COMM 641
Web Programming Beginning

Zak Ruvalcaba / Emily Kay / Paul Richard

Lecture 4

This week at a glance…

This week at a glance…

- Introduction to CSS
  - What is CSS?
  - Why Should I Use CSS?
  - How CSS Works
  - 3 Ways of Defining Styles
- Selector Types
  - CSS 1 Selector Types
- Exploring the CSS Properties
  - Font Properties
  - Background Properties
  - Block Properties
  - Box Properties
  - Border Properties
  - List Properties
  - Positioning Properties

Introduction to CSS

Introduction to CSS

What is CSS?

- Created and formally adopted in 1997, Cascading Style Sheets, or CSS, are a Web Development technique for creating consistent **designs** for Web pages.

- CSS was introduced to allow developers to separate their content from their design. This allows HTML to perform more of the function that it was originally intended to be used for - the markup of content, without worry about the design and layout.

- CSS, unlike HTML, gives the Web developer much more control over the layout and design than HTML ever did by outlining hundreds of styling possibilities for a Web page.

- Categories of design exist for type, backgrounds, lists, block/paragraph formatting, borders, links, element positioning, and more.

Introduction to CSS

Why should I use CSS?

- **Consistency:** by editing a single CSS file, you can make site-wide design changes in seconds.
- **Responsive:** CSS lets you output to multiple formats quickly.
- **Cacheable:** external CSS files are cached by browsers, improving load time.
- **Self-describing:** CSS lets you use logical names for page elements. It's self-describing which makes it easy to work with.
- **Flexible:** CSS lets you do things normal HTML doesn't.
- **Optimization:** Pseudo classes cut down on the need to use scripting languages.
- **Clean code:** external style sheets result in less confusing code which speeds up the process of line placement. Clean code is more accessible to search engines, thereby improving their ability to spider your content, leading to higher rankings.
- **Standardization:** If your goal is to create valid HTML5 web pages, you'll have to use it anyway.

Introduction to CSS

How CSS works

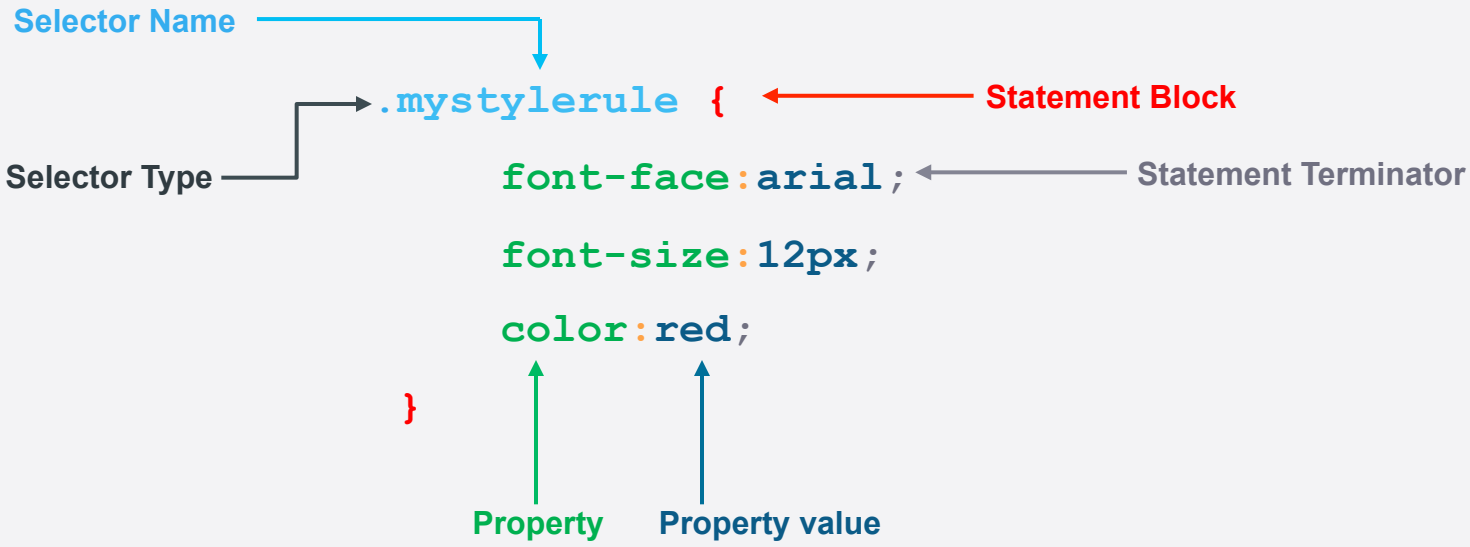
The general process involved with CSS is simple:

