Cascading Style Sheets (CSS)

Sridhar Alagar

Motivation

What font and size does <h2>Motivation</h2> generate?

Some default from the browser

Can override defaults with attributes

<h2>Motivation<h2>

Style Sheets were added to:

Specify style to use rather than browser default

Avoid coding style for every element

Separate content from format/presentation

Separate content from style

Content to display in html file

Format (how) to display is in a separate CSS file

Link using an element attribute

<h2 class="head2">Motivation<h2>

Style for all <h2> elements can be defined once

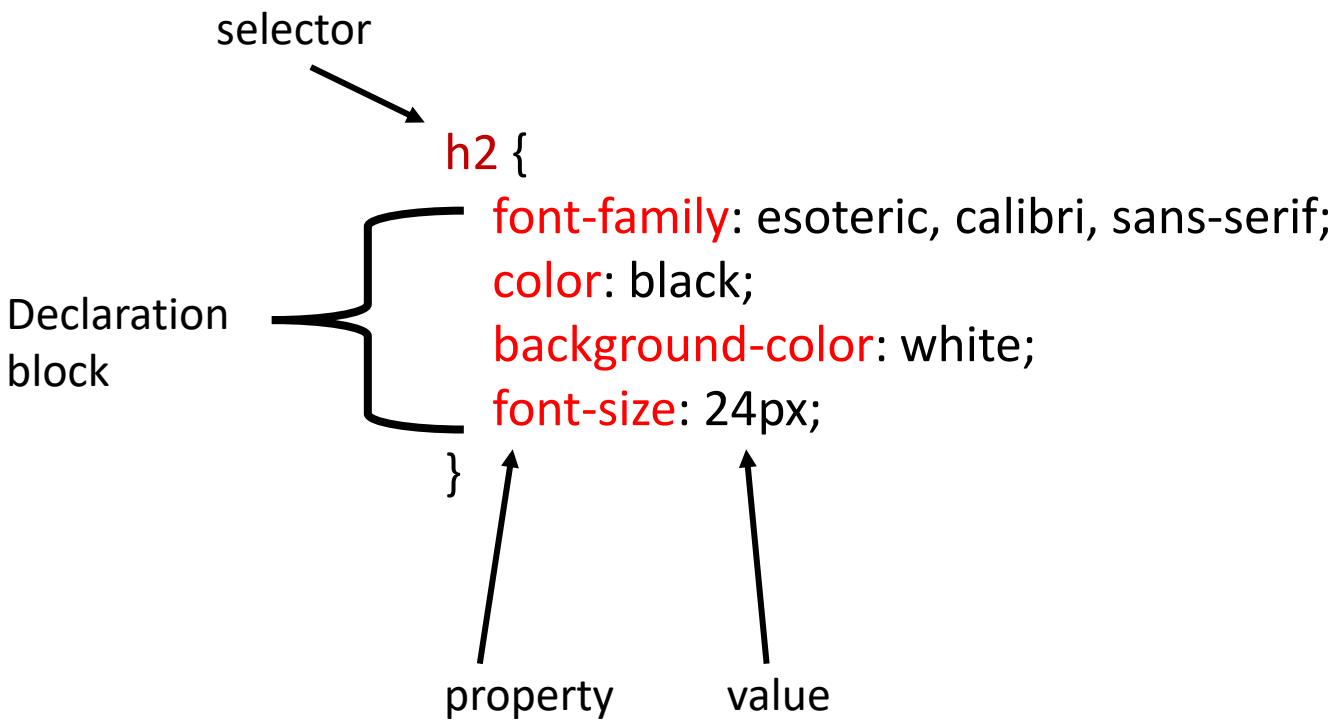
Want to change color or size of all <h2>?

