CSS Layout

Lesson 5

Learning Objectives

- Explore how to select nested elements to apply styling.
- Style a webpage using classes and ids
- Experiment and predict effects of floats & clearing CSS positioning
- Apply multi-column layouts using CSS Flexbox to develop a webpage.

Agenda

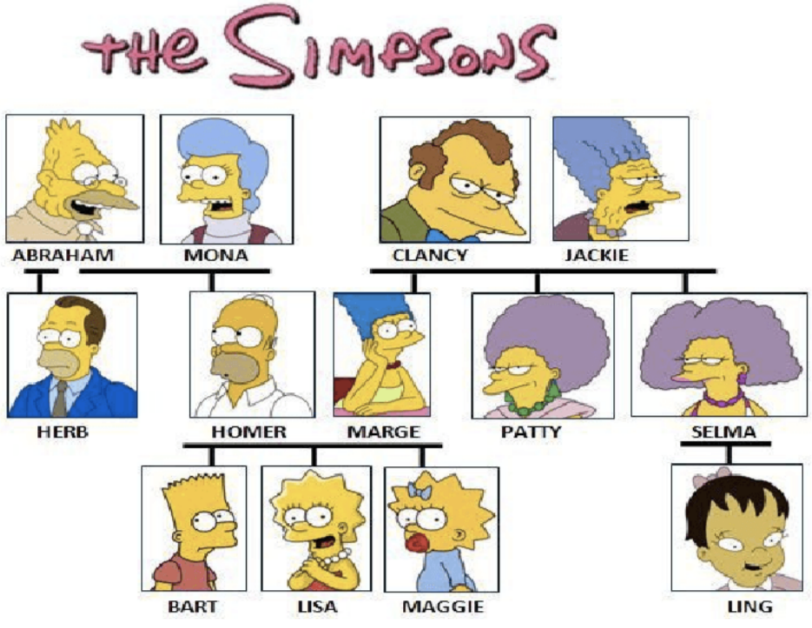
- Nested Elements
- Classes & IDs
- Floats
- Flexbox
- Lab Time

Nested Elements

Page structure - Relationships

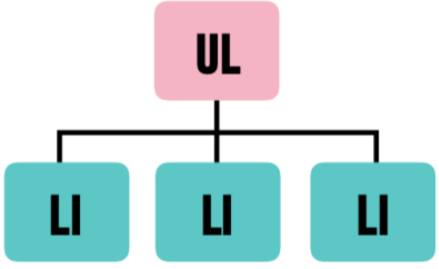
Relationships

We often use the same terms we would use to describe relationships between family members to describe the relationships between HTML elements.



Parent/Child

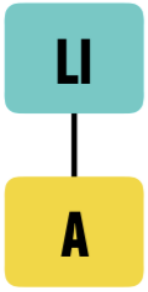
- Here, we can say that our <ul> element is the **parent** of our <li> elements
- We can also say that our <li> elements are **children** of the <ul> element




```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>The Evolution of Denim</title>
    </head>
    <body>
        <h1>The Evolution of Denim</h1>
        <ul>
            <li>Dark Wash <a href="linkgoeshere">Jeans</a></li>
            <li>Stone Wash</li>
            <li>Chambray</li>
        </ul>
    </body>
</html>
```

Nested Elements

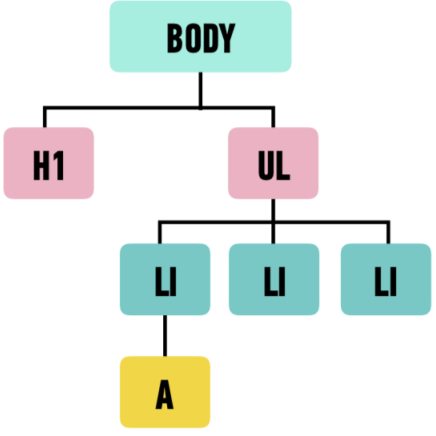
- Similarly, we can have an <a> tag that is nested inside, or wrapped by, our <li> element




```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>The Evolution of Denim</title>
    </head>
    <body>
        <h1>The Evolution of Denim</h1>
        <ul>
            <li>Dark Wash <a href="linkgoeshere">Jeans</a></li>
            <li>Stone Wash</li>
            <li>Chambray</li>
        </ul>
    </body>
</html>
```

Nested Elements (cont'd)

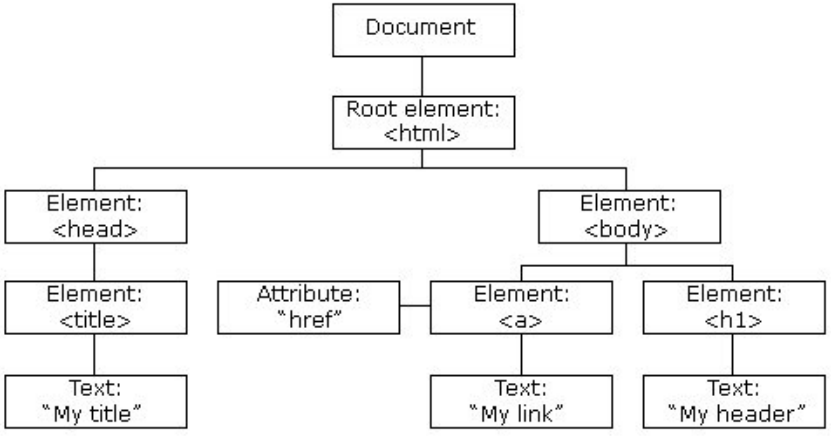
- By extension, we can say that all of our HTML content is "wrapped" by the body, or "nested" inside the body since they are within the open/close <body> tags




```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>The Evolution of Denim</title>
    </head>
    <body>
        <h1>The Evolution of Denim</h1>
        <ul>
            <li>Dark Wash <a href="linkgoeshere">Jeans</a></li>
            <li>Stone Wash</li>
            <li>Chambray</li>
        </ul>
    </body>
</html>
```

DOM Tree

This should look familiar, right?



Nested Selectors

Most Common CSS Selectors

SELECTOR:	MEANING:	EXAMPLE:
TYPE	Selects an element	`a {}`
DESCENDANT	Selects an element that is a **descendent** of another element	`p a {}`
UNIVERSAL	Selects **all** elements in a document	`* {}`
MULTIPLE	Select multiple elements	`h1, h2 {}`

Nested Selector Example

Which element here will be italic?

```
p a {
    font-style: italic;
}
```

Nested Selector Example (cont'd)

- The **last element** in the selector string is *always* the one that we are styling.
- Try reading right-to-left!

```
p a {
    font-style: italic;
}
```

Code Along

Open: **starter_code/nested_selectors**

Key Objective

Open the **lesson_05_css_layouts/starter_code/nested_selectors** folder and style the nested selectors on the page together.

Timing:

- **5 minutes** - style the nested elements on the page

Directions:

- Change font color for all em tags inside of p tags to be lightblue
- Change font color for blockquote to be #333
- Change font color for all em tags inside of p tags inside of blockquotes to be lightgreen

Classes & IDs

Targeting specific elements

Classes & **IDs** are HTML attributes that allow us to add "labels" to elements so we can target them in our CSS.

ID Attribute

- **Ids** are used to target *one specific element*
- Each element can only have one id
- **Important**: two elements on the same page cannot have the same id

```
<nav id="main-nav">Content</nav>
```

```
#main-nav {
    text-align: center;
}
```

Class Attribute

- **Classes** are used to group elements together
- Each element can have multiple classes (separated by a space)
- Multiple elements can have the same class name

```
<div class="alert message">Content</div>
```

```
.alert {
    color: red;
    font-size: 20px;
}
```

Good to know...

