Intro to CSS(3)

Prerequisites

Intro to HTML

You should have completed the Intro to Internet & HTML before taking Intro to CSS.

What you'll learn

- History of CSS
- Anatomy of a style
- External, Internal, Embedded, Inline Styles
- Selectors: Tag Selectors, Class Selectors, ID Selectors, Group Selectors, Universal Selector, Descendant Selectors, Pseudo Classes & Pseudo Elements, Child Selectors, Adjacent Sibling Selector, Attribute Selectors
- New CSS3

What you'll learn in this module.

1. What is CSS?
2. CSS syntax

What you won’t learn

Every single CSS property/value since the Big Bang.

To get complete resources of all CSS(3) properties, use online resources
https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
http://www.tutorialspoint.com/css/css_references.htm

What is CSS?

- CSS: Cascading Style Sheets

Why is is “Cascading”, and not just “Style Sheets”?

CSS gives you creative control over the layout and design of your web pages.
A style = A rule describing how to format a particular portion of a web page
A style sheet = a set of these styles.
CSS makes HTML look good, but it is not the same as HTML. It is a different language altogether

Why are stylesheets called cascading?
The style sheet with the highest priority controls the content display. Declarations not set in the highest priority source are passed on to a source of lower priority, such as the user agent style. This process is called cascading.
http://en.wikipedia.org/wiki/Cascading_Stylesheets#Sources

What is CSS?

Why is it “Cascading”, and not just “Style Sheets”?

Why are stylesheets called cascading?
The style sheet with the highest priority controls the content display. Declarations not set in the highest priority source are passed on to a source of lower priority, such as the user agent style. This process is called cascading.
http://en.wikipedia.org/wiki/Cascading_Stylesheets#Sources

Plus, SS wouldn’t sound quite right ;)

History of CSS

The origins of CSS...

External stylesheets have existed since the SGML days in the 1980s.

As HTML evolved, designers gained more control over site appearance. But the resulting HTML code grew much more complex.

Since the early days of SGML, coders realized that the structure of a document (what it contained) should be separate from the presentation of a document (how it was viewed). The user should decide the ideal presentation, while the developer should decide what the presentation should contain.

For example, the user may pick one style of presentation for printing a document, another style for viewing a document on screen, and a third for editing a document.

In the 1990s, 9 different stylesheet languages were proposed. Of those nine, two were chosen as the foundation for what would later be known as CSS (Cascading Style Sheets).

One of the distinguishing features of CSS is that it could be split into multiple files. One stylesheet could cascade (or inherit) from another, allowing you to mix style preferences together. The site designer could have one set of preferences, then the user could “embrace-and-extend” that design as he saw fit.

History of CSS

- CSS1 (1996)
- CSS2 (1998)
- CSS3 (1998-?)
- CSS 2.1 (2004-2011)
- CSS4

- CSS 1, introduced in 1996, laid the groundwork for Cascading Style Sheets. The basic structure of a style, the selector concept, and most of the CSS properties were all in that very first version: fonts, colors, text alignment, etc.
- In 1998, CSS 2 added new features, including the ability to target your CSS to different printers, monitors, and other devices and the ability to position elements wherever you wanted.
- CSS 3 was first proposed in 1998 and it still under development. CSS 3 broke CSS 2 into over 50 separate modules. Only 4 have reached “recommendation” status and a few others are close. Although the W3C still has to finalize this standard over all, some web browsers are already adopting a few of its new guidelines and features. www.css3.info
- CSS 2.1 was proposed in 2004 to help address issues caused by bugs in CSS2 and browsers having only partial or incorrect implementations. It reached recommendation status in 2011.
- Since CSS3 broke CSS into separate modules, each module has been allowed to evolve independently. Some CSS3 modules have evolved to level 4, and some new modules have already been proposed. So while there isn’t a formal CSS4 project, the set of level 4 modules can be collectively referred to as CSS4.

And just for fun, here a complete reference to CSS4 (check the date before investing

too much time!)
http://webdesign.tutsplus.com/articles/css4-is-coming-what-you-need-to-know--webdesign-11521

HTML B.C. (Before CSS)

```
  <p>
  <strong>
    <font color="#0066FF" size="5" face="Verdana,Arial, Helvetica, sans-serif">
      Wyncode
    </font>
  </strong>
  </p>
  <p>
  <font color="#FF3300" size="4" face="Georgia,Times New Roman, Times,serif">
    <em><strong>Code School</strong></em>
  </font>
</p>
```

What's wrong with this markup?

