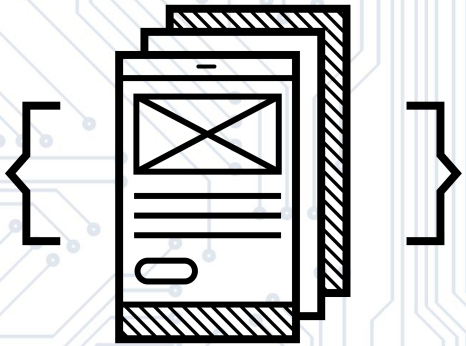
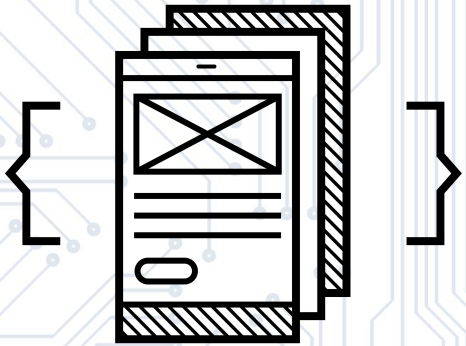


CSS Flexbox and the Flexible Box Model

UX/UI Design

Lesson 18.1

45 minutes



CSS Flexbox and the Flexible Box Model

UX/UI Design

Lesson 18.1

Today’s Objectives

By the end of class today, you will:

Create CSS Flexbox containers and set them to display as a row or a column.

Position CSS Flexbox items inside containers to create clean and fluid layouts.

Nest CSS Flexbox containers to control elements contained inside them.

Apply CSS Flexbox skills in a coding activity called Jake's Eatery.

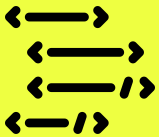
Front-End Development Units

Today we'll cover the Flexible Box Model and how it makes web developers' lives easier.

UI Design Process

Dev GRIDS

HTML



HTML Structure

CSS 3

Bootstrap 4

JavaScript/ jQuery

Git/GitHub/ GitHub Pages

Responsive Web Design

Atomic Design Mobile First

Debugging

Templates and Layout and Pages

Web Prototyping

UI Testing and A/B Testing

Introducing CSS Flexbox

What is CSS Flexbox?

What is CSS Flexbox?

Previously, we used the following display types for layout:

01

`block`

Has a specified height & width

Takes up a whole line

02

`inline`

Only uses auto height & width

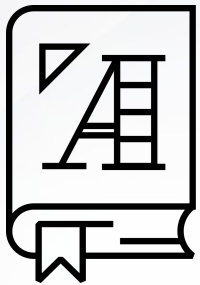
Shares a line with other elements

03

`inline-block`

Has a specified height & width

Shares a line with other elements



CSS Flexbox is a separate layout model for creating responsive rows or columns.

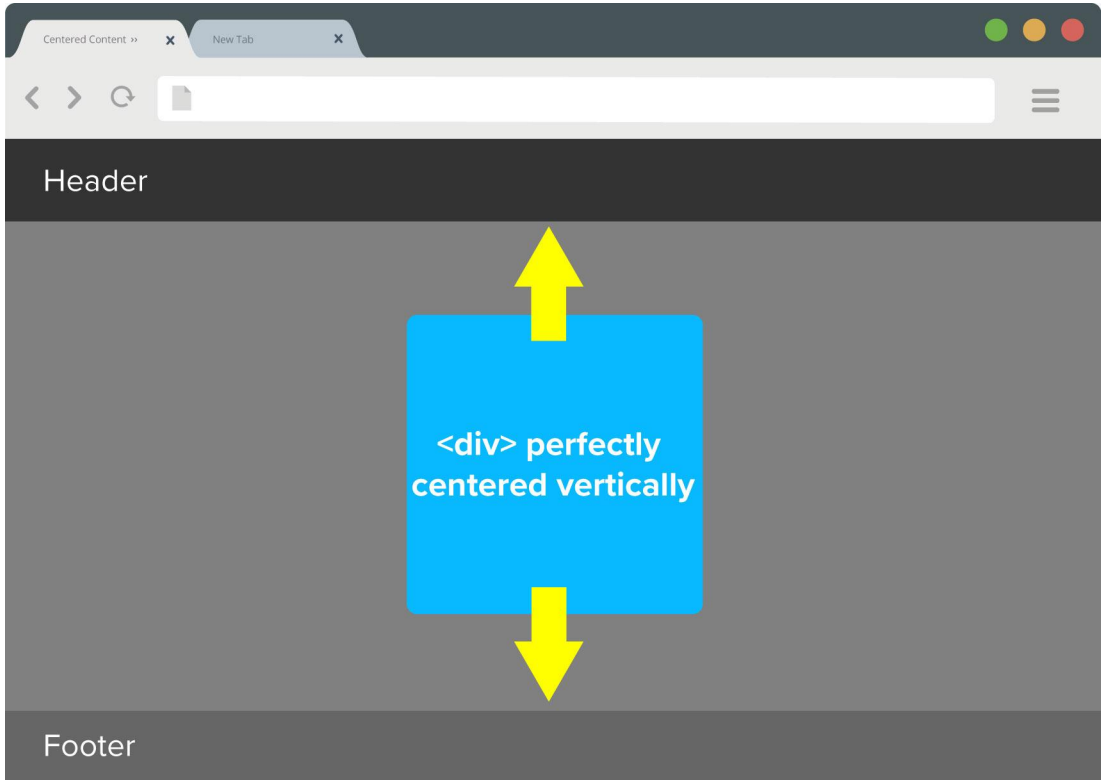
Flexbox elements are flexible; they automatically grow, shrink, squish together, or spread out to fill the available space.