Easy…change at one place

Adding style to HTML

```
<head>
<link href="filename" type="text/css" rel="stylesheet" />
<style type="text/css">
      body {
            font-family: sans-serif; color: red;
      }
</style>
</head>
<body>
      <div style="color: black;…">
</body>
```

CSS file consists of several rules



Selector type – universal

```
* {
    border: 2px solid black;
}
```

Selects every element in the document

Selector – tag name

```
<p>I am big and blue</p>
```

```
p {
   font-family: calibri, sans-serif;
   color: blue;
   font-size: 24px;
}
```

Selector – class

```
<p class="shade">Shade me</p>
```

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

```
<h4 class="shade">Shade me too</p>
```

Selector – tag and class

```
h4.shade {
    font-size: 18px;
}
```

```
<p class="shade">Shade me</p>

<h4 class="shade">Shade me too</p>
```

Selector – id attribute

```
#unique {
    color: pink;
    font-size: 24px;
}
```

```
<p id=“unique”>I am pink<\p>
```

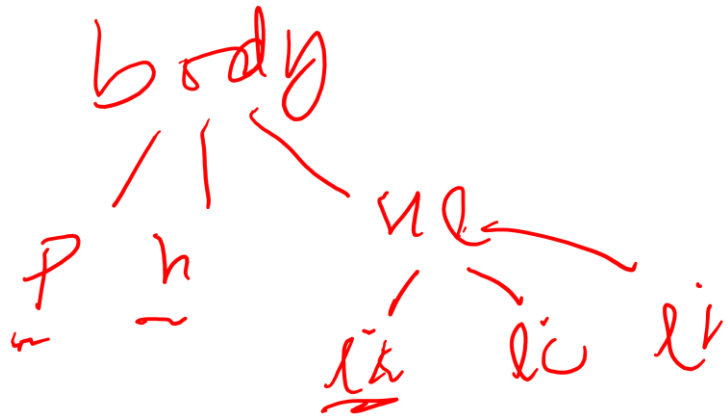
Selector – [attribute]

```
*[name] {
  color: 24px;
}

h2[name] {
  font-size: 24px;
}
```

```
<h2  name="foo">Foo<\p>
<p  name="foo">I am foo<\p>
```

Combinators



Combines two selectors

Allows to select specific element(s) based on the document hierarchy

Relationship:

descendants

children

siblings

Combinators - Descendant

```
<div class="div"><p>Text in .div</p></div>
<p>Text not in .div</p>
```

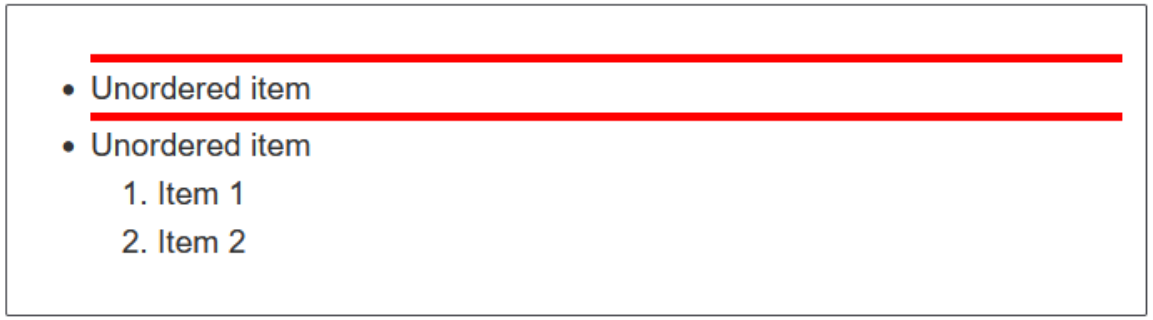
```
.div p {
    color: red;
}
```

Text in .div
Text not in .div

Combinators - Child

```
<ul>
  <li>Unordered item</li>
  <li>Unordered item
    <ol>
      <li>Item 1</li>
      <li>Item 2</li>
    </ol>
  </li>
</ul>
```

```
ul>li {
 border-top: 5px solid red;
}
```



Combinators – sibling, adjacent sibling

```
<div>
  <p>I am paragraph inside a div</p>
</div>
<p>I am a paragraph outside a div</p>
<h3>I am a level three heading</h3>
<p>I am a paragraph outside a div</p>
```

```
div ~ p {
/* selects all <p> siblings of div */
 color: green;
}

div + p {
/* selects  <p> immediately after div */
 color: green;
}
```

Pseudo class Selectors

Selects element that are in a specific state

hover – apply rule when mouse is over element

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

a:link, a:visited – apply rule when link is visited or not visited

```
a:visited {
    color: green;
}
```

```
a:link {
    color: blue;
}
```