- Avoid class and ID names that are similar to names already used by HTML and CSS
- Avoid "classitus"! Only use classes when you need to
- BEST PRACTICE: Add a class or ID to the parent and use nested selectors when possible, instead of adding a class or id to each individual element in a section.

Resources

- The Difference Between ID and Class (CSS Tricks)
- CSS Naming Conventions that will save you Hours of Debugging (Medium Article)

Code Along

Open: **starter_code/error_message**

Key Objective

Open the **lesson_05_css_layouts/starter_code/error_message** folder and style the page using classes and ids.

Timing:

- **7 minutes -** style the page using classes and ids together
- **3 minutes -** add a success div and style it differently

Directions:

- Add different background color, border, and padding to the notification ID and message class
- Add a new background color and border to the error class and the success class (you do)

Nested Selectors w/ Classes & IDs

```
.contact p a {
    color: red;
}

#main-title h1 {
    font-size: 36px;
}
```

Complex Selectors

There are a number of other ways to select elements in CSS. Check out the resource below:

Complex CSS Selectors Resource

Slack Check #1

How would you select the **a** elements inside of the main-nav class?

```
<header>
    <h1><a href="">Nested Selectors</a></h1>

    <nav class="main-nav">
        <a href="">Home</a>
        <a href="">About</a>
        <a href="">Contact</a>
        <a href="">Blog</a>
    </nav>
</header>
```

Slack Check #2

How would you select the **span** element inside of the paragraph inside of the section?

```
<header>
    <p class="content-text">Welcome to my <span>Amazing</span> website!</p>
</header>
<section>
    <h2>About Me</h2>
    <p class="content-text">Lorem ipsum <span>consectetur adipisicing</span> elit.</p>
    <p>Consectetur adipisicing elit</p>
    <button>Read More</button>
</section>
```

Slack Check #3

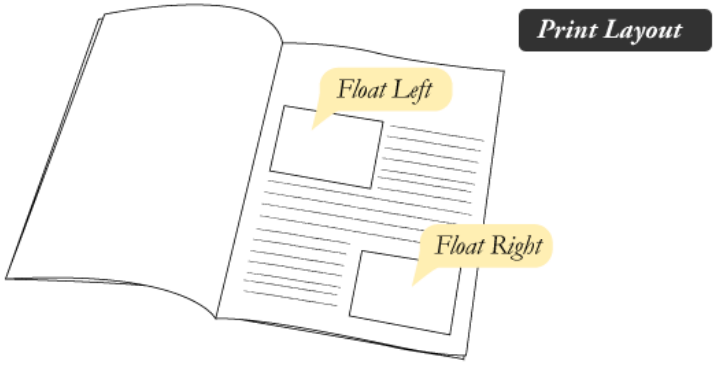
How would you select the button with the id of **about-button** element inside of the section?

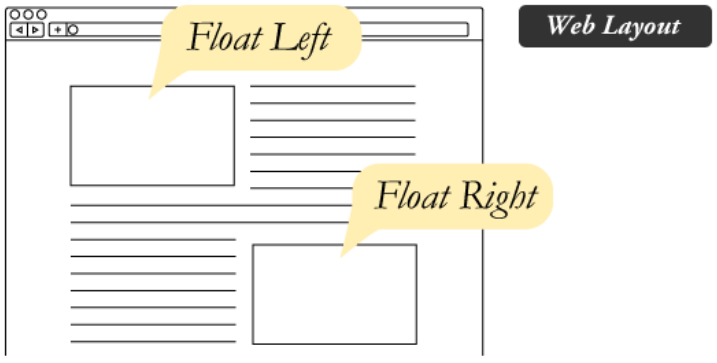
```
<section id="about">
    <h2 id="about-heading">About Me</h2>
    <p id="about-paragraph">Lorem ipsum dolor sit amet.</p>
    <button id="about-button">Read More</button>
</section>
```

CSS Float

Float

The CSS **float** property is used for positioning and layout on web pages. It essentially allows certain content to flow around the floated element.





Float Properties

```
img {
    /* The element floats to the left of its container */
    float: left;

    /* The element floats to the right of its container */
    float: right;

    /* Default. The element does not float (will be displayed just where it occurs in the text) */
    float: none;
}
```

Clear

The CSS **clear** property specifies what elements can float beside the cleared element and on which side.

```
p {
    /* No floating elements allowed on the left side */
    clear: left;

    /* No floating elements allowed on the right side */
    clear: right;

    /* No floating elements allowed on either the left or the right side */
    clear: both;

    /* Default. Allows floating elements on both sides. */
    clear: none;
}
```

Code Along

Open: Basic Float Codepen

Floated Layouts

In the past, developers commonly used the CSS float property to create multi-column layouts.

Code Along

Open: Floated Layouts Codepen

Intro to Flexbox

Traditional Multi-column Layouts

In the past, developers used floats to build their multi-column grid layouts, but Flexbox is now considered best practice for grid layouts.

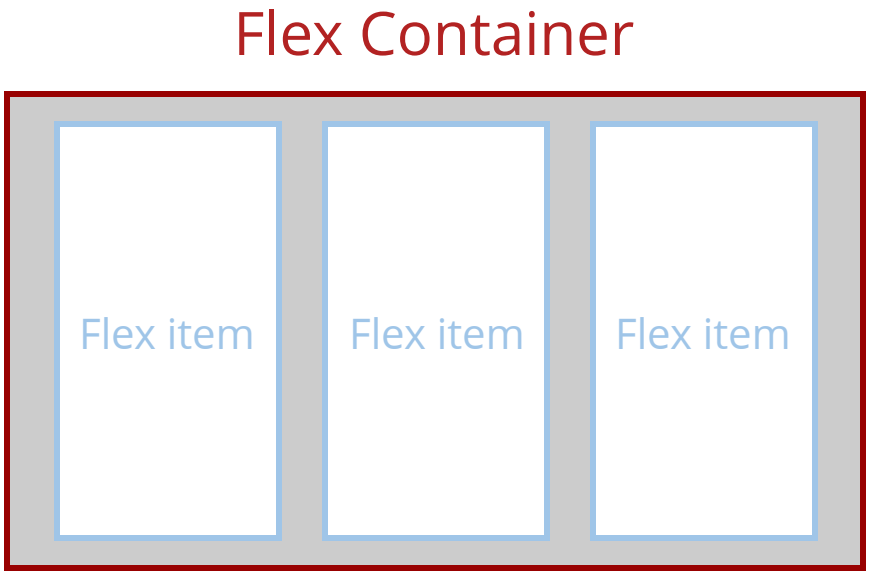
In addition to creating incredible structure on our page, Flexbox solves a lot of layout issues that float either causes or cannot solve (or both!).

What is Flexbox?

Flexible Box Layout Module

The Flexible Box Layout Module -- or **Flexbox** -- makes it easier to design flexible responsive layout structure without using float or positioning.

