CSS – Cascading Style Sheets

Agenda:

Day 3

* Syntax for CSS
* Selectors (Grouping / Combining)
* Margins and Padding - How to space things out.
* Forms
* CSS - Lists
* Borders - Things that go around things.
* Putting It All Together - Throwing all of the above ingredients into one spicy hotpot.

**Fundamentals of HTML and CSS**

***Learning Objectives:****In this lesson, students learn the fundamentals of HTML, XHTML and CSS in order to understand the foundations of creating the style and structure of their web pages. Students learn how to use document types (DOCTYPE) to help them validate their web pages and for use as a troubleshooting tool. Additionally, students build a simple web page in order to understand how to add images, create hyperlinks and work with both internal and external style sheets.*

**CSS Layout**

***Learning Objectives:****In this lesson, students learn the fundamentals of creating a two column, fixed-width layout using HTML and CSS. Students learn the function of a CSS “reset” file and how to add it to their pages. The core concepts of CSS layout are covered in this lesson. This includes the use of the <div> element, as well as the CSS float and clear properties, to create columns on the page.*

CSS exists to style your HTML — to **present** your content. The language of **Cascading Style Sheets** is different to HTML yet just as simple and user friendly.

Options for course review:

* Text Editors: notepad, notepad++,
* Text Editors w/results: W3Schools Tryit, Practiceboard

# **Unstyled Example**

# Educational Technology Blog

a. Educational Technology

b. [Google](http://www.google.com)

### Learning CSS

If you write web pages, Cascading Style Sheets (CSS) is a great language to know.

It lets you add fonts, colors, background images, and more to your pages.

Furthermore, it separates your content from your presentation, so you can change them independently. Trust me, that's a good thing.

Without CSS, you can read the text on this page just fine. And if you write CSS for this page, you can apply that same CSS to other pages to make them look similar. Cool.

**Styled Example**

***Educational Technology Blog***

a. Educational Technology

b. Google

**Learning CSS**

If you write web pages, Cascading Style Sheets (CSS) is a great language to know.

It lets you add ***fonts***, **colors**, background images, and more to your pages.

Furthermore, it separates your content from your presentation, so you can change them independently. Trust me, that's a good thing.

Without CSS, you can read the text on this page just fine. And if you write CSS for this page, you can apply that same CSS to other pages to make them look similar. Cool.

## Examples

* Now we're going to look at some examples of CSS.
* Pretty much any website you look at uses CSS, but the *CSS Zen Garden* website is a great example.
* This website shows how you can apply **different presentation styles to the same content.** Demonstrate the website from your browser, and show how clicking on different links changes the look.

**Student Exploration**
Tell Students

* Now I will let you explore this website. As you change between designs, keep in mind **what changes** and **what doesn't change.**
* Open your computers and go to www.csszengarden.com

**Discussion**
Ask the students, **what changed and what did not change?**

* font color, size, family; background color and image; layout of elements on the page; etc. **change**
* content (text) **does not change**

## Resources

## Overview - [www.csszengarden.com](http://www.csszengarden.com)

Cheat Sheet / CSS References - http://www.phpforkids.com/references/cheat-sheet-css.php

## Definition and Usage

### [CSS Syntax](http://www.cssbasics.com/css-syntax/)

The syntax for CSS is different than that of (X)HTML markup. Though it is not too confusing, once you take a look at it. It consists of only 3 parts.

selector { property: value }

The selector is the (X)HTML element that you want to style. The property is the actual property title, and the value is the style you apply to that property.

Each selector can have multiple properties, and each property within that selector can have independent values. The property and value are separated with a colon and contained within curly brackets. Multiple properties are separated by a semi colon.

body {
  background: #eeeeee;
  font-family: “Trebuchet MS”, Verdana, Arial, serif;
}

CSS Syntax

A CSS rule-set consists of a selector and a declaration block:



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

In the following example all <p> elements will be center-aligned, with a red text color:

Example

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

Grouping Selectors

If you have elements with the same style definitions, like this:

h1 {
    text-align: center;
    color: red;
}

h2 {
    text-align: center;
    color: red;
}

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

It will be better to group the selectors, to minimize the code.

To group selectors, separate each selector with a comma.

In the example below we have grouped the selectors from the code above:

h1, h2, p {
    text-align: center;
    color: red;
}

Combining Selectors

You can combine elements within one selector in the following fashion.

h1, h2, h3, h4, h5, h6 {
  color: #009900;
  font-family: Georgia, sans-serif;
}

Inheritance

When you nest one element inside another, the nested element will inherit the properties assigned to the containing element. Unless you modify the inner elements values independently.

For example, a font declared in the body will be inherited by all text in the file no matter the containing element, unless you declare another font for a specific nested element.

body {font-family: Verdana, serif;}

Now all text within the (X)HTML file will be set to Verdana.

If you wanted to style certain text with another font, like an h1 or a paragraph then you could do the following.

h1 {font-family: Georgia, sans-serif;}
p {font-family: Tahoma, serif;}

**CSS ID Selector**

IDs are similar to [classes](http://www.cssbasics.com/css-ids/chapter_3_css_class.html), except once a specific id has been declared it cannot be used again within the same (X)HTML file.