1. CSS works by outlining style rules
2. Style rules contain properties and property values that define how an element on your web page will look
3. Style rules are then applied to elements on your web page to stylize that element.

Introduction to CSS

How CSS works

The following example outlines the basic structure of a style rule (class):



Introduction to CSS

How CSS works

Once the style rule has been created, it'll reside within a style sheet. That style sheet will then by linked into the web page via the <link> tag. Selectors are then applied to elements in your Web page to stylize them.

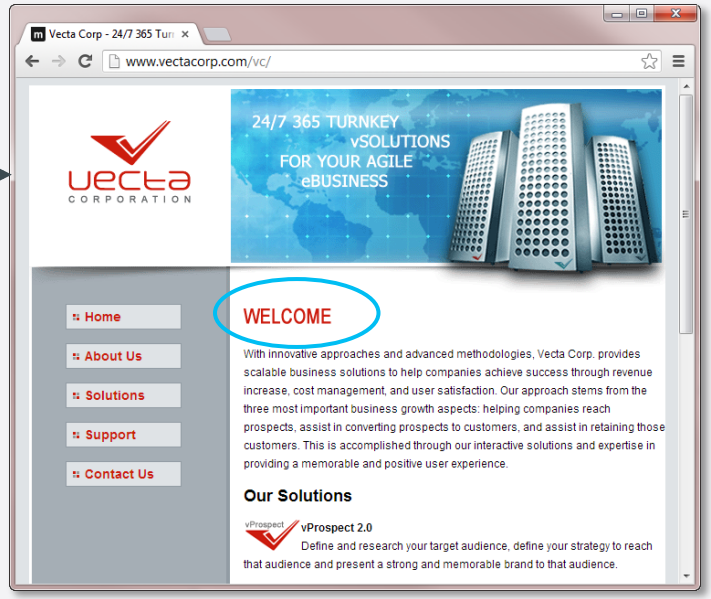
```
.heading {
     color:red;
     font-size:1.6em;
}
```

Style Sheet
styles.css

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

<h1 class="heading">
     Welcome
</h1>
```

Web Page
index.html



Introduction to CSS

Three ways of defining styles

As mentioned in the previous presentation, there are 3 ways of defining style sheets. They are:

- External
- Document-wide (Embedded)
- Inline

Introduction to CSS

External style sheets

By far the most common method for outlining your styles is to use an external style sheet. External style sheets are created in a separate file and then linked into all of the Web pages in your site. This allows you to centrally and globally manage all of the styles for every element in your site.

Introduction to CSS

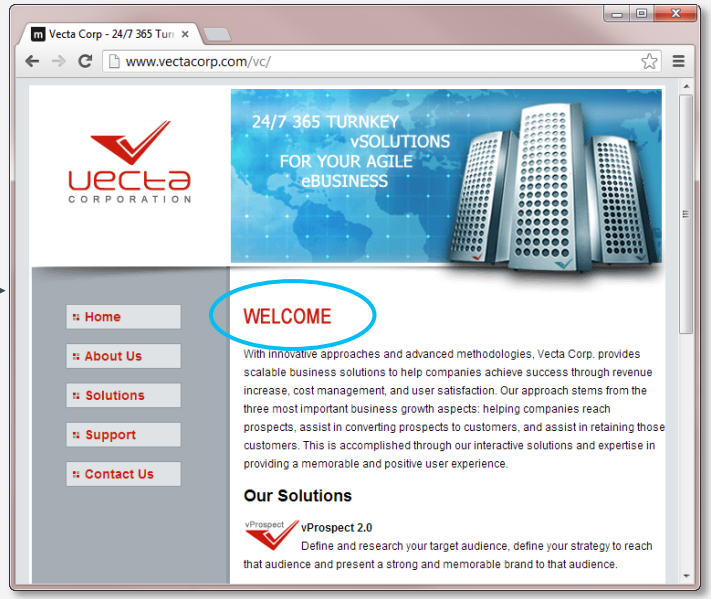
Document-wide style sheets

A second way of defining a style sheet would be to use document-wide styles. In this method, styles are placed within a <style> tag which resides within the <head> tag of the web page. The style is still utilized by the tag in exactly the same way as the external method but the style only becomes available to that web page and no others within the site.