Flexbox Layout



Flexbox HTML Structure

```
<section>

    <article>
        <h3>Title 1</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>

    <article>
        <h3>Title 2</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>

    <article>
        <h3>Title 3</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>

</section>
```

Flex Container

The parent wrapper is our **flex container**

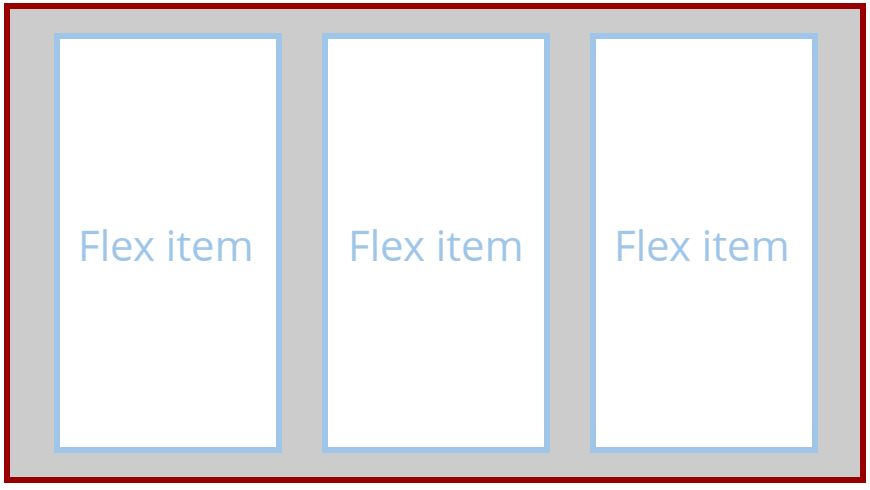
```
<section>

    <article>
        <h3>Title 1</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>

    <article>
        <h3>Title 2</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>

    <article>
        <h3>Title 3</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>

</section>
```

Flex Container

Flex Items

The direct children are the **flex items**

```
<section>
    <article>
        <h3>Title 1</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>
    <article>
        <h3>Title 2</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>
    <article>
        <h3>Title 3</h3>
        <p>Lorem ipsum dolor sit elit.</p>
    </article>
</section>
```



Multi-column Layouts

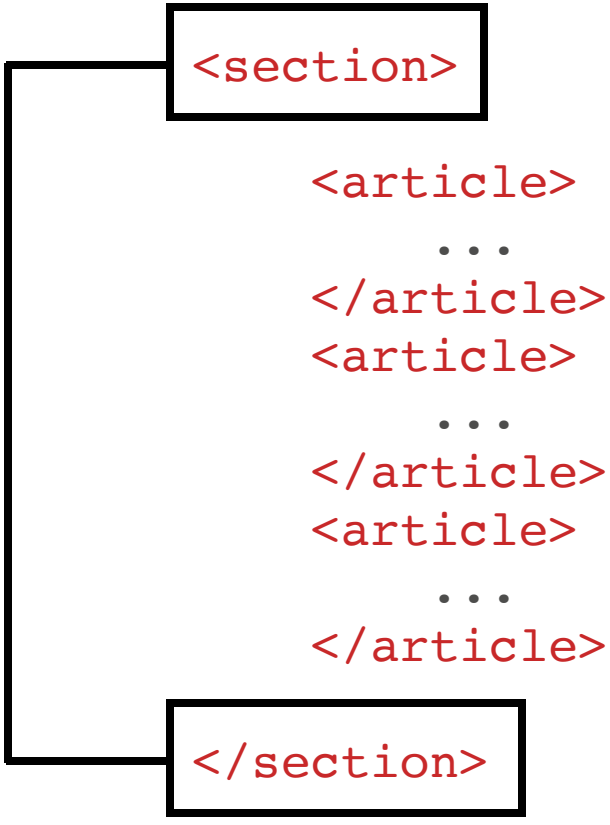
Code Along

Open: Intro to Flexbox

How does Flexbox Work?

Parent wrapper (flex container)

In order to get started with Flexbox, you need to make your wrapper element into a flex container:



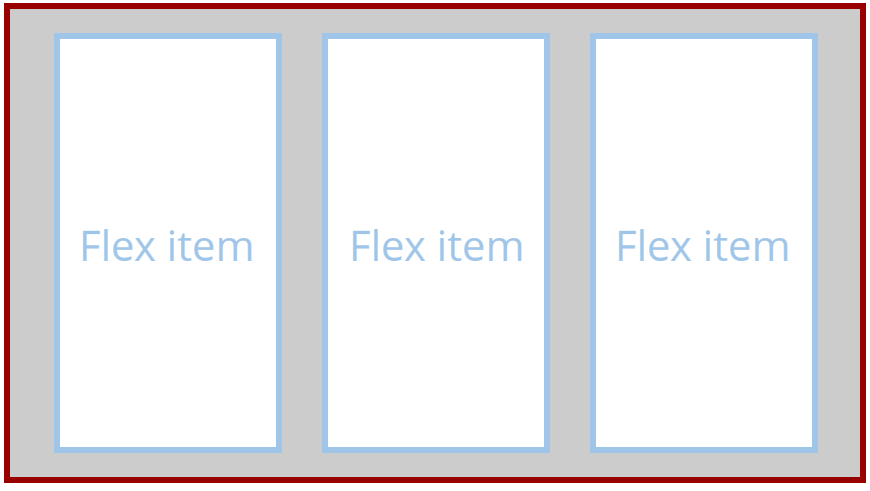

```
section {
    display: flex;
}
```

By default, items are arranged along the main axis, from left to right.

Creating a Flexbox

```
display: flex;
```

Flex Container



Multi-column Layout Columns

To create flex container, simply add:

```
display: flex;
```

Then, set a width in percentages on the flex items.

Key Objective:

Open **starter_code/beatles_columns** folder. Work through steps to create a multi-column layout

Timing:

- **10 minutes** - work on the in-class lab
- **~2 minutes** - Review as a class

Directions:

1. Identify where the flex container/items need to go
2. Add applicable HTML and CSS to achieve a 2-column layout that resembles the included screenshot.

BEATLES – DRIVE MY CAR

LYRICS

Asked a girl what she wanted to be
She said baby, "Can't you see
I wanna be famous, a star on the screen
But you can do something in between"

Baby you can drive my car
Yes I'm gonna be a star
Baby you can drive my car
And maybe I love you

I told a girl that my prospects were good
And she said baby, "It's understood
Working for peanuts is all very fine
But I can show you a better time"

I told that girl I can start right away
And she said, "Listen baby I got something to say
I got no car and it's breaking my heart
But I've found a driver and that's a start"

Baby you can drive my car
Yes I'm gonna be a star
Baby you can drive my car
And maybe I love you
Beep beep'm beep beep yeah
Beep beep'm beep beep yeah
Beep beep'm beep beep yeah
Beep beep'm beep beep yeah (fade out)

ALBUM INFO

Rubber Soul is the sixth studio album by English rock band the Beatles, released on 3 December 1965.

- **Album:** Rubber Soul
- **Release Date:** December 3, 1965

Break time!

Let's take 5-10 minutes to decompress...