Pseudo class Selectors for position

Selects child element(s) based on position in their parent container

li:first-child – selects when <li> is the first child

li:last-child – selects when <li> is the last child

li:nth-child(i) – selects when <li> is the i^{th} child

p:first-of-type – selects the first <p> inside its parent

p:last-child – selects the last <p> inside its parent

li:nth-child(i) – selects the i^{th} <p> inside its parent

Properties

Many properties control the style of an element

Some frequently used properties are related to

- Color
- Size
- Font
- Position
- Visibility

There are many more…

Color Properties: color & background-color

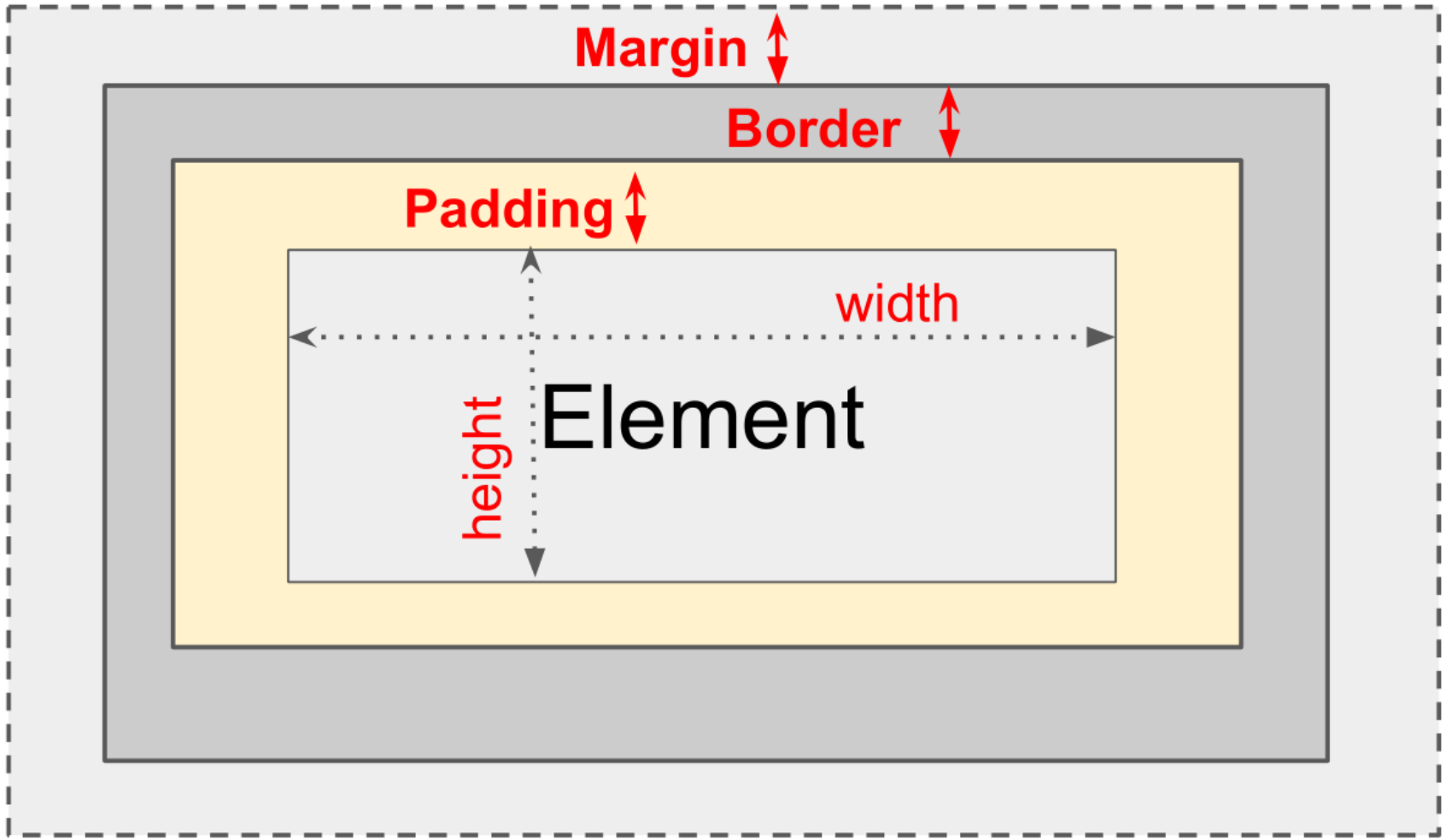
Any color is a combination of RGB of varying intensities (0 to 255)

