Unit 4:
Cascading Style Sheets (CSS)

By:Padmavati Sarode

Motivation

- HTML markup can be used to represent
  - Semantics: h1 means that an element is a top-level heading
  - What are Semantic Elements? **A semantic element clearly describes its meaning to both the browser and the developer**. Examples of non-semantic elements: and - Tells nothing about its content. Examples of semantic elements: <form> , <table> , and <article> - Clearly defines its content.
  - Semantic HTML or semantic markup is **HTML that introduces meaning to the web page rather than just presentation**. For example, a <p> tag indicates that the enclosed text is a paragraph. This is both semantic and presentational because people know what paragraphs are, and browsers know how to display them.

Style Sheet Languages

- Cascading Style Sheets (CSS)
- Applies to (X)HTML as well as XML documents in general XML stands for eXtensible Markup Language
- XML is a markup language much like HTML
- XML was designed to store and transport data
- XML was designed to be self-descriptive
- XML is a W3C Recommendation
- Extensible Stylesheet Language (XSL)
  - Often used to transform one XML document to another form, but can also add style

CSS Introduction

- A styled HTML document

produced by the style sheet `style1.css`:

```
body  { background-color:lime }

p     { font-size:x-large; background-color:yellow }
```

CSS Introduction

```
<!DOCTYPE html
        PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>
      CSSHelloWorld.html
    </title>
    <link rel="stylesheet" type="text/css" href="style1.css"
          title="Style 1" />
    <link rel="alternate stylesheet" type="text/css" href="style2.css"
          title="Style 2" />
  </head>
  <body>
    <p>
      Hello World!
    </p>
  </body>
</html>
```

CSS Introduction

```
<!DOCTYPE html
        PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>
      CSSHelloWorld.html
    </title>
    <link rel="stylesheet" type="text/css" href="style1.css"
          title="Style 1" />
    <link rel="alternate stylesheet" type="text/css" href="style2.css"
          title="Style 2" />
  </head>
  <body>
    <p>
      Hello World!
    </p>
  </body>
</html>
```

Types of CSS (Cascading Style Sheet)

- Cascading Style Sheet(CSS) is used to set the style in web pages that contain HTML elements. It sets the background color, font-size, font-family, color, … etc property of elements on a web page.
  There are three types of CSS which are given below:

- Inline CSS
- Internal or Embedded CSS
- External CSS
- **Inline CSS:** Inline CSS contains the CSS property in the body section attached with element is known as inline CSS. This kind of style is specified within an HTML tag using the style attribute.

Example of Inline CSS

```
<!DOCTYPE html>
<html>
    <head>
        <title>Inline CSS</title>
    </head>

    <body>
        <p style = "color:#009900; font-size:50px;
                font-style:italic; text-align:center;">
            GeeksForGeeks
        </p>

    </body>
</html>
```

Internal or Embedded CSS:

This can be used when a single HTML document must be styled uniquely. The CSS rule set should be within the HTML file in the head section i.e the CSS is embedded within the HTML file.

```
<!DOCTYPE html>
<html>
   <head>
      <title>Internal CSS</title>
      <style>
         .main {
            text-align:center;
         }
         .GFG {
            color:#009900;
            font-size:50px;
            font-weight:bold;
         }
         .geeks {
            font-style:bold;
            font-size:20px;
         }
```

```
        </style>
    </head>
    <body>
        <div class = "main">
            <div class ="GFG">GeeksForGeeks</div>

            <div class ="geeks">
                A computer science portal for geeks
            </div>
        </div>
    </body>
</html>
```

External CSS: External CSS contains separate CSS file which contains only style property with the help of tag attributes (For example class, id, heading, … etc). CSS property written in a separate file with .css extension and should be linked to the HTML document using **link** tag. This means that for each element, style can be set only once and that will be applied across web pages.