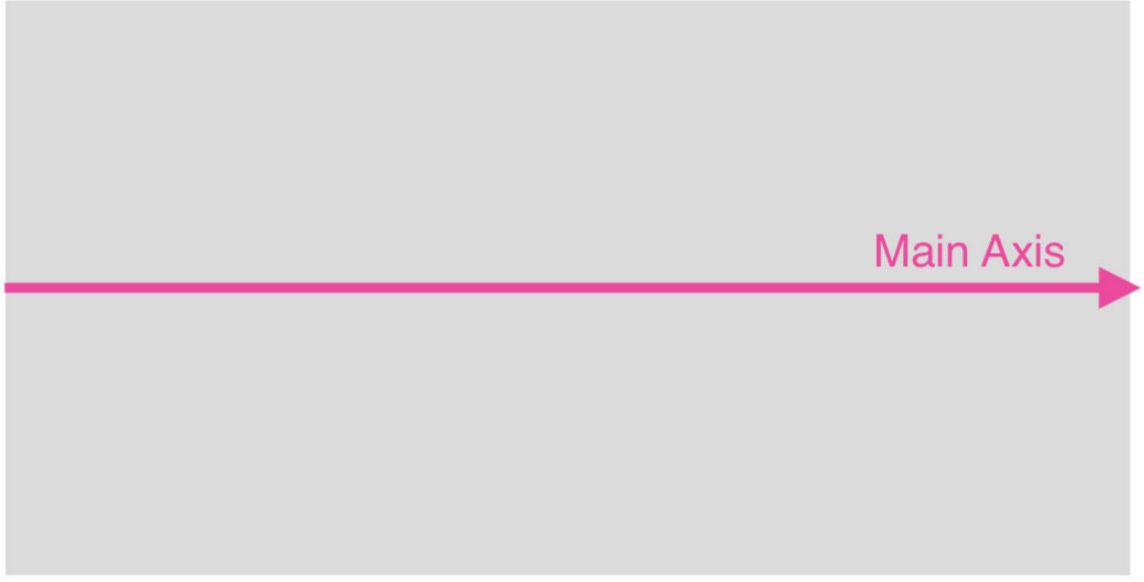
Flexbox CSS

Code Along

Open: Flexbox CSS

justify-content

The **justify-content** property controls how you align items on the main axis.

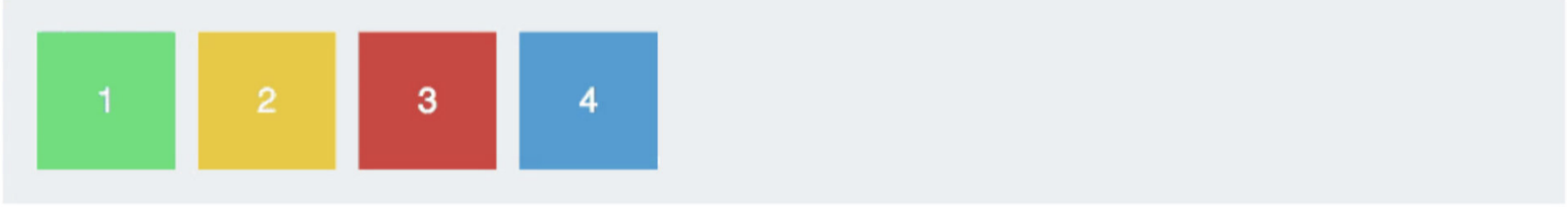


justify-content Property

```
flex-container-element {
    display: flex;

    /* Default value. Items are positioned at the beginning of the container */
    justify-content: flex-start;

    /* Items are positioned at the end of the container */
    justify-content: flex-end;

    /* Items are positioned at the center of the container */
    justify-content: center;

    /* Items are positioned with space between the lines */
    justify-content: space-between;

    /* Items are positioned with space before, between, and after the lines */
    justify-content: space-around;
}
```

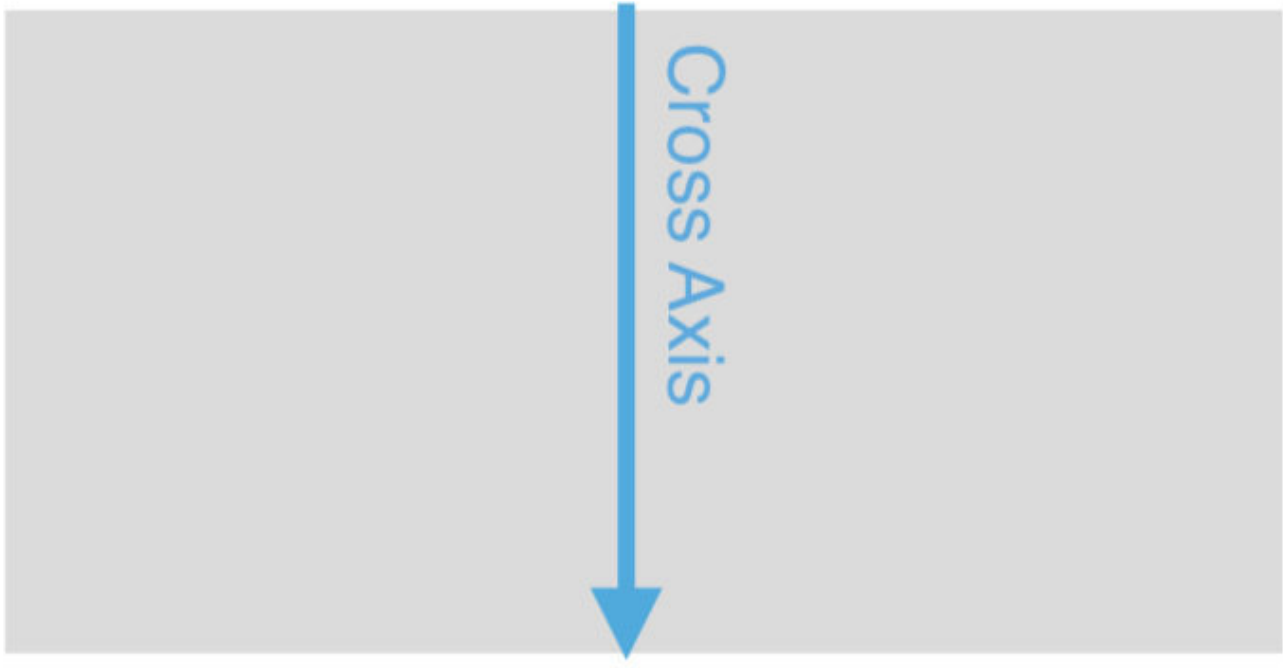
justify-content Example

```
justify-content: flex-start;
```



align-items

The **align-items** property controls how you align items on the cross axis.

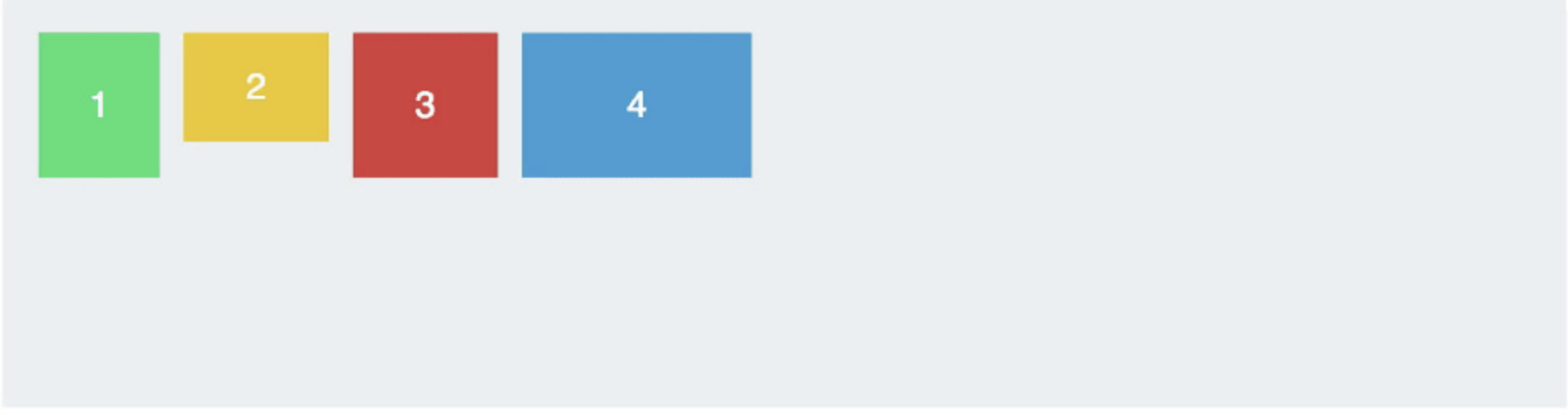


align-items Property

```
flex-container-element {
    display: flex;

    /* Items are positioned at the beginning of the container */
    align-items: flex-start;

    /* Items are positioned at the end of the container */
    align-items: flex-end;

    /* Items are positioned at the center of the container */
    align-items: center;

    /* Default. Items are stretched to fit the container */
    align-items: stretch;

    /* Items are positioned at the baseline of the container */
    align-items: baseline;
}
```

