Lab 02: *CSS* Frontend

CSS Crash Course

CSS

Requires: Chrome Browser

Table of Content: *Implementing a Simple (HTTP) Web Server*

# of Parts	Duration	Topic	Page
Introduction	*10 minutes*	**Lab Introduction** Define CSS, Declarations, Properties	2
Goal 0	*0 minutes*	**Setup HTML/CSS Files** Advice for designing & writing a Interactive Story	4
Goal 1	*10 minutes*	**CSS Selectors** element selector, class selector, id selector	6
Goal 2	*10 minutes*	**CSS Colors** background colors, font colors	8
Goal 3	*10 minutes*	**CSS Fonts** font family, font size, font types	10
Goal 4	*10 minutes*	**CSS Spacing** padding, margin, outline, border	12
Goal 5	*10 minutes*	**CSS Sizing** width, height, max-width, max-height	14
Goal 6	*10 minutes*	**CSS Backgrounds** background image, repeat image, fixed image, gradients	16
Goal 7	*10 minutes*	**CSS Customizations** pseudo-class selectors, text decorations	18
Goal 8	*10 minutes*	**CSS Aligning with Flexbox** center, left, right, between, around, evenly	20
Goal 9	*10 minutes*	**CSS Aligning with Gridview** column templates, row templates	22
End	*10 minutes*	**Concluding Notes** Summary and Submission notes	24
Homework	*n/a*	**Style your own Interactive Story** Use CSS to style your own story	

Lab Introduction

Prerequisites

None. You must have the Chrome browser, and a code editor.

Motivation

Understand CSS and use it to style various properties of the HTML in the browser viewport

Goal

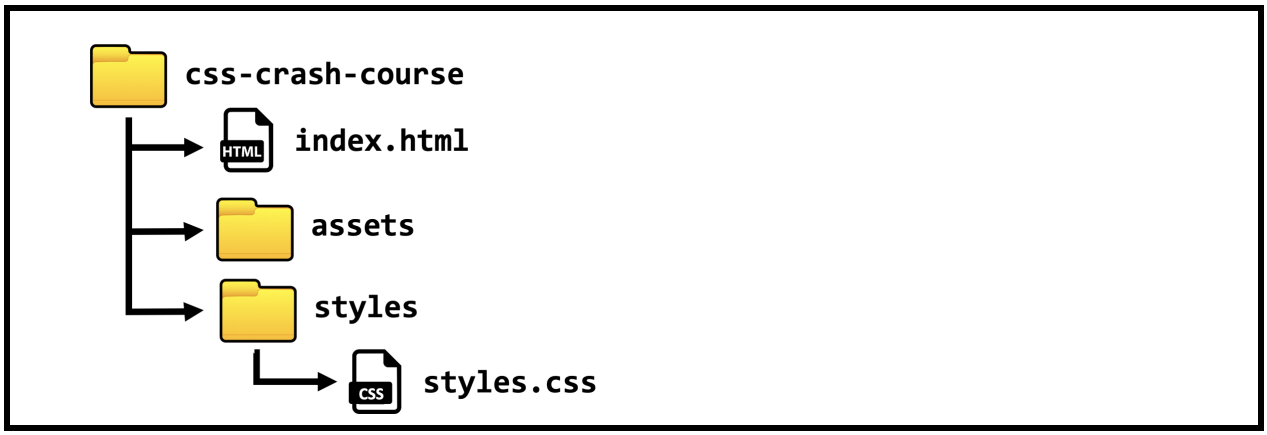
Change the colors, fonts, spacing, sizes, background, & alignment of the default styles of HTML

Learning Objectives

- Structure of a CSS file
- CSS Declarations
- CSS Properties
- CSS Selectors
- Flexbox vs Gridview
- Google Fonts API

Project Architecture:

Start this project by downloading the starter files from github.
Create all necessary files and folders as illustrated below.



Download Starter files:

https://github.com/scalemailted/css-crash-course/archive/master.zip

Concepts

Cascading Style Sheets (CSS)

CSS is the standard styling language for styling Web pages. CSS consists of a series of declarations. CSS is linked in the HTML's head element.

- **Declarations:** All CSS statements are defined as declarations. The basic format of a declaration consist of: selector { property: value; }
- **Selectors:** A selector is used by CSS to select a HTML element for styling. There are 3 basic types of selectors: element selector, class selector, id selector.
- **Structure of CSS file:** CSS declarations are all defined consecutively one after the other. A style applied to an HTML element cascades down to all of its children, hence CSS

Fonts

There are browser supported fonts & Google fonts

- **Generic types:** There are five generic types of fonts: Serif, Sans-Serif, Monospace, Cursive, Fantasy. Each has its own use case.
- **Google fonts**: A free library of over 1000+ fonts that work in browsers.

Responsive Web Design

Responsive web design makes your web page look good on all devices. Landscape mode on laptops and portrait mode on phones both have varying sized viewports. A responsive styling changes the display to best suit the device.

