CSS

Selectors & Properties

Anatomy of a Website

Your Content

+ **HTML**: Structure

+ **CSS**: Presentation

= Your Website

A website is a way to present your content to the world, using HTML and CSS to present that content & make it look good.

CSS: What is it?

CSS = Cascading Style Sheets

CSS is a "style sheet language" that lets you style the elements on your page.

CSS is embedded inside HTML, but it is not HTML itself.

CSS: What can it do?

text

color

size

position

Anatomy of CSS

CSS consists of "style rules". Each style rule consists of a "selector" and "declarations" of property-value pairs:

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

body {
  color: yellow;
  background-color: black;
}
```

CSS in HTML

CSS can be embedded in HTML in several ways. One way is to include all CSS in a `style` tag, usually inside the `head` tag:

```
<html>
<head>
<style>
body {
  color: yellow;
  background-color: black;
}
</style>
</head>
```

Selectors

The Selector

```
selector {
  property: values;
}
```

The selector is used to select which elements in the HTML page will be given the styles inside the curly braces.

Types of Selectors

1. element
2. id
3. class
4. position in document

Types of Selectors: element

```
p {
}
```

Selects all p elements in the entire document.

Types of Selectors: id

```
#header {
}
```

Selects any element with the id “header", e.g.

```
<p id="header"></p>
```

Element ids are unique, so that should only be one element.

The "#" is how you tell CSS "this is an id."

Types of Selectors: class

```
.warning {
  color: red;
}
```

Selects any element with the class name “warning”, e.g.

```
<p class="warning"></p>
```

Multiple elements can have the same class name.

The "." is how you tell CSS "this is a class name."

Types of Selectors: class

An element can have only 1 id but multiple classes, so you can take advantage of that for greater flexibility.

```
.intro {
  background-color: blue;
}
.desc {
  color: red;
}
```

```
<p class="desc intro">hi</span> -> hi
<p class="desc">hi</span> -> hi
<p class="intro">hi</span> -> hi
```

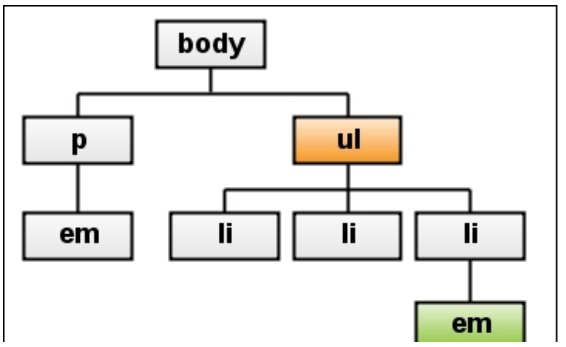
Types of selectors: position in document

```
ul em {
  color: yellow;
}
```

Selects any `em` element that's a descendant of a `ul` element.

The " " (space) is how you say "find a descendant."

```
<body>
 <ul>
  <li><em>Hi</em></li>
 </ul>
 <p><em>Bye</em></p>
</body>
```



Types of selectors: id + position

```
#related-brands li {
  color: gray;
}
```

Selects any <li> element that is a descendant of any element with an id that equals "related-brands."

```
<ul id="related-brands">
  <li>Rice Krispies
  <li>NutriGrain
</ul>
```

Tip: Try it out in the Selectoracle.

Types of selectors: element + class

```
li.special {
  color: yellow;
}
```

Selects any `li` element with a class attribute that contains the word "special".

```
<ul>
  <li>Rice Krispies
  <li class="special">NutriGrain
</ul>
```

Warning: If you place a space after the ".", it'll look for descendants of `li` with class of "special."

Types of selectors: ALL!

```
#header div.content form p em.required {
  color: white;
}
```

Selects any `em` element with a class attribute that contains the word "required" that is a descendant of a `p` element that is a descendant of a `form` element that is a descendant of a `div` element with a class attribute that contains the word "content" that is a descendant of any element with an id attribute that equals "header".

Types of selectors: pseudo classes

A set of "pseudo classes" can style anchor elements depending on their state.

```
a:link { /* unvisited link */
  color: red;
}
a:visited { /* visited link */
  color: blue;
}
a:hover { /* moused over link */
  color: green;
}
a:active { /* current link */
  color: purple;
}
a:focus {  /* focused link */
  color: purple;
}
```

Grouping selectors

You can group selectors to apply the same style to all of the selectors by separating them with commas:

```
a:hover, a:active:, a:focus {
  background-color: yellow;
}
```

The selector: Cascade rules

Generally:

- id is more specific than a class, class is more specific than element.
- the longer the selector, the more specific it is
- If style rules are equally specific, the last one wins!

Example:

```
#a #b h1 { color: red; } /* winner! */
#a h1 { color: blue; }
```

Properties

Property Value Pairs

Each property can have one or more comma separated values.

```
font: italic 12px sans-serif;
color: #333;
background-color: red;
```

Properties: color

The "color" property changes the text color. Colors can be specified either by name, for the most common colors, or by hexadecimal value.

```
color: red;
color: #ff0000;
color: rgb(255, 0, 0);
```

This property is inherited, which means it'll also be applied to all descendant elements but can be overridden by more specific rules. This rule makes all the body text red unless specified otherwise:

```
body {
  color: red;
}
```

Properties: background-color

The "background-color" property changes the background color. Besides the BODY element, all elements default to a transparent background.