Example: The file given below contains CSS property. This file save with .css extension. For Ex: **geeks.css**

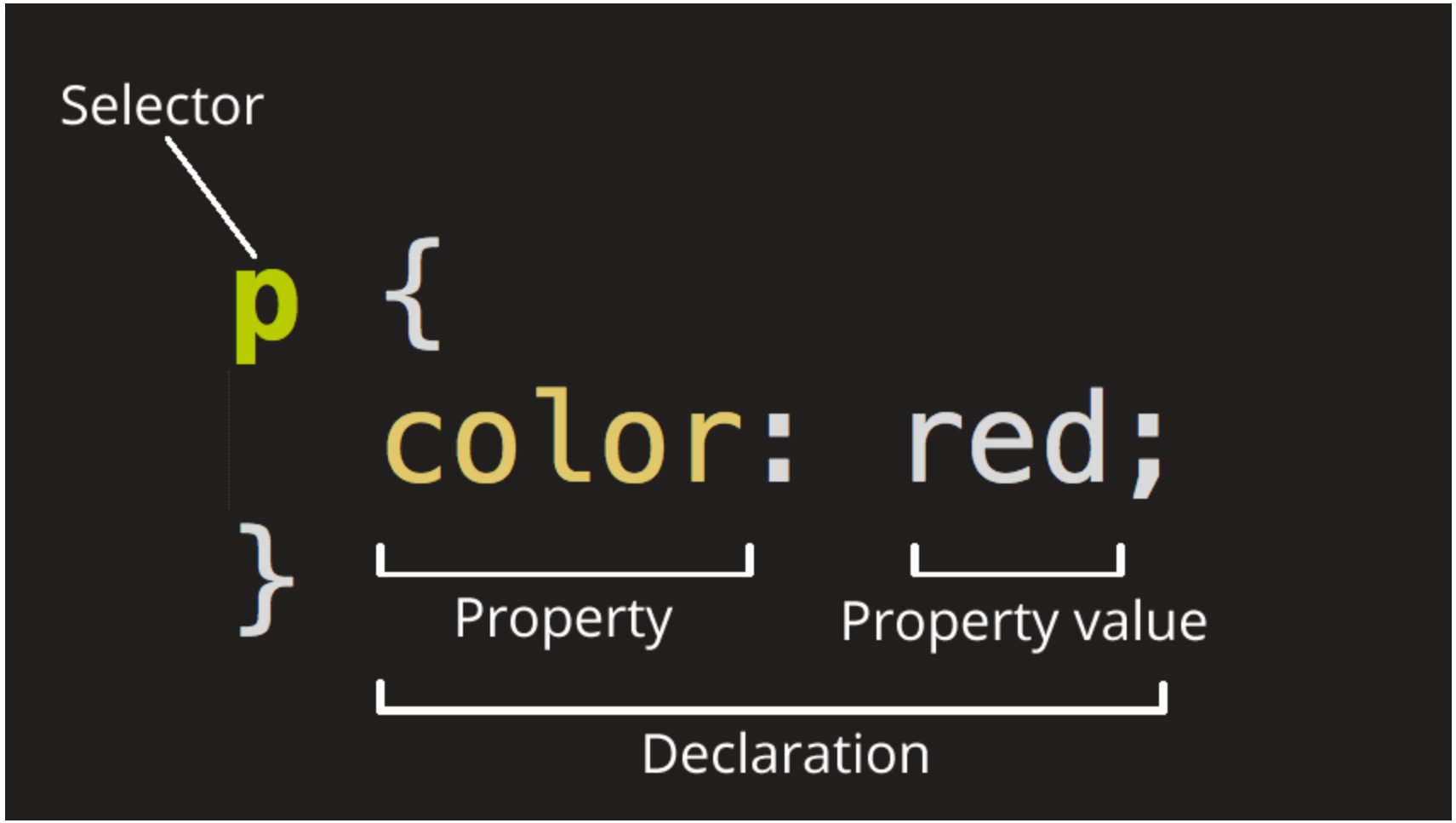
```
body {
background-color:powderblue;
}
.main {
 text-align:center;
}
.GFG {
color:#009900; font-size:50px; font-weight:bold;
 }
#geeks {
font-style:bold; font-size:20px;
 }
```

Properties of CSS: Inline CSS has the highest priority, then comes Internal/Embedded followed by External CSS which has the least priority. Multiple style sheets can be defined on one page. If for an HTML tag, styles are defined in multiple style sheets then the below order will be followed.