align-items Example

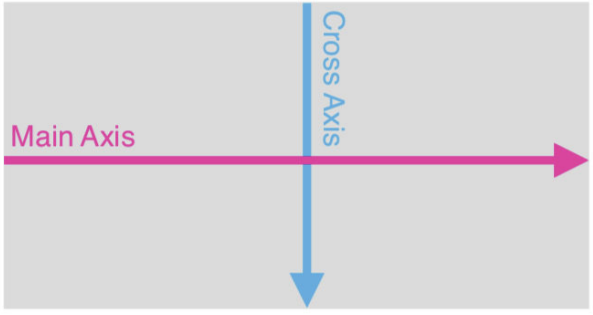
```
align-items: flex-start;
```



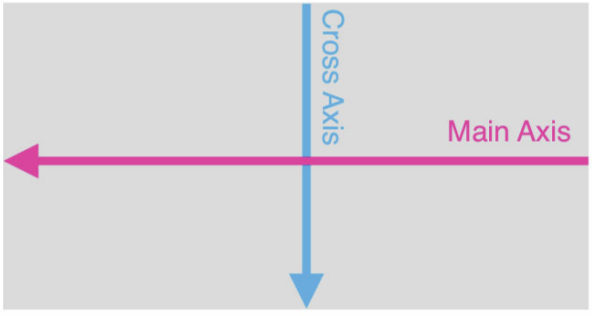
flex-direction

The **flex-direction** property lets you rotate the main axis.

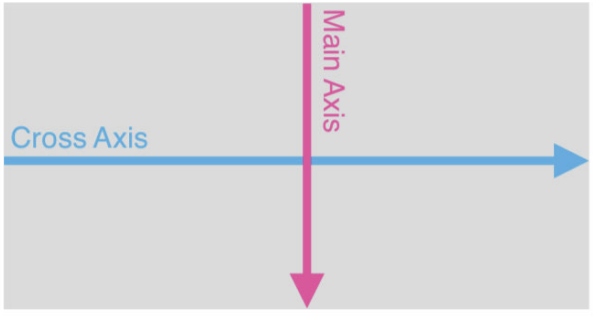
`flex-direction: row;`



`flex-direction: row-reverse;`



`flex-direction: column;`



`flex-direction: column-reverse;`

flex-direction Property

```
flex-container-element {
    display: flex;

    /* Default value. The flexible items are displayed horizontally, as a row */
    flex-direction: row;

    /* Same as row, but in reverse order */
    flex-direction: row-reverse;

    /* The flexible items are displayed vertically, as a column */
    flex-direction: column;

    /* Same as column, but in reverse order */
    flex-direction: column-reverse;
}
```

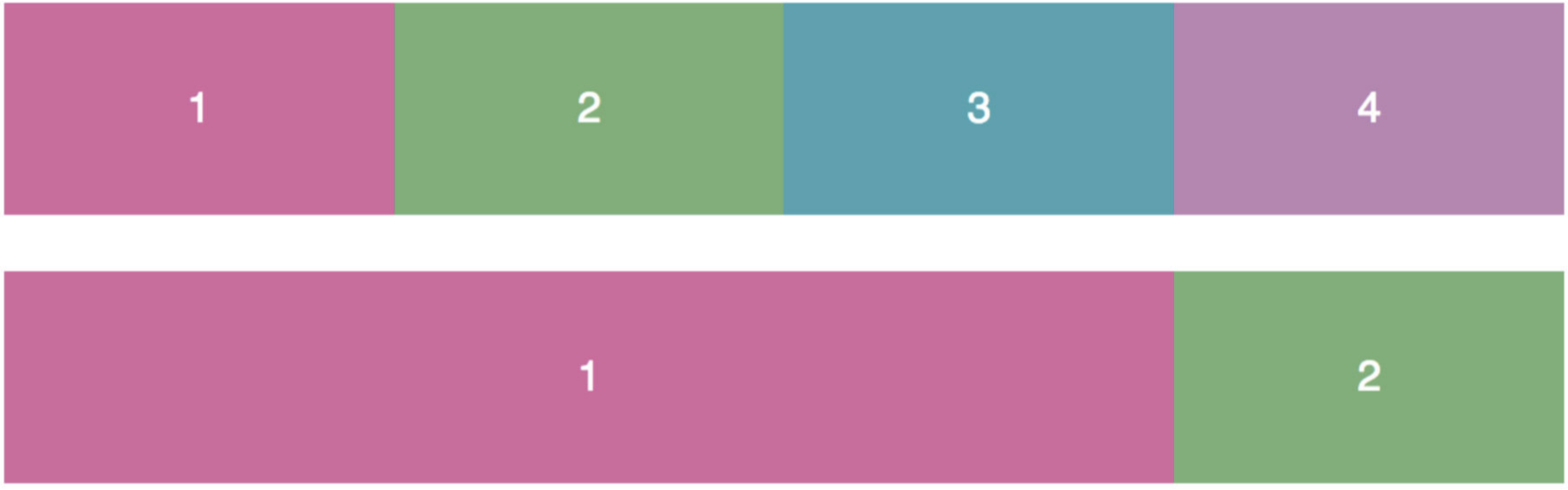
flex-direction Example

```
flex-direction: row;
```

