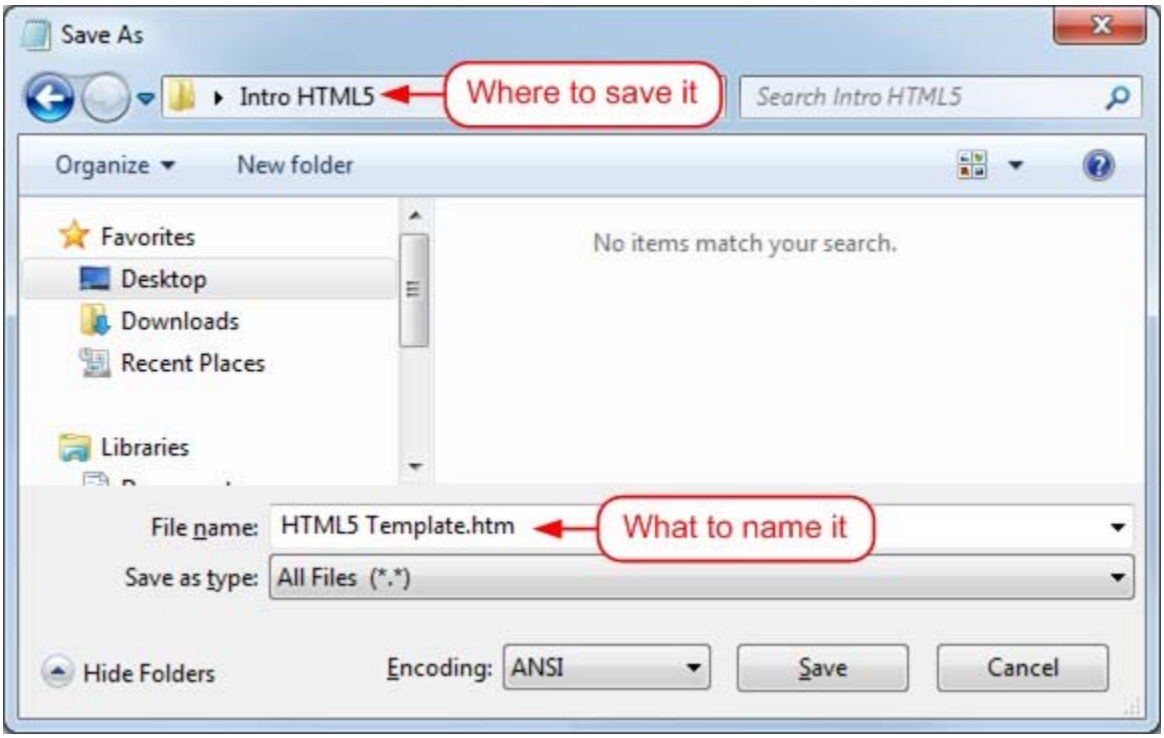
2. Type (or copy and paste) all of the required HTML 5 tags into the document, like this:




Required HTML5 tags in Notepad

3. Click the **Close** (X) button in the upper-right corner of editor's program window, and choose **Yes** or **Save** when asked about saving your changes.

4. In the Save As dialog box, click **Intro HTML5** to specify where you want to save the file. If your Save As dialog box window is much smaller than the one shown here and you can't choose Intro HTML5 from the available files, click **Browse Folders**, and then double-click **Desktop** (if you put Intro HTML5 on your desktop) or **Documents** (if you put it in your Documents folder). Then double-click the **Intro HTML5**folder to choose it as the folder where you want to save the file.

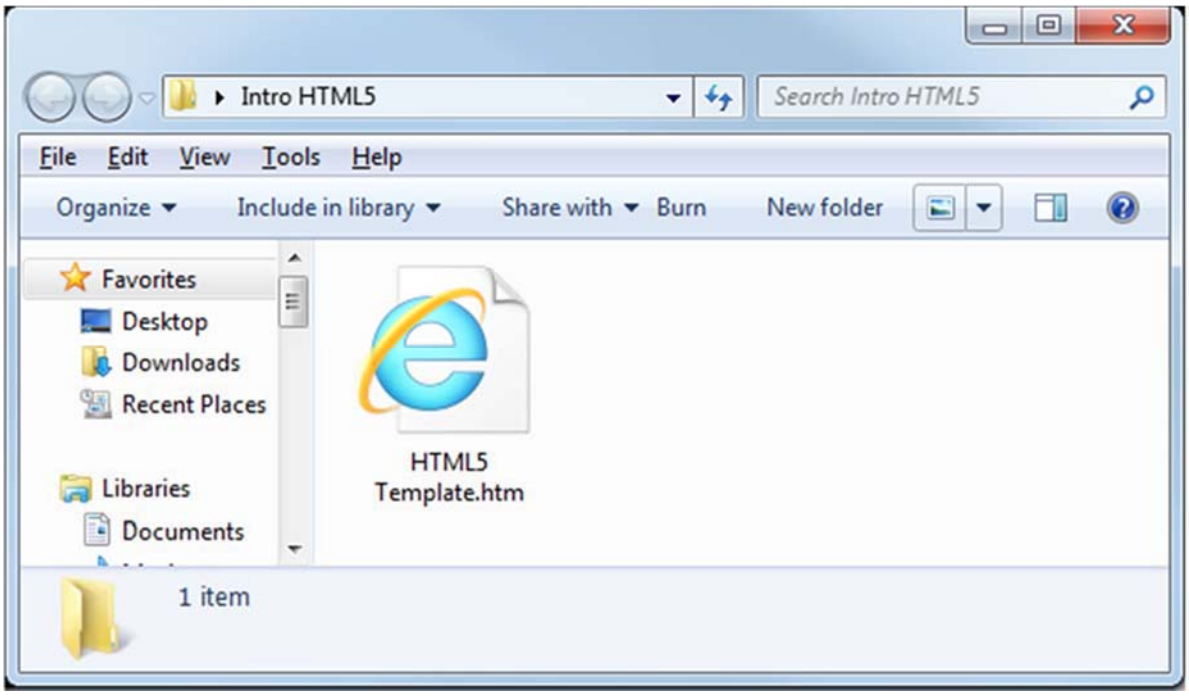



Save as HTML5 Template.htm in your Intro HTML5 folder

5. For the File name, make sure you type *HTML5 Template.htm*. For the Save As Type, you can choose the default **Text** document, or switch to **All Files**. It shouldn't really matter as long as you type *.htm* at the end of the filename. The default Encoding of ANSI is fine, as that's common for English, especially when writing HTML or CSS code.

6. Click **Save** when you're ready. Notepad closes, and the new file saves to the folder you specified with the filename you specified.

7. To verify that you did it correctly, get to wherever you put the Intro HTML5 folder (your Desktop or Documents folder). Then double-click the Intro HTML 5 folder to open it. Once that folder is open, you should see an icon for the HTML5 Template.htm file. Your icon may look different depending on your operating system, default browser, and current view settings. But you should see an icon with the filename HTML5 template in that folder.




Save as HTML5 Template.htm in your Intro HTML5 folder

In this example, the icon is a blue lowercase e because that's the icon for the Internet Explorer browser, and I have that set as the default browser. Your icon may look more like the icon for your own default browser. In terms of the size and general appearance of the icon, that's a view setting. If you don't know how to work those but you'd like a quick lesson, see the Lesson 1 FAQs (click **Resources** near the top or bottom of any page in the course, then click **Frequently Asked Questions**).

This concludes the assignment for Lesson 1. Please don't try to complete the Lesson 2 assignment until you complete Lesson 2. To see the dates when lessons are released, click the **Lessons** link near the top or bottom of this page.