```
<head>
<style>
   .heading {
      color:red;
      font-size:1.6em;
   }
</style>
</head>

<body>
<h1 class="heading">Welcome</h1>
</body>
```

Web Page
index.html



Introduction to CSS

Combining external with document-wide style sheets

Of course, you could also combine external and document-wide style sheets. Consider the following scenario:

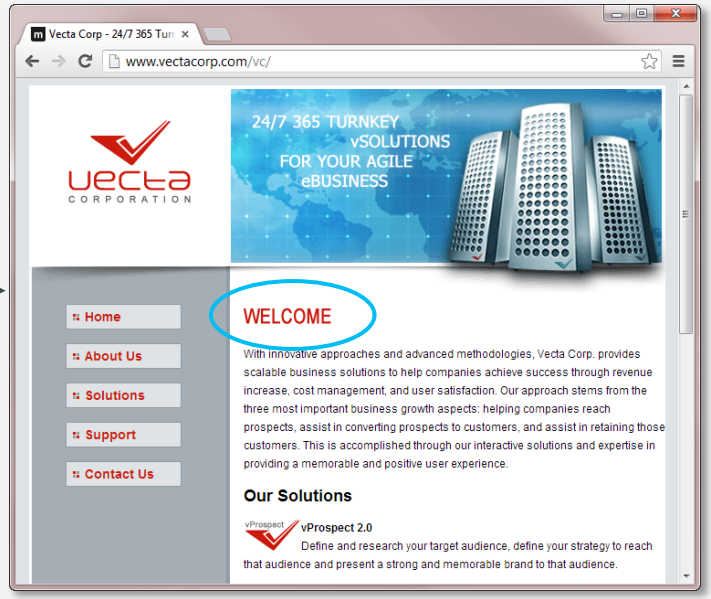
Introduction to CSS

Inline styles

Inline styles are a quick way of applying style to elements within your web pages. The upside to using inline styles is that you can quickly switch to the source and apply a style to any tag. The downside however is that the style is only applied to that tag and no others within the page or site. Furthermore, because inline styles are applied to tags, they tend to get lost and forgotten about throughout the design process.