Why Learn CSS Flexbox?

Simple Layout Requirements

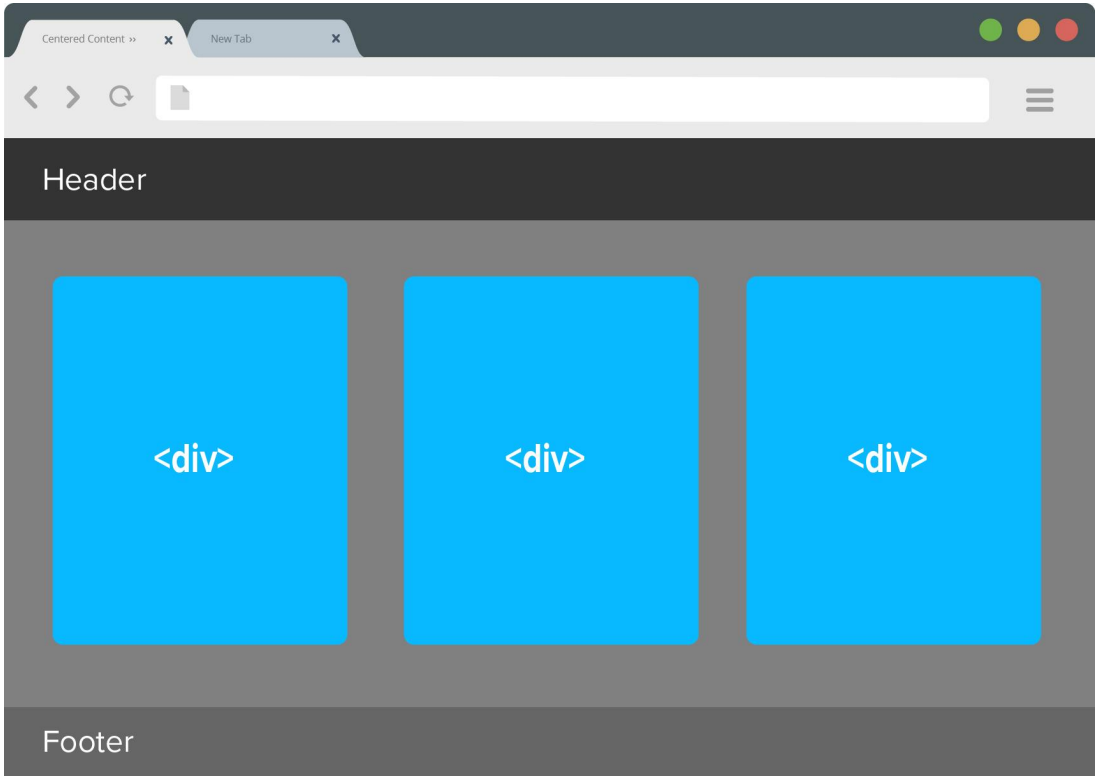
Vertically centering elements.