- As Inline has the highest priority, any styles that are defined in the internal and external style sheets are overridden by Inline styles.
- Internal or Embedded stands second in the priority list and overrides the styles in the external style sheet.
- External style sheets have the least priority. If there are no styles defined either in inline or internal style sheet then external style sheet rules are applied for the HTML tags.

CSS Properties

Anatomy of a CSS ruleset



The whole structure is called a ruleset. (The term ruleset is often referred to as just rule.) Note the names of the individual parts:

Selector

This is the HTML element name at the start of the ruleset. It defines the element(s) to be styled (in this example, <p> elements). To style a different element, change the selector.

Declaration

This is a single rule like color: red;. It specifies which of the element's properties you want to style.

Properties

These are ways in which you can style an HTML element. (In this example, color is a property of the <p> elements.) In CSS, you choose which properties you want to affect in the rule.

Property value

To the right of the property—after the colon—there is the **property value**. This chooses one out of many possible appearances for a given property. (For example, there are many color values in addition to red.)

Note the other important parts of the syntax:

Apart from the selector, each ruleset must be wrapped in curly braces. ({})
Within each declaration, you must use a colon (:) to separate the property from its value or values.
Within each ruleset, you must use a semicolon (;) to separate each declaration from the next one.
To modify multiple property values in one ruleset, write them separated by semicolons, like this:

```
p {
  color: red;
  width: 500px;
  border: 1px solid black;
}
```

Selecting multiple elements

You can also select multiple elements and apply a single ruleset to all of them. Separate multiple selectors by commas. For example:

```
p,
li,
h1 {
  color: red;
}
```

Different types of selectors

There are many different types of selectors. The examples above use **element selectors**, which select all elements of a given type. But we can make more specific selections as well. Here are some of the more common types of selectors:

Selector name	What does it select	Example
Element selector (sometimes called a tag or type selector)	All HTML elements of the specified type.	p selects <p>
ID selector	The element on the page with the specified ID. On a given HTML page, each id value should be unique.	#my-id selects <p id="my-id"> or <a id="my-id">
Class selector	The element(s) on the page with the specified class. Multiple instances of the same class can appear on a page.	.my-class selects <p class="my-class"> and <a class="my-class">
Attribute selector	The element(s) on the page with the specified attribute.	img[src] selects <img src="myimage.png"> but not <img>
Pseudo-class selector	The specified element(s), but only when in the specified state. (For example, when a cursor hovers over a link.)	a:hover selects <a>, but only when the mouse pointer is hovering over the link.

Class and ID selectors in CSS

In CSS, **class** and **ID selectors** are used to identify various HTML elements. The main benefit of setting class or ID is that you can present the same HTML element differently, depending on its class or ID.

Class selector

The **class selector** selects elements with a specific class attribute. It matches all the HTML elements based on the contents of their class attribute. The . symbol, along with the class name, is used to select the desired class.