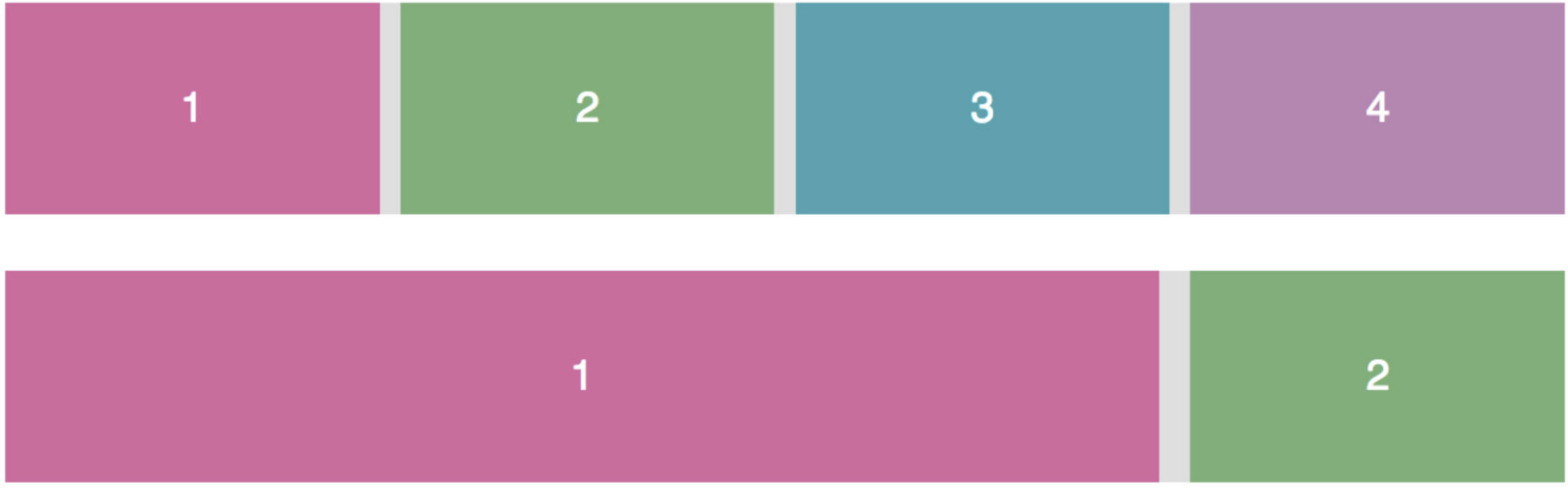
Code Along

Open: Flex Practice Codepen

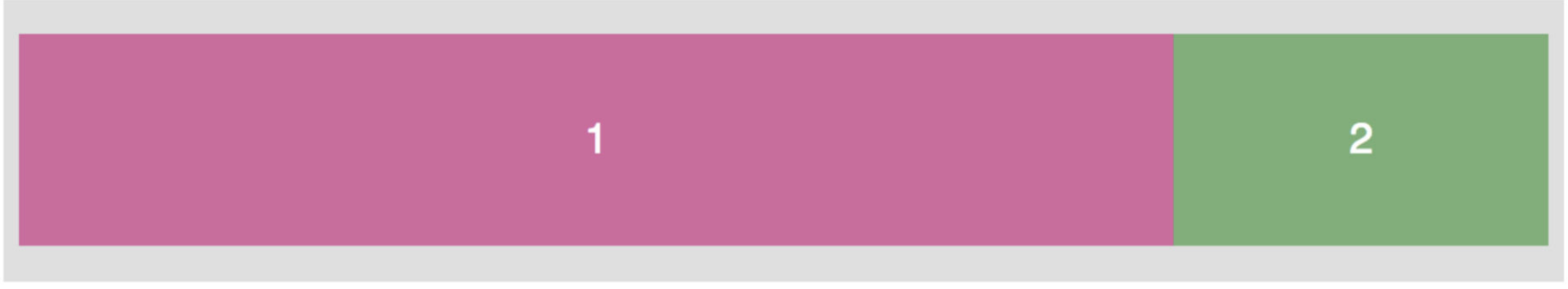
Part 1



Part 2



Bonus



Lab Time

Open: **starter_code/layout_lab**

Directions:

1. Open the **lesson_05_css_layouts/starter_code/layout_lab** folder
2. Create the layouts found in the PDF file from scratch

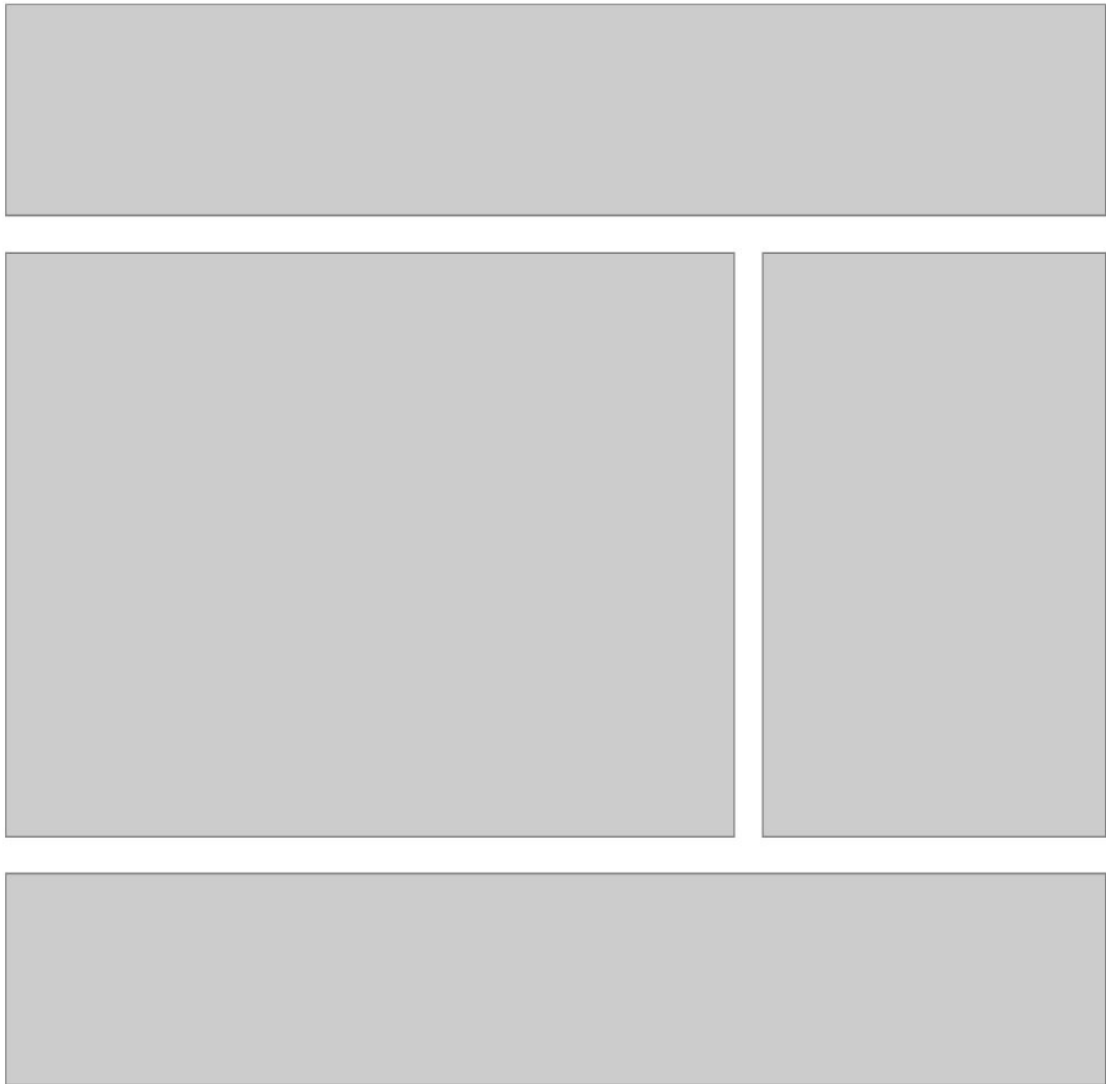
Timing:

- **10 minutes** - do the first layout together
- **50 minutes** - develop the other 4 layouts based on your own