(`margin: auto` only works for horizontal centering)



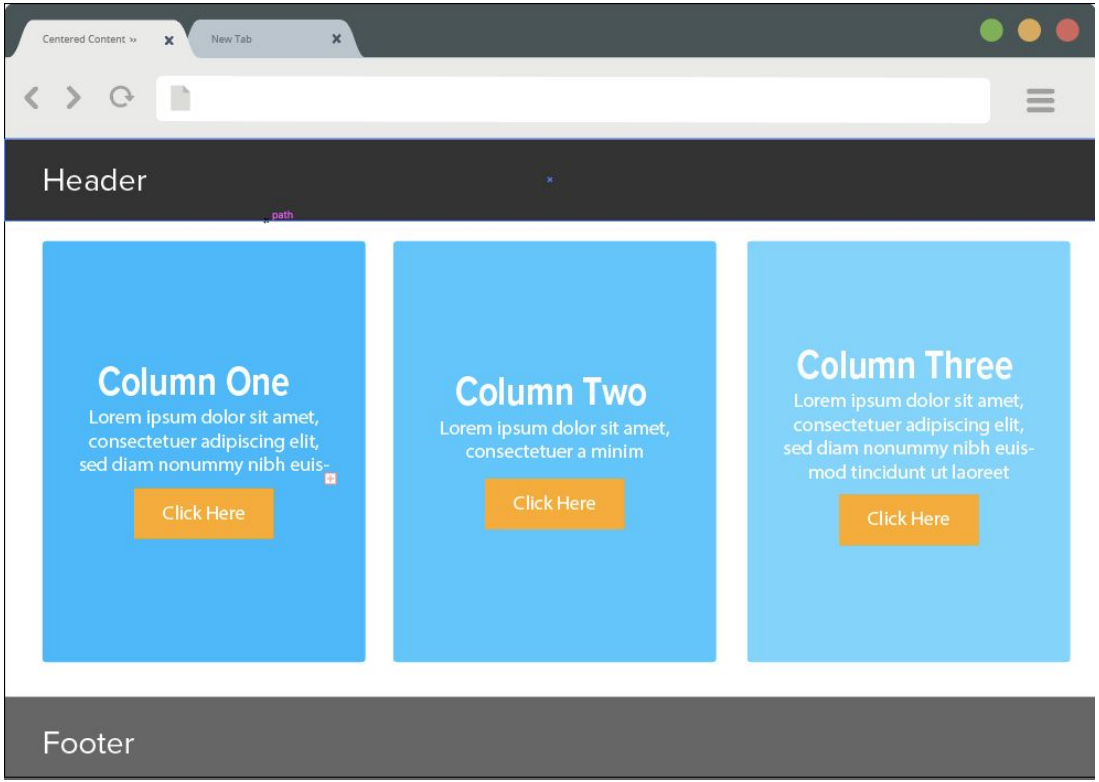
Simple Layout Requirements

Spacing and sizing elements equally according to the available space (especially in responsive layouts).



Simple Layout Requirements

Maintaining consistent container sizes with inconsistent content sizes.



How Can We Use CSS Flexbox?

Flex Containers and Flex Items

To start using Flexbox, we can use the rule `display: flex` to transform any element into a **flex container**.

Any elements inside of a flex container are considered **flex items**. These will automatically align themselves in a responsive row.

HTML

```
<section class="parentContainer">
  <div class="flexChild">
    "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
  </div>
  <div class="flexChild">
    "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
  </div>
  <div class="flexChild">
    "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
  </div>
</section>
```

CSS

```
.parentContainer {
  display: flex;
  background-color: lightblue;
  padding: 15px;
}
.flexChild {
  background-color: red;
  padding: 10px;
  margin: 10px;
  height: 200px;
}
```

What If We Want a Column?