```
background-color: black;
background-color: #000000;
background-color: rgb(0,0,0);

body {
  background-color: yellow;
}

table {
  background-color: #FFCC00;
}
```

Finding colors

To find colors or color schemes for your webpage, you can use:

- Desktop graphics programs: Adobe Photoshop, Mac Colors
- Online tools: RGBTool, Web Color Visualizer , Color Scheme Designer, ColourLovers, Kuler
- Browser extensions: Eye Dropper for Chrome, ColorZilla for FireFox

Property: font-family

The "font-family" property specifies the font family (or "font face") of the text. You can specify either a specific font name or a generic family name (serif, sans-serif, monospace, cursive).

```
font-family: "Times New Roman";
font-family: sans-serif;
```

A comma separated list of font families can be specified if you want the browser to prefer one but use the others as backup options.

```
font-family: "Times New Roman", serif;
font-family: "Arial", sans-serif;
font-family: Courier, monospace;
```

Finding fonts

Several websites exist that help you pick fonts for your website and to customize it with non-default fonts:

- TypeTester
- TypeKit
- FontFonter
- Good Web Fonts
- Typotheque

Property: font-size

The "font-size" property specifies the size of a font. It can be specified as a fixed size in various units, a percentage, or as a predefined keyword.

```
font-size: 1.5em;
font-size: 12px;
font-size: 100%;
font-size: larger;
```

Property: font-size (em)

The "em" unit lets you set the size of the text relative to the text around it. This makes the page resize nicely in proportion if the user changes their default font-size. The default size is "1em".

```
p {
  font-size: 0.9em;
}

strong {
  font-size: 1.5em;
}
```

Hello **there**!

Property: font-size (px)

The "px" unit lets you size font in terms of pixels, which is the unit also used to size images and other elements. It is easier to understand than em, but doesn't work as well when printing or resizing.

```
h2 {
  font-size: 17px;
}
```

Normal Headline

Resized Headline

Property: font-size (keywords)

There are various keywords that can be used if you're not as worried about the precise sizing: xx-small, x-small, small, medium, large, x-large, xx-large.

```
p.footnote {
  font-size: small;
}
```

There are also two relative keywords, that act similar to em's in setting size relative: smaller, larger.

```
p.intro {
  font-size: larger;
}
```

Property: font-size (%)

The size can also be specified as a percentage, which works similar to "ems", and can be used in conjunction with other units.

```
body {
  font-size: 12px;
}

h1 {
  font-size: 200%;
}

h1 a {
  font-size: 75%;
}
```

Header Link

Text!

Property: font-style

The "font-style" property specifies the font style of the text, either "normal" by default or "italic".

```
font-style: italic;
```

Italicized text.

Property: font-weight

The "font-weight" property specifies the thickness of the font. The default is "normal" and the typical override is "bold". You can also specify "bolder", "lighter", or a number from 100 to 900.

```
font-weight: bold;
```

Bold text!

"Shorthand" properties

A "shorthand" property in CSS lets you specify multiple properties in one property, for conciseness purposes. Instead of specifying each "font-" property separately, you can bundle them up in one "font" property.

```
table.geeky {
  font-weight: bold;
  font-style: italic;
  font-size: 10px;
  font-family: sans-serif;
}
```

Those four rules can be written as:

```
table {
  font: italic bold 10px sans-serif;
}
```

The "font" property expects values ordered as font-style, font-variant, font-weight, font-size, and font-family, and any of the first three can be left off.

Other text properties

```
line-height: 10px;
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis aliquam convallis ligula ut suscipit. Phasellus hendrerit, enim scelerisque...

```
text-align: center;
```

Centered text.

```
text-decoration: underline;
```

Underlined text

```
text-indent: 20px;
```

Indented text.

CSS Style

Comments

Comments will be ignored by the browser and are useful for documenting your styles to other humans or commenting out rules.

```
/* Comment here */
p
{
  margin: 1em; /* Comment here */
  padding: 2em;
  /* color: white; */
  background-color: blue;
}

/*
  multi-line
  comment here
*/
```

CSS files

CSS can also be defined in a separate file, and imported using either the `link` element or the `@import` statement in CSS.

Linked stylesheet:

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

Imported stylesheet:

```
<style>
@import url("main.css")
</style>
```

Inline CSS

CSS styles can be assigned to individual HTML elements via the `style` attribute:

```
<tag style="property1: value; property2: value;"> </tag>
```

```
<h1 style="color:red; text-decoration: underline;">Hello</h1>
```

Inline styles will have precedence over all other styles. Inline CSS should be avoided as it increases maintenance cost and blurs line between presentation and content.

Coding Conventions

It is possible to write CSS like this:

```
selector{property:values}selector{property:values}
```

But it's preferred to write it like this:

```
selector {
  property: values;
}

selector {
  property: values;
}
```

Notice: space between "selector" and "{", 2 spaces before property-value pair, space after "property:", semi-colon after "values", line between rules.

Naming Conventions

Some rules to follow when making IDs and class names:

- Describe the content, not the presentation ("warning", not "redbox").
- Use all lowercase, and hyphens when needed for readability ("header-info", not "headerInfo").
- Use hyphens to show that a class or ID is part of something else. (e.g. "footer", "footer-copyright", and "footer-logo").