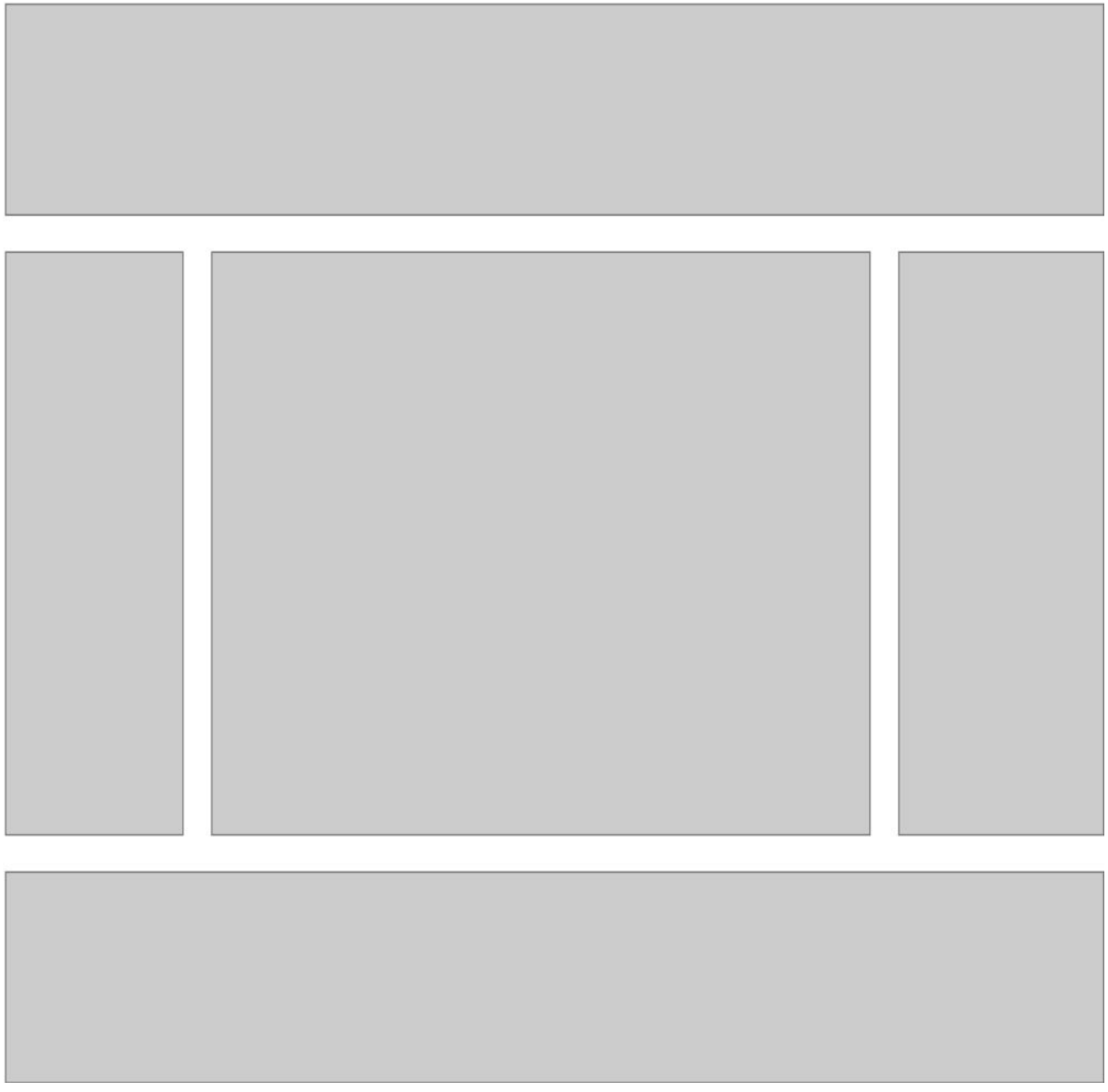
Tips:

- First, determine which parent element will become your flex container, and which elements will become the flex items.
- Hint: it may not be visible