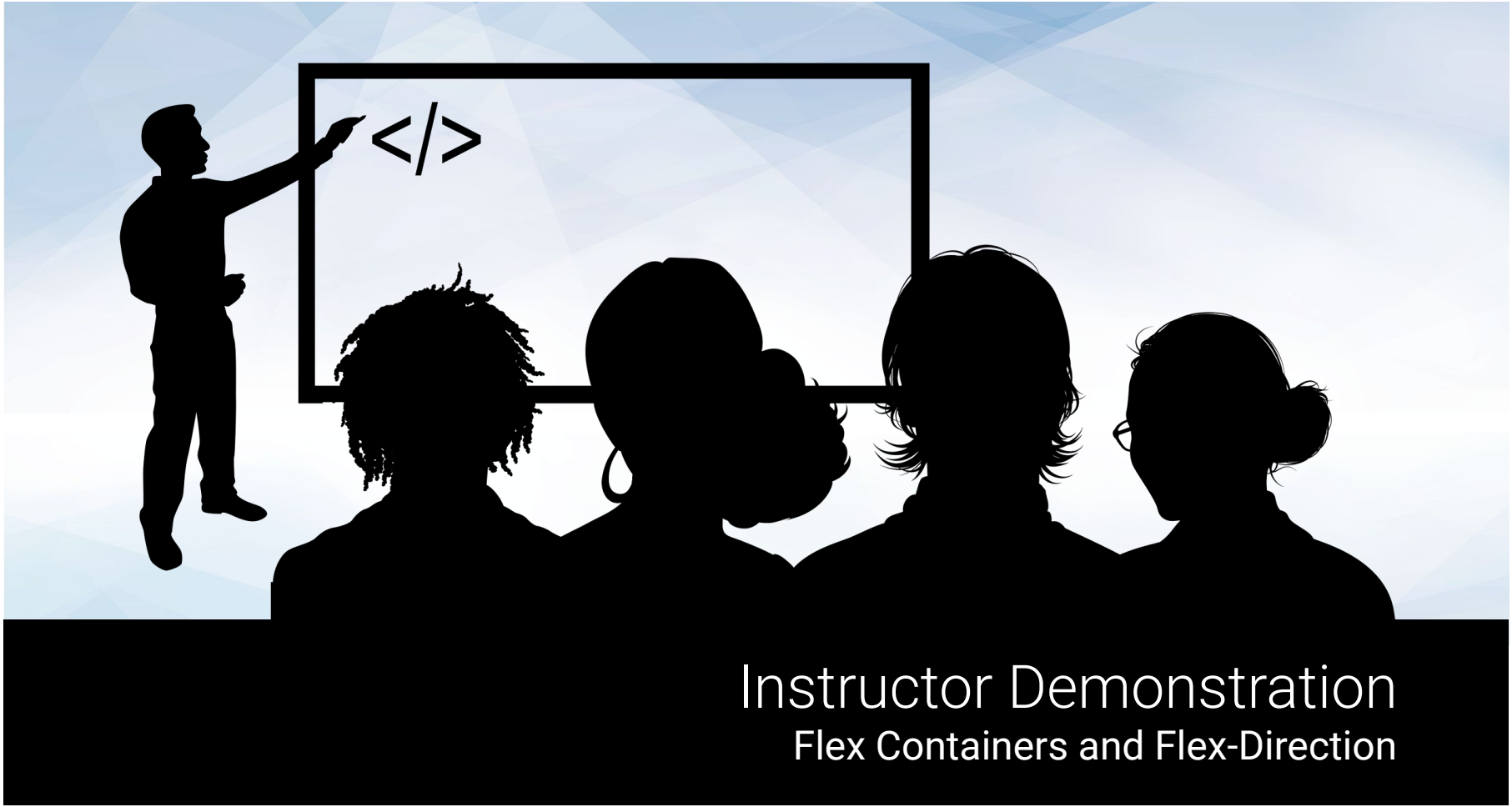
If we want a column instead of a row, we can add the `flex-direction` property. Note that `row` is the default value, so this is only necessary for flex columns.

The `column` value stacks the flex items vertically (from top to bottom):

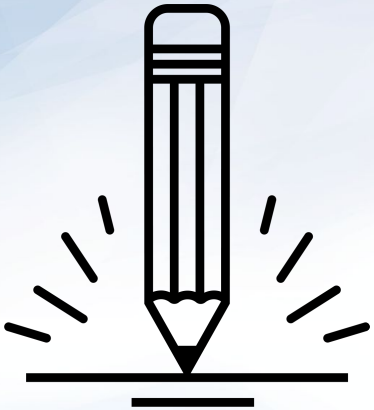
```
.flex-container {
  display: flex;
  flex-direction: column;
}
```

The `row` value stacks the flex items horizontally (from left to right):

```
.flex-container {
  display: flex;
  flex-direction: row;
}
```



https://codepen.io/matt-boisseau/pen/zYqVaxv



Activity:

Your First CSS Flexbox Layout

(Instructions sent via Slack)

Suggested Time:
15 minutes

Let's Review: Your First CSS Flexbox Layout

- Why do we use flex?
- What kind of layouts would use flex? One-dimensional or two?
- What does it mean to create a one-dimensional layout?

