Introduction to CSS

computer on desk

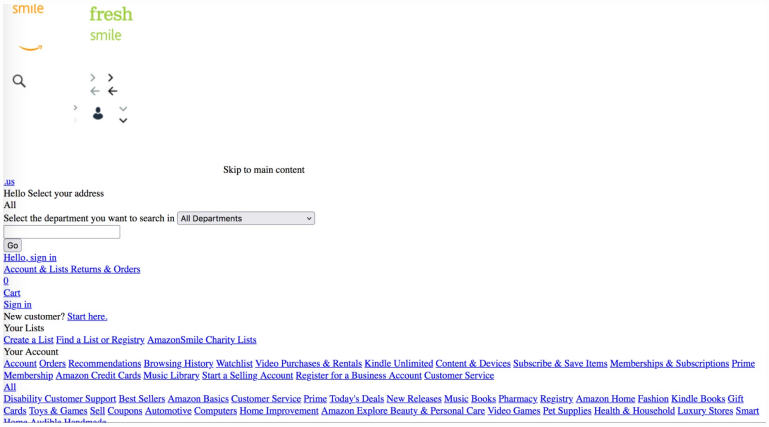
This lesson is intended to introduce you to the basics of CSS, or Cascading Style Sheets. You will learn what it are used for, how to incorporate it into your project, and how learning CSS can enhance your career.

Goals

By the end of this case you will learn:
* What is CSS (Cascading Style Sheets)?
* What is it used for?
* Origins and definition
* CSS Fundamentals
- Structure
- Style rules
- Selectors
* How to add CSS to your project

What is CSS and what is it used for?

As you journey around the world wide web, you may ask yourself, “how can websites look so different from one another?” The answer is **Cascading Style Sheets**, or CSS. CSS allows differing visual appearances for pages on the web. Fonts, colors, and spacing are all handled by style rules inherited by a web page established in the CSS. If you ever see a web page that looks like the following, it is likely the CSS styles were not loaded, or inappropriately applied.



website without CSS formatting

CSS allows developers to bring uniquely designed web pages to the web and enhance the visual experience it offers. Fonts, colors, and spacing are all handled by style rules managed by CSS and loaded to a web page.

The above image is the Amazon.com home page with no CSS applied. This demonstrates that no matter the website, big or small, proper CSS is critical to providing a good user experience.

To view a web page without CSS, do the following:

– Download and open Firefox web browser
– Go to a page
– In the `'View'` menu, go to Page Style
– Select `'No Style'`

Exercise 1:

Origins and Definition

CSS stands for Cascading Style Sheets. CSS was first proposed in 1994 by Hakon Wium Lie, the first version following in 1996. It is a style sheet language which is used to determine the appearance and formatting of a document written in a markup language. It is typically used with Hyper Text Markup Language (HTML) to style web pages and user interfaces. It can also be used with any kind of document written with Extensible Markup Language [XML], like plain XML, Scalable Vector Graphics (SVG) or XML User Interface Language (XUL).

Along with HTML, CSS is also commonly paired with JavaScript to create user interfaces for web and mobile sites and applications. JavaScript is also used for websites that are not applications, like profile pages and brochures.

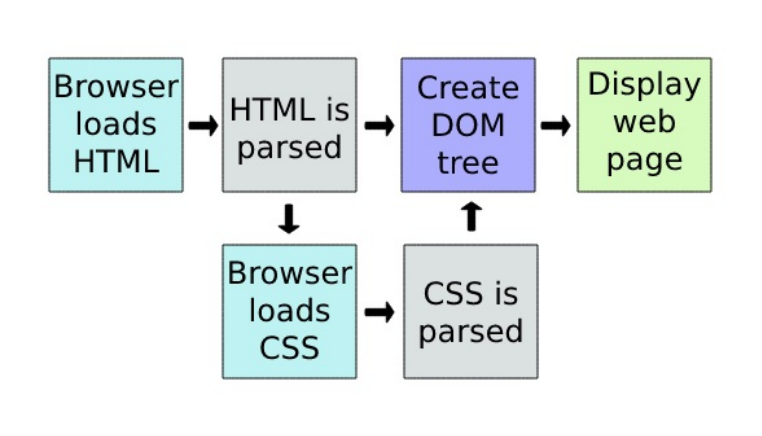
You can learn more about CSS here.

How does CSS Work?

When a browser loads a page, it combines the page's content with its style sheet. It processes the documents in stages. Different browsers load HTML and CSS in differing ways, and while the below list is simplified, it should give a general timeline on what happens when the page is loaded. The documents are processed in the following stages:

1. HTML is received from the web and loaded to the browser
2. The browser converts the HTML file into a Document Object Model (DOM). The DOM represents the HTML file in the global browser namespace.
3. The browser then grabs the resources linked to the HTML document. These include, videos, images, scripts, and CSS files.
4. The browser parses the CSS, sorting the different rules by their selectors, which include: elements, classes, IDs, and other variations. Based on the selector it finds, the browser works out which rules are applied to which nodes in the DOM. It then attaches the styles to them, as suggested by the CSS.
5. The resulting object is called the 'render tree,' which includes the visible styled html elements.
6. The page is shown on the screen (this stage is referred to as "painting").

Refer to the visual below:



stages of processing documents with HTML and CSS

Exercise 2:

||| (Correct order: 7,6,1,5,9,8,2,4,3)

Structure, syntax and properties

Structure and Syntax

The basic syntax includes a **selector** with its associated declaration block. A declaration block consists of two things; a CSS **property**, and its **value**. Below is a sample of CSS.

Example 1
|files|index.html|style.css|
|———|———-|———-|
|index.html
style.css|`<!DOCTYPE html>`
`<html>`
`<head>`
`<link rel="stylesheet" href="style.css">`
`</head>`
`<body>`
`<p>Hello world!</p>`
`</body>`
`</html>`|**p{color:green;}**|

In the example above, `'p'` refers to the **selector**, `'color'` is the **property** being affected, and `'green'` is the **value** being applied. This CSS rule will change the color of the paragraph's text to green.

Properties

CSS properties allow developers to change how elements look by assigning styles – or, in the case of animations, behavior – to each HTML element. CSS properties are incredibly powerful, and enable developers to introduce variety and design into web pages:

Examples of commonly used properties:

files	index1.html	style1.css

index.html
style1.css
	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<link rel="stylesheet" href="style1.css">`
	`</head>`
	`<body>`
	`<p`
		.roboto{font-family:'Roboto', sans-serif;}
		
		.green{color: green;}

```
class="roboto">Hello
world!</p>
</body>
</html>
```

```
.left{text-align: left;}
```

- `'font-family'` is used to set the font of text and header elements. This example will change the font of the element with class `'.roboto'` to Roboto, with a default font-family of `'sans-serif'`
- `'color'` colors text and ‘background-color’ colors the background of the element
- `'text-align'` is used to set the alignment of an element, and can be set to center, left, or right. This example will align the element to the left.

Comments and selectors

Comments

Comments are a helpful way to leave a message to yourself, or whoever is working on the style sheet in the future. Describing what classes do, or what their goal should be can quickly ease confusion.

To add a comment, add `<!--` and begin typing. Once your comment is complete, type `-->` to close the comment.

