

CSS

Cascading Style Sheets

What is CSS?

- **CSS** stands for **Cascading Style Sheets**
- CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
- CSS **saves a lot of work**. It can control the layout of multiple web pages all at once
- External stylesheets are stored in **CSS files**

Three ways to Insert CSS

- External style sheet
- Internal style sheet
- Inline style

External Style Sheet

- With an external style sheet, you can change the look of an entire website by changing just one file!
- Each page must include a reference to the external style sheet file inside the `<link>` element.
- External styles are defined with the `<link>` element, inside the `<head>` section of an HTML page.

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

External Style Sheet cont.

- An external style sheet can be written in any text editor.
- The file should not contain any html tags.
- The style sheet file must be saved with a .css extension.
- Common common style sheet file name is “mystyle.css” or “style.css”

Internal Style Sheet

- An internal style sheet may be used if one single page has a unique style.
- Internal styles are defined within the `<style>` element, inside the `<head>` section of an HTML page.

```
<head>
  <style>
    body {
      background-color: # linen;
    }

    h1 {
      color: # maroon;
      margin-left: 40px;
    }
  </style>
</head>
```

Inline Styles

- An inline style may be used to apply a unique style for a single element.
- To use inline styles, add the style attribute to the relevant element.
- The style attribute can contain any CSS property.
- Inline styles are defined with the **style** attribute of the relevant element.
- Inline style has the highest priority, and will override external and internal styles.

```
<h1 style="color: blue;margin-left:30px;">This is a heading</h1>
```

CSS Syntax

- A CSS rule-set consists of a selector and a declaration block:

```
h1 {  
  color: blue;  
}
```

- The selector points to the HTML element you want to style.
- The declaration block contains one or more declaration separated by semicolons.
- Each declaration includes a CSS property name and value, separated by a colon.
- A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

CSS Selectors

- CSS selectors are used to “find” (or select) HTML elements based on their element name, id, class, attribute, and more.

The element Selector

- The element selector selects elements based on the element name.

```
p {  
    text-align: center;  
    color:  red;  
}
```

The id Selector

- The id selector uses the id attribute of an HTML element to select a specific element.
- The id of an element should be unique within a page, so the id selector is used to select one unique element!
- To select an element with a specific id, write a hash (#) character, followed by the id of the element.

```
#para1 {  
    text-align: center;  
    color:  red;  
}
```

The class Selector

- The class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character, followed by the name of the class.

```
.center {  
    text-align: center;  
    color:  red;  
}
```

The class Selector cont.

- You can also specify that only specific HTML elements should be affected by a class.

```
p.center {  
    text-align: center;  
    color:  red;  
}
```

The class Selector cont.

- HTML elements can also refer to more than one class.

```
<p class="center large">This paragraph refers to two classes.</p>
```

CSS Comments

- Comments are used to explain the code, and may help when you edit the source code at a later date.
- Comments are ignored by browsers.
- A CSS comment starts with `/*` and ends with `*/`
- Comments can also span multiple lines.

```
p {  
  color: red;  
  /* This is a single-line comment */  
  text-align: center;  
}  
  
/* This is  
a multi-line  
comment */
```

CSS Colors

- Colors are specified using predefined color names, RGB, HEX, HSL, RGBA, or HSLA values.

Color Names

- In HTML, a color can be specified using a color name.
- For example, “blue” would give the color blue, “orange” would give the color “orange”, “violet” would give the color “violet”.

Text Color

- You can set the color of text by using the `color` property.

```
h1 {  
  color:  blue;  
}
```

Color Values

- In HTML, colors can also be specified using RGB values, HEX values, HSL values, RGBA values, and HSLA values
- For example, the RGB value of the color “Tomato” would be defined like this:

```
h1 {  
  color:  rgb(255, 99, 71);  
}
```

- The previous example is the same as:

```
h1 {  
  color:  tomato;  
}
```

RGB Value

- In HTML, a color can be specified as an RGB value, using this formula:

rgb(red, green, blue)

- Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255.
- For example, ***rgb(255, 0, 0)*** is displayed as red, because red is set to its highest value (255) and the others are set to 0.
- To display the color black, all color parameters must be set to 0, like this:
rgb(0, 0, 0).
- To display the color white, all color parameters must be set to 255, like this:
rgb(255, 255, 255).

HEX Value

- In HTML, a color can be specified using a hexadecimal value in the form:

#rrggbb

- Where rr (red), gg (green), and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).
- For example, ***#ff0000*** is displayed as red, because red is set to its highest value (ff) and the others are set to the lowest value (00).

