Course Overview – Introduction

Welcome to an introduction to HTML Cascading Style Sheets (CSS), an optional no credit course that will cover the basics and prepare you for future classes. The goal of this course is to introduce the core concepts of CSS so you can start applying CSS to your (X)HTML pages. CSS gives you the power to style and lay out web sites so they are more usable and compact, better looking, better structured and easier to maintain.

In this course, you will learn about the basics of CSS in order to format your pages for today's modern web browsers. Course assignments are provided to allow you the opportunity to practice the concepts covered. There is no textbook required, however you may want to look into purchasing the next book for your next HTML or web design course. This will be good reference material for this course.

Here are the major areas you will cover:

- **Week 1:** What is CSS and Why Use It
- **Week 2:** Style Sheets
- **Week 3:** Laying out Pages with CSS

Week 1 Overview – Orientation

In your first week, you will learn how to write your first style sheets. You start with a definition of CSS and why it is fundamental to any professional web site.

Next, you will learn proper CSS syntax structure.

Finally, you will learn about the various types of CSS selectors and when it is best to use each (as well as when it is not).

What is CSS and Why Is It Important?

What is CSS?

HTML is the predominant markup language for describing Web pages. HTML stands for Hyper Text Markup Language, and is a language, which makes it possible to present information on the Internet. HTML tags were originally designed to define the content of a document. The layout of the document was supposed to be taken care of by the browser, without using any formatting tags.

However a the early Web browsers continued to add new HTML tags and attributes (like the <font> tag and the color attribute), to format page content, it became more and more difficult to create and maintain Web sites where the content of HTML documents was clearly separated from the document's presentation layout. To solve this problem, the World Wide Web Consortium (W3C)—the non-profit, standard setting consortium, responsible for standardizing HTML—added Cascading Style Sheets to the HTML 4.0 specification.

Invented in 1997 as a supplement the existing HTML standard, Cascading Style Sheets (CSS) have become the standard professional method for laying out Web pages and formatting content. Many HTML tags that were the only way to define the look and feel of a Web page, such as the <font> tag, e have been replaced by CSS.

All major browsers support CSS, and are the standard method of formatting

HTML and CSS. Text is un-arguably the most important element of any Web page. Modern Web site design requires the use of CSS for formatting text, specifying fonts, applying styles to text, setting line heights and letter spacing, styling drops caps and improving text legibility .

HTML documents. CSS is a Web designer's dream tool. It enables designers to be completely consistent with the look and feel of their Web pages and provide more control over the layout and design than straight HTML ever did.

There are three parts to CSS:

1. CSS styles
2. CSS placement
3. CSS style cascade

You will take a look at each of these elements in turn, but for now, let's answer the question "Why use CSS?"

Why Use CSS

1. CSS Solves a Common Problem

HTML tags were originally designed to define only the content of a document. Once designers got their hands on the Web they wanted to control the look and feel more as well. Earlier, the layout of the document was controlled by the browser without the need for formatting tags. This is why you need to be aware of the browser's default style control, which you will get to latter. As the Web matured, it became more difficult to create Web sites where the content of HTML documents were separate from the document's presentation layout and look and feel. To solve this problem, the World Wide Web Consortium (W3) created CSS and added it to the HTML 4.0 specification.

2. CSS Can Save a Lot of Work

CSS is a breakthrough in Web design because it allows developers to control the style and layout of multiple Web pages all at once. CSS defines **how** HTML elements are to be displayed. Styles are normally saved in **external .css** files. This enables Web designers to define a style for each HTML element and apply it to as many Web pages as desired. To make a global change, simply change the style, and all elements in the site are updated automatically.

3. Multiple Styles Cascade into One

Style sheets allow style information to be specified in many ways. Styles can be specified inside a single HTML element (or tag), inside the <head> element of an HTML page, or in an external CSS file. Even multiple external style sheets can be referenced inside a single HTML document. These capabilities give CSS the power to style and layout Web sites so they are more usable and compact.