CSS Resources

Here are resources & examples for CSS:

References: https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
Tutorials: https://developer.mozilla.org/en-US/docs/Web/CSS#tutorials
Examples: https://www.awwwards.com/
Playground: https://codepen.io/

Goal 0: Setup HTML/CSS Files

'Approach' → Plan phase

This lab needs a HTML file: **`index.html`** in project folder & a CSS file: **`styles.css`** in the **styles** folder

---

'Apply' → Do phase

Step 1: Initialize HTML with Head & Body elements

index.html

```
<!-- This is the HTML for Lab 2: CSS Crash Course -->
<html>
   <head> </head>
   <body> </body>
</html>
```

Step 2: *(HTML)* Head: Favicon & Title for Browser tab

index.html → <head>

```
<head>
    <title> CSS Crash Course </title>
    <link rel="icon" href="assets/css-favicon.png" />
</head>
```

Step 3: *(HTML)* Body: Heading element

index.html → <body>

```
<body>
    <h1> CSS Crash Course </h1>
    <hr>
    <!-- Goal 1: CSS Selectors -->
    <!-- Goal 2: CSS Colors -->
    <!-- Goal 3: CSS Fonts -->
    <!-- Goal 4: CSS Spacing -->
    <!-- Goal 5: CSS Sizing -->
    <!-- Goal 6: CSS Backgrounds -->
    <!-- Goal 7: CSS Customizations (to Default HTML Styles) -->
    <!-- Goal 8: CSS Aligning with Flexbox -->
    <!-- Goal 9: CSS Aligning with Gridview -->
</body>
```

Step 4: *(CSS)*: Comments

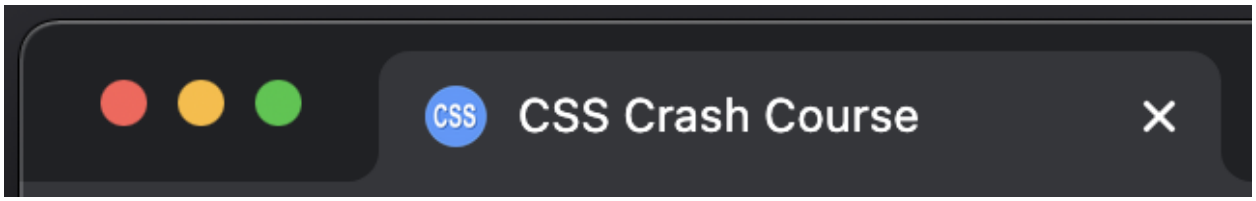
styles/styles.css

```
/*This is the CSS for Lab 2: CSS Crash Course */

/* Goal 1: CSS Selectors */
/* Goal 2: CSS Colors */
/* Goal 3: CSS Fonts */
/* Goal 4: CSS Spacing */
/* Goal 5: CSS Sizing */
/* Goal 6: CSS Backgrounds */
/* Goal 7: CSS Customizations (to Default HTML Styles) */
/* Goal 8: CSS Aligning with Flexbox */
/* Goal 9: CSS Aligning with Gridview */
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.

Browser Tab



Browser Viewport

Goal 1: CSS Selectors

'Approach' → Plan phase

There are 3 basic types of CSS selectors: Element selector, Class selector, and ID selector.

'Apply' → Do phase

Step 1: *(HTML)*: Link to **`styles.css`** as a stylesheet

index.html → <head>

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

Step 2: *(HTML)*: Add new HTML, some have `class` or `id` attributes

index.html → <body>

```
<!-- Goal 1: CSS Selectors -->
<h3> Goal 1: Selectors </h3>
<p> The body is set to gray using the element selector </p>
<p class='class-selector'> This element was selected by its class name </p>
<p id='id-selector'> This element was selected by its identifier </p>
<hr>
```

Step 3: *(CSS)*: Add an element selector, class selector, & id selector

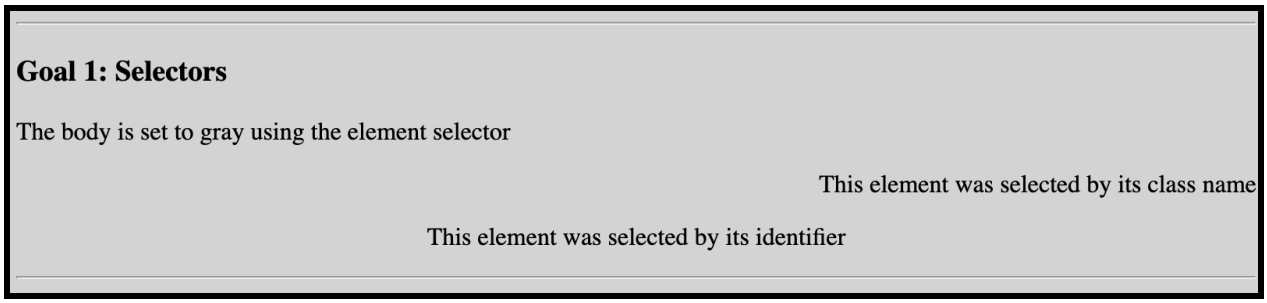
styles/styles.css

```
/* Goal 1: Selectors */
body {
   background-color: lightgray;
}

.class-selector{
   text-align: right;
}

#id-selector{
   text-align: center;
}
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.

Browser Viewport → Goal 1: Selectors



Goal 2: CSS Colors

'Approach' → Plan phase

CSS has styles for background colors and font colors. Colors be declared by name, rgb, or hex values

---

'Apply' → Do phase

Step 1: *(HTML)*: Add new HTML, some with two `class` values

index.html → <body>

```
<!-- Goal 2: CSS Colors -->
<h3> Goal 2: Colors </h3>
<p class='bg-green text-green'> This is green background with green text </p>
<p class='bg-blue text-red'> This is blue background with red text </p>
<p class='bg-red text-blue'> This is red background with blue text </p>
<p class='bg-dark text-white'> This is black background with white text </p>
<hr>
```

Step 2: *(CSS)*: Class selectors for background colors

styles/styles.css → / Goal 2: CSS Colors */*

```
/*Goal 2: CSS Colors*/

/*background colors*/
.bg-white{
   background-color: rgb(255,255,255);
}

.bg-green{
   background-color:rgb(0,255,0);
}

.bg-red{
   background-color: rgb(255,0,0);
}

.bg-blue{
   background-color: rgb(0,0,255);
}

.bg-yellow{
   background-color: rgb(255,255,0);
}

.bg-dark{
   background-color: rgb(0,0,0);
}
```