HTML A.C. (After CSS)

```
<p>Wyncode Code School</p>
<p>Learning to code.</p>
```

Refactoring

Anatomy of a Style

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

A style/rule is made up of two elements:

- the web page element to format (the selector) and
- the formatting instructions (the declaration block). Between the opening and closing braces of a declaration block, you can add one or more declarations (formatting instructions).

Every declaration has two parts, a property and a value, and ends with a semicolon.

Anatomy of a Style

```
p { color: red; }
```

CSS offers a wide range of formatting options, called properties. A property is a word—or a few hyphenated words—indicating a certain style effect. Most properties have straightforward names like font-size, margin-top, and text-align. For example, the background-color property sets a background color.

Mini Quiz 1

```
h1 {
   font-style:italic;
}
```

Match the following

1. Selector
2. Property
3. Value

Mini Quiz 2

```
h 1 {
   font-style:italic
   text-decoration:underline
```

Find the error(s) in the code below

External Style Sheets

An external style sheet must have a name ending with **".css"**

Add external stylesheet to the `<head>`...
`</head>`

- An external style sheet collects all your style information in a single file that you then link to a web page with just a single line of code.
- You can attach the same external style sheet to every page in your website, providing a unified design. It also makes a complete site makeover as easy as editing a single text file.
- External style sheets help web pages load faster: When a web browser downloads an external style sheet, it stores the file in your visitor's computer cache. When your visitor hops to other pages on the site that use the same external style sheet, there's no need for the browser to download the style sheet again.

External Style Sheets

Option 1

```
<link rel="stylesheet" type="text/css"
href="css/global.css">
```

Option 2

```
<style type="text/css">
@import url(css/global.css);
</style>
```

Quiz: How can you add more than 1 external style sheet?

Internal Style Sheets

```
<style type="text/css">
h1 {
color: #FF7643;
}
</style>
</head>
<body>...
```

An internal style sheet is a collection of styles that's part of the web page's code. It always appears between opening and closing HTML <style> tags in the page's <head> portion.

Internal Stylesheets are also known as embedded stylesheets.

Internal Style Sheets

What are the pros and cons of internal stylesheets?

- Internal style sheets are easy to add to a web page and provide an immediate visual boost to your HTML.
- But they aren’t the most efficient method for designing an entire website composed of many web pages. For one thing, you need to copy and paste the internal style sheet into each page of your site—a time-consuming chore that adds bandwidth-hogging code to each page.
- Internal style sheets are even more of a hassle when you want to update the look of a site as every page's styles must be updated

CSS Comments

A CSS comment begins with "/*", and ends with "*/", like this:

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

Comments are used to explain your code, and may help you when you edit the source code at a later date. Comments are ignored by browsers.

Inline Styles

```
<h1 style="color: #C7AA8D;">
```

When you type a CSS rule directly into a page's HTML, you're creating an inline style.

Inline styles offer none of the time-and bandwidth-saving benefits of style sheets, so the pros hardly ever use them. If you absolutely must change the styling on a single element on a single page, then you may want to resort to an inline style. (For example, when creating HTML-formatted email messages, it's usually best to use inline styles. That's the only way to get CSS to work in Gmail, for instance.)

Stylesheet Order

What color is the paragraph?

```
<link href="blue.css" type="text/css" rel="
stylesheet"/>
<style>
p{color: red;}
</style>
<link href="green.css" type="text/css" rel="
stylesheet"/>
```

Style Order Matters!

Stylesheet order

blue.css:

```
p{color: blue;}
```

green.css

```
p{color: green;}
```

Edit the order of the internal & external stylesheets to demo the effect of the latest style being included being used.

Tag Selectors

- AKA element selectors
- apply to every occurrence of an HTML tag on a web page
- For example, when you want to format every paragraph of text on a page using the same font, color, and size, you merely create a style using `p` (as in the `<p>` tag) as the selector.
- Tag selectors bear the exact same name as the tag they style—`p`, `h1`, `table`, `img`, etc.