Suggested Time: 5 minutes

Alignment with CSS Flexbox

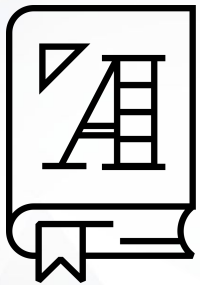
Alignment with CSS Flexbox

Being able to align content in flex is one of the most attractive features of flexbox. We'll discuss the following two CSS properties:

01 `align-items`

02 `justify-content`

The `align-items` Property



The `align-items` **property** defines the default behavior of flex items as they fill the container and specifies their layout within that container along the cross axis (top-bottom).

The `align-items` Property

The following five values are used often:

01 `flex-start` aligns items at the beginning of the container (or top).

02 `flex-end` aligns items at the end of the container (bottom).

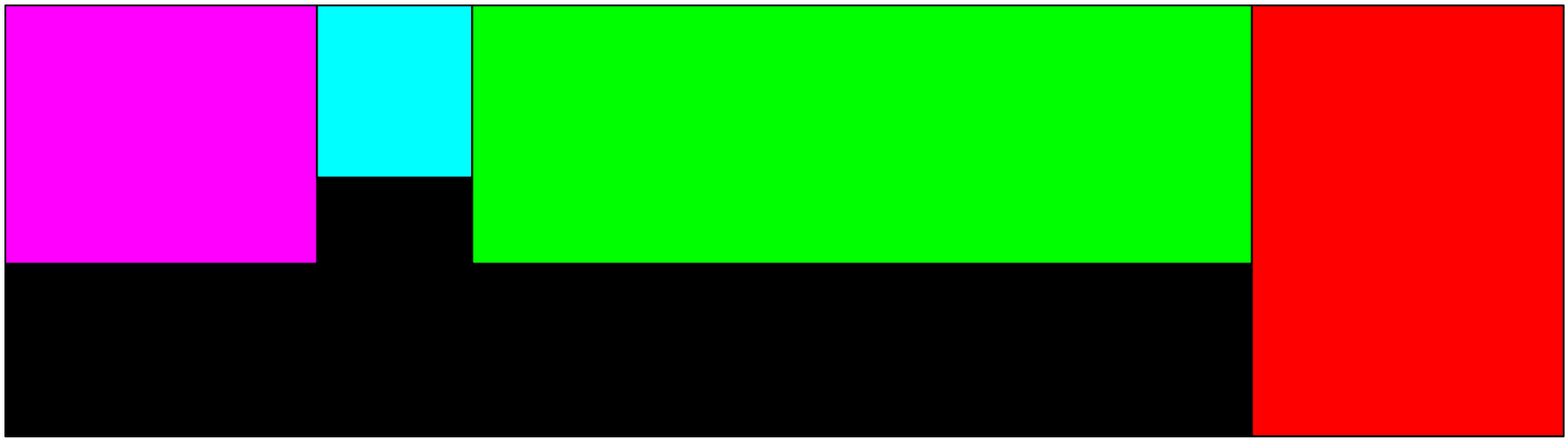
03 `center` centers content vertically in its parent container.

04 `baseline` aligns items so that their baselines align.

05 `stretch` items stretch to fill the container top-bottom or left-right.

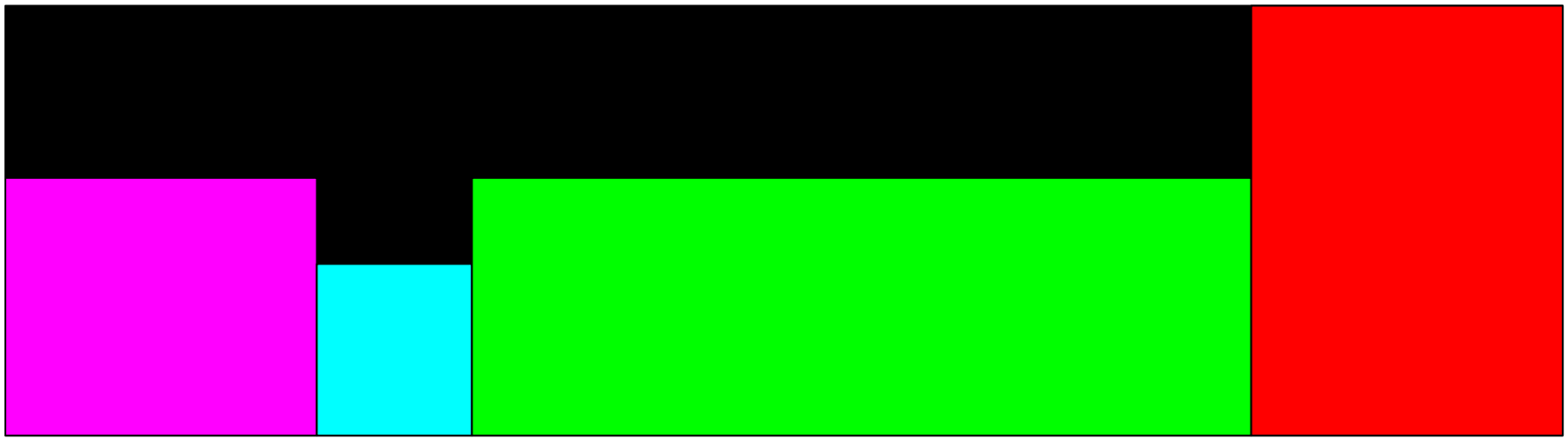
flex-start


```
.flex-container {
    align-items: flex-start;
}
```