Example 2

```
<!-- This is an HTML comment -->

/* This is a CSS comment */
```

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<!-- This is an
html comment -->
</body>
</html>
```

```
/* This is a css comment
*/
```

Selectors

Selectors are used to target specific HTML elements. There are multiple types of CSS selectors. They include:

- **Element** selectors target HTML elements based on their name.
- **Id** selectors target HTML elements based on their attribute.
- **Class** selectors target HTML elements based on a class attribute.
- **Universal** selectors target all elements present on the page.

Example 3

Element:

files	index-element.html	style-element.css
	`<!DOCTYPE html>` `<html>`	

index-element.html style-element.css	<head> <link rel="stylesheet" href="style- element.css"> </head> <body> Hello world! </body> </html>	**p { text-align: left; color: blue;}**

In the example above, all the HTML `'p'` elements will be aligned to the left and the color of their text will be changed to blue.

ID:

files	index-id.html	style-id.css
index-id.html style-id.css	<!DOCTYPE html> <html> <head> <link rel="stylesheet" href="style-id.css"> </head> <body> Hello world! </body> </html>	**#redtext**{ color:red; text-align:center;}

In the example above, the HTML element with the ‘id=“red_text”’ attribute will have the rule applied to it, and will be made red and aligned to the center. ID’s must be unique and target only one element.

Classes, universal and external CSS

Classes

files	index3.html	style3.css
index3.html style3.css	<!DOCTYPE html> <html> <head> <link rel="stylesheet" href="style3.css"> </head> <body> <h1 class="right">Hello world!</h1> </body> </html>	**.right**{ text-align:right; color: green;}

In this case, all HTML elements are aligned to the right with text colored green, or 'made to align right' with text colored green.

Universal:

The universal selector, represented by "*" selects every HTML element in the document.

files	index-universal.html	style-universal.css
index-universal.html style-universal.css	<!DOCTYPE html> <html> <head> <link rel="stylesheet" href="style- universal.css"> </head> <body> Hello world! </body> </html>	*{text-align: right; color: blue;}

The above will make every HTML element align right, and align to the right with text colored blue.

Exercise 3: Multiple Choice

In the example above, we demonstrated styling `'id'` and `'class'` attributes. In the examples below, which `'id'` attribute will turn the paragraph text blue and add 100 pixels of margin to the top and bottom of the element?

How to add CSS to your project

There are three standard ways to insert CSS into your HTML document.

- External CSS
- Internal CSS
- Inline CSS

External CSS
External CSS is contained in a separate file. You can change the entire look and feel of the webpage just by loading it. External loading respects the principle of separating structure and styling. It keeps the rules in a self-contained document that is easy to work on. It is also enables developers to swap out entire stylesheets `(or rulesets)`easily.

External CSS is loaded with a single `<link>` tag inserted in the `<head>` element in the HTML. It is useful to note that `<link>` does not need a closing tag.

Example 4

files	index.html	style.css

index.html
style.css
	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<link`
	`rel="stylesheet"`
	`href="style.css">`
	`</head>`
	`<body>`
	`<p>Hello world!</p>`
	`</body>`
	`</html>`
		p{color:green;}

Internal and inline

Internal CSS

Defined in the `<style>` section of the HTML, internal CSS may be used if one single HTML page has a unique style. There are useful attributes with this tag. For example, the `'media'` attribute allows the developer to load a separate stylesheet for mobile phones, and the `'title'` attribute gives the user the option of selecting a different style from the browser's `'view'` menu. Internal CSS speeds up load time. Load above-the-fold CSS internally to reduce load time.

More information can be found here.

Example 5

files	index2.html

index.html	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<style>body{padding:1%;}*`
	`{box-sizing: border-box;}`
	`</style>`
	`</head>`
	`<body>`
	`<p>Hello world!</p>`
	`</body>`
	`</html>`

Inline CSS

Many content management systems (CMS) use inline CSS; but if it's not necessary, too much inline CSS is against best practices. It's also used a lot in HTML emails.

Example:

files	index4.html

	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<p`
	`style="color:blue;">This`

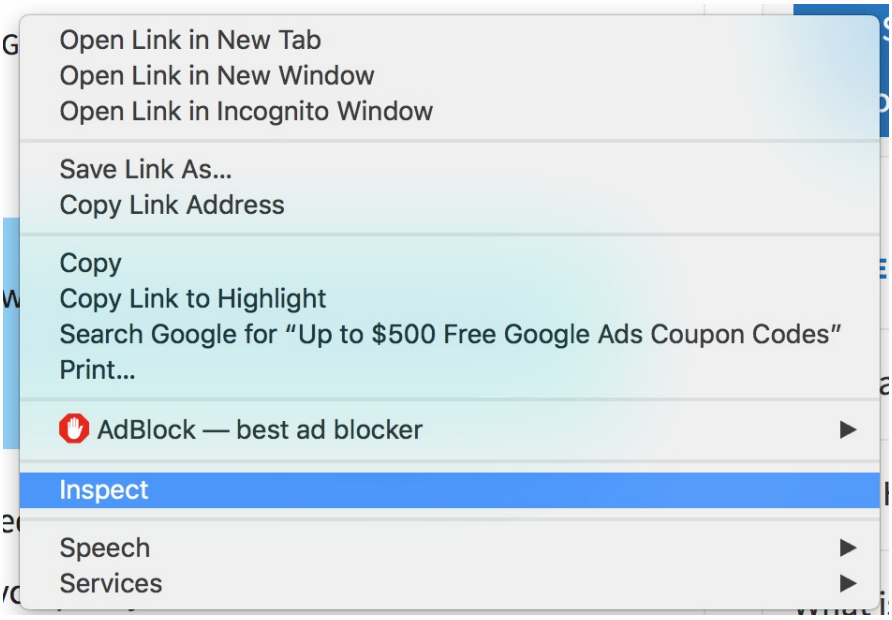
index4.html	`text is blue.</p>`
	`</head>`
	`<body>`
	`<p>Hello world!</p>`
	`</body>`
	`</html>`

Exercise 4

|||**Answer:** False. Using classes to apply styles is more efficient as it allows classes to be reused across multiple components. Inline CSS should be used only when necessary.

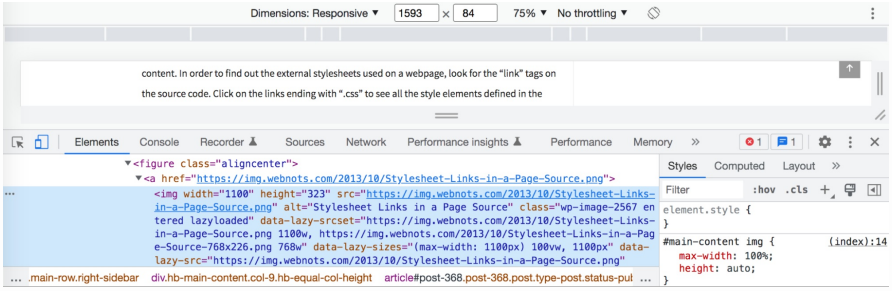
Bonus: How to View CSS in Console

The Inspector shows all of the CSS. In order to view the HTML content, inline and internal styles of a webpage, open the webpage in Chrome browser. Right click any place on the page and select `"Inspect"` option as shown in the picture below:



This will open a new window which will show the marked up HTML content and styles of each element used on that webpage. Some sites will show you a clear source view, but others will show the source code without line breaks and spaces.

Depending on your browser, you will be shown the following:



The currently applied CSS rules are located within the `"Styles"`tab. You can see the above element, an image, was given the following properties. `"Max-width"` was set to 100%, meaning the image will take up the full amount of space allotted to it. `"Height"` was set to auto, meaning the height will be determined based on the final width of the image.

Being able to access the console and check which style rules apply to an element is key to troubleshooting and testing how the browser handles the CSS.

Tip! Most browsers have a shortcut to view the console (F12).

Conclusion & Takeaways

- CSS allows for variety in the visual appearance of webpages on the web today.
- CSS can be added to a website inline, internally and externally.
- Every website in production today uses some kind of CSS, which means there are a lot of career opportunities for front end developers who know it well.

Attribution

Unsplash free wallpaper images, retrieved Nov 10, 2022 (https://unsplash.com/photos/m_HRfLhgABo)
How CSS is structured (https://linuxhint.com/css-structure/)
W3 School CSS (https://www.w3schools.com/css)
Javatpoint: what is CSS (https://www.javatpoint.com/what-is-css)
Mozilla: How CSS works - DOM tree image (https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/How_CSS_works)
W3 add css (https://www.w3schools.com/css/css_howto.asp)

Use CSS to Create a Restaurant Menu

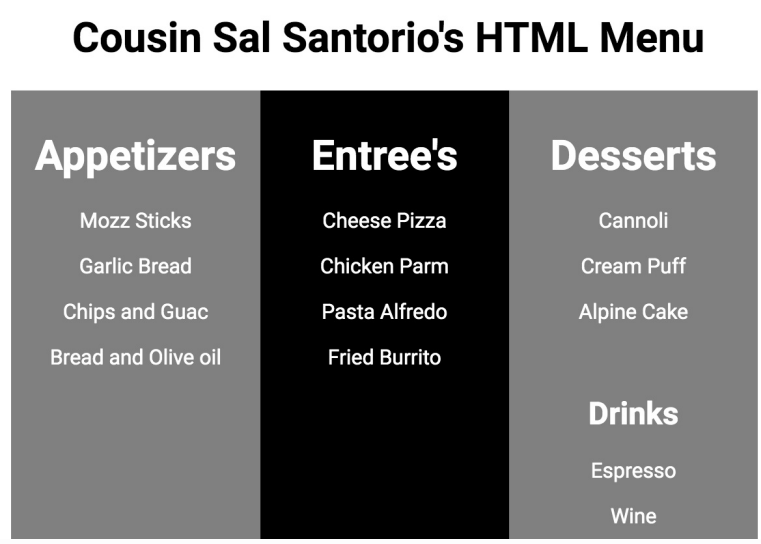
Goals

By the end of this guided exercise you will be able to:
1. Write CSS code to add styles to a basic restaurant menu
2. Practice CSS : properties, selectors, and values

Introduction

Restaurant Menu

A small business, Cousin Sal Santorio's Restaurant, wants to revamp their online menu. By the end of this exercise, you will have practice using CSS selectors to change element style properties. Currently, the menu looks like this:



menu displayed without CSS

Cousin Sal wants a more colorful menu with a new font and an image in the background.

Instructions

Use the HTML below and the CSS you have learned so far to change the style of Cousin Sal's menu. Add a new font that will be used by the entire menu, make each of the columns more colorful, and add a background image.

Starting code:

files	index.html	style.css

index.html
style.css
	<!DOCTYPE html>
	<html>
	<head>
	<style>
	body {
	padding: 1%;
	}
	*{
	box-sizing: border-
	box;
	}
	.left, .right,
	.center {
	float: left;
	width: 33%;
	color: white;
	padding: 10px;
	height: 500px;
	text-align: center;
	}
	.left {
	background-color:
	grey;
	}
	.right {
	background-color:
	grey;
	}
	.center {
	background-color:
	black;
	}
	.container:after {
	clear: both;
	}
	</style>
	</head>
	<body>
	<h1 style="text-
	align:	