Quiz: What's a problem with tag selectors?

Class Selectors

```
.special {
  color:#FF0000;
  font-family:"Comic Sans MS", cursive;
}
<p class="special">I'm special!</p>
```

- Class selectors are applied to some elements only
- Example: you can create a class style named .warning and then apply it only to a paragraph containing warnings, without affecting any other paragraphs.
- Unlike tag selectors, which limit you to the existing HTML tags on the page, you can create as many class selectors as you like and put them anywhere you want.
- All class selector names must begin with a period.
- CSS permits only letters, numbers, hyphens, and underscores in class names.
- Class names are case-sensitive

Class Selectors

```
.special {
  color:#FF0000;
  font-family:"Comic Sans MS", cursive;
}
```

Quiz: Is the style being applied to this?

```
<p class="Special">I'm special!</p>
```

Nope. Class names are case-sensitive!

ID Selectors

```
#header {
  font-size: 36px;
}

<div id="header">I'm big!</div>
```

- Use ID selector for identifying a unique part of a page, like a banner, navigation bar, or the main content area.
- Just like a class selector, you create an ID by giving it a name in CSS, and then you apply it by adding the ID to your page's HTML code.
- You should apply only a single ID to a single HTML tag.

ID Selectors

```
#header {
  font-size: 36px;
}
```

Quiz: Is this style being applied here?

```
<div id="Header">I'm big?</div>
```

No.

Hint: Use lowercase for all class and ID selectors!

ID Selectors

Uh-oh...what happens if you do this?

```
<div id="header">I'm big!</div>
<div id="header">I'm big too!</div>
```

Even though ID selectors are *supposed* to be unique, your browser won't explode if you have several elements with the same id.

Class VS ID selectors

HTML	CSS
<div class=""xyz"> ... 	.xyz{...}
	#xyz{...}

http://en.wikiversity.org/wiki/Web_Design/CSS_Classes

Class VS ID Selectors

- To use a style several times on a page, you must use classes.
- Use IDs to identify sections that occur only once per page.

Consider using an ID selector to sidestep style conflicts, since web browsers give ID selectors priority over class selectors. For example, when a browser encounters two styles that apply to the same tag but specify different background colors, the ID's background color wins.

Group Selectors

```
h1, h2, h3, h4, h5, h6 { color: #2D958F }
```

Quiz: Can group selectors be used for multiple classes?

- Sometimes you need a quick way to apply the same formatting to several different elements.
- For example, maybe you'd like all the headers on a page to share the same color and font. Creating a separate style for each header—h1, h2, h3, h4, and so on—is way too much work. In that case, you can use group selectors:

Universal Selector

```
* {text-decoration:line-through;}
```

- CSS also gives you a sort of über group selector - the universal selector.
- An asterisk (*) is universal selector shorthand for selecting every single tag in a page.

Descendant Selectors

```
<body>
  <h1>
    Wyncode is <strong>Ruby</strong>
  </h1>
  <p>Wyncode is <strong>HTML</strong>.</p>
</body>
```

Descendent selectors let you format a tag based on its relationship to other tags.

- the <body> tag is an ancestor for all of the tags inside of it, the <h1>, <p>, and tags
- the <p> tag is a descendent of the <body> tag
- A parent is the closest ancestor. <p> is the parent of
- A child is a tag that's directly enclosed by another tag. <p> is a child of <body> but isn't.
- Tags that are children of the same tag are called siblings. <h1> and <p> are siblings.

Descendant selectors

In order to format the color of the strong text in headings only, use:

```
h1 strong {color: red;}
```

Quiz: Could you achieve the same result with classes/IDs? What are the pros and cons?

Descendant Selectors

You don’t have to describe the entire lineage of the tag you want to format.

```
<div>
  <ul>
    <li>
      <a href="#">Link</a>
    <li>
  </ul>
</div>
```

Valid descendant selectors to target the link “Link”:
div ul li a
ul li a
li a
div a
div ul a
div li a

Descendant selectors

```
p.intro a { color: yellow; }

<p class="intro">
  <a href="#">Link</a>
</p>
```

You can build complex descendent selectors combining different types of selectors.