Syntax

```
.class-name {
  /* Define properties here */
}
```

ID selector

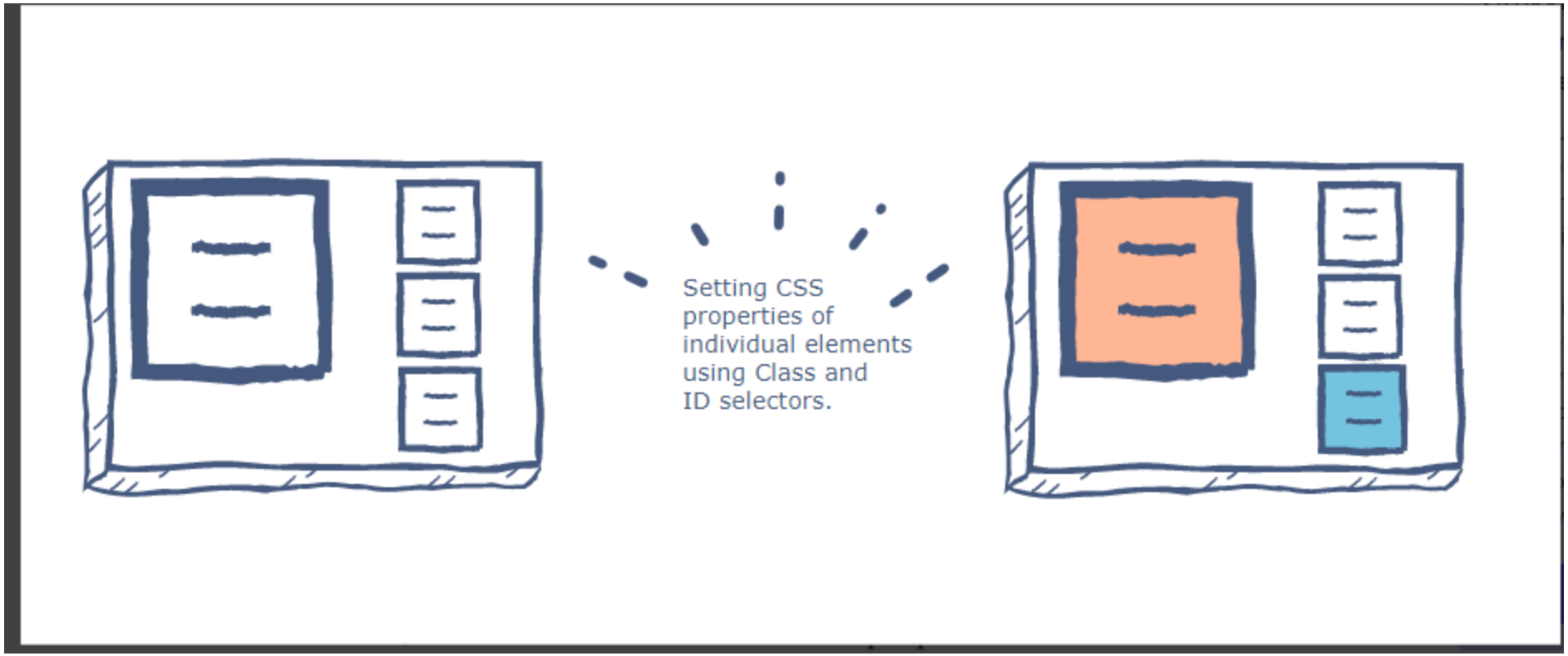
The **ID selector** matches an element based on the value of its id attribute. In order for the element to be selected, its ID attribute must exactly match the value given in the selector. The # symbol and the id of the HTML element name are used to select the desired element.

Syntax

```
#idname {
  /* Define properties here */
}
```

The difference between Class and ID selector

The difference between an ID and a class is that an ID is only used to identify one single element in our HTML. IDs are only used when one element on the page should have a particular style applied to it. However, a class can be used to identify more than one HTML element.


Setting CSS properties of individual elements using Class and ID selectors.

Code

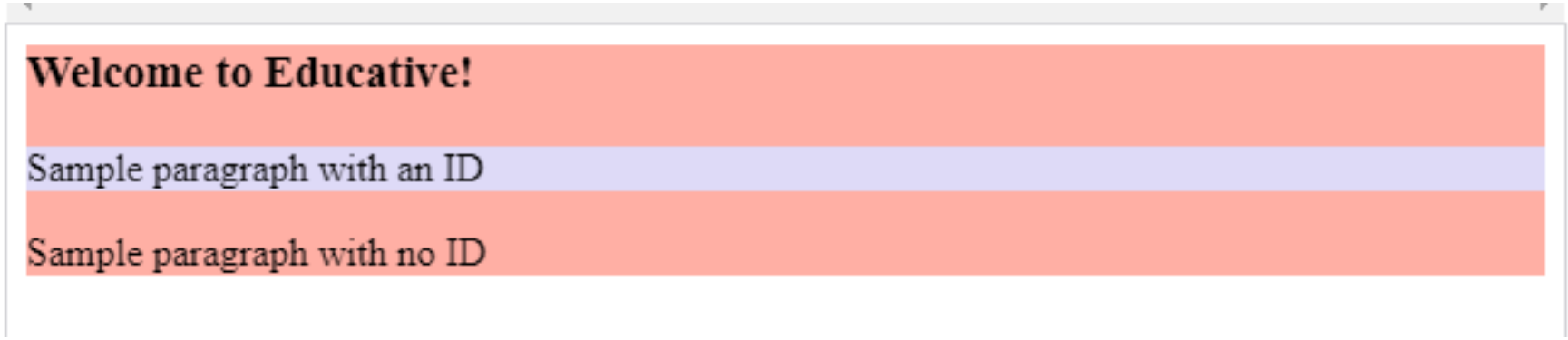
The following code demonstrates the difference between *class* and *ID selectors*, and how to set their individual properties. The code sets a class for the div and, therefore, all the elements within the div will have this class. However, one of the paragraphs is assigned an ID which allows individual properties to be set for this particular paragraph. These properties are set using the appropriate class and ID selectors.

```
<html>
 <head>
 </head>
 <body>
  <div class="main">
   <h3>Welcome to Educative!</h3>
   <p id="demo">Sample paragraph with an ID</p>
   <p> Sample paragraph with no ID</p>
  </div>
 </body>
</html>
```

```
main {
  background-color: #FFAFA4;
}

#demo {
  background-color: #DEDAF7;
}
```