IDs are generally used to style the layout elements of a page that will only be needed once, whereas I use [classes](http://www.cssbasics.com/css-ids/chapter_3_css_class.html) to style text and such that may be declared multiple times.

The main container for this page is defined by the following.

<div id=”container”>
Everything within my document is inside this division.
</div>

I have chosen the id selector for the “container” division over a class, because I only need to use it one time within this file.

Then in my CSS file I have the following:

#container{
  width: 80%;
  margin: auto;
  padding: 20px;
  border: 1px solid #666;
  background: #ffffff;
}

**Example**

<!DOCTYPE html>

<html>

<head>

<style>

#para1 {

 text-align: center;

 color: red;

}

</style>

</head>

<body>

<p id="para1">Hello World!</p>

<p>This paragraph is not affected by the style.</p>

</body>

</html>

**Selectors: Example 1**

<!DOCTYPE html>

<html>

<head>

<style>

body {

 color: red;

}

h1 {

 color: #00ff00;

}

p.ex {

 color: rgb(0,0,255);

}

</style>

</head>

<body>

<h1>This is heading 1</h1>

<p>This is an ordinary paragraph. Notice that this text is red. The default text-color for a page is defined in the body selector.</p>

<p class="ex">This is a paragraph with class="ex". This text is blue.</p>

**Selector/Class: Example 2**

Times

Serif

Arial

Helvet

verdana

<!DOCTYPE html>

<html>

<head>

<style>

p.serif {

 font-family: "Times New Roman", Times, serif;

}

p.sansserif {

 font-family: Arial, Helvetica, sans-serif;

}

</style>

</head>

<body>

<h1>CSS font-family</h1>

<p class="serif">This is a paragraph, shown in the Times New Roman font.</p>

<p class="sansserif">This is a paragraph, shown in the Arial font.</p>

</body>

</html>

**Margins / Padding**

The margin property declares the margin between an (X)HTML element and the elements around it. The margin property can be set for the top, left, right and bottom of an element. (see example below)

  margin-top: length percentage or auto;
  margin-left: length percentage or auto;
  margin-right: length percentage or auto;
  margin-bottom: length percentage or auto;

As you can also see in the above example you have 3 choices of values for the margin property

* length
* percentage
* auto

You can also declare all the margins of an element in a single property as follows:

  margin: 10px 10px 10px 10px;

**Aligning, Centering, … images**

<!DOCTYPE html>

<html>

<head>

<style>

body {

 background: #ffffff url("img\_tree.png") no-repeat right top;

 margin-right: 200px;

}

</style>

</head>

<body>

<h1>Hello World!</h1>

<p>Now the background image is only shown once, and it is also positioned away from the text.</p>

<p>In this example we have also added a margin on the right side, so that the background image will not disturb the text.</p>

</body>

</html>

**Margin Examples**

<!DOCTYPE html>

<html>

<head>

<style>

p {

 background-color: yellow;

}

p.ex {

 border: 1px solid red;

 margin-top: 100px;

 margin-bottom: 100px;

 margin-right: 150px;

 margin-left: 80px;

}

</style>

</head>

<body>

<h2>Using Individual margin Properties:</h2>

<p>This is a paragraph with no specified margins.</p>

<p class="ex">This paragraph has a top and bottom margin of 100px, a left margin of 80px, and a right margin of 150px.</p>

</body>

</html>

The class selector allows you to style items within the same (X)HTML element differently. Similiar to what I mentioned in the introduction about inline styles. Except with classes the style can be overwritten by changing out stylesheets. You can use the same class selector again and again within an (X)HTML file.

Spans are very similar to divisions except they are an inline element versus a block level element. No linebreak is created when a span is declared.

You can use the span tag to style certain areas of text, as shown in the following:

<span class=”italic”>This text is italic</span>

Then in my CSS file:

.italic{
  font-style: italic;
}

The final result is: *This text is italic.*

**Padding**

Padding is the distance between the border of an (X)HTML element and the content within it.

Most of the rules for [margins](http://www.cssbasics.com/css-padding/chapter_7_css_margins.html) also apply to padding, except there is no “auto” value, and negative values cannot be declared for padding.

  padding-top: length percentage;
  padding-left: length percentage;
  padding-right: length percentage;
  padding-bottom: length percentage;

As you can also see in the above example you have 2 choices of values for the padding property

* length
* percentage

You can also declare all the padding of an element in a single property as follows:

  padding: 10px 10px 10px 10px;

If you declare all 4 values as I have above, the order is as follows:

1. top
2. right
3. bottom
4. left

If only one value is declared, it sets the padding on all sides. (see below)

  padding: 10px;

**Example**

<!DOCTYPE html>

<html>

<head>

<style>

p.one {

 border: 1px solid red;

 background-color: yellow;

 padding-top: 50px;

 padding-right: 30px;

 padding-bottom: 50px;

 padding-left: 80px;

}

</style>

</head>

<body>

<h2>Using Individual padding Properties:</h2>

<p>This is a paragraph with no specified padding.</p>

<p class="one">This paragraph has a top and bottom padding of 50px, a left padding of 80px, and a right padding of 30px.</p>

</body>

</html>

## **Border**

CSS Border Properties

The CSS border properties allow you to specify the style, width, and color of an element's border.

