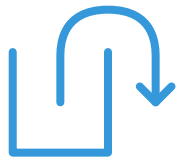

CSS

Cascading Style Sheets

Exegetic Analytics
2019-09-08

What is CSS?

> CSS is what makes the internet look good.

HTML

- HTML = HyperText Markup Language
- What is the content of the page?

CSS

- CSS = Cascading Style Sheets
- How should the contents appear?

Inline Style

```
<html>
  <body>
    <!-- Nobody does this anymore! -->
    <p style="font-weight: bold;">I'm bold!</p>
  </body>
</html>
```

Embedded Styles

```
<html>
  <head>
    <style>
      /* Styles go here! */
    </style>
  </head>
</html>
```

Linked Styles

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

Structure of CSS

CSS is simply a set of rules.

Each rule has the following structure:

```
<selectors> {
  <declarations>
}
```

Selectors

Selectors are used to target specific portions of the document.

"To which elements does this apply?"

Declarations

The declaration block contains one or more declarations, each of which has the form:

```
<property>: <value>;
```

"What must be done with the selected elements?"

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

The *selector* is `p`. This tells us what portion of the document the rule will be applied to.

The *property* is `font-color` and the *value* is `red`.

The effect of this rule is that all paragraph content (within `p` tags) will be rendered as red text.

The Selector Zoo

Tags

CSS

```css
p {
  color: red;
}
strong {
  background-color: yellow;
}
```

HTML

```html
<p>
  All paragraph text will be red. Bold text
  will also have a <strong>yellow</strong>
  background.
</p>
```

Identifier

An identifier rule begins with a "#" and targets one specific element in the document.

CSS

```css
#introduction {
  font-style: italic;
}
```

HTML

```html
<div id="introduction">
  This is the introduction.
</div>
<div>
  This is not the introduction!
</div>
```

An identifier is unique within a document.

Class

A class rule begins with a ".".

CSS

```css
.big {
  font-size: 2rem;
}
.huge {
  font-size: 4rem;
}
```

HTML

```html
<p>
  This is normal text.
</p>
<p class="big">
  This is big text.
</p>
<p class="huge">
  This is HUGE text!
</p>
```

A class rule can be applied to multiple elements in a document.

Grouping

Multiple selectors can be grouped together, separated by a ",".

CSS

```css
h1, h2 {
  font-style: italic;
}
```

HTML

```html
<h1>Chapter Heading</h1>
<h2>First Section</h2>
<h2>Second Section</h2>
```

The rules will be applied to *all* of the selectors listed in the group.

Descendants

Descendant selectors identify tags which are descendants (but not necessarily *direct* descendants!) in the document tree.

CSS

```
.outer p {
}
```

HTML

```
<div class="outer">
  <p>Some paragraph text.</p>
  <div class="inner">
    <p>Some more paragraph text.</p>
  </div>
</div>
```

Children

Child selectors identify tags which are *direct* descendants of a parent tag.

CSS

```
.outer > p {
}
```

HTML

```
<div class="outer">
  <p>Some paragraph text.</p>
  <div class="inner">
    <p>Some more paragraph text.</p>
  </div>
</div>
```

Adjacent Sibling

Adjacent sibling selectors identify tags which are at the same level in the document tree and adjacent to each other.

CSS

```
.big + p {
}
```

HTML

```
<p class="big">
  This is big text.
</p>
<p>Some paragraph text.</p>
<p>Some more paragraph text.</p>
```

Attribute

Attribute selectors identify tags which have specific attribute values.

CSS

```
img[width="50%"] {
}
```

HTML

```
<img width="50%">
<img width="100%">
```

Attribute Pattern

You can also match on attribute patterns.

CSS

```css
/* "begins with" */
a[href^="mailto://"] {
}
```

```css
/* "ends with"  */
a[href$=".pdf"] {
}
```

```css
/* "contains"   */
img[src*="logo"] {
}
```

HTML

```html
<a href="http://">
<a href="mailto://">
<a href="ftp://">
```

```html
<a href="logo.png">
<a href="report.pdf">
<a href="report.docx">
```

```html
<img src="avatar.png">
<img src="company-logo.png">
<img src="banner.jpg">
```

Pseudo-Classes

First, Last & Only Child

CSS

```css
p:first-child {
  /* */
  /* */
  /* */
}
```

```css
p:last-child {
  /* */
  /* */
  /* */
}
```

```css
p:only-child {
  /* */
}
```

HTML

```html
<div>
  <p></p>
  <p></p>
  <p></p>
</div>
```

```html
<div>
  <p></p>
  <p></p>
  <p></p>
</div>
```

```html
<div>
  <p></p>
</div>
```

nth Child

The `:nth-child(n)` selector matches the `n`th child element, regardless of type.

CSS

```
li:nth-child(2) {
  /* */
  /* */
  /* */
}
```

```
li:nth-child(odd) {}
li:nth-child(even) {}
```

```
li:nth-child(3n) {}
li:nth-child(2n+1) {}
```

HTML

```
<ul>
  <li></li>
  <li></li>
  <li></li>
</ul>
```

`n` can be:

- a number
- a keyword or
- a formula.

Formula is of form `an + b` with `n` starting at 0.

nth of Type

The `:nth-of-type(n)` selector matches the `n`th element of a specific type (with the same parent!).

CSS

```css
table:nth-of-type(2) {
  /* */
  /* */
  /* */
  /* */
}
```

HTML

```html
<p></p>
<table></table>
<p></p>
<table></table>
<p></p>
<table></table>
```

Let the Styling Commence!