Step 3: *(CSS)*: Class selectors for text colors

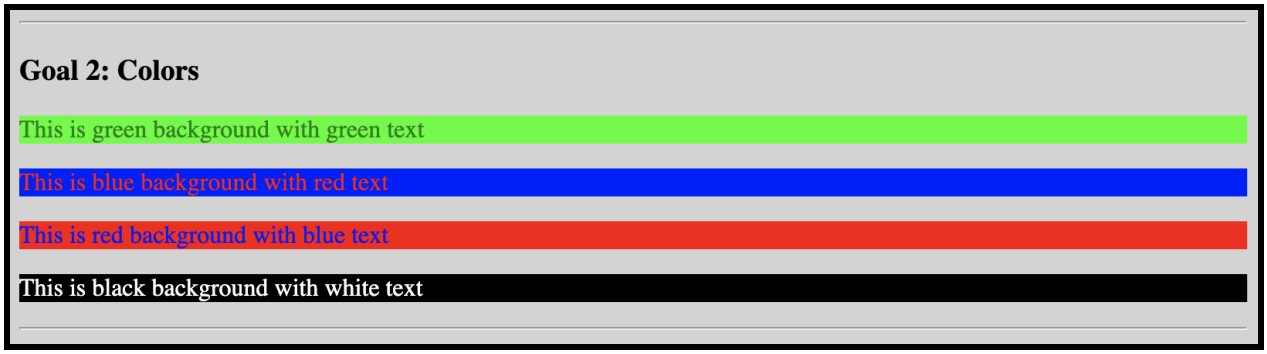
styles/styles.css → `/* Goal 2: CSS Colors */`

```css
/*font colors*/
.text-red{
   color: red;
}

.text-blue{
   color: blue;
}

.text-green{
   color: green;
}

.text-white{
   color: white;
}
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.

Browser Viewport → Goal 2: Colors



Goal 3: CSS Fonts

'Approach' → Plan phase

CSS has styles for fonts, font sizes, font thicknesses. Google provides a free library with 1000+ fonts

---

'Apply' → Do phase

Step 1: *(HTML)*: Link Bangers font from Google fonts as stylesheet

index.html → `<head>`

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

Step 2: *(HTML)*: Add new HTML under the goal 3 comments

index.html → `<body>`

```
<!-- Goal 3: CSS Fonts -->
<h3> Goal 3: Fonts </h3>
<p>
    <span class='bold'>This is bold text</span>
    <span class='italic'>This is italicized text</span>
    <span class='large-text'>This is large text</span>
    <span class='small-text'>This is small text</span>
    <span class='arial-font'>This is a browser font called arial</span>
    <span class='bangers-font'>This is a google font called bangers</span>
</p>
<hr>
```

Step 3: *(CSS)*: Class selectors for font properties

styles/styles.css → `/*Goal 3: CSS Fonts*/`

```
/* Goal 3: CSS Fonts */

/* font properties */
.bold{
   font-weight: bold;
}

.italic{
   font-style: italic;
}

.large-text{
   font-size: 22px;
}

.small-text{
   font-size: small;
}
```

Step 4: *(CSS)*: Class selectors for font families: Browser font & Google font

*styles/styles.css → /*Goal 3: CSS Fonts*/*

```
/* font families */

.arial-font{
   font-family: 'Arial';
}

.bangers-font{
   font-family: 'Bangers';
}
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.

Browser Viewport → Goal 3: Fonts

Goal 4: CSS Spacing

'Approach' → Plan phase

HTML elements spacing defined by box model: [margin [border [padding [element] padding] border] margin]

'Apply' → Do phase

Step 1: *(HTML)*: Add new HTML under the goal 4 comments

index.html → *<body>*

```
<!-- Goal 4: CSS Spacing →

<h3> Goal 4: Spacing </h3>

<div class='outline border'> outline versus border on element </div>
<div class='margin outline border'> margins are on the outside of the outline/border </div>
<div class='padding outline border'> paddings are on the inside of the outline/border </div>

<!-- This is a container div to give spacing to all its children elements -->
<div class='container outline'>
    <h4> Heading in a container </h4>
    <img src='assets/400x100.png'>
    <p> Text in a container </p>
</div>
<hr>
```