Style Syntax and Structure

Let's now write our first CSS style. CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties. CSS syntax is made up of three parts: a selector, a property and a value:

```
selector {property: value}
```

The selector is normally the HTML element (or tag) that you wish to define, the property is the attribute that you wish to change, and each property can take a value. The property and value are separated by a colon, and surrounded by curly braces:

```
body {color: black}
```

```
p {font-family: "sans serif"}
```

Note: If you wish to specify more than one property, you must separate each property with a semicolon. This example shows how to define a center-aligned paragraph, with a red text color:

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

To make the style definitions more readable, you can describe each property on is own line, like this:

```
p
{
text-align: center;
color: black;
font-family: arial
}
```

Grouping

You can also group selectors. Separate each selector with a comma. In the example below, all the header elements are grouped together on a single line. In this case all header elements will be displayed in green text color:

```
h1,h2,h3,h4,h5,h6
{
color: green
}
```

CSS Comments

Like in HTML, comments can be used to explain code, helping you edit the source code at a later date. Comments are ignored by browsers. A CSS comment begins with "/*", and ends with "*/", like this:

```
/* This is a comment */
p
{
text-align: center;
/* This is another comment */
color: black;
font-family: arial
}
```

Writing and Applying Your First Styles

Writing Your First Style

Open the boilerplate document that you created from the *Looking Ahead* sidebar. You can also copy and paste this content into another HTML page.

Looking Ahead: Assignment 1

To complete the assignments for this course you will need to write basic HTML

Change the title tag inside the head tag to get a more descriptive page title.

```
<title>First Styles</title>
```

You will write our first selector in the <head> section of our document. Write an opening style tag.

```
<style>
```

Now redefine the <body> so that by default, all the text on your page is purple.

Do this by writing the tag, then a curly brace, a property such as the color, a colon, then the value, purple, a semicolon, then a closing curly brace.

```
body {color: purple; }
```

Add a few lines and then add another selector, say the <p> tag and repeat the process, adding additional properties and a value as such:

```
p {   font-size: 36px;
font-weight: bold;
color: blue;
}
```

Close your style tag:

```
</style>
```

You now have your first styles that you can apply to text on this page.

Apply the Styles

To apply these styles, simply type some text between your opening and closing body tags in your HTML. The body style will automatically be used to format this text, in this example, the text will be colored purple.

Next add a <p> tag to your HTML page. Add some content between the opening and closing paragraph tags. The p style you defined previously in your HTML head section with CSS *<style>* will be used to style the text, in this case it will be blue, 36 pixels in size, and bold.

Check your results by opening this file in your browser; it should look like the image below.

code using a text editor like Windows Notepad or TextEdit on the Macintosh. To prepare for your assignments, let's write what is called a “boiler-plate" HTML document that can be used for all of your assignments here. This document can also be copied and pasted into ANY future HTML document.

The first thing you need in your boiler-plate HTML is to use a DOCTYPE to declare what type of HTML or XHTML you're using. The DOCTYPE lets browsers know what to expect and tells validators how to judge your code in order to check its syntax .

```
<!DOCTYPE html PUBLIC "-//W3C//DTD
XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html
xmlns="http://www.w3.org/1999/xhtml">
```

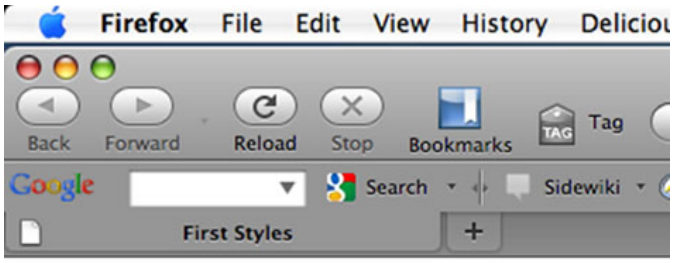
Next you need to tell the browser that you will "speak" to it in the language of HTML. This is done with the tag <html>

Type "<html>" in the next line of your document in Notepad. As you may recall from a previous HTML class, <html> is an opening tag and must be closed with a closing tag when you are finished typing HTML. So to make sure you don't forget the HTML close tag now type "</html>" a couple of lines down and write the rest of the document between these opening and closing tags <html> and </html>.

The next thing your document needs is a "head", which provides information about your document, and a "body", which is the content of the document. Since HTML is always very logical, the head (<head> and </head>) is on top of the body (<body> and </body>).

You have made your first simple Web page. Your document should now look like this:

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



Here is the final complete code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>First Styles</title>

<style>

body {color: purple; }
p {
font-size: 36px;
font-weight: bold;
color: blue;
}

</style>

</head>

<body>
body text

<p>paragraph text </p>
</body>
</html>
```

One of the trickiest things about CSS is that it requires the use of ending tags (as

```
<html
xmlns="http://www.w3.org/1999/xhtml">

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

What you have made here can also be the basic template for all your future HTML documents. Now let's add content to your Web page.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD
XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html
xmlns="http://www.w3.org/1999/xhtml">

<head>
<title>My first Web page</title>
</head>
<body>
<p>Hello! This is my first Web page.</p>
</body>
</html>
```

does XHTML, HTML5 and XML). For example, if you have a style applied to a paragraph tag , the browser may not know where to end that style if you do not include the ending tag .

Introduction to Selectors

There are many ways to apply styles across an HTML document, but often you want to use a style on some of the tags in an HTML document, but not all instances of the tag in HTML. Or, you may want to create a style that you can apply across several tags in a web page, without repeating the style rule. To do these types of selective style applications, you will use selectors.

In CSS, *selectors* are used to declare which elements a style applies to. Selectors are a kind of match expression, which can apply to all elements of a specific type, or to only those elements which match a certain attribute.
For example, in our previous p style, the *p* selector applies to all paragraph tags in the document.