```
center;">Cousin Sal
Santorio's HTML
Menu</h1>
<div
class="container">
<div class="left">
<h1>Appetizers</h1>
<p>Mozz Sticks</p>
<p>Garlic Bread</p>
<p>Chips and Guac</p>
<p>Bread and Olive
oil</p>
</div>
<div class="center">
<h1>Entree's</h1>
<p>Cheese Pizza</p>
<p>Chicken Parm</p>
<p>Pasta Alfredo</p>
<p>Fried Burrito</p>
</div>
<div class="right">
<h1>Desserts</h1>
<p>Cannoli</p>
<p>Cream Puff</p>
<p>Alpine Cake</p>
<br>
<h2>Drinks</h2>
<p>Espresso</p>
<p>Wine</p>
</div>
</div>
</body>
</html>
```

Detailed Instructions

1. To begin, take the internal CSS in the code provided and create an external style sheet.
   Using what we've learned, load the stylesheet in the `<head>` section. By managing your CSS in a separate file you can use the same CSS on multiple web pages, without having to change every page when you want to change the CSS.

2. Adding a Font
   To begin, look over the code provided and determine how to best add a new font that will be used by every element. For this exercise, we want to use the font family `'Roboto'`. Since this is an external font available on Google Fonts, we will first have to import it into our style sheet. To do so, copy and past the following at the very top of **style.css**, above all of our other styles:

```
@import url('https://fonts.googleapis.com/css2?
family=Roboto:ital,wght@0,100;0,300;0,400;0,500;0,700;0,900;1,10
0;1,300;1,400;1,500;1,700;1,900&display=swap');
```

Now, what would be the best way to set this as the font for all our elements?

Hint: the `"font-family"` attribute is what is used to set a font for an element.

Question 1:

|||**Answer:** The `'body,'` or universal selector, `'*'` will affect every element on the page.

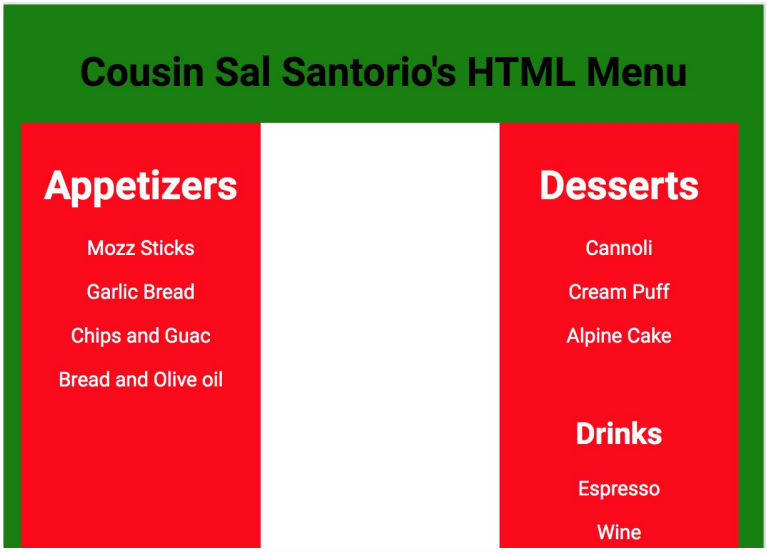
Once the CSS is added correctly, the menu will look like this:

menu with CSS to improve font

3. Colors
   Sal loves the new font, but he also wants to add color to the menu. As Sal is Italian, he wants to theme the colors around the Italian flag.

Italian flag colors

Next: set the background color of the menu to Green. Then, make the left and right columns Red. Finally, set the middle column to White. If you have done it correctly, the menu will now look like the following:



menu with CSS to improve font and color

Big improvement! However, during the color change, we lost the text in the middle column! Also, the black title font on the green background doesn't have enough contrast.

4. Change the text in the middle column to black. Use inline CSS to change the color of the menu title to white. If you have done it correctly, it should look like this:

menu with text color improved to display with colors

Note: While we want to practice inline CSS, in most cases, it is better to change the CSS in an external stylesheet.
Also, due to cascading, you can set the color for an element a second time in its class, which due to being read second will be the one ultimately applied. That is the case in this step for the middle column text.

The entire CSSOM is usually built before any inline CSS gets parsed, since

external stylesheets are introduced in the

tag and the CSSOM is built quickly. If you want a property in your stylesheet CSS to override the inline CSS, you should add ‘!important’ to it.

Bonus

Final touches
Sal loves the new menu. It's already bringing them new business. However, he wants to move the drink menu into the middle column and give it more space. Make adjustments to the HTML and CSS to achieve this effect. If done correctly, it will look like this:

menu with drinks moved to the middle column

Our menu is complete! It is colorful, has a new font, and lays out the content according to our client's specifications.

Bonus: If you want to explore more of what CSS can do in this exercise, try to achieve the following:

- Add a background image to the menu
- Add borders to each of the columns, making sure all the text is within the border
- Add a drop shadow to the headings
- Make the columns responsive (this concept is not introduced until the next lesson)

Conclusion & Takeaways

- CSS is highly flexible, and can make a page look different in many ways
- It is important to practice and play around with sizing, colors, and fonts
- CSS classes provide an efficient way to add the same CSS to more than one element.

To learn more about the difference between ids and classes, click here.

Attribution

7. Tutorialspoint: CSS Grids (https://www.tutorialspoint.com/how-to-create-a-3-column-layout-grid-with-css)
8. CSS Classes W3 (https://www.w3schools.com/html/html_classes.asp)
9. Mozilla Specificity: https://developer.mozilla.org/en-US/docs/Web/CSS/Specificity

Box Model

Goals

By the end of this case you will learn:

- What Box Model is
- How Box Model affects elements
- Properties that affect Box Model

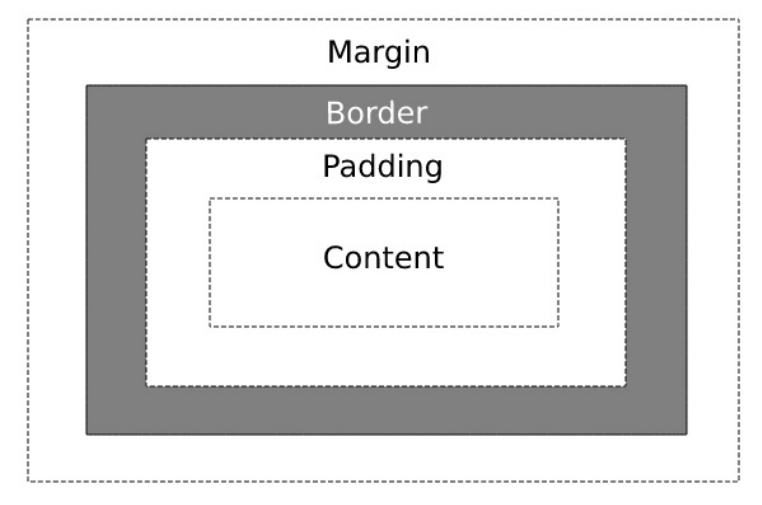
Introduction

Now that we have learned the basics around CSS and how it works, it is time to dive into more complex systems that allow you to create powerful, well designed web pages.

Systems around CSS

Box Model

In HTML, all elements are considered **boxes**. This means each element has visual space around it that separates it from other elements. The amount of surround space and CSS rules that govern it are referred to as the **Box Model**. With CSS, we use the box model to visualize the size of elements in order to better understand how their size is impacted by changes in content, padding, borders, and margins.



box model diagram of margin, border, padding, and content

What affects the dimensions of our box?

Content size

More than anything, the content on the page (video, copy, images) has a major impact on the size of element boxes. Boxes change with the size of your content. Changing font sizes or using a bigger image will take up more space, increasing the size of the box.

Block and inline boxes

There are two types of boxes in CSS; block boxes and inline boxes. The type refers to how that kind of box behaves in page flow and relation to boxes on the page.

You can set values for the display type using the **display** property, which has various values.

To learn more about box types click here.

Exercise 1

|||Answer: The element's size depends on the height and width properties of the parent box, and the parent's parent box. Answers saying to reduce the size of the image or the amount of text are best. Saying the font size should be reduced is not a great answer, as it will negatively affect the user experience.

Padding

Adding padding to an element creates a transparent empty space around the content. If you were working on a button that was too close to other elements on the page, you could add a padding of 10px to create space around it. It is also possible to add padding to specific sides of an element.

CSS has properties that specify the padding on each side of an element:

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

There is also a shorthand version of the rule.

If the padding property has **four** values:

padding: 20px 10px 50px 20px;
- top padding is 20px
- right padding is 10px
- bottom padding is 50px
- left padding is 20px

If the padding property has **three** values:

padding: 5px 20px 50px;
- top padding is 5px
- right and left paddings are 20px
- bottom padding is 50px

If the padding property has **two** values:

padding: 50px 25px;
- top and bottom paddings are 50px
- right and left paddings are 25px

Margin

Margin allows the user to add space outside the border of the box. This is usually used to give an element extra visual space, and can be added to specific sides of the box. Margin is given to elements much like padding is. The shorthand syntax for margin is the same as padding.