OUTPUT



Fonts and text

Add the <link> element somewhere inside your index.html's head (anywhere between the <head> and </head> tags).
It looks something like this:

```
<link href="https://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet" />
```

1.This code links your page to a style sheet that loads the Open Sans font family with your webpage.
2.Next, delete the existing rule you have in your style.css file. It was a good test, but let's not continue with lots of red text.
3.Add the following lines (shown below), replacing the font-family assignment with your font-family selection from What will your website look like?. The property font-family refers to the font(s) you want to use for text. This rule defines a global base font and font size for the whole page. Since <html> is the parent element of the whole page, all elements inside it inherit the same font-size and font-family.

```
html {
 font-size: 10px; /* px means "pixels": the base font size is now 10 pixels high */
 font-family: "Open Sans", sans-serif; /* this should be the rest of the output you got from Google Fonts */
}
```

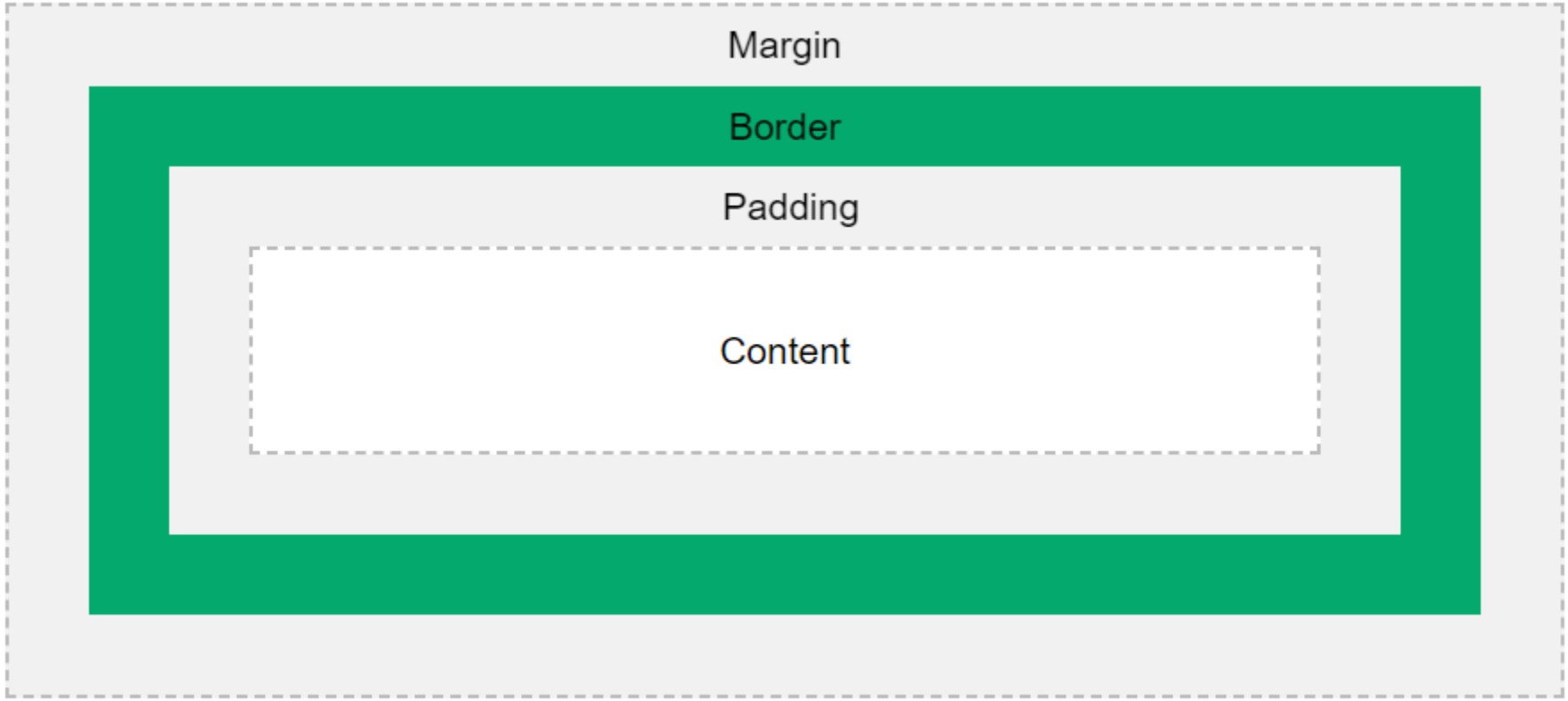
Now let's set font sizes for elements that will have text inside the HTML body (<h1>, <li>, and <p>). We'll also center the heading. Finally, let's expand the second ruleset (below) with settings for line height and letter spacing to make body content more readable.

```
h1
 { font-size: 60px; text-align: center; }
 p, li
{
font-size: 16px; line-height: 2; letter-spacing: 1px;
}
```