Step 2: *(CSS)*: Class selectors for box model stylings

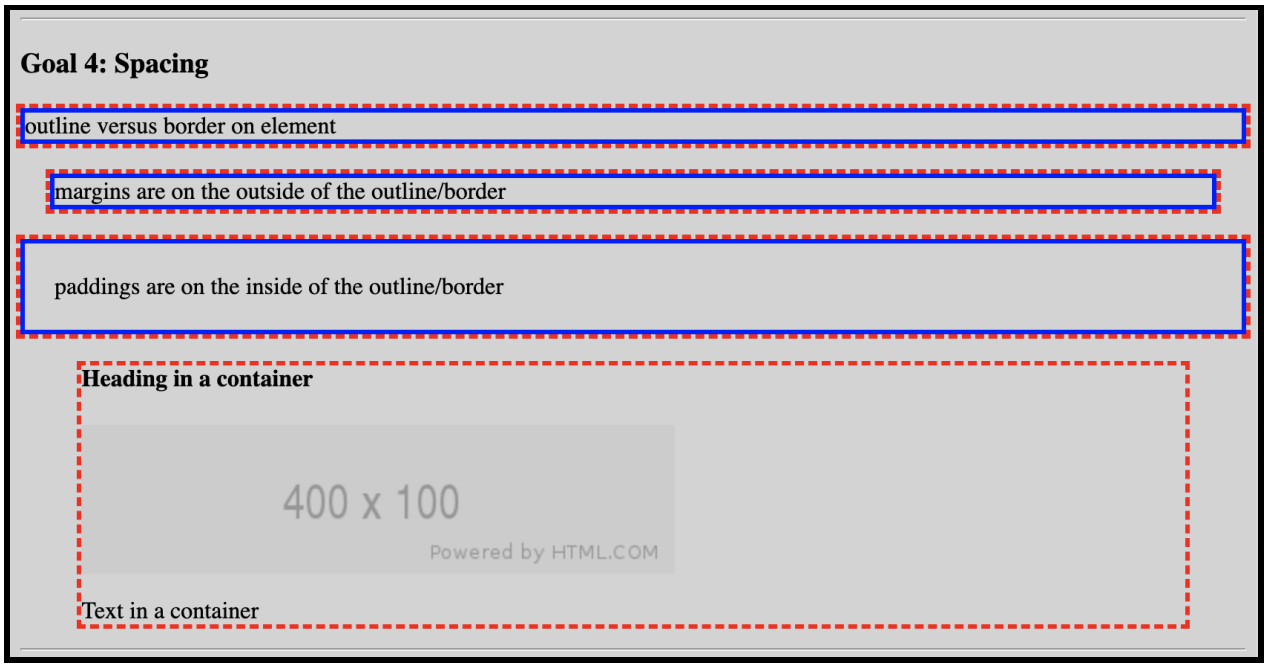
styles/styles.css → */*Goal 4: CSS Spacing*/*

```
/* Goal 4: CSS Spacing */

.outline{
   outline-width: 3px;
   outline-style: dashed;
   outline-color: red;
}

.border{
   border-width: 3px;
   border-style: solid;
   border-color: blue;
}

.padding{
   padding: 20px;
}

.margin{
   margin: 20px;
}

.container{
   width: 90%;
   margin: auto;
}
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.

Browser Viewport → Goal 4: Spacing



Goal 5: CSS Sizing

'Approach' → Plan phase

Width/Height property forces all element into a size whereas Max Width/Height applies only if its too big

---

'Apply' → Do phase

Step 1: *(HTML)*: Add new HTML under the goal 5 comments

index.html → *<body>*

```
<!-- Goal 5: CSS Sizing -->

<h3> Goal 5: Sizing </h3>

<p> This image is normally too big to fit in the viewport </p>
<img class='shrink-to-viewport' src='assets/4000x1000.png'>

<p>This image already fits in the viewport so its not resized</p>
<img class='shrink-to-viewport' src='assets/400x100.png'>

<p> This image fills the viewport width regardless if its too small or too big </p>
<img class='force-full-viewport' src='assets/400x100.png'>
<hr>
```

Step 2: *(CSS)*: Class selector for dynamically resizing elements

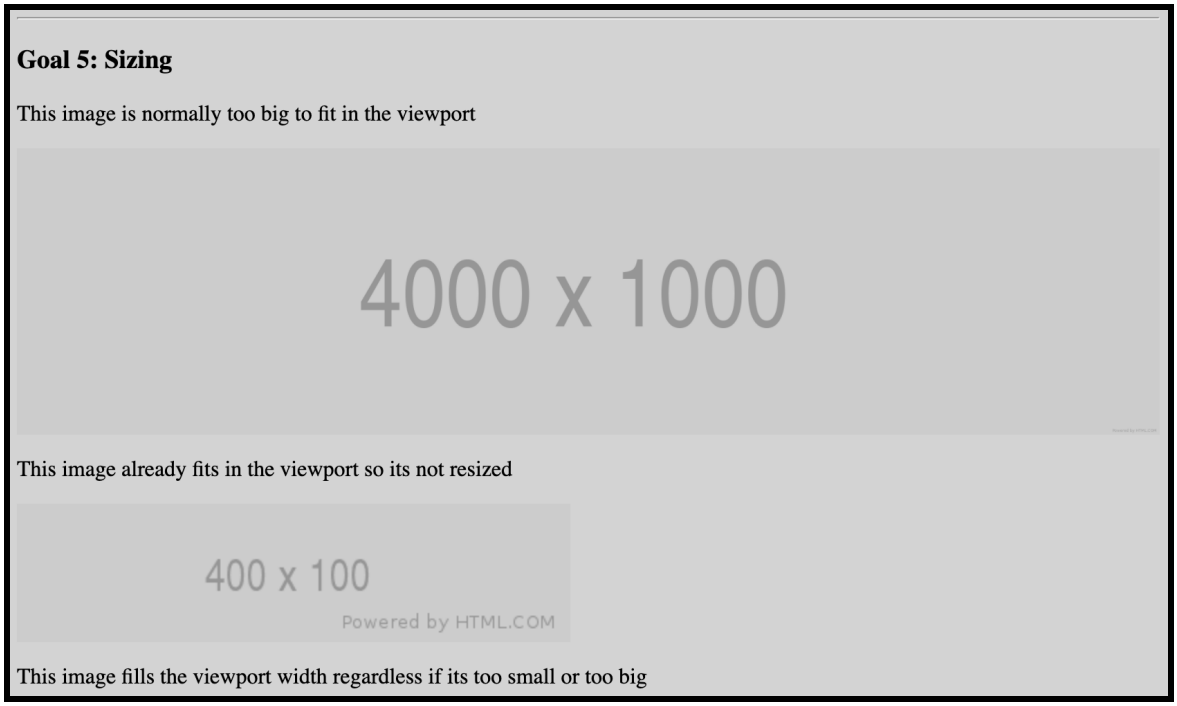
styles/styles.css → */*Goal 5: CSS Sizing*/*