You can set the color, style and width of the borders around an element in one declaration by using the border property.

border: 1px solid #333333;

Values:

* color
* style
* width

**Bordered Inputs**

Use the border property to change the border size and color, and use the border-radius property to add rounded corners:

input[type=text] {
    border: 2px solid red;
    border-radius: 4px;
}

input[type=text] {
    border: none;
    border-bottom: 2px solid red;
}

**Example 1**

<!DOCTYPE html>

<html>

<head>

<style>

p {

 border-top-style: dotted;

 border-right-style: solid;

 border-bottom-style: dotted;

 border-left-style: solid;

}

</style>

</head>

<body>

<p>2 different border styles.</p>

</body>

</html>

**Example 2**

<!DOCTYPE html>

<html>

<head>

<style>

p.dotted {border-style: dotted;}

p.dashed {border-style: dashed;}

p.solid {border-style: solid;}

p.double {border-style: double;}

p.groove {border-style: groove;}

p.ridge {border-style: ridge;}

p.inset {border-style: inset;}

p.outset {border-style: outset;}

p.none {border-style: none;}

p.hidden {border-style: hidden;}

p.mix {border-style: dotted dashed solid double;}

</style>

</head>

<body>

<h2>The border-style Property</h2>

<p>This property specifies what kind of border to display:</p>

<p class="dotted">A dotted border.</p>

<p class="dashed">A dashed border.</p>

<p class="solid">A solid border.</p>

<p class="double">A double border.</p>

<p class="groove">A groove border.</p>

<p class="ridge">A ridge border.</p>

<p class="inset">An inset border.</p>

<p class="outset">An outset border.</p>

<p class="none">No border.</p>

<p class="hidden">A hidden border.</p>

<p class="mix">A mixed border.</p>

</body>

</html>

**Lists**

**HTML Lists and CSS List Properties**

In HTML, there are two main types of lists:

* unordered lists (<ul>) - the list items are marked with bullets
* ordered lists (<ol>) - the list items are marked with numbers or letters

The CSS list properties allow you to:

* Set different list item markers for ordered lists
* Set different list item markers for unordered lists
* Set an image as the list item marker
* Add background colors to lists and list items

**Example**

<!DOCTYPE html>

<html>

<head>

<style>

ul.a {

 list-style-type: circle;

}

ul.b {

 list-style-type: square;

}

ol.c {

 list-style-type: upper-roman;

}

ol.d {

 list-style-type: lower-alpha;

}

</style>

</head>

<body>

<p>Example of unordered lists:</p>

<ul class="a">

 <li>Coffee</li>

 <li>Tea</li>

 <li>Coca Cola</li>

</ul>

<ul class="b">

 <li>Coffee</li>

 <li>Tea</li>

 <li>Coca Cola</li>

</ul>

<p>Example of ordered lists:</p>

<ol class="c">

 <li>Coffee</li>

 <li>Tea</li>

 <li>Coca Cola</li>

</ol>

<ol class="d">

 <li>Coffee</li>

 <li>Tea</li>

 <li>Coca Cola</li>

</ol>

</body>

</html>

**Student will create a HTML/CSS web page**

The web page will consist of tags presented in class today:

* Borders
* Padding
* Margins
* Lists
* Selectors (Grouping)
* Alignment

**Forms**

The look of an HTML form can be greatly improved with CSS:

Top of Form

First NameLast NameState             

Bottom of Form

* Styling Input Fields Use the width property to determine the width of the input field:

## Padded Inputs Use the padding property to add space inside the text field.

**Example**

<!DOCTYPE html>

<html>

<style>

input[type=text], select {

 width: 100%;

 padding: 12px 20px;

 margin: 8px 0;

 display: inline-block;

 border: 1px solid #ccc;

 border-radius: 4px;

 box-sizing: border-box;

}

input[type=submit] {

 width: 100%;

 background-color: #4CAF50;

 color: white;

 padding: 14px 20px;

 margin: 8px 0;

 border: none;

 border-radius: 4px;

 cursor: pointer;

}

input[type=submit]:hover {

 background-color: #45a049;

}

div {

 border-radius: 5px;

 background-color: #f2f2f2;

 padding: 20px;

}

</style>

<body>

<h3>Using CSS to style a HTML Form</h3>

<div>

 <form action="action\_page.php">

 <label for="fname">First Name</label>

 <input type="text" id="fname" name="firstname">

 <label for="lname">Last Name</label>

 <input type="text" id="lname" name="lastname">

 <label for="country">State</label>

 <select id="country" name="country">

 <option value="australia">Australia</option>

 <option value="canada">Canada</option>

 <option value="usa">USA</option>

 </select>

 <input type="submit" value="Submit">

 </form>

</div>

</body>

</html>

**Lesson Reflections – Part 2**

**Student will create a HTML/CSS web page**

The web page will consist of tags presented in class today:

* Forms
* Lists