Descendant selectors

```
p.intro a { color: yellow; }
p .intro a { color: yellow; }
```

Which one of these styles applies to this HTML:

```
<p>
  <div class="intro">
    <a href="#">Link</a>
  <div>
</p>
```

PLEASE NOTE:
The use of spaces is extremely important for descendant selectors.

Quiz: Can you replace the above with .intro a?

Pseudo Classes/Elements

selector:pseudo-class { ... }

Pseudo-classes are selectors that let you target elements themselves, or a special state of the element – like :hover, :first-child, :nth-of-type(n), :empty, :root, :lang, etc. **Pseudo-elements** on the other hand allow you to target a piece of an existing element, something which is not part of the document element tree.

Pseudo classes for links

Pseudo classes for links:

- a:link
- a:visited
- a:hover
- a:active

- a:link selects any link that your guest hasn't visited yet, while the mouse isn't hovering over or clicking it. This style is your regular, unused web link.
- a:visited is a link that your visitor has clicked before, according to the web browser's history.
- a:hover lets you change the look of a link as your visitor passes the mouse over it.
- a:active lets you determine how a link looks as your visitor clicks. In other words, it covers the brief nanosecond when someone's pressing the mouse button, before releasing it.

:focus

```
input:focus {
  background-color: #FFFFCC;
}
<body>
  <input type="text" />
</body>
```

- :focus applies when the visitor does something to indicate her attention to a web page element - usually by clicking or tabbing into it.
- When a visitor clicks in a text box on a web form, she puts the focus on that text box.

:first-letter, :first-line

- `:first-letter`
  - pseudo-element lets you create a drop cap
- `:first-line`
  - targets the first line

:first-letter, :first-line

```
p:first-letter{
  color:red;
}
<p>Lorem ipsum</p>
<p>Lorem ipsum</p>
```

Quiz: use the first-line pseudo class.

:before, :after

```
p:before {
    content: "A tip!";
}
<p>paragraph content</p>
<p>paragraph content</p>
<p>paragraph content</p>
<p>paragraph content</p>
```

:before, :after:
this pseudo-element lets you add content before or after a given element

:before Example

```
blockquote:before {
  display: block;
  content: "\201C";
  font-size: 80px;
  position: absolute;
  left: -20px;
  top: -20px;
}
```

http://www.webmaster-source.com/2012/04/24/pure-css-blockquote-styling/

:first-child

```
li:first-child{font-weight: bold;}

<ul>
  <li>First</li>
  <li>Second</li>
</ul>
```

The :first-child pseudoelement lets you select and format just the first of however many children an element may have

:after or ::after?

Which one should you use?

```
element:after  { style properties }
element::after { style properties }
```

In the CSS3 spec the authors wanted a clearer distinction between the two, so they decided that **pseudo-elements should start with a double colon**. But for the sake of compatibility browsers should also parse the single colon versions of the four pseudo-elements that existed before CSS3 – `::first-line`, `::first-letter`, `::before` and `::after`. Internet Explorer 8 and lower don't understand the double colon notation however, so most people just keep using a single ":"

https://developer.mozilla.org/en-US/docs/Web/CSS/::after
http://bricss.net/post/10768584657/know-your-lingo-pseudo-class-vs-pseudo-element

Child Selectors

```
body h1{ color: green;}
body > h1{color: red;}

<body>
  <h1>Descendant</h1>
  <div><h1>Child</h1></div>
</body>
```

Unlike a descendent selector, which applies to all descendents of a tag (children, grandchildren, and so on), the child selector lets you specify the first decendent, the immediate child.

For example, the selector body > h1 selects any <h1> tag that’s a immediate child of the <body> tag, not the grandchild.

Adjacent Sibling Selector

```
h1 + p {
font-style:italic;
}

<h1>Heading</h1>
<p>Adjacent Sibling Paragraph</p>
<p>Another Paragraph</p>
```

Sometimes you need to select a tag based not on its parent tag but on its surrounding siblings—the tags that share a common parent. A tag that appears immediately after another tag in HTML is called an adjacent sibling.

Attribute Selectors

```
<input type="text" />
<input type="checkbox" />

input[type="text"]{
  border: 1px green solid;
}
```

CSS provides a way to format a tag based on any attributes it has.