```
/* Goal 5: CSS Sizing */

.force-full-viewport{
   width:100%;
   height:100%;
}

.shrink-to-viewport{
   max-width:100%;
   max-height:100%;
}
```

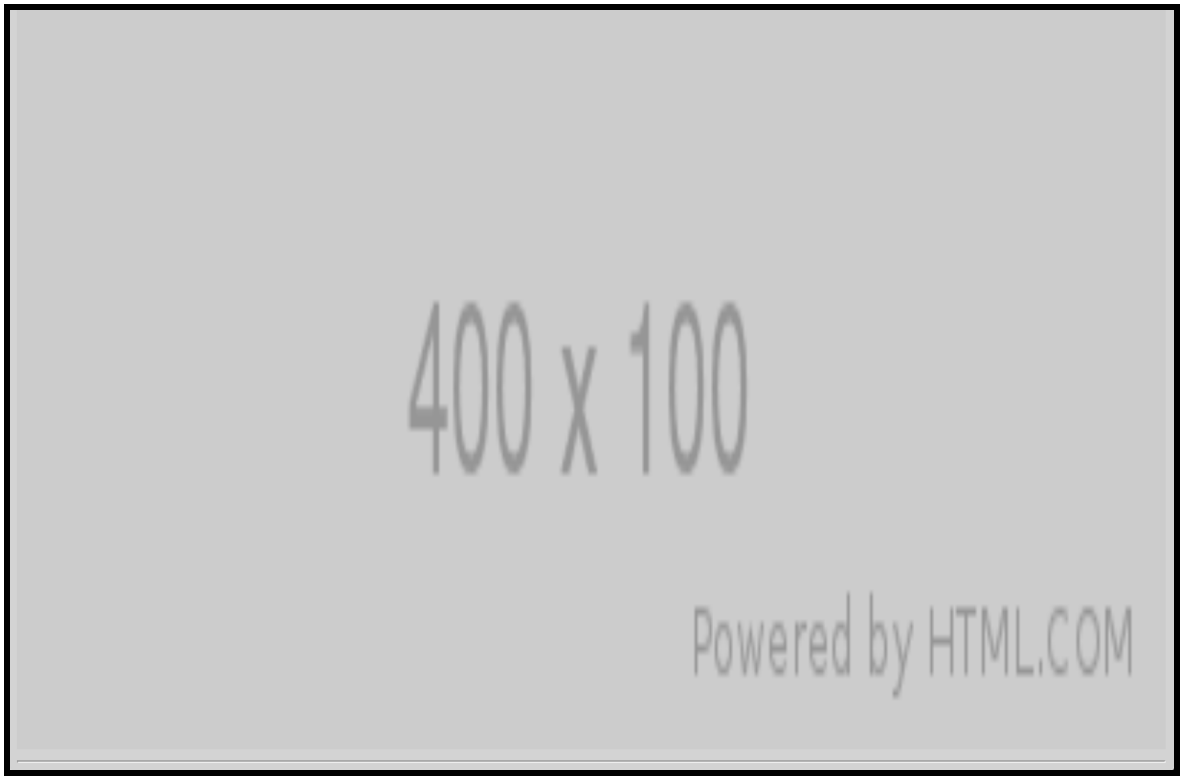
'Assess' → Test phase

Open the **`index.html`** page in the browser.

Browser Viewport → Goal 5: Sizing



Browser Viewport → Last image fills the viewport



Goal 6: CSS Backgrounds

'Approach' → Plan phase

CSS has styles for the background with colors, images, and gradients.

'Apply' → Do phase

Step 1: *(HTML)*: Add new HTML under the goal 6 comments

index.html → `<body>`

```
<!-- Goal 6: CSS Backgrounds -->
<h3> Backgrounds </h3>

<p> The background is an image </p>
<div class="background-image container outline">
    <h4> Heading on top of background image </h4>
    <img src='assets/400x100.png'>
    <p> Text on top of background image </p>
</div>

<p> The background is an image & does not repeat </p>
<div class="background-image-without-repeat container outline">
    <h4> Heading on top of background image </h4>
    <img src='assets/400x100.png'>
    <p> Text on top of background image </p>
</div>

<p>The background is an image & is fixed in place</p>
<div class="background-image-fixed container outline">
    <h4> Heading on top of background image </h4>
    <img src='assets/400x100.png'>
    <p> Text on top of background image </p>
</div>

<p>The background is a gradient of colors</p>
<div class="background-gradient container outline">
    <h4> Heading on top of background image </h4>
    <img src='assets/400x100.png'>
    <p>Text on top of background image</p>
</div>
<hr>
```

Step 2: *(CSS)*: class selectors for background image, *path via 'styles' folder*

styles/styles.css → `/*Goal 6: CSS Backgrounds*/`

```
/* Goal 6: CSS Backgrounds */
.background-image{
   background-image: url('../assets/bg-image.jpg');
}