CSS Box Model

In CSS, the term "box model" is used when talking about design and layout.
The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content.
The image below illustrates the box model:

Explanation of the different parts:
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
The box model allows us to add a border around elements, and to define space between elements.



Example

Demonstration of the box model:

```
div {
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
```

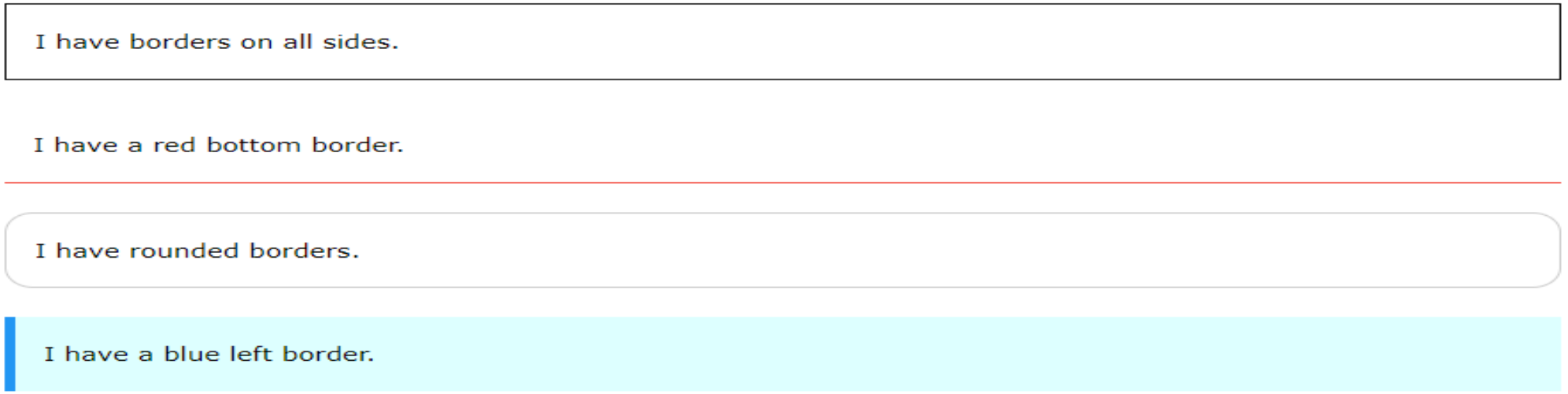
CSS Border Style

The `border-style` property specifies what kind of border to display.