```
<body>
<h1 style="color:red;font-size:1.6em;">
    Welcome
</h1>
</body>
```

Web Page
index.html



Selector Types

Selector Types

Applying style to your web pages

Now that you understand that styles are made up of style rules, and style rules are stored in style sheets (usually), let’s turn our attention to the various types of selectors that you’ll use to build your style rules.

Selector Types

Introduction to selector types – CSS 1

Selector Type	Example	Description
`.class`	.heading	Selects all elements with class="heading".
`#id`	#heading	Selects all elements with id="heading".
`element`	h1	Selects all <h1> elements.
`element, element`	div, h1	Selects all elements and <h1> elements.
`element element`	div h1	Selects all <h1> elements inside elements.
`:active`	a:active	Selects the active link.
`::first-letter`	p::first-letter	Selects the first letter of every <p> element.
`::first-line`	p::first-line	Selects the first line of every <p> element.
`:hover`	a:hover	Selects links on mouse over.
`:link`	a:link	Selects all unvisited links.
`:visited`	a:visited	Selects all visited links.

Selector Types

Introduction to selector types – CSS 2

Selector Type	Example	Description
`*`	*	Selects all elements.
`element>element`	div > p	Selects all <p> elements where the parent is a element.
`element+element`	div + p	Selects all <p> elements that are placed directly after elements.
`[attribute]`	input[checked]	Selects all <input> elements that have been checked.
`[attribute=value]`	input[type=text]	Selects all <input> elements where the type attribute's value is "text".
`[attribute~=value]`	a[title~=vecta]	Selects all <a> elements with a title attribute containing the word "vecta".
`[attribute|=value]`	a[lang|=en]	Selects all <a> elements with a lang attribute value starting with "en".
`::after`	p::after	Insert content after every <p> element.
`::before`	p::before	Insert content before every <p> element.
`:first-child`	p:first-child	Selects every <p> element that is the first child of its parent.
`:focus`	input:focus	Selects the <input> element which has focus.

Exploring the CSS Properties

Exploring the CSS Properties

Font properties

Font properties can be used to set styling for type on a web page. Properties include the ability to change the font face, font size, color, style, weight, variant, and more. Font properties are some of the most basic properties in CSS and one's that you'll use fairly often.

Exploring the CSS Properties

Font properties (CSS 1 and CSS 2)

Property	Description	Values
`color`	Sets the color of text.	Hex color or color by name
`font`	A generic property that allows you to set the font style, variant, weight, size, line-height, and font family within a single style rule.	font-style, font-variant, font-weight, font-size, line-height, font-family
`font-family`	Sets the font family for the text.	Font family
`font-size`	Sets the size of the text.	Font size in either pixels, percent, points, inches, millimeters, picas, em, etc.
`font-style`	Sets the style of the text.	normal, italic, oblique
`font-variant`	Allows you to convert the text to small caps.	normal, small-caps
`font-weight`	Sets the "boldness" of the text.	normal, bold, bolder, lighter, and 100-900
`text-decoration`	Sets the special styling that should be applied to text.	none, underline, overline, line-through, blink

Exploring the CSS Properties

Font properties - example

This example uses a type selector to set various font properties for the <body> tag:

CODE

```
body {
    font-family:Arial,Helvetica,sans-serif;
    color:#DFE3E6;
    font-size:1em;
}
```

You can also use the generic font property to set all of these properties on one line. Notice that color isn't a value of the font property and therefore will still need to be added within its own line:

CODE

```
body {
    font:1em Arial,Helvetica,sans-serif;
    color:#DFE3E6;
}
```

Exploring the CSS Properties

Background properties

Background properties can be used to set background colors and images for elements on a web page or the web page itself. Properties include the ability to change the background color, set a background image, set how the background image should repeat, and more.

Exploring the CSS Properties

Background properties (CSS 1 and CSS 2)

Property	Description	Values
`background`	A generic property that allows you to set the background color, image, repeat, attachment, and position.	background color, image, repeat, attachment, position
`background-attachment`	Sets whether the background image should scroll with the page or remain fixed.	scroll, fixed
`background-color`	Sets the background color for an element.	hex color or color by name
`background-image`	Associates a background image with an element.	url for the image, none
`background-position`	Sets the position of the background image in relation to the element.	percentage, length, left, center, or right
`background-repeat`	Sets how the background image should repeat within the element.	repeat, repeat-x, repeat-y, no-repeat

Exploring the CSS Properties

Background properties - example

This example uses a type selector to set various background properties for the <h1> tag:

CODE

```
h1 {
    background-color:#DFE3E6;
    background-image:url('icon.gif');
    background-repeat:no-repeat;
    background-position:7px center;
}
```

You can also use the generic background property to set all of these properties on one line:

CODE