.background-image-without-repeat{
   background-image: url('../assets/bg-image.jpg');
   background-repeat: no-repeat;
   background-position: center;
}
```

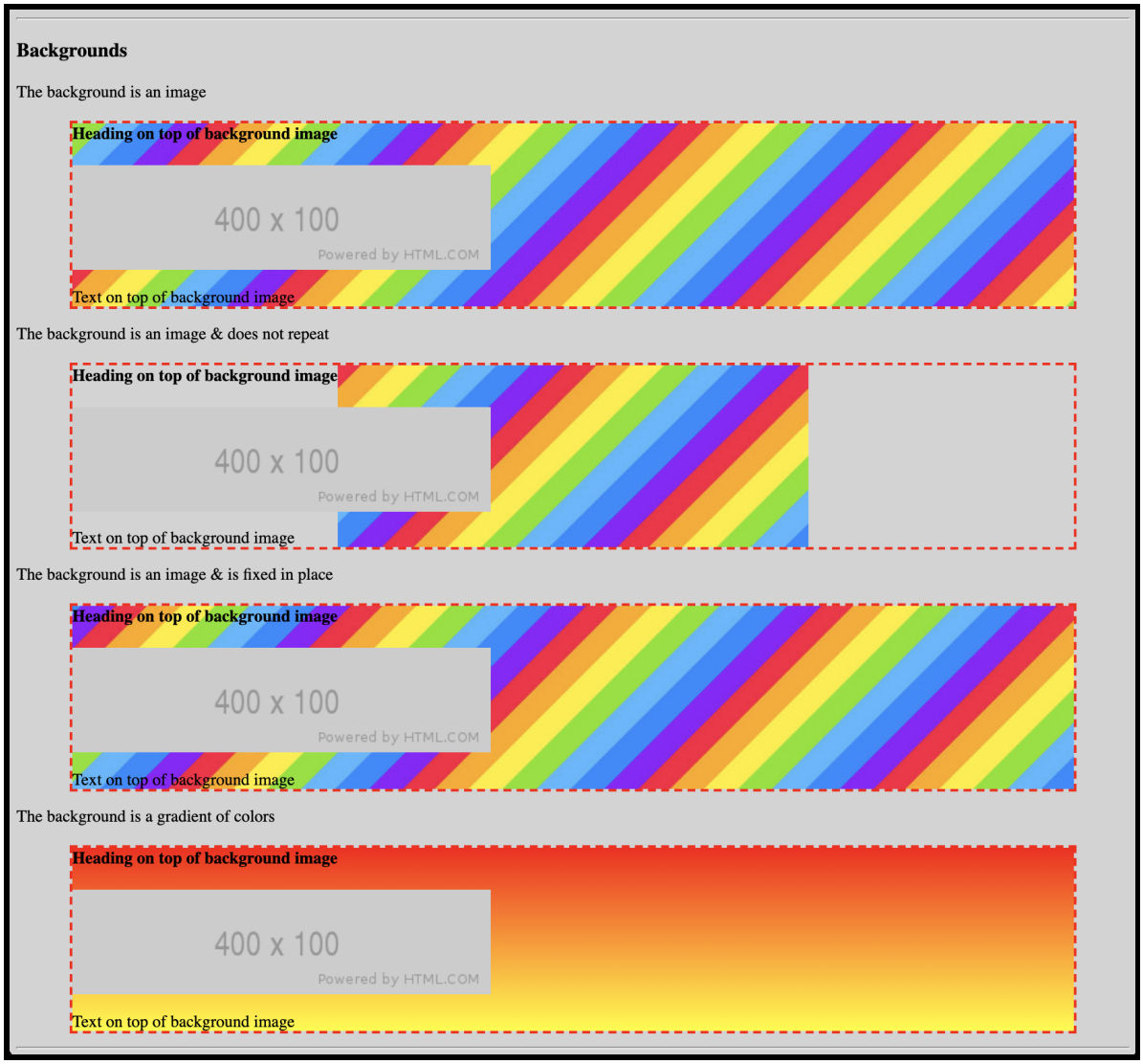
Step 3: *(CSS)*: class selectors: fixed background image & color gradient

styles/styles.css → `/*Goal 6: CSS Backgrounds*/`

```
.background-image-fixed{
   background-image: url('../assets/bg-image.jpg');
   background-attachment: fixed;
}

.background-gradient{
   background-image: linear-gradient(red, yellow);
}
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.



Goal 7: CSS Customizations

'Approach' → Plan phase

CSS customizes default styles for all the common HTML elements. Pseudo-class selectors target 'states'

'Apply' → Do phase

Step 1: *(HTML)*: Add new HTML under the Goal 7 comments

index.html → <body>

```
<!-- Goal 7: CSS Customizations -->
<h3> Customizations </h3>

<div class='container'>
    <p>
        <span class='bold'>Anchors:</span>
        Removes the default underline & changes text to red when mouse hovers over
    </p>
    <a href='#'>Hover over me!</a>

    <p>
        <span class='bold'>Buttons:</span>
        Change background color to red when mouse hovers over
    </p>
    <button>Hover over me!</button>

    <p>
        <span class='bold'>Lists:</span>
        Indent all List Items & replace bullets with images
    </p>
    <ul>
        <li>Item 1</li>
        <li>Item 2</li>
        <li>Item 3</li>
    </ul>

    <p>
        <span class='bold'>Tables:</span>
        Row padding and alternating table row colors
    </p>
    <table>
        <tr>
            <th>1</th>
            <th>2</th>
            <th>3</th>
        </tr>
        <tr>
            <td>4</td>
            <td>5</td>
            <td>6</td>
        </tr>
        <tr>
            <td>7</td>
            <td>8</td>
            <td>9</td>
        </tr>
    </table>
</div>
<hr>
```