- Pre-defined colors: red, blue, green, pink, etc. (140 standard names)
- 8-bit hex number for RGB: #ff0000 → RED
- 0 to 255 to specify intensities: rgb(255, 165, 0) →ORANGE
- Percentage intensities: rgb(0,50%,50%) → TEAL

```
h4 {
   color: #000;
   background-color: rgb(255, 165, 0);
}
```

Heading 4

Box Model – everything in CSS has a box around it



Total element width =
width +
left padding +
right padding +
left border +
right border +
left margin +
right margin

Margin & Padding
Transparent

Size units

Absolute	
2px	pixels
1mm	millimeters
2cm	centimeters
0.2in	inches
3pt	printer point 1/72 inch
Relative	
2em	2 times the element's current font size
3rem	3 times the root element's current font size

Size Properties: Element, padding, border, margin

width - Override element defaults
height

padding-top
padding-right
padding-bottom
padding-left

margin-top
margin-right
margin-bottom
margin-left

```
border-bottom-color
border-bottom-style
border-bottom-width
border-left-color
border-left-style
border-left-width
border-right-color
border-right-style
border-right-width
```
etc.

padding

```
.box{
    border: 5px solid red;
}
```

Font properties

Many properties related to font

font-family – font to use

font-size – size of the font

font-style – font face(italics, oblique, etc.)

font-weight – boldness of the font

```
p{
   color: teal;
   background: color;
   border: 1px solid black;
   font-family: Arial, Helvetica, sans-serif;
   font-size: 20px;
   font-style: oblique;
   font-weight: bold;

}
```

Check the font

Additional common properties

background-image: image for elements background

background-repeat: images repeated or shown only once based on value

Alignment properties – text-align, vertical-align, align-item

text-decoration: underline, overline, line-through

cursor: set the cursor when over the element (e.g., crosshair )

Basic document flow

I am a basic block level element. My adjacent block level elements sit on new lines below me.

By default we span 100% of the width of our parent element, and we are as tall as our child content. Our total width and height is our content + padding + border width/height.

We are separated by our margins. Because of margin collapsing, we are separated by the size of one of our margins, not both.

Inline elements like this one and this one sit on the same line along with adjacent text nodes, if there is space on the same line. Overflowing inline elements will wrap onto a new line if possible (like this one containing text), or just go on to a new line if not, much like this image will do:

display property – controls element visibility

display: none - element is not displayed and doesn't take up space

display: block - element is treated as block element
display: inline - element is treated as inline element
display: inline-block – doesn't break into a new line, but behaves like block
display: flex – element is treated as flex container
display: grid - element is treated as grid container

visibility: hidden - element is not displayed and takes up space

Flexbox and grid layout

display: flex – element is treated as flex container

- Layout in one dimension (row/column)
- Items flex to fill additional space and shrink to fit smaller space
- Very useful for web app layout
  - Space divided equally among all element
  - Align different sizes
  - Very useful for handling different window and display sizes

display: grid - element is treated as grid container

- layout in rows and columns

Positioning

position:

- Choose the position of the element

position: absolute

- Positioned on an absolute location within the containing element

position: relative

- Positioned relative to its normal position

Some CSS issues

Inheritance

- Some properties are inherited, e.g., font-size
- Some (borders, background) are not inherited

Specificity

When multiple rules match, specific rule wins

```
<span>Text1</span>                    span.test { color: green }
<span class="test">Text2</span>       span { color: red }
```

Sources

1. MDN - Learn to style HTML using CSS
2. Fundamentals of Web Development. 3rd Edition by Randy Connolly and Ricardo Hoar.
3. Stanford CS 142 Web Application – Lectures
4. CSS selectors cheat sheet for beginners