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

Example 1

files	index.html	styles.css

index.html
styles.css
	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<link`
	`rel="stylesheet"`
	`href="style.css">`
	`</head>`
	`<body>`
	`<p`
	`class="inside">Hello`
	`world.</p><p`
	`class="outside">Hello`
	`world.</p>`
	`</body>`
	`</html>`
		.inside{padding:100px 200px;}
		.outside{margin:100px;}

In the above example, the paragraph given the inside class will have 100px of padding added to the top and bottom of the element, and 200px added to the left and right. The paragraph given the outside class will have 100px of margins on all sides.

Note: margin can be negative and padding cannot.

Padding is meant to extend space around content, which is relevant in cases when the box styling is visible, for example when the box has a background color.

Collapsing Margins

Top and bottom margins are occasionally combined (referred to as collapsed) into a single margin value whose size is the largest of the individual margins. This behavior is known as margin collapsing. Click here to learn more.

Exercise 2

Border

It can seem counterintuitive, but borders do take up some space in the box model. The border wraps around the **content** and **padding** of the element.

Border Style
The border-style property allows the user to display more than one kind of border.

- solid - Creates a solid border
- dashed - Creates a dashed border
- dotted - Creates a dotted border
- double - Creates a double border
- groove - Creates a grooved border
- ridge - Creates a ridged border
- inset - Creates an inset border
- outset - Creates an outset border
- hidden - Creates a hidden border
- none - Removes any border

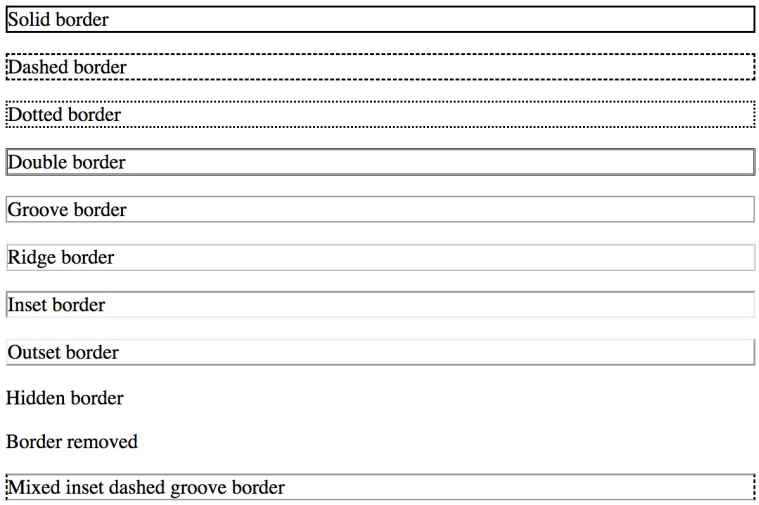
Examples of each style

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

Mix of styles

If you would like a mix of styles, use the following syntax
.mix {border-style: dotted dashed solid double;}

files	index.html	styles.css
index.html styles.css	<!DOCTYPE html> <html> <head> <link rel="stylesheet"href="style.css"> </head> <body> Hello world. Hello world. Hello world. Hello world. Hello world. Hello world. Hello world. Hello world. Hello world. Hello world. </body> </html>	.solid {border-style: solid;} .dashed {border-style: dashed;} .dotted {border-style: dotted;} .double {border-style: double;} .groove {border-style: groove;} .ridge {border-style: ridge;} .inset {border-style: inset;} .outset {border-style: outset;} .hidden {border-style: hidden;} .none {border-style: none;} .mix {border-style: dotted dashed solid double;}



examples of how each border style appears

Exercise 3

|||Answer: False. Border types can be mixed and matched freely.

Conclusion & Takeaways

- Box model defines how elements relate to each other in terms of sizing and layout
- Properties such as margin, padding, and border, along with the content itself, determine the elements size
- Shorthand syntax is helpful, and efficient

Attribution

10. W3 Tutorials: Box Model (https://www.w3schools.com/css/css_boxmodel.asp)

Functions

Goals

By the end of this case you will:
- Learn about CSS functions
- Learn how to use functions in your project

Introduction

Functions

CSS, like object-oriented programming languages, uses functions. **CSS functions allow the developer to compute styles with code and reuse the computation throughout the stylesheet.** Some examples of computations are calculating values or changing elements of the HTML file. CSS functions can be inserted anywhere you would place a value.

Example 1

files	index.html	style.css

index.html
style.css
	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<link`
	`rel="stylesheet"`
	`href="style.css">`
	`</head>`
	`<body>`
	`<div class="example">`
	`<p>Hello world!</p>`
	`</div>`
	`</body>`
	`</html>`
		.example {background:
		url(picture.png);}

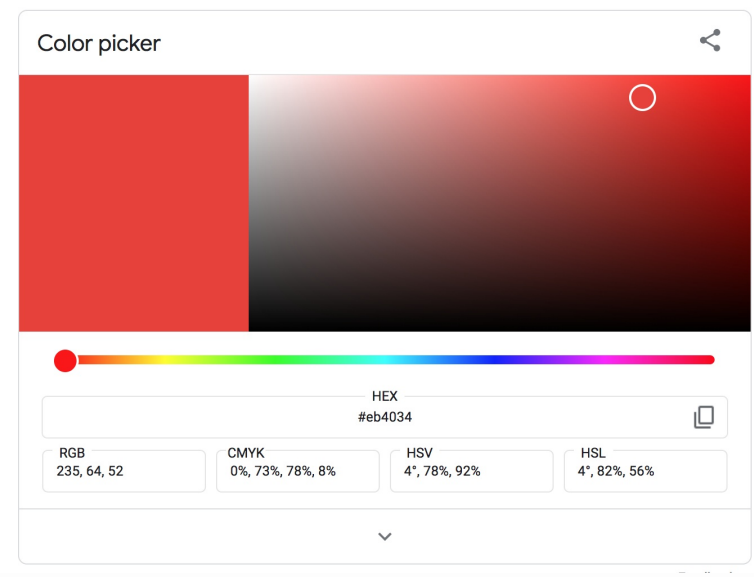
The example above uses the CSS function `url()`. We feed a relative path argument that leads to the image in your file structure into the parentheses. The `url()` function then imports the image at the end of the path into the `background-image` property for this CSS declaration. This will change the background of the element affected by the CSS selector.

Exercise 1:

|||Answer: background:
url(https://cdn.pixabay.com/photo/2018/08/14/13/23/ocean-3605547__340.jpg);

Color

Functions can do many things, here are a few additional examples.



color picker diagram

Color is a commonly used function in CSS.

files	index1.html	style1.css

index1.html
style1.css
	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<link`
	`rel="stylesheet"`
	`href="style1.css">`
	`</head>`
	`<body>`
	`<div class="example">`
	`<p>Hello world!</p>`
	`</div>`
	`</body>`
	`</html>`
		.example {color:
		rgb(255, 0, 0);
		color: rgba(255, 0, 0,
		0.5);
		color: rgb(255 0 0 /
		0.5);}

The color function allows you to use numerical values to determine the red `(r)`, green `(g)`, blue `(b)`, and alpha `(a)` levels of a color.

For example, a blue color with a hex value of `#0000ff` can be represented using rgba with the code `'rgba(0,0,255,1)'`, where the first position takes the red value, between 0 and 255, the second position takes the green

value, the third position takes the blue value, and the final position takes the `'alpha'` value. When `'alpha'` = 0, the color is fully transparent, when it equals 1, it is fully opaque.

The alpha value of 1 means that it is fully opaque. This means it won't show any content behind the color. If we change the alpha value to 0.5, the color will be 50 percent transparent. If you use an `rgb()` function instead of `rgba()`, you do not need to supply an alpha value.

Exercise 2

|||Answer: B

Common CSS functions

Other common CSS functions include:
- `attr()`
- This function allows you to reach into HTML, grab an attribute's content, and push it to the CSS property
- Commonly used in print stylesheets, `attr()` is used to show the URL of a link after its text
- `calc()`
- This function takes two arguments and calculates a result from the operator (+, -, *, /) that is supplied
- `calc()` allows users to perform calculations for specifying CSS values. It's useful for determining length, integers, numbers, time, frequencies, percentages, and angles, and more
- `var()`
- This function allows developers to refer to variables.