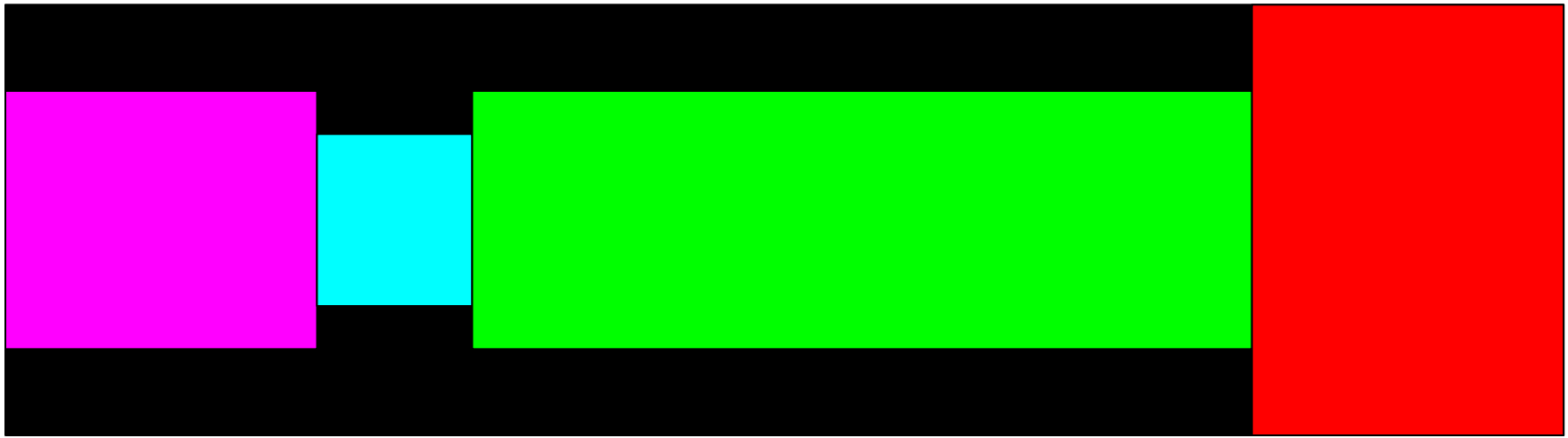
flex-end


```css
.flex-container {
    align-items: flex-end;
}
```



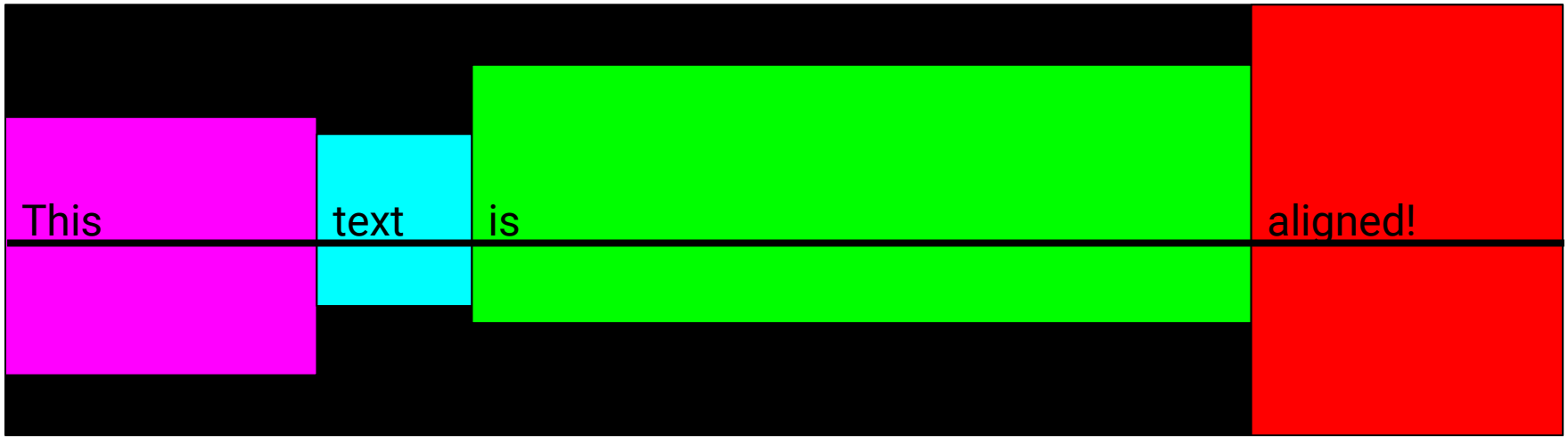
center

```css
.flex-container {
    align-items: center;
}
```