RGBA Value

- RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color.
- An RGBA color value is specified with:

rgba(red, green, blue, alpha)

- The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all).
- For example, ***rgba(255, 99, 71, 0.4)***

CSS Backgrounds

- CSS background properties:
 - background-color
 - background-image
 - background-repeat
 - background-attachment
 - background-position

Background Color

- You can set the background color for HTML elements by using the `background-color` property.

```
h1 {  
    background-color:  blue;  
}
```


Background Image

- The `background-image` property specifies an image to use as the background of an element.
- By default, the image is repeated so it covers the entire element.
- The background image for a page can be set like this:

```
body {  
    background-image: url("paper.gif");  
}
```

Background Image - no-repeat

- Showing the background image only once is specified by the `background-repeat` property.

```
body {  
    background-image: url("img_tree.png");  
    background-repeat: no-repeat;  
}
```

CSS Fonts

- The CSS font properties define the font family, boldness, size, and the style of a text.

Font Family

- The font family of a text is set with the font-family property.
- The `font-family` property should hold several font names as a “fallback” system. If the browser does not support the first font, it tries the next font, and so on.
- Start with the font you want, and end with a generic family, to let the browser pick a similar font in the generic family, if no other fonts are available.
- If the name of a font family is more than one word, it must be in quotation marks, like: “Times New Roman”.
- More than one font family is specified in a comma-separated list:

```
p {  
    font-family: "Times New Roman", Times, serif;  
}
```

Font Style

- The `font-style` property is mostly used to specify italic text.
- This property has three values:
 - Normal - The text is shown normally
 - italic - The text is shown in italics
 - Oblique - The text is “leaning” (oblique is very similar to italic, but less supported)

```
p.normal {  
  font-style: normal;  
}  
  
p.italic {  
  font-style: italic;  
}  
  
p.oblique {  
  font-style: oblique;  
}
```

Font Size

- The `font-size` property sets the size of the text.
- Being able to manage the text size is important in web design. However, you should not use font size adjustments to make paragraphs look like headings, or headings look like paragraphs.
- Always use the proper HTML tags, like `<h1>` - `<h6>` for headings and `<p>` for paragraphs.
- If you do not specify a font-size, the default size for normal text, like paragraphs, is 16px (16px=1em)

```
h1 {  
    font-size: 40px;  
}
```

CSS Borders

- The CSS border properties allow you to specify the style, width, and color of an element's border.

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

Border style cont.

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

```
p.dotted {border-style: dotted;}
p.dashed {border-style: dashed;}
p.solid {border-style: solid;}
p.double {border-style: double;}
p.groove {border-style: groove;}
p.ridge {border-style: ridge;}
p.inset {border-style: inset;}
p.outset {border-style: outset;}
p.none {border-style: none;}
p.hidden {border-style: hidden;}
p.mix {border-style: dotted dashed solid double;}
```

Border Width

- The `border-width` property specifies the width of the four borders.
- The width can be set as a specific size (in px, pt, cm, em, etc) or by using one of the three predefined values: thin, medium, or thick.
- The `border-width` property can have from one to four values (for the top border, right border, bottom border, and the left border).

```
p.one {  
  border-style: solid;  
  border-width: 5px;  
}  
  
p.two {  
  border-style: solid;  
  border-width: medium;  
}  
  
p.three {  
  border-style: solid;  
  border-width: 2px 10px 4px 20px;  
}
```

Border Color

- The `border-color` property is used to set the color of the four borders.
- The color can be set by:
 - Name - specify a color name, like “red”
 - Hex - specify a hex value, like “#ff0000”
 - Rgb - specify a rgb value, like “rgb(255, 0, 0)”
 - Transparent
- The `border-color` property can have from one to four values (for the top border, right border, bottom border, and the left border).
- If `border-color` is not set, it inherits the color of the element.

```
p.one {  
    border-style: solid;  
    border-color:  red;  
}
```

Border - Individual Sides

- From the previous examples, you have seen that it is possible to specify a different border for each side.
- In CSS, there are also properties for specifying each of the borders (top, right, bottom, and left)

```
p {  
  border-top-style: dotted;  
  border-right-style: solid;  
  border-bottom-style: dotted;  
  border-left-style: solid;  
}
```


Border - Shorthand Property

- As you saw from the previous examples, there are many properties to consider when dealing with borders.
- To shorten the code, it is also possible to specify all the individual border properties in one property.
- The **border** property is a shorthand property for the following individual border properties:
 - border- width
 - Border-style (required)
 - Border-color

```
p {  
  border: 5px solid red;  
}
```

Rounded Borders

- The `border-radius` property is used to add rounded borders to an element

```
p {  
    border: 2px solid  red;  
    border-radius: 5px;  
}
```

CSS Margins

- The CSS margin properties are used to create space around elements, outside of any defined borders.
- With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

Margin - Individual Sides

- CSS has properties for specifying the margin for each side of an element:
 - margin-top
 - margin-right
 - margin-bottom
 - margin-left
- All the margin properties can have the following values:
 - Auto - the browser calculates the margin
 - Length - specifies a margin in px, pt, cm, etc.
 - % - specifies a margin in % of the width of the containing element
 - Inherit - specifies that the margin should be inherited from the parent element
- Negative values are allowed.