- For example, if you define a variable with the value 50 (`--number1: 50;`) you could then use `var()` to refer to it during calculations.
- Ex. `var(--number1) * 10` would equate to 500

- `lang()`
  - This function is used to set language-specific quotes
  - This is useful for tasks like internationalization
- `:not()`
  - This selector will select anything that isn't what is specified
  - Example: You could target everything that isn't paragraph text with `:body:not(p)`
  - This is an example of a pseudo class
    - To learn more about pseudo classes, click here.
- `hsl()` and `hsla()`
  - Like `rgb()` and `rgba()`, `hsl()` and `hsla()` allow you to determine color
  - Instead of using red, green, and blue, `hsl()` and `hsla()` use hue (h), saturation (s), and lightness (l)

Exercise 3

|||Answer: A. A is the properly formatted function, and will result in 50*2, which is 100.

Conclusion & Takeaways

CSS functions are a powerful tool when creating web pages. They introduce functionality into CSS, and enable developers to compute CSS properties. When utilized well, they can make your code base more efficient.

Attribution

1. LogRocket: CSS functions (https://blog.logrocket.com/how-when-use-css-calc-tutorial-examples/)
2. CSS-Tricks Functions (https://css-tricks.com/complete-guide-to-css-functions/)

Cascades

Goals

By the end of this case you will learn:
- CSS cascade
- Some best practices to follow when coding cascading stylesheets

Introduction

Cascades

Recall that CSS stands for ***cascading* style sheets**. The 'cascade' in 'cascading style sheets' refers to a core concept in CSS.

Inevitably, you will find that the CSS you thought was applied to an element is not working as expected. When two CSS rules assign different values to the same property of an HTML element, the cascade (along with another core concept in CSS called "specificity") determines which rule takes precedence.

Example 1

files	index.html	styles.css

index.html
styles.css
	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<link`
	`rel="stylesheet"`
	`href="style.css">`
	`</head>`
	`<body>`
	`<p>This is a`
	`paragraph.</p>`
	`</body>`
	`</html>`
		p {color: green;}
		p {color: red;}

Exercise 1 Quick Response
|||Answer: as the second css property defined the p color as red, the paragraph will be red.

Specificity

You may run into a case where two selectors are targeting the same HTML element. That is where the concept of **specificity** comes in. Specificity is a measure of how specific a selector's selection will be. When different CSS selectors target the same property of an HTML element, specificity determines the value that gets applied.

- An **element selector** is *less* specific. Element selectors select all elements of that type, so it has less weight.
- A **class selector** is *more* specific. Class selectors select only the elements on a page that have a specific class value, so it has more weight.

Example 2

files	index.html	styles.css

index.html
styles.css
	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<link rel="stylesheet" href="style-specificity.css">`
	`</head>`
	`<body>`
	`<p class="paragraph">This is a paragraph.</p>`
	`</body>`
	`</html>`
		.paragraph {color: green;}
		p {color: blue;}

Exercise 2 Quick Response

|||Answer:The paragraph's text will be green because the selector 'paragraph' is a class selector. This applies even though the element selector came last. Class selectors are more specific than element selectors.

Additional notes on specificity:

- Adding `!important` to an element will override everything else applied
  - Ex: `color: green !important;`
- Inline CSS has the second-highest specificity. It will override everything besides `!important`
- The 'id' selector has the highest specificity among selectors but lower than `!important` and inline CSS
- The class selector (as well as attribute selector, and pseudo-class) has lower specificity than id, and `!important`

- The element selector, and pseudo-elements are next, below class selectors
- The universal selectors have the lowest specificity
  External sheets with repeated rules will be the last rules will be implemented.

Exercise 3

|||Answer: False. Among selectors, the ID has the highest level of specificity, but both inline CSS and the !important property have higher specificity.

Conclusion & Takeaways

Learning CSS will empower you to create attractive, professionally styled, websites. Knowing how to position and style your elements, and how they interact to populate the page, will give you the knowledge needed to begin creating your own content.

In the next lesson, you will learn about the CSS grid, and some best practices you can use to avoid common pitfalls when designing web pages.

Attribution

1. Mozilla: Learn Cascades (https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Cascade_and_inheritance)

Grid Layout intro

Goals

By the end of this case you will learn about:
- Grid layout
- Elements
- Columns, rows, gaps, lines
- Grid container
- Align content

Introduction

Up to this point in the lesson, you have learned how to apply basic CSS styles and functions to change the appearance of your HTML web pages.

Now, you will dive deeper into CSS grid alignment.

You will also learn some best practices you can follow when using CSS grid to push your project to the next level.

Grid Layout

CSS Grid Layout allows you to divide an HTML page into regions while defining the relationships between regions, in terms of relative size and position. Grid layout enables an author to arrange elements into columns and rows, which can simplify the process of building and modifying responsive web pages.

grid layout example diagram

Elements

First, create a grid container by declaring `display: grid;` or `display: inline-grid;` on an element. When you do, the direct children of the element become grid items. You position content by moving the newly created grid items around with CSS properties.

Example 1
In this example, there is a containing 'div' with a class of wrapper. Inside are five child elements.

files	index.html	style.css
index.html style.css	<!DOCTYPE html> <html> <head> <link rel="stylesheet" href="style.css"> </head> <body> one two three four five </body> </html>	

Next, make the .wrap a grid container.

files	index.html	style.css
index.html style.css	<!DOCTYPE html> <html> <head> <link rel="stylesheet" href="style.css"> </head> <body> one two three four five 	.wrap {display: grid;}

```
</body>
</html>
```

Now the elements are defined as a grid! You can now arrange them with grid properties.

files	index2.html	style2.css
index2.html style2.css	<!DOCTYPE html> <html> <head> <link rel="stylesheet" href="style2.css"> </head> <body> one two three four five </body> </html>	.wrap {display: grid; grid-template-columns: 200px 200px 200px;}

This example will display each of the grid elements as a 200px wide block.

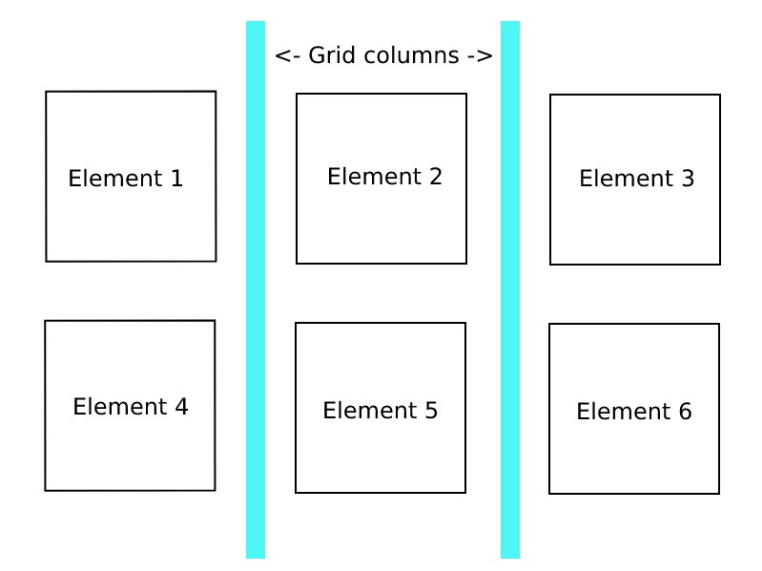
Exercise 1

|||Answer: C The ‘grid-template-columns’ and ‘grid-template-rows’ properties apply to the parent element. To arrange the child elements in the grid, use ‘grid-row’ and ‘grid-column’ on the child elements.

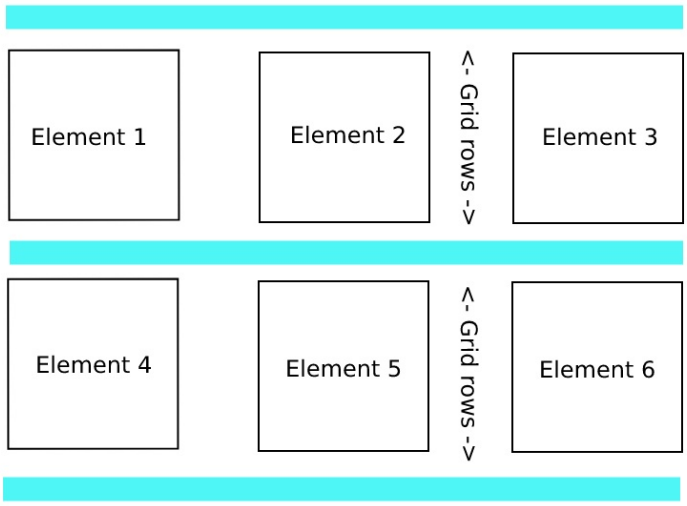
Columns, rows, gaps, and lines

Having established the basic CSS grid container, you can now use grid properties to set the relative positions of the child elements. One of these properties is the CSS `'grid-gap.'` There are three variations of this property. You can see them listed below.

- `grid-column-gap`: Adjust the gap spacing between grid container columns.
- `grid-row-gap`: Adjust the gap spacing between grid container rows.
- `grid-gap`: This is a shorthand property for both columns and rows in a grid container.



grid columns diagram



grid rows diagram

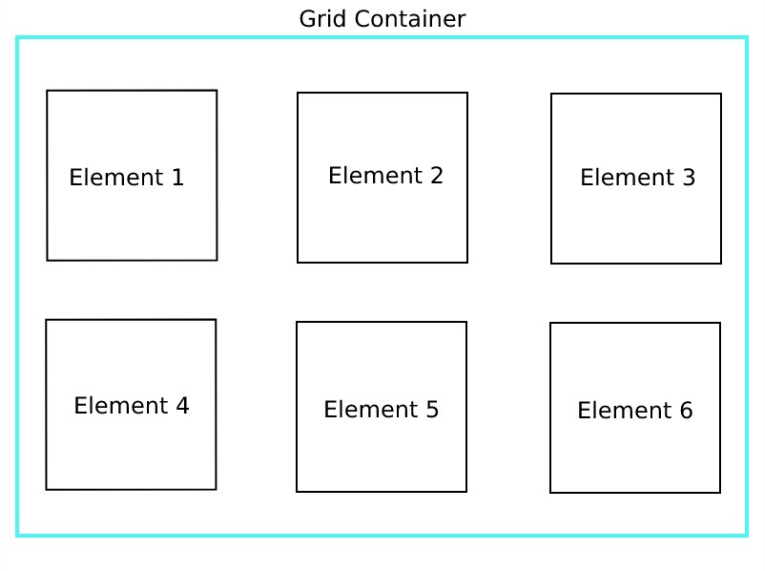
Exercise 2

|||Answer: D

Grid container

To make an HTML element behave as a grid container, you have to set the display property to `grid` or `inline-grid`.

Grid containers consist of grid items, placed inside columns and rows.



grid container diagram

The grid-template-columns Property

The `grid-template-columns` property defines the number of columns in your grid layout, and it can define the width of each column.

The value is a space-separated list, where each value defines the width of the respective column.

If you want your grid layout to contain 4 columns, specify the width of the 4 columns, or `"auto"` if all columns should have the same width. `"auto"` will lead to each column taking up as much space (or as little) as allowed. To set 4 columns of equal width, `'1fr 1fr 1fr 1fr'` is more specific. Click here to learn more.

Example 2
This will create a grid with four columns:

files	index3.html	style3.css

```
<!DOCTYPE html>
<html>
<head>
```

index3.html style3.css	<link rel="stylesheet" href="style3.css"> </head> <body> one two three four </body> </html>	.grid-container{ display: grid; grid-template-columns: auto auto auto auto;}

Exercise 3

|||Answer: True Note: While this is true, if you use ‘auto’ with fractions ‘auto’ will expand to take up the remaining space.

Conclusion & Takeaways

Grid is an important part of modern web design. It allows for content to be easily managed and modified. It’s used often in responsive web design.

Attribution

1. W3 CSS grid (https://www.w3schools.com/css/css_grid.asp)
2. CreativeBlog: CSS Grid (https://www.creativebloq.com/advice/a-comprehensive-guide-to-using-css-grid)
3. Udacity: Using CSS Grid (https://www.udacity.com/blog/2021/06/css-grid-layout-lines-and-gaps-explained.html)
4. Mozilla CSS Grid Basics (https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout/Basic_Concepts_of_Grid_Layout)
5. Unsplash free images (https://unsplash.com/photos/_Y3IuVbPpmU)

Add styles to previous pages

Goals

By the end of this lesson you will know:
- How to best incorporate CSS in different forms:
- External CSS
- Internal CSS
- Inline CSS

Introduction

Clearly written CSS that respects separation of style and structure is easier to work on. Poorly written CSS can create confusion and redundancy. In this lesson, we will go over some best practices you can use to write better CSS.

How to Best Incorporate CSS

Below are the three ways of inserting a style sheet into your HTML file:

- External CSS
- Internal CSS
- Inline CSS

In most cases, using an external stylesheet is the best way to use CSS. An external style sheet allows the CSS to be edited independently of the HTML file. It can then be added easily to multiple web pages or projects. Using internal and inline CSS can make editing a project more difficult, as tracking down where each rule was applied in the document can take a lot of time.

External CSS

It's easier to add and remove entire rulesets with external CSS files. Each HTML page must reference the appropriate external CSS file inside the `<link>` element, within the head section. Multiple external stylesheets, that reference the same elements, also overwrite one another in a cascade.

Reminder: the `<link>` element has no closing tag.

Example 1

files	index.html	style.css

index.html style.css	<!DOCTYPE html> <html> <head> <link rel="stylesheet" href="style.css"> </head> <body> Heading one Sample copy. </body> </html>	h1{color:purple;} p{color:green;}

In the example above, the `style.css` file will be loaded into the HTML, and all rules will be applied to the associated elements.

Internal

Internal CSS

Internal styles, inserted into the HTML document with the `<styles> </styles>` tag, may be used when the web page is unique.

In content management systems, internal styles are often written into HTML templates programmatically. This means that when a user creates a page, the HTML code is automatically generated with the necessary style elements. Additionally, it ensures that all pages have a consistent look and feel.

The internal stylesheet is defined within the opening and closing tags of a `<style>` element, inside the head section in the HTML.

files	index1.html

index1.html	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<style>body{background-`
	`color: #ffffcc;}h1`
	`{color: #cc3300;margin-`
	`left: 40px;}</style>`
	`</head>`
	`<body>`
	`<h1>This is a`
	`heading</h1>`
	`<p>This is a paragraph.`
	`</p>`
	`</body>`
	`</html>`

Inline

Inline CSS

Inline CSS may be used to apply a unique style for a single element.

To use inline styles, add the style attribute directly in the relevant element. Single property changes are inserted between quotation marks. Multiple property changes are delimited with semicolons, as in the stylesheet syntax. **Inline CSS should be avoided.** It defeats the purpose of separating structure from content. Basically, you should be aware of inline styles, you may occasionally see or use inline styles, but it is poor practice. Whenever possible, use a `.css` style sheet for adding styles to your html, even if you only have a small number of things to style.

files	index2.html

index2.html	`<!DOCTYPE html>`
	`<html>`
	`<body>`
	`<h1 style="color: blue;`
	`text-align:`
	`center;">This is a`
	`heading</h1>`
	`<p`
	`style="color:red;">This`
	`is a paragraph.</p>`
	`</body>`
	`</html>`

Exercise 1

|||Answer: B. While internal CSS is usable, external CSS files are the fastest way to swap between rule sheets.

Keep your code neat

Here are some additional tips to keeping your CSS code neat and readable:
1. Use clear comments.
2. Create classes for reusable components.
3. Stick to one color format: Hex, HSLA, or RGBA.
4. Use a company level CSS naming scheme.
5. Introduce an intelligent system for declaring and using variables, functions, and other language features.

Writing readable CSS

Writing readable CSS

There are two ways you will see CSS formatted: *Single-line*, and *Multi-line*. The CSS language doesn't care which one is used. Typically, developers find it more readable to have each property and value pair on a new line.

Single-line:

files	index3.html	style3.css

index3.html
style3.css
	`<!DOCTYPE html>`
	`<html>`
	`<head><link`
	`rel="stylesheet"`
	`href="style3.css">`
	`</head>`
	`<body>`
	`<h1>Heading</h1>`
	`</body>`
	`</html>`
		.scarlet { color: red; }
		h1{color: black;
		padding-top:100px }

Multi-line:

files	index3.html	style3.css

index3.html
style3.css
	`<!DOCTYPE html>`
	`<html>`
	`<head><link`
	`rel="stylesheet"`
	`href="style3.css">`
	`</head>`
	`<body>`
	`<h1>Heading</h1>`
	`</body>`
	`</html>`
		.scarlet {
		color: red;
		}
		
		h1 {
		color: black; padding-
		top:100px
		}

Exercise 2

Conclusion & Takeaways

CSS is a powerful tool that is critical to understand if you want to create world-class web pages. Both the systems around it, as well as best practices must be followed to avoid issues with scalability, responsiveness, as well as

accessibility. This is just the start, but the topics covered in this week will be a foundation for your web creation journey.

Attribution

1. Mozilla Code Organization (https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Organizing#tips_to_keep_your_css_tidy)
2. X-code: 6 best practices (https://x-team.com/blog/css-best-practices/)

Grid of Artists

Goals

By the end of this lesson you will:
- Take a series of images, and add them to a grid
- Add captions below each artist
- Style the elements in the rows and columns
- Practice coding in css

Introduction

Now that we have learned about CSS, it is time to create a page using the files provided. Create a 4x4 grid with photos of artists. Add a caption below each image. You should also style the elements in the grid.
>

Instructions

1. First, you will want to create your grid container in the body of the HTML

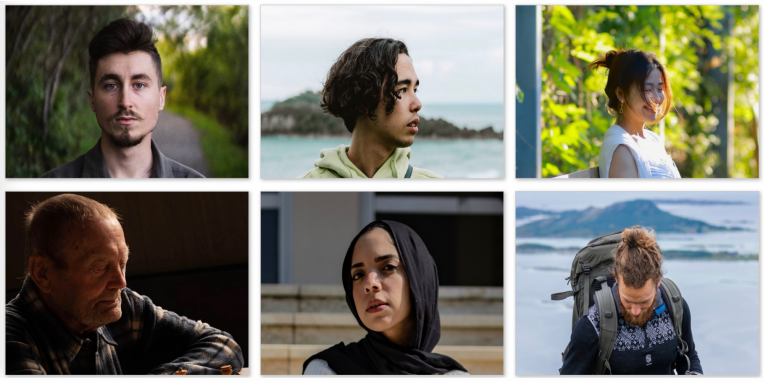
files	index.html	styles.css

index.html	`<!DOCTYPE html>`
	`<html>`
	`<head>`
	`<link`
	`rel="stylesheet"`
	`href="style.css">`
	`</head>`
	`<body>`
	`<main class="grid">`
	`</main>`
	`</body>`
	`</html>`	

2. Add the CSS for your grid in an external stylesheet. Make sure there are gaps between the grid columns and rows. The grid contents should take up the full width of their respective grid cells. Ensure there is a drop shadow applied to each image.

3. Next, Add in your images to the grids in the body of the HTML

Once all your images are added, it should begin to look like the following:

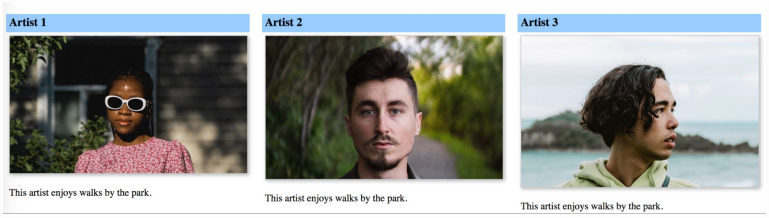


sample grid of artist photos

These are our artists! But we know nothing about them, or who they are.

4. Next, we want to add in headings above each of the artists to introduce them, as well as a paragraph below to hold a description or comment about each image. Make sure the heading text is centered, has a background color, and a border.
5. Use `<h2>` tags for the headings. Typically there is still only one `<h1>` tag per page, usually the page title. If there are no `<h2>` tags, there should be no `<h3>` tags, and so on...

The page should look like the following.

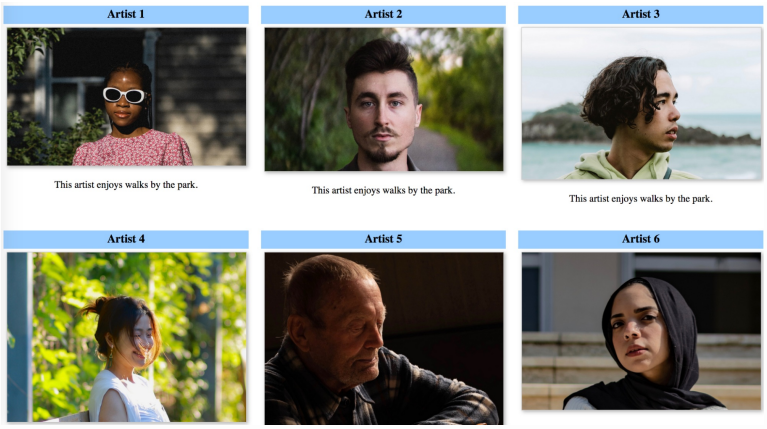


sample grid of artist photos with captions

6. Next, tighten up the look and feel of the boxes. Center align both the headings and captions in the grid container.

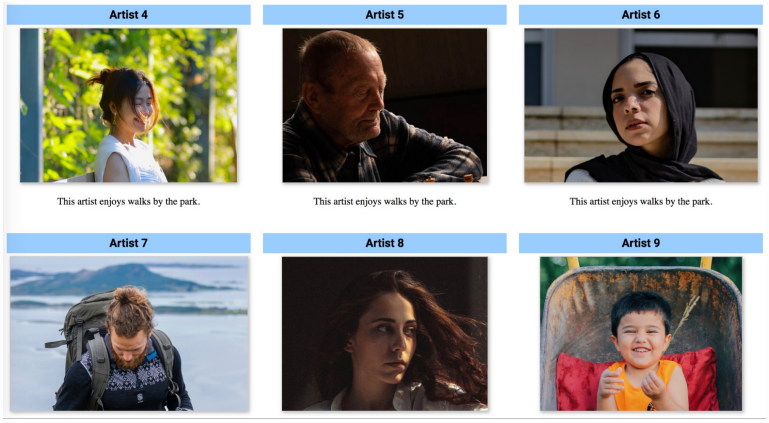
This class can be added to both header and paragraph elements:

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



sample grid of artist photos with captions and centered text

7. And the grid is ready! Time for some finishing touches. Make sure the image heights are aligned, and that they are centered below their heading. Additionally, change the header heading font to Roboto. If students want, they can add in names and bios for the artists. Aligning the images can be done in a number of ways. In this case, I added the center class to the divs that contain the images and adjusted the height of the `<img>` element to 250px.



sample grid of artist photos with adjusted grid margins

We have our grid!
>

Challenge yourself!

If you want to take the exercise further, try the following:

– Add an introduction section consisting of a title, and paragraph
– Try and have them make this section visually distinct from the rest using borders and background colors
– Add a background image to the grid
– Add links to the images, as well as hover effects

Conclusion & Takeaways

If you want a career in digital advertising or web development it is helpful to learn CSS. Learning CSS will also give you a better understanding of the web pages that make up the modern web. In this lesson, we have put together all of the material that we've covered this week by creating a complex CSS-formatted web page.

Project 3: Add CSS

Adding CSS

> So far, your personal webpage consists solely of html, possibly with in-line styling. Now that we have gotten comfortable with css, you are going to add css styling to your website! This means writing a css styling file in order to make your webpage more visually engaging.
>
> For your personal website, choose and add:
> - a font family
> - indentation
> - borders
> - backgrounds and colors to text headings and paragraphs
> - grid organization to text and/or images
>
> When complete, share your html and css code. Be sure your css file is "linked" to your html file. These files have been set up for you here, as well as a preview pane. As your website grows, the preview pane will not be as useful and it is likely you will want to open your preview as a web page so you can see how it truly looks.
>
> Things to keep in mind:
> - make your page aesthetically pleasing
> - make your page easy to read and navigate

"Code is like humor. When you have to explain it, it's bad." – Cory House