```
p {
font-size: 36px;
font-weight: bold;
color: blue;
}
```

If this style was defined in a separate style document, called an external *style sheet*, then it would apply to all the paragraphs in all the pages that use this style sheet. In contrast, the class selector .bodyblue applies only to those document elements that have been styled with a class named “bodyblue”.

There are various types of CSS selectors or Selector Types:

- **Custom CSS (Class) styles:** create a customized style with the set attributes. These class styles can be applied to *any* tag.
- **HTML Base/Tag styles:** redefine the formatting for a particular tag, such as a tag or a <h1> tag. All text formatted with either a p tag or h1 tag is immediately updated.
- **Advanced CSS Selector styles:** redefine the formatting for:
    - A particular *combination* of tags (for example, td h2 applies whenever an h2 header appears *inside* a table cell) and pseudo-class styles (for example, a:link, a:hover, a:visited)
    - A specific ID attribute (for example, #myStyle applies to all tags that contain the attribute-value pair id="myStyle)
- In addition to these, a set of ***pseudo-classes*** can be used to define further behavior. Probably the best-known of these is :hover, which applies a style only when the user hovers the mouse cursor over a link.

  A *pseudo-class* selects entire elements, such as :link or :visited, whereas a*pseudo-element* makes a selection that may consist of partial elements, such as :first-line or :first-letter.

The *ID selector* refers to an ID used in a tag, e.g., <span *ID*="masthead">. Likewise, the *class selector* refers to a class, as in <span *class*="first">.

Base/Tag Selectors

As a Web designer, you will want to use all of these selectors at one time or another and the key is to learn them well enough to know under which circumstances one selector will work better than another.

Many designers confuse these various sectors and often use only one to the exclusion of the others and fail to utilize the strengths of each. So before we go any further, let’s review these selectors with examples and more descriptions, starting with the **Base**

Selector also called **Tag Selectors**

Base Selector/Tag Selector

Most of the styles covered so far are this type of selector. The TAG selector takes an existing tag and redefines some, or all, of its default properties to style the whole element, such as the <p> tag

This example styles ALL paragraphs to red.

```
P {color: #F00;}
```

You can also define CUSTOM selectors known as ID or CLASS selectors. These can both be applied to HTML elements as simple attributes and they provide much greater control over the formatting of your HTML content.

The Class Selector

The class attribute is available to almost every (X)HTML tag.

```
<p class="bluetext">

Or

<div class="bordered">
```

The class selector is always defined with a period before the class name and it is important that you always include the dot when naming your class selectors.

```
.bordered {
border : 1px solid black ;
}
```

You can also set a class selector to only specific elements that have that class. Do this by placing the type selector before the period in the class selector without any spaces:

```
div1.bordered {
border : 2px solid red ;
}
```

In this case only the div1 tag will get the solid red border that is 2 pixels wide.
The class selector allows you to set multiple styles to the same element or tag in a document. For example, you might want to have certain sections of your text styled in different colors. You would then assign your paragraphs with classes like this:

```
<style type="text/css">
p.blue {background-color: #0000ff;}
p.red {background-color: #ff0000;}
</style>
```

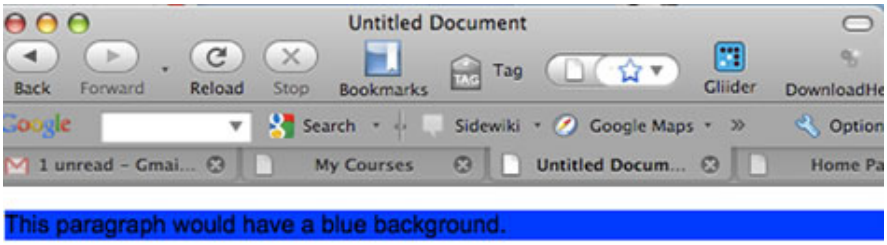
When you want to call the various classes you would use the CLASS attribute within the paragraph tag. Any tag that did not have a

class attribute would be given the default style for that tag. For example:

```
<p class="blue">
This paragraph would have a blue background.
</p>
```

```
<p>
And this paragraph would default back to the default paragraph style.
</p>
```

Try this now yourself. The results should look like the image below.



If you want to use a single class across multiple HTML elements, simply remove the HTML element from the beginning of the style call (but be sure to leave the period (.) in place) like this:

```
<style type="text/css">
.blue {background-color: #0000ff;}
.red {background-color: #ff0000;}
</style>
```

These two classes are then available to any element that needs them:

```
<p class="blue">
This paragraph has a blue background.
</p>
<h2 class="blue">This h2 content also has a blue background.</h2>
```

The ID Selector

The ID selector allows you to give a name to a specific style without associating it to any particular tag or other HTML element. However, each ID can only be used ONCE in an HTML document and is unique to the page.

You associate an ID tag the same way you associate classes, that is, within the element that should have that style. The id selector is defined with a # preceding its name.

```
#blue {color: #000099;}
<p id="blue">This will create blue text</p>
```

You can name your ID tags any way you like, however it is good form to provide your ID tags with names that you will remember and understand a few weeks or months later.

The style rule below will match the element with an id attribute of “blue_color with a value of "blue":

```
#blue_color {color: blue}
```

The style rule below will match the p element with the id "paragraph1" which centers the text content inside the paragraph and colors it red:

```
p#paragraph1
{
text-align: center;
color: red
}
```

Never start an ID name with a number and also note that HTML standards require that your ID names all be unique.

Context Selectors

Contextual selectors consist of two or more simple selectors separated by white space.

This contextual selector is constructed to show that the rule will be applied only when you add the h3 style tag to content inside a table cell:

```
td h3 {color: blue; }
```

In this case, all text inside tables and formatted with the h3 tag will be blue. However, If the browser does not find an exact match (in this case it only finds <h3> elements outside of <td> elements), it will NOT apply the styles dictated by the contextual selector to them.

Class vs. ID

When to Use Each Selector

It is often hard to decide when to use a class and when to use an id for an element. Here is an easy way to think of the difference:

Use a class selector if:

1. The style is used in various places throughout the document.
2. The style is very general.

For example, use classes to style certain words or sentences red or to center some text.

Use an id selector if:

1. The style will be used only once in each document.

2. The style is specific to a certain area of the document.

For example, use ids to create headers or navigation elements with div tags or to style all your text in the footer section of each document.

Remember that an id can only appear **once** in any HTML document. Once you've used the id, it should not be used again on that page.

As you start using CSS and apply your styles to page content you will learn much more about all three of these selectors, but let's summarize here:

- Classes are a very flexible method for applying your CSS rules again and again within a page. Use Classes to control elements that need to be used in many places on your pages.
- Use Classes to control elements that belong to a group and to also enhance the behavior and styling of an ID.
- Don't use classes for main structural elements within a page, such as headers and navigation sections.
- Before creating a new class, make sure that redefining an HTML tag using a Tag selector cannot do the same thing. You do not want to get class happy and overuse them. Most Web designers use Tag selectors to style *standard styles* that are used throughout the site and classes to style *exceptions* to these site style standards.
- Use IDs to style structural sections or portions of a page.
- Avoid using IDs when there is more than one property or setting required for the CSS rule.
- Do not use IDs for element that may multiply in the future such as additional images, link styles, or paragraphs.

Week 1 Assignment

Writing your first styles

To write CSS, you will be writing HTML tags with CSS notation added in the HTML file. Open your simple text software (Notepad on your PC or Text Edit on your Mac) or Dreamweaver using the code window only. Write the HTML and CSS needed for this assignment and then save as a .htm or .html file. Check your work in your web browser. Please make sure that you have no extra spaces or missing syntax, this will cause errors. Also make sure that you are not using HTML for styling.

Write Styles and HTML that :
1) Use the boiler-plate HTML code provided in the Looking Ahead: Assignment 1 to create a basic Web document and to correctly add your style definitions to this document. 1.

2) Write a style that formats all heading tags (H to H6) with the Arial bold font, red in color, and left aligned. In addition, size the H1 tag at 32 px and the H3 tag at 10 px.

a. Make sure your styles all include a descriptive comment, a selector, a property, and a value.

b. Group your styles where doing so makes them easier to read and always add a liberal amount of comments to explain each style and make them easier to edit at a later date.

3) Apply your H1, H2 and H3 styles to 3 different paragraphs of text on an HTML page.