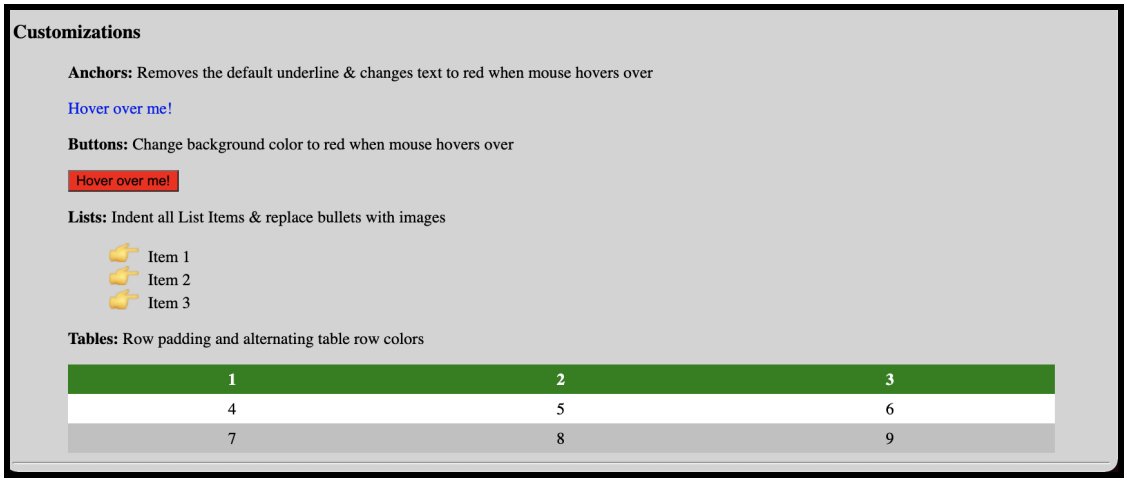
Step 2: *(CSS)*: class selectors to customize all common HTML elements

styles/styles.css → `/*Goal 7: CSS Customizations*/`

```css
/*Goal 7: CSS Customizations*/
a {
   text-decoration: none;
 }

a:hover{
   color:red;
 }

button:hover {
   background-color: red;
}

ul {
   list-style-image: url('../assets/list-item-image.png');
   list-style-position: inside;
}

table {
   border-collapse: collapse;
   width: 100%;
}

th, td {
   text-align: center;
   padding: 5px;
}

tr:nth-child(even){
   background-color: white;
}

tr:nth-child(odd){
    background-color: silver;
}

th {
   background-color: green;
   color: white;
}
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.



Goal 8: CSS Aligning with Flexbox

'Approach' → Plan phase

CSS aligns HTML elements using flexbox model, responsive to viewport's size.

'Apply' → Do phase

Step 1: *(HTML)*: Add new HTML under the Goal 8 comments

index.html → <body>

```
<!-- Goal 8: CSS Aligning with Flexbox -->
<h3> Aligning with Flexbox </h3>

Flexbox to left align block-level elements
<div class='left border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>

Flexbox to center align block-level elements
<div class='center border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>

Flexbox to right align block-level elements
<div class='right border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>

Flexbox to justify align block-level elements, maximizing space between elements
<div class='justify-space-between border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>

Flexbox to justify align block-level elements, centering space between elements
<div class='justify-space-around border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>

Flexbox to justify align block-level elements, maximizing space between elements
<div class='justify-space-evenly border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>
<hr>
```

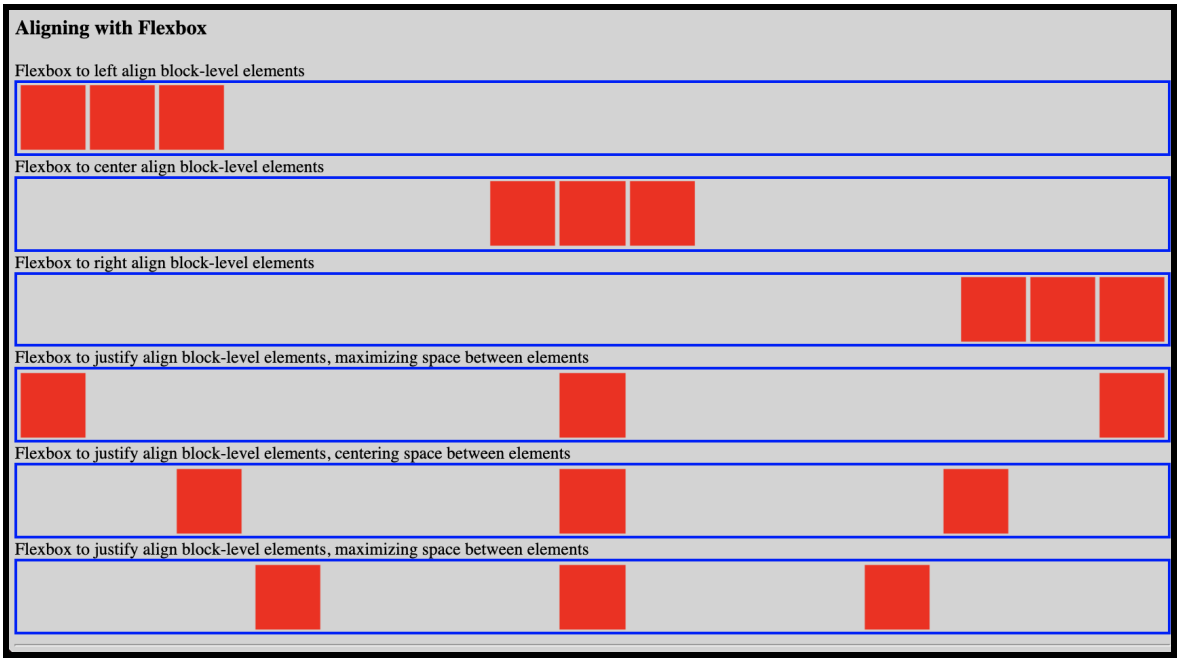
Step 2: *(CSS)*: class selectors to customize all common HTML elements

styles/styles.css → `/*Goal 8: CSS Aligning with Flexbox*/`

```
/*Goal 8: CSS Aligning with Flexbox*/
.center{
   display: flex;
   justify-content: center;
}

.left {
   display: flex;
   justify-content: flex-start;
   }

.right{
   display: flex;
   justify-content: flex-end;
}

.justify-space-between {
   display: flex;
   justify-content: space-between;
}

.justify-space-around{
   display: flex;
   justify-content: space-around;
}

.justify-space-evenly{
   display: flex;
   justify-content: space-evenly;
}
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.



Goal 9: CSS Aligning with Gridview

'Approach' → Plan phase

CSS aligns HTML elements using Gridview model, allowing content defined by rows and columns.

'Apply' → Do phase

Step 1: *(HTML)*: Add new HTML under the Goal 9 comments

index.html → *<body>*

```
<!-- Goal 8: CSS Aligning with Gridview -->
<h3> Aligning with Grid </h3>

Gridbox with 1 column
<div class='grid-1col border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>

Gridbox with 2 columns
<div class='grid-2col border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>

Gridbox with 3 columns
<div class='grid-3col border'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
    <img src='assets/square-tile.gif'>
</div>
<hr>
```

Step 2: *(CSS)*: class selector for a grid with 1 column and auto rows

styles/styles.css → */*Goal 9: CSS Aligning with Gridview*/*

```
/* Goal 8: CSS Aligning with Gridview */

.grid-1col{
   display: grid;
   grid-template-columns: repeat(1, 1fr);
   grid-template-rows: auto;
   justify-items: center;
}
```

Step 3: *(CSS)*: class selector for grid with 2 columns and auto rows

styles/styles.css → `/*Goal 9: CSS Aligning with Gridview*/`

```
.grid-2col{
   display: grid;
   grid-template-columns: repeat(2, 1fr);
   grid-template-rows: auto;
   justify-items: center;
}
```

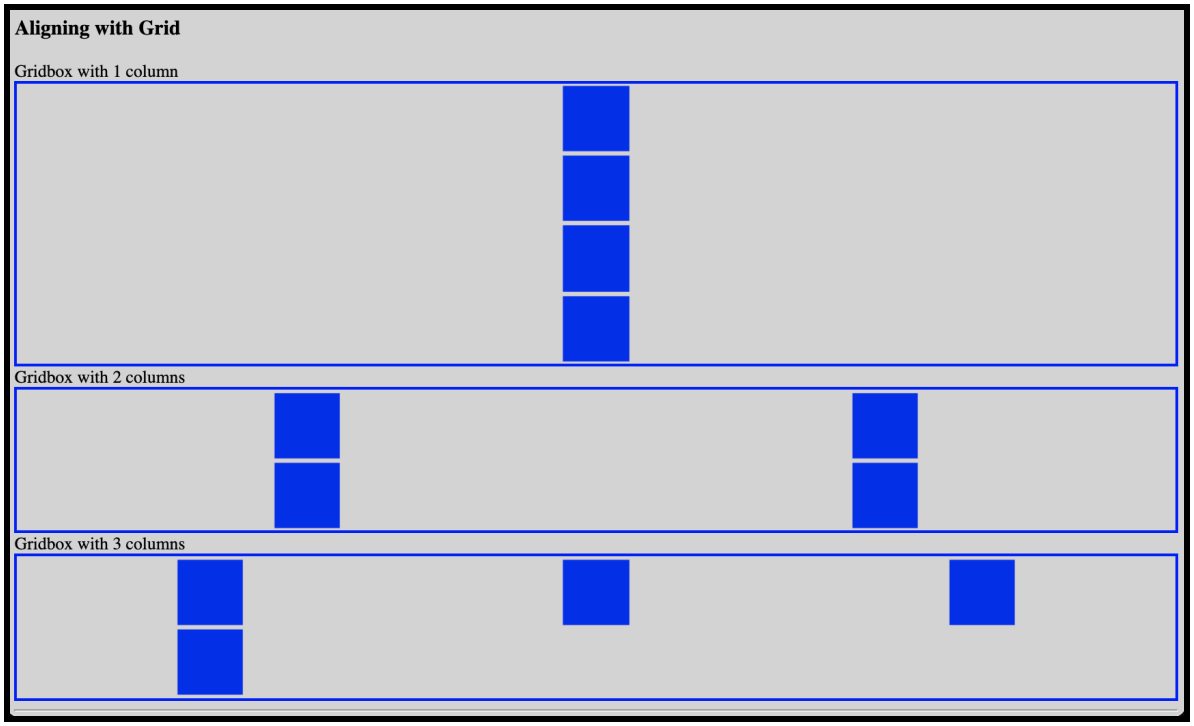
Step 4: *(CSS)*: class selector for grid with 3 columns and auto rows

styles/styles.css → `/*Goal 9: CSS Aligning with Gridview*/`

```
.grid-3col{
   display: grid;
   grid-template-columns: repeat(3, 1fr);
   grid-template-rows: auto;
   justify-items: center;
}
```

'Assess' → Test phase

Open the **`index.html`** page in the browser.



Conclusions

Final Comments

In this lab you learned to implement CSS. This required defining CSS declarations along with CSS selectors. You should understand the difference between flexbox display & gridview display. This lab covered many CSS properties such as: color, background-color, font-family, font-size, font-type, padding, margin, outline, border, width, height, background-image, text-decoration.

Future Improvements

- Practice with additional CSS properties not found in this lab
- Try learning about CSS media queries

Lab Submission

Compress your project folder into a zip file and submit on Moodle.

Homework 2:

Add CSS styling to your own unique Interactive Story from Homework 1. Use a single CSS file to define styles for your multiple HTML files. You must create beautiful looking, visually appealing, and thematic styles for your entire story.

Homework Bonus:

Showcase bonus. You can receive up to 20 bonus points if your styles are compelling and novel.
I'll publish all showcase projects on UNO's web page as a demo for future students. You should cite such projects on your resume.

End.