```
h1 {
    background:#DFE3E6 url('icon.gif') no-repeat 7px center;
}
```

Exploring the CSS Properties

Block properties

Block properties (also known as type properties) can be used to set how text/type will be handled within a web page or other nested elements. Properties include the ability to set word spacing, letter spacing, vertical alignments, text alignments, indenting, white spacing, and more.

Exploring the CSS Properties

Block properties (CSS 1 and CSS 2)

Property	Description	Values
`word-spacing`	Sets the amount of space in between words.	normal, length
`letter-spacing`	Sets the amount of space in between letters.	normal, length
`line-height`	Sets the amount of space in between lines of text.	normal, numeric value, length, percentage
`vertical-align`	Sets the vertical alignment of elements.	baseline, sub, super, top, text-top, middle, bottom, text-bottom, percentage, and length
`text-align`	Sets the horizontal alignment of text.	left, right, center, justify
`text-indent`	Allows you to indent the first line of text in a block container.	normal, small-caps
`text-transform`	Control's the capitalization of text.	capitalize, uppercase, lowercase, none
`white-space`	Specifies how white space inside an element is handled.	normal, pre, nowrap, pre-wrap, and pre-line

Exploring the CSS Properties

Block properties - example

This example uses a type selector to set various text properties for the <body> tag:

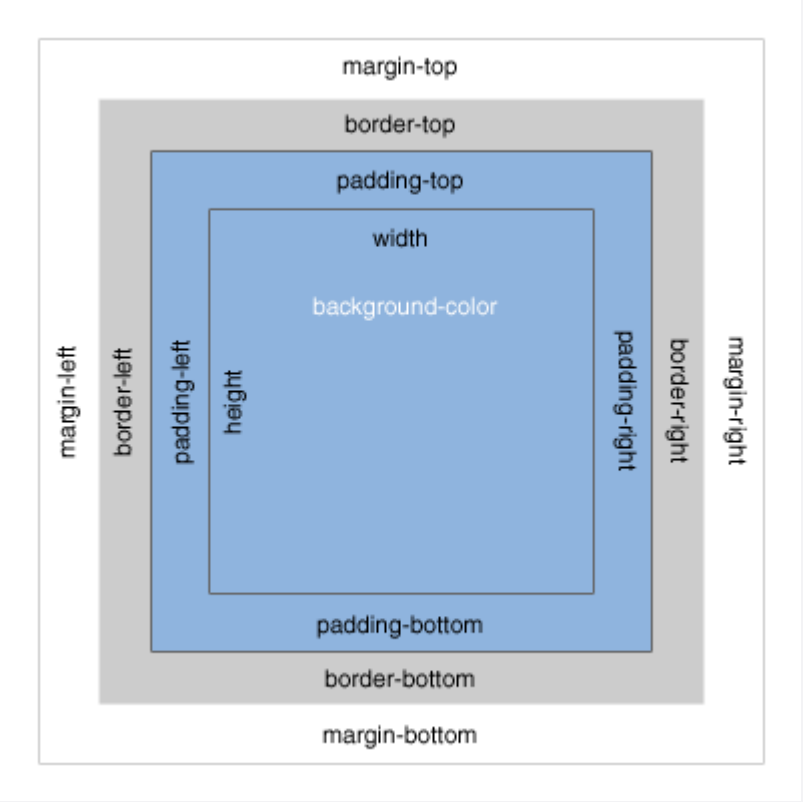
CODE

```
body {
    font-family:Arial,Helvetica,sans-serif;
    font-size:1em;
    line-height:2em;
    text-align:center;
    vertical-align:top;
}
```

Exploring the CSS Properties

Box properties

Most HTML elements can be considered boxes. In CSS, the term "box model" refers to the design and layout of these elements using various CSS properties that control margins, borders (covered in the next presentation), paddings, widths, heights, and more. Box properties are an important part of web design and you'll find that you'll be using these properties with great regularity regardless of whether you're using the DIV+CSS or the HTML5+CSS method of design and layout (covered later).



Exploring the CSS Properties

Box properties (CSS 1 and CSS 2)

Property	Description	Values
`margin<-side>`	Sets the amount of margin for a side of the box.	Width in pixels or percent
`padding<-side>`	Set the amount of padding for a side of the box.	Width in pixels or percent
`width`	Sets the width of the box.	length, percentage, auto
`min-width`	Sets the minimum width of the box.	length, percentage, auto
`max-width`	Sets the maximum width of the box.	length, percentage, auto
`height`	Sets the height of the box.	length, percentage, auto
`min-height`	Sets the minimum height of the box.	length, percentage, auto
`max-height`	Sets the maximum height of the box.	length, percentage, auto

Exploring the CSS Properties

Box properties - example

This example uses a type selector to set various box properties for the <section> tag:

CODE

