CASCADING STYLE SHEETS

ANDREAS DRANIDIS

CSS - CASCADING STYLE SHEETS

- One of the three core languages of the Web (HTML, CSS, Javascript)
- Invented in 1996 by W3C - became popular after 2000. Latest version: CSS3
- CSS is the presentation language of the Web
- The ‘Driving’ Problem:
  - What font type and size does <h1>Heading</h1> generate?
    Answer: The default format from the browser
  - Early HTML - Override defaults with attributes.
    <table border="2" bordercolor="black">
  - Style sheets were added to address these issues.
    Specify style to use rather than browser default.
    Not have to code styling on every element.

CSS TO THE RESCUE

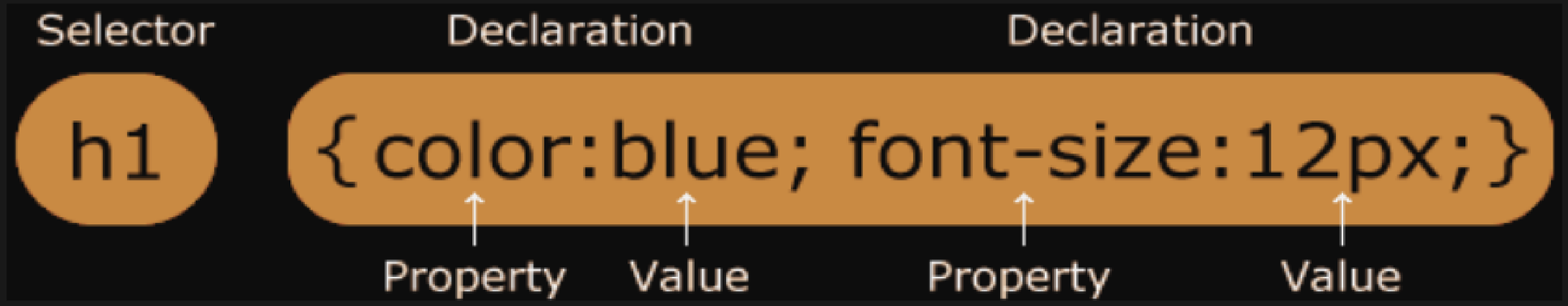
- **Key concept:** *Separate style from content*
- Content is in HTML files
- Formatting information in .css code
- Link a presentation styling with element(s)
  Selectors such as HTML tags, element ids, or through a class
- Define once for multiple pages
  DRY principle (Don’t Repeat Yourself)

PROS AND CONS OF CSS

- Reusability and time saving
- Consistency in design and easy maintenance
- Bandwidth reduction - faster loading time
- Search engine friendly
- Respond to varying screen sizes - multiple device compatibility

- Compatibility - Cross-browser issues
- Complexity and confusion for beginners

ANATOMY OF A RULESET



- The selector points to the HTML element you want to style.
- The declaration section assigns rules to the selector. It can have one or more rules separated by semicolons.
- Each declaration has 'a property and value' pair, separated by a colon. A declaration ends with semicolon and the whole block is surrounded by curly brackets.

CSS PROPERTIES

There exists so many properties.

MDN Web Docs - Most common CSS reference
MDN Web Docs - Complete CSS reference

WHERE TO ADD CSS

- **External** - having the CSS in a separate file. The most common and most useful method

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

- **Embedded** - within an HTML document by placing CSS inside a <style> element contained inside the HTML <head>. Less efficient than external.

```
<head>
  <style>
    h1 {
      color: blue;
      background-color: yellow;
    }
    </style>
</head>
```

- **Inline** - within a ‘style’ attribute of a single HTML element. Avoid it.

```
<h1 style="color: blue; background-color: yellow;">Hey</h1>
```

SIMPLE CSS SELECTORS

- **Type selector:** (also called tag, name or element). It selects an HTML tag/element.

```
span {
  background-color: yellow;
}
```

- **Class selector:** starts with a full stop (.) and selects everything in the HTML with that class applied to it.

```
.highlight {
  background-color: yellow;
}
```

- **ID selector:** starts with a # and targets only one element - used once per page. Use a little as possible.

```
#vote-tick {
  background-color: yellow;
}
```

- Selectors can be combined with comma

ADVANCED CSS SELECTORS

- **Combinator selectors** - select elements based on a specific relationship between them

```
/* descendant */
div p {
  background-color: yellow;
}
/* child */
div > p {
  background-color: yellow;
}
```

- **Pseudo-class selectors** - select elements based on a certain state

```
/* mouse over link */
a:hover {
  color: red;
}
```

- **Pseudo-elements selectors** - select and style a part of an element

```
p::first-line {
  color: red;
  font-variant: small-caps;
}
```

- **Attribute selectors** - select elements based on an attribute or attribute value

```
a[target] {
  background-color: yellow;
}
```

CSS BOX MODEL

Each HTML element consists of four elements:

Content

The content of the box, where text and images appear

Padding

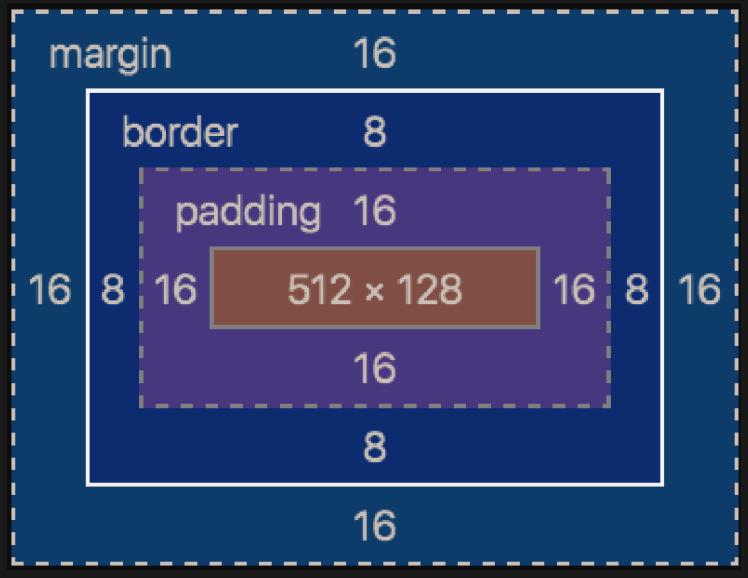
Clears an area around the content. The padding is transparent

Border

A border that goes around the padding and content

Margin

Clears an area outside the border. The margin is transparent



CSS SPECIFICITY

If there are two or more CSS rules that point to the same element, the selector with the highest specificity value will "win", and its style declaration will be applied to that HTML element.

Think of specificity as a score/rank that determines which style declaration are ultimately applied to an element.

```
#demo {color: blue;}
.test {color: green;}
p {color: red;}

<p id="demo" class="test" style="color: pink;">Hello World!</p>
```

Calculating specificity: Start at 0, add 100 for each ID value, add 10 for each class value (or pseudo-class or attribute selector), add 1 for each element selector or pseudo-element.

Note: Inline style gets a specificity value of 1000, and is always given the highest priority!
Note 2: There is one exception to this rule: if you use the !important rule, it will even override inline styles!

A: h1 color: red;
B: h1#content color: orange;
C: <h1 id="content" style="color: pink;">Heading</h1>

CSS FRAMEWORKS

Bootstrap and friends

In essence, a CSS framework comprises several CSS stylesheets ready for use by web developers and designers. The stylesheets are prepped for use for standard web design functions: setting colors, layout, fonts, navbars, etc.

But is Bootstrap necessary or even suggested for beginners?

1. Make sure you really know CSS
2. Learn Bootstrap
3. Study Bootstrap code, you'll learn some layout foundations and there are actually a lot of interesting tricks
4. Use your own CSS (perhaps combined with a bit of Bootstrap)

AND ALWAYS REMEMBER...