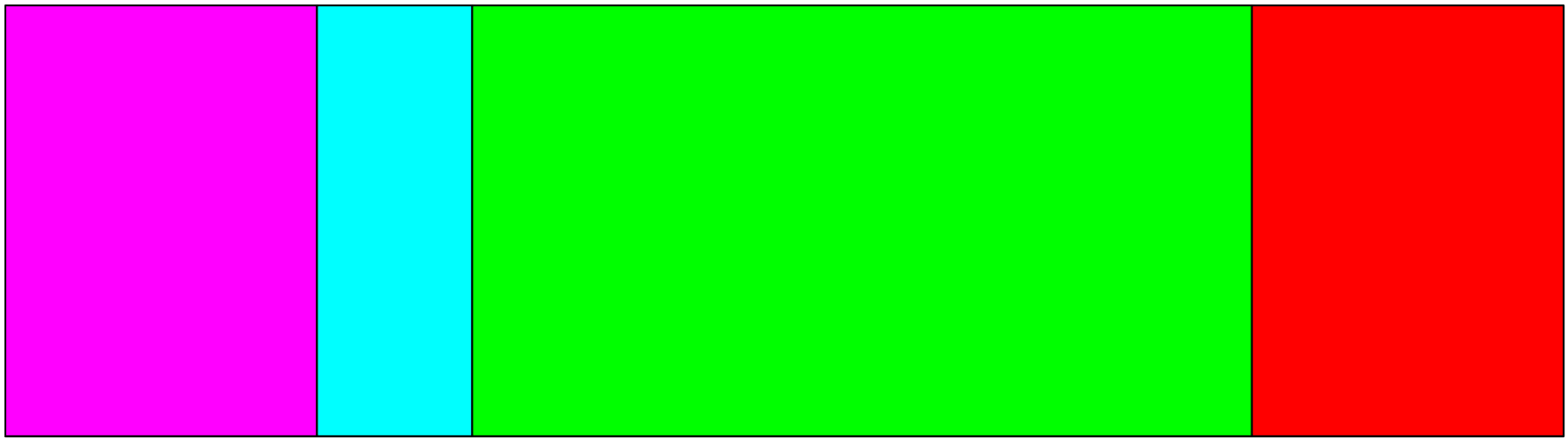
baseline

```css
.flex-container {
    align-items: baseline;
}
```

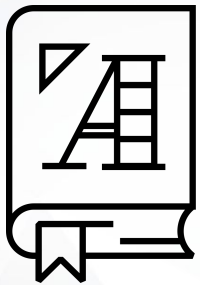


stretch


```css
.flex-container {
    align-items: stretch;
}
```



The `justify-content` Property



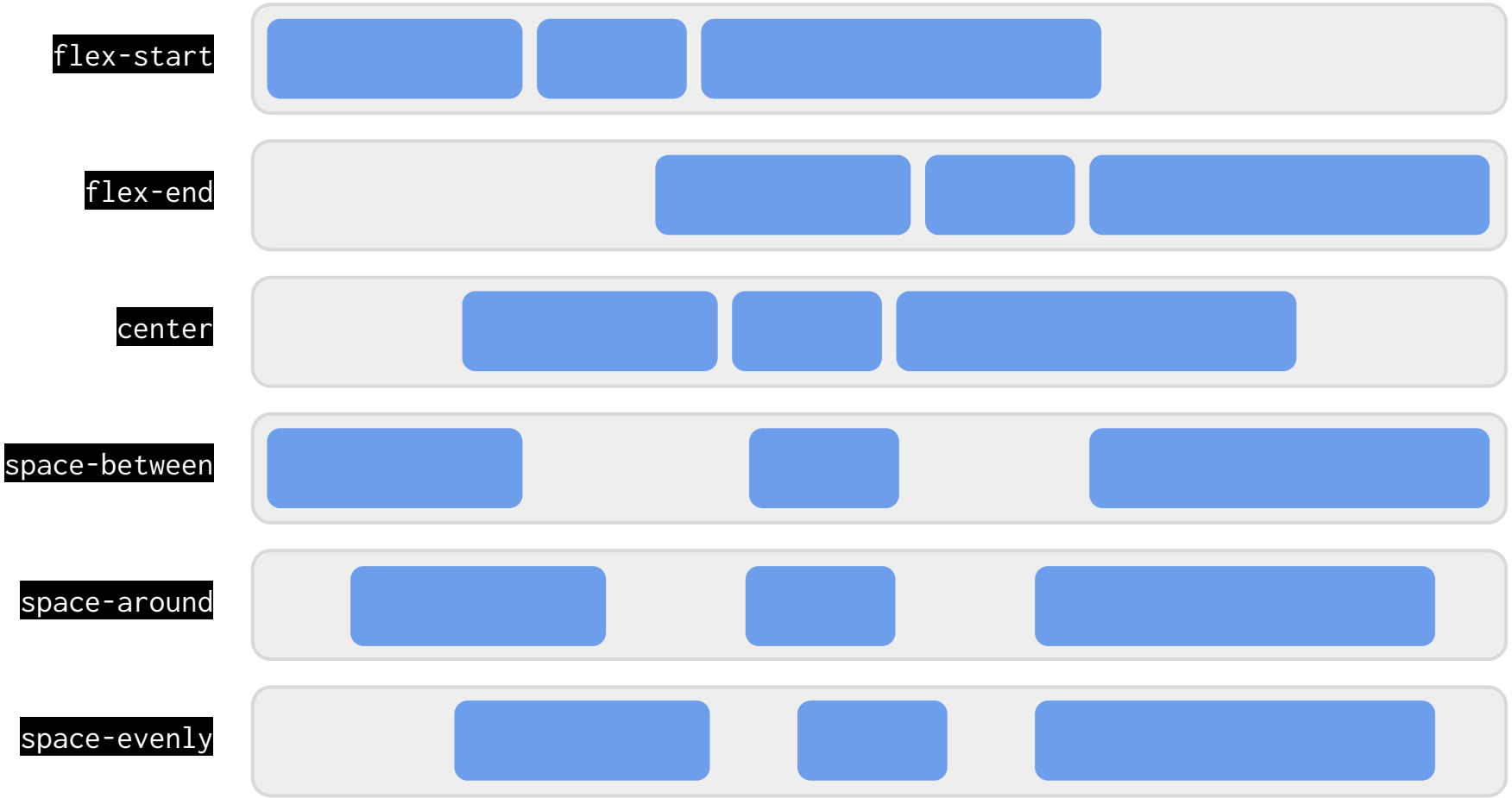
The CSS property `justify-content` is used to distribute content across the left and right side of containers, very similar to `margin: 0 auto`.

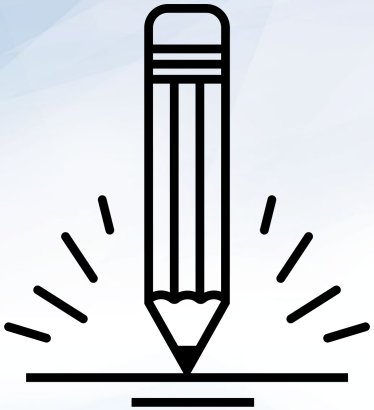
The `justify-content` Property

`justify-content` positions elements left to right, as opposed to top or bottom.

There are six values:

- `flex-start` flex items are aligned at the start line (this is the default).
- `flex-end` flex items are aligned at the end of the flex container.
- `center` flex items are aligned in the center of the container similar to `margin: 0 auto;`.
- `space-between` items are evenly distributed in the line; first item is on the start line, the last item is on the end line.
- `space-around` items are evenly distributed in the line with equal space around them.
- `space-evenly` items are distributed so that the spacing is the same between any two adjacent alignment subjects, before the first alignment subject and after the last alignment subject.


flex-start
flex-end
center
space-between
space-around
space-evenly



Activity:

Aligning CSS Flexbox Items