The Cascade

- Many times, you'll wonder why a page element looks the way it does
- Example: Multiple styles are applied to an element, thru internal and external style sheets - who wins?
- The Cascade governs which styles get applied to an element

Nearest Ancestor Wins

If a tag doesn't have a specific style applied to it, then, in the case of any conflicts from inherited properties, the nearest ancestor wins.

```
div{color:red;}
p{color:green;}
<div><p>What color?</p></div>
```

Conflicting Styles

- A tag selector is worth 1 point.
- A class selector is worth 10 points.
- An ID selector is worth 100 points.
- An inline style is worth 1,000 points.

```
p { color: red; }
.info { color: green; }
#head { color: pink; }

<p class="info" id="head">Welcome!</p>
```

Quiz: What color is applied to the text?

Conflicting Styles

```
p .link { color: blue; }
.link a { color: red; }

<p class="link">Visit <a class="link" href="#"
>website.</a></p>
```

Quiz:

- What color is applied to the link?
- What happens when styles are reversed?

Stylesheet Placement

The style appearing last wins.
p .link { color: blue; } - external stylesheet
.link a { color: red; } - external stylesheet

Visit <a class="link" href="#"
>website.

TIP: It’s best to list any external style sheets first, and then include any internal styles

!important

```
a { color: green !important; }
a { color: red; }

<a href="#">Visit website.</a>
```

- CSS provides a way of overruling specificity entirely.
- Simply insert !important after any property to shield it from specificity-based overrides.

Formatting text

- Formatting Text
  - font-family
  - color
  - font-size: px, em, keywords, percentages
  - spacing
  - text-align, text-indent

Box Model

- margin
- padding
- border
- content

Each HTML element is a “box” which consists of content, with margins, paddings, and borders.

The box model refers to how the dimensions of an element are calculated taking into account these properties.

To calculate the total width, you must add up all properties that apply (the margin does not add up to the width)
http://jsbin.com/tatuq/1/edit?html,css,output

Note: CSS3 allows you to define different methods for box sizing through the box-sizing property http://css-tricks.com/box-sizing/)

Floats

```
float:left;
float:right;
float: none;
```

HTML normally flows from the top of the browser window down to the bottom, one headline, paragraph, or block-level element on top of another.

The float property moves an element to either the left or right. In the process, content below the floated element moves up and wraps around the float

When a box is taken out of normal flow, all content that is still within normal flow will ignore it completely and not make space for it

Floats

Example:

```
div{
float:right;
width: 250px;
}
<aside>Sidebar</aside>
<p>Paragraph Text</p>
```

If you can imagine a floated element as a box that physically lifts itself off the ground, flies to the left or the right until it can go no further, and then plops itself back down on the ground, you've got the right idea (think Terran bases in StarCraft). The floated element blows right by your text, and then your text repositions itself to wrap around the floated stuff once it has "landed".

Floats always position themselves in relative to other floats.

Floats

Floated elements move to the left or right edge of their containing element; by default the browser window. However if you float an element that’s inside another tag then the float will go to the left or right edge of that container tag.

Example: Add a wrapper to the previous example that has a width of 500px and is centered.

Note: When you float block-level elements, you should also set the width property for that element.

clear

```
clear: left;
clear: right;
clear: both;
```

The clear property instructs an element to not wrap around a floated item. By clearing an element, you essentially force it to drop down below the floated item. Also, you can control which type of float (left or right) is cleared or force a style to simply ignore both types of floats.

New in CSS3

- border radius
- box shadow
- text shadow
- box sizing
- multiple backgrounds
- opacity
- gradients

Media Queries

```
@media screen and (min-width: 480px)
{
/*CSS rules that apply IF
**we are rendering on a screen
**and the window is at least 480px wide
*/
}
```

CSS3 media queries are logical expressions that evaluate the current values of media features in the user's browser. If the media query expression evaluates as TRUE, the contained CSS is applied.
CSS media queries determine width and show different amount of columns as width decreases
http://mediaqueri.es/
http://jsbin.com/tokel/1/edit?html,output

Twitter Bootstrap

CSS:
http://getbootstrap.com/css/

The following link adds the Bootstrap CSS::
<link href="http://getbootstrap.com/dist/css/bootstrap.css" rel="stylesheet" type="
text/css" />

http://jsbin.com/wifab/1/edit?html,output