The following values are allowed:

- `dotted` - Defines a dotted border
- `dashed` - Defines a dashed border
- `solid` - Defines a solid border
- `double` - Defines a double border
- `groove` - Defines a 3D grooved border. The effect depends on the border-color value
- `ridge` - Defines a 3D ridged border. The effect depends on the border-color value
- `inset` - Defines a 3D inset border. The effect depends on the border-color value
- `outset` - Defines a 3D outset border. The effect depends on the border-color value
- `none` - Defines no border
- `hidden` - Defines a hidden border

The `border-style` property can have from one to four values (for the top border, right border, bottom border, and the left border).



Ex:

html { background-color: #00539f; } !This rule sets a background color for the entire page.

- width (of an element).
- background-color, the color behind an element's content and padding.
- color, the color of an element's content (usually text).
- text-shadow sets a drop shadow on the text inside an element.
- display sets the display mode of an element.

Styling the body

```
body {
 width: 600px;
margin: 0 auto;
background-color: #ff9500;
padding: 0 20px 20px 20px;
border: 5px solid black;
}
```

There are several declarations for the <body> element. Let's go through these line-by-line:

•margin: 0 auto; When you set two values on a property like margin or padding, the first value affects the element's top *and* bottom side (setting it to 0 in this case); the second value affects the left *and* right side. (Here, auto is a special value that divides the available horizontal space evenly between left and right). You can also use one, two, three, or four values, as documented in Margin Syntax.

•background-color: #FF9500; This sets the element's background color. This project uses a reddish orange for the body background color, as opposed to dark blue for the <html> element. (Feel free to experiment.)

•padding: 0 20px 20px 20px; This sets four values for padding. The goal is to put some space around the content. In this example, there is no padding on the top of the body, and 20 pixels on the right, bottom and left. The values set top, right, bottom, left, in that order. As with margin, you can use one, two, three, or four values, as documented in Padding Syntax.

•border: 5px solid black; This sets values for the width, style and color of the border. In this case, it's a five-pixel–wide, solid black border, on all sides of the body.

•width: 600px; This forces the body to always be 600 pixels wide.

Centering the image

```
img {
display: block;
margin: 0 auto;
 }
```

Next, we center the image to make it look better. We could use the `margin: 0 auto` trick again as we did for the body. But there are differences that require an additional setting to make the CSS work. The `<body>` is a **block** element, meaning it takes up space on the page. The margin applied to a block element will be respected by other elements on the page. In contrast, images are **inline** elements, for the auto margin trick to work on this image, we must give it block-level behavior using `display: block;`.

Working with HTML Block and Inline Elements

Every HTML element has a default display value, depending on what type of element it is.
There are two display values: block and inline.

Block-level Elements

A block-level element always starts on a new line, and the browsers automatically add some space (a margin) before and after the element.
A block-level element always takes up the full width available (stretches out to the left and right as far as it can).
Two commonly used block elements are: `<p>` and `<div>`.
The `<p>` element defines a paragraph in an HTML document.
The `<div>` element defines a division or a section in an HTML document.

Example

```
<p>Hello World</p>
<div>Hello World</div>
```

Here are the block-level elements in HTML:

<address>	<article>	<aside>	<blockquote>	<canvas>	<dd>	
<dl>	<dt>	<fieldset>	<figcaption>	<figure>	<footer>	<form>
<h1>-<h6>	<header>	<hr>		<main>	<nav>	<noscript>
	<p>	<pre>	<section>	<table>	<tfoot>	
<video>						

Inline Elements

An inline element does not start on a new line.

An inline element only takes up as much width as necessary.

This is a element inside a paragraph.

Example

Hello World

Here are the inline elements in HTML:

<a>	<abbr>	<acronym>	<b>	<bdo>	<big>	
<button>	<cite>	<code>	<dfn>		<i>	<img>
<input>	<kbd>	<label>	<map>	<object>	<output>	<q>
<samp>	<script>	<select>	<small>			<sub>
<sup>	<textarea>	<time>	<tt>	<var>		

Note: An inline element cannot contain a block-level element!

Advanced CSS

Featured snippet from the web

Advanced CSS is **a set of tools and techniques that help you create the modern websites that employers and clients are looking for**. These skills help you make websites more responsive more easily so, whatever kind or size of device someone is using to view your site, it looks fantastic and works well.