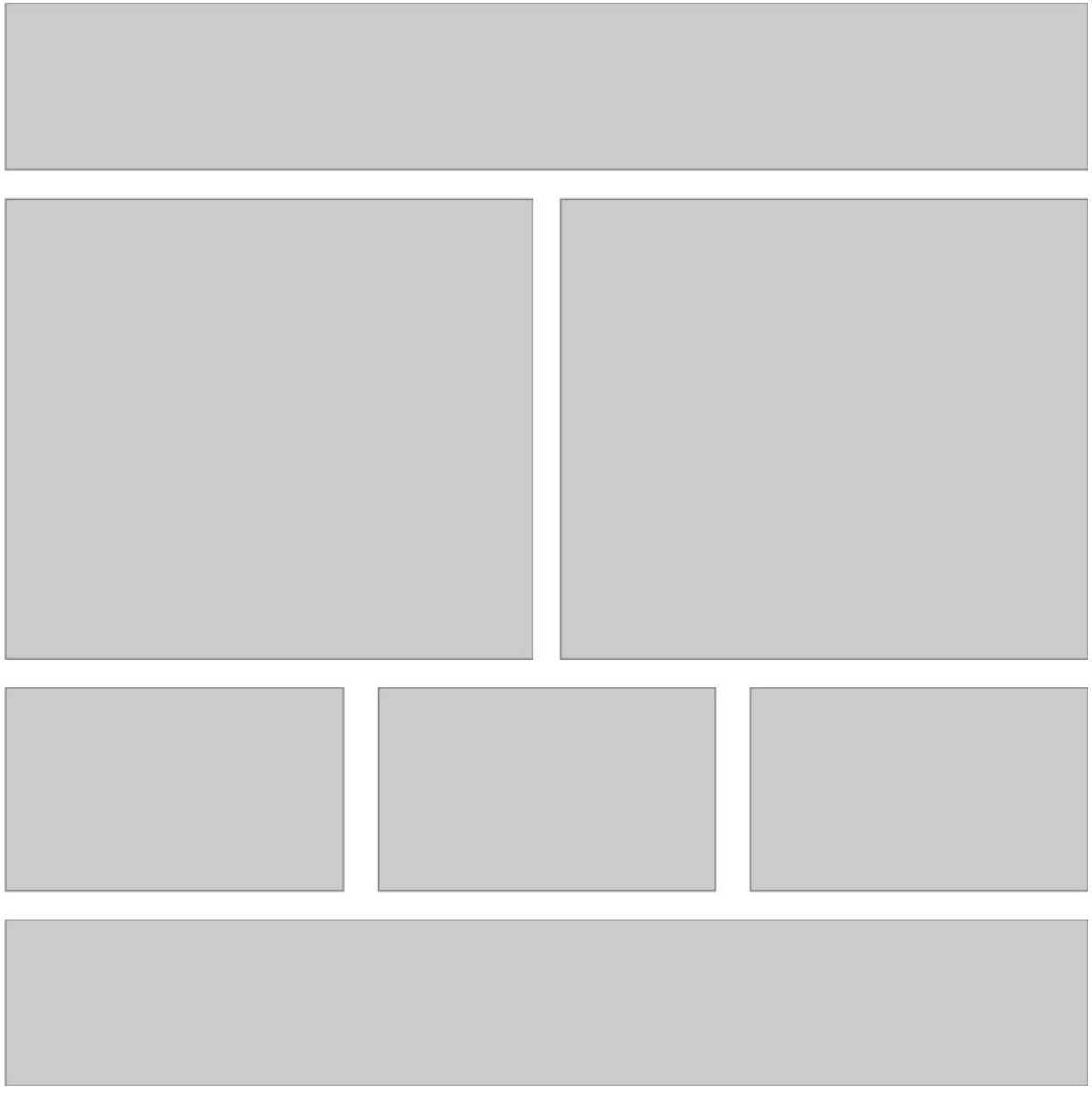
Two-Column



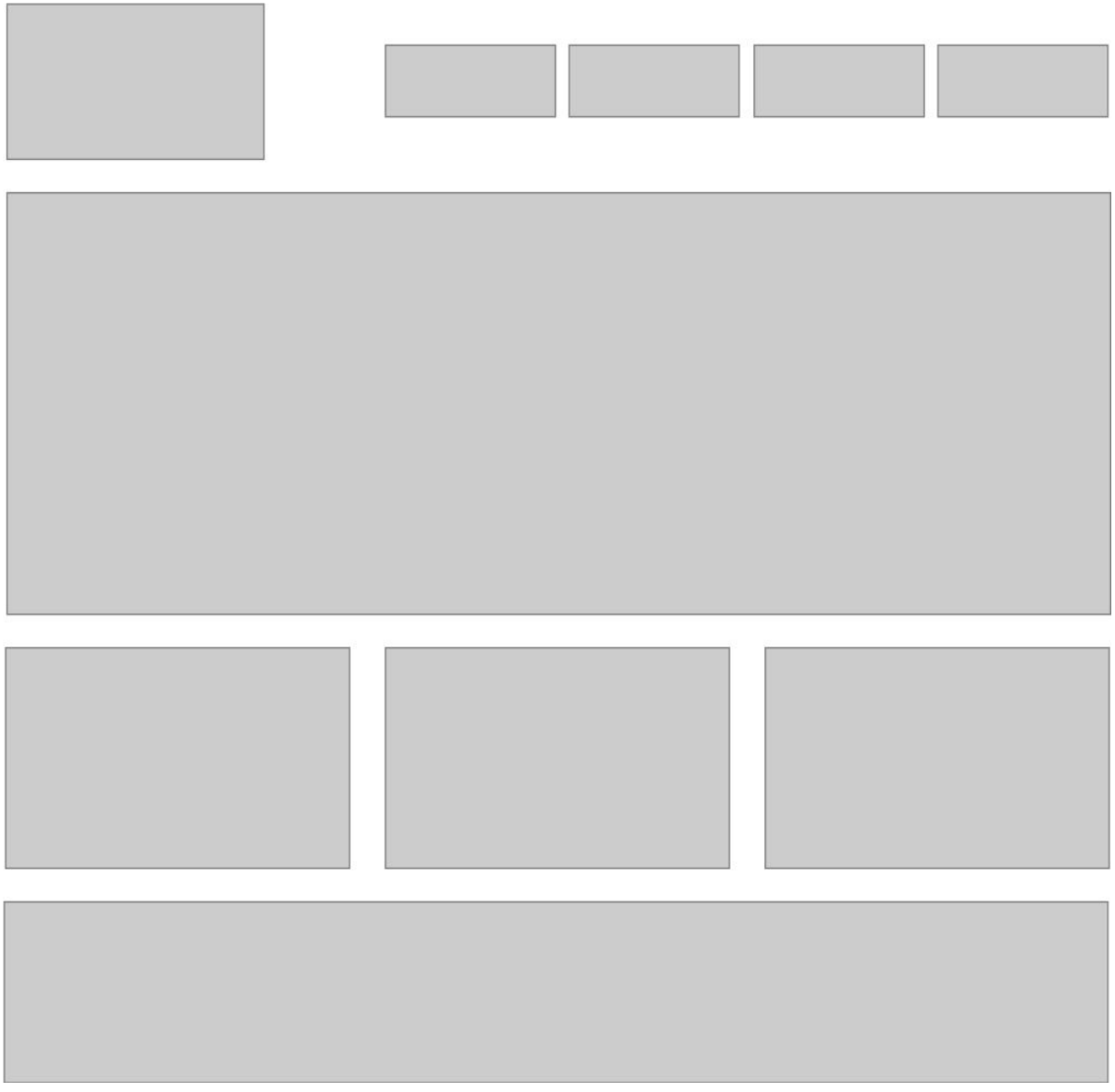
Three-Column



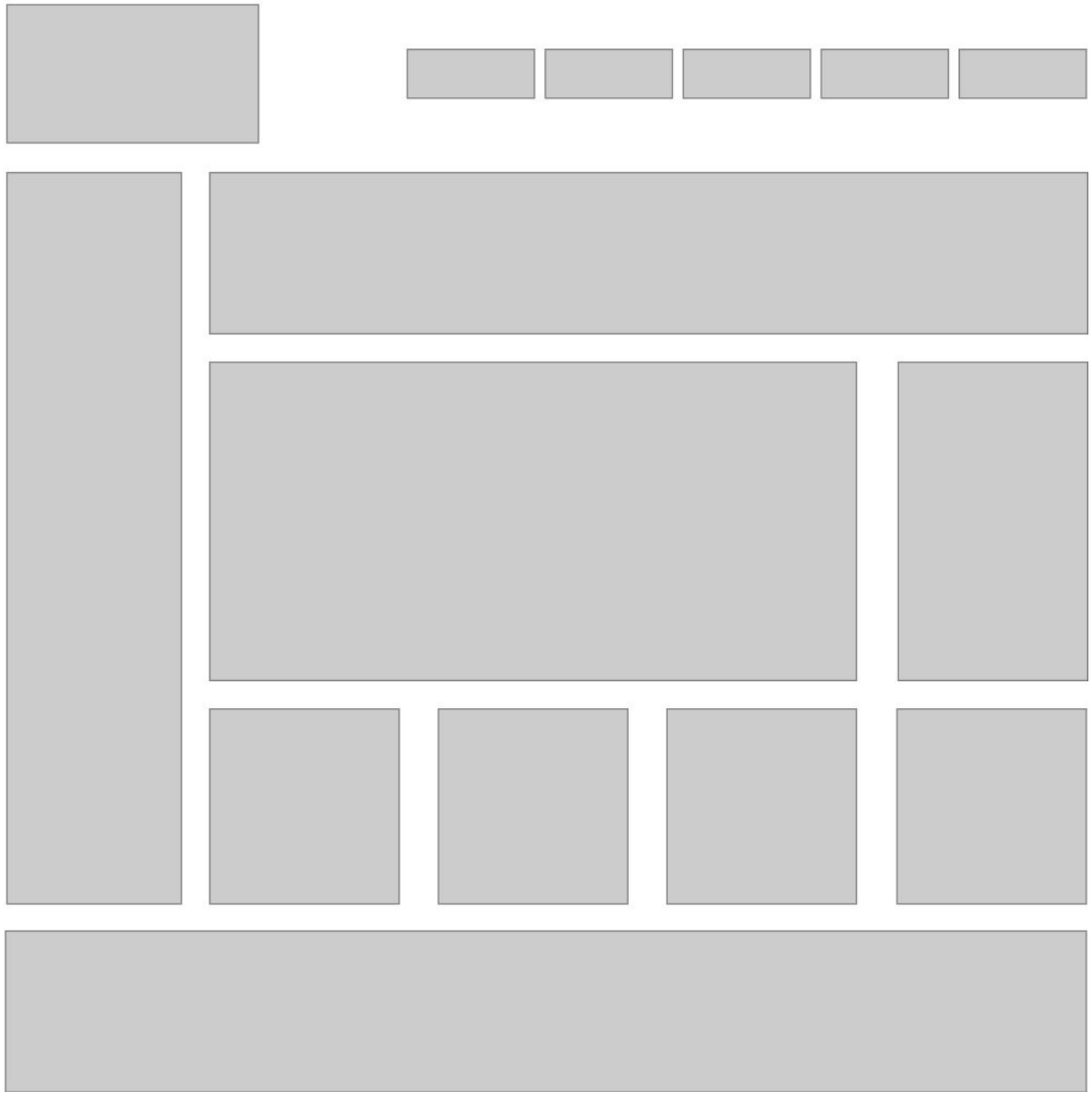
Simple Marketing



Marketing with Nav



Magazine



Exit Tickets

Take 5-10 minutes to give us some

(Link is in Slack Room)

Learning Objectives Review

- We explored how to select nested elements to apply styling.
- We styled a webpage using classes and ids
- We experimented and predicted effects of floats & clearing CSS positioning
- We applied multi-column layouts using CSS Flexbox to develop a webpage.

Week 2 HW Due Wednesday

- **Due Wednesday, May 8th by 11:59pm ET**
- Once you upload your HW to your repo, please PM the TA with a link to the Assignment folder.
- This is how we will know whether an assignment has been completed or not
- Once graded, the TA will reply with feedback

Next Class...

Lesson 6 - CSS Layouts Lab