Margin - Individual Sides cont.

- Set different margins for all four sides of a `<p>` element

```
p {  
    margin-top: 100px;  
    margin-bottom: 100px;  
    margin-right: 150px;  
    margin-left: 80px;  
}
```

Margin - Shorthand Property

- To shorten the code, it is possible to specify all the margin properties in one property.
- The margin property is a shorthand property for the following individual margin properties:
 - Margin-top
 - Margin-right
 - Margin-bottom
 - Margin-left
- Top (25px), Right (50px), Bottom (75px), Left (100px)::

```
p {  
    margin: 25px 50px 75px 100px;  
}
```

CSS Padding

- The CSS padding properties are used to generate space around an element's content, inside of any defined borders.
- With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

Padding - Individual Sides

- CSS has properties for specifying the padding for each side of an element:
 - padding-top
 - padding-right
 - padding-bottom
 - padding-left
- All the padding properties can have the following values:
 - Length - specifies a padding in px, pt, cm, etc.
 - % - specifies a padding in % of the width of the containing element
 - Inherit - specifies that the padding should be inherited from the parent element
- Negative values are not allowed.

Padding - Individual Sides

- Set different padding for all four sides of a `<div>` element:

```
div {  
    padding-top: 50px;  
    padding-right: 30px;  
    padding-bottom: 50px;  
    padding-left: 80px;  
}
```

Padding - Shorthand Property

- To shorten the code, it is possible to specify all the padding properties in one property.
- The padding property is a shorthand property for the following individual padding properties:
 - padding-top
 - padding-right
 - padding-bottom
 - padding-left
- Top (25px), Right (50px), Bottom (75px), Left (100px):

```
div {  
    padding: 25px 50px 75px 100px;  
}
```

CSS Layout - The display Property

- The `display` property is the most important CSS property for controlling layout.

The display Property

- The display property specifies if/how an element is displayed.
- Every HTML element has a default display value depending on what type of element it is.
- The default display value for most elements is block or inline.

Block-level Elements

- A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).
- Examples of block-level elements:
 - `<div>`
 - `<h1>` - `<h6>`
 - `<p>`
 - `<form>`

Inline Elements

- An inline element does not start on a new line and only takes up as much width as necessary.
- Examples of inline elements:
 - ``
 - `<a>`
 - ``

Display: none;

- `display: none;` is commonly used with Javascript to hide and show elements without deleting and recreating them.
- The `<script>` element uses `display: none;` as default.

Override The Default Display Value

- As mentioned, every element has a default display value. However, you can override this.
- Changing an inline element to a block element, or vice versa, can be useful for making the page look a specific way, and still follow the web standards.
- A common example is making inline `` elements for horizontal menus:

```
li {  
    display: inline;  
}
```

Override The Default Display Value cont.


- The following example displays `<a>` elements as block elements:

```
a {  
  display: block;  
}
```

CSS Layout - Horizontal & Vertical Align

Center Align Elements

- To horizontally center a block element (like `<div>`), use `margin: auto;`
- Setting the width of the element will prevent it from stretching out to the edges of its container.
- The element will then take up the specified width, and the remaining space will be split equally between the two margins
- Center aligning has no effect if the width property is not set (or set to 100%).

```
.center {  
    margin: auto;  
    width: 50%;  
    border: 3px solid  green;  
    padding: 10px;  
}
```

Center Align Text

- To just center the text inside an element, use `text-align: center;`

```
.center {  
    text-align: center;  
    border: 3px solid  green;  
}
```



Center an Image

- To center an image, set left and right margin to `auto` and make it into a block element.

```
img {  
    display: block;  
    margin-left: auto;  
    margin-right: auto;  
    width: 40%;  
}
```

Center Vertically - Using padding

- There are many ways to center an element vertically in CSS. A simple solution is to use top and bottom padding:

```
.center {  
    padding: 70px 0;  
    border: 3px solid  green;  
}
```

Assignment 2.2 - Recreate HTML & CSS Site

- Recreate this [webpage](#) with HTML and CSS.