```
section {
    width:460px;
    min-height:200px;
    padding:5px;
    margin-right:20px;
}
```

Exploring the CSS Properties

Border properties

Use the collection of border properties to set the visible border/outline style, color, and width for elements within your web page. Additionally, you can use the new CSS3 box-shadow and/or border-radius properties to set an elements drop shadow or corner curve properties respectively.

Exploring the CSS Properties

Border properties (CSS 1 and CSS 2)

Property	Description	Values
`border<-side><-color>`	Sets the color for the border of a box.	Hex color or color by name
`border<-side><-style>`	Sets the style for the border of a box.	none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset
`border<-side><-width>`	Sets the width for the border of a box.	Width in pixels or percent
`outline`	Generic property that sets the outline color, style, and width of an element within a single selector. Outline is different than border in that outline is not part of the element's dimension. Therefore, changing the width and height of an element will not affect the element's outline like it would border.	outline-color, outline-style, outline-width
`outline-color`	Sets the color of the outline.	Hex color or color by name, invert
`outline-style`	Sets the style of the outline.	none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset
`outline-width`	Sets the width of the outline.	thin, medium, thick, length

Exploring the CSS Properties

Border properties - example

This example uses a type selector to set border top and bottom properties for the <section> tag:

CODE

```
section {
    border-top:solid 1px #929CA4;
    border-bottom:solid 1px #DFE3E6;
}
```

If the properties are the same, you can use the generic border property instead:

CODE

```
section {
    border:solid 1px #929CA4;
}
```

Exploring the CSS Properties

List properties

Use the collection of list properties to set how you want the marker for a particular list item to appear. If you do not want a glyph to appear for the marker, you may also choose to completely hide the marker as well.

Exploring the CSS Properties

List properties (CSS 1 and CSS 2)

Property	Description	Values
`list-style`	Generic property that allows you to set the list style type, position, and image within a single style rule.	list-style-type, list-style-position, list-style-image
`list-style-image`	Sets the image that will be used as the list item marker.	url to the image, none
`list-style-position`	Specifies the position of the marker with respect to the box.	inside, outside
`list-style-type`	Specifies the appearance of the list item marker as long as no list style image is set.	disc, circle, square, decimal, decimal-leading-zero, lower-roman, upper-roman, lower-greek, lower-latin, upper-latin, armenian, georgian, lower-alpha, upper-alpha, none

Exploring the CSS Properties

List properties - example

This example uses a descendant selector to set various list properties for the <li> tag:

CODE

```
ul li {
    list-style-type:upper-roman;
    list-style-position:outside;
}
```

Exploring the CSS Properties

Positioning properties

Use the positioning properties to set how a box is positioned on the page. Several properties exist for creating multi-column layouts as well as stacked boxes. You can also use clip and overflow properties to set how content within a box is treated either by clipping off portions of the box, hiding excess content altogether, etc.

Exploring the CSS Properties

Positioning properties (CSS 1 and CSS 2)

Property	Description	Values
`position`	Sets the type of positioning to apply to an element.	static, relative, absolute, fixed
`visibility`	Specifies whether boxes are visible or hidden. Hidden elements will still take up space on the page. To hide an element completely, set the display property to none.	visible, hidden, collapse
`z-index`	Specifies the "stacking level" of an absolutely positioned element.	auto, numeric value
`overflow`	Specifies whether content of a box is clipped when content within it overflows the box's dimensions.	visible, hidden, scroll, auto
`left, right, top, bottom`	Specifies how far a box's margin edge is offset from the element's containing block.	length, percentage, auto
`clip`	Applies to elements that are absolutely positioned. Allows you to "clip off" or remove a portion of an element.	Rectangle (top, right, bottom, left), auto
`float`	Specifies how a box should float in relation to other elements.	left, right, none
`clear`	Specifies the sides of a box that may not be adjacent to an earlier floated box.	none, left, right, both
`display`	This powerful property allows you to change the default display characteristics of an element.	inline, block, list-item, inline-block, table …, none

Exploring the CSS Properties

Positioning properties - example

This example sets positioning properties for two tags. The two elements will appear side-by-side of one another:

CODE

```
div          {float:left;background-color:silver;}
div #myDiv1  {width:100px;height:100px;}
div #myDiv2  {width:50px;height:50px;;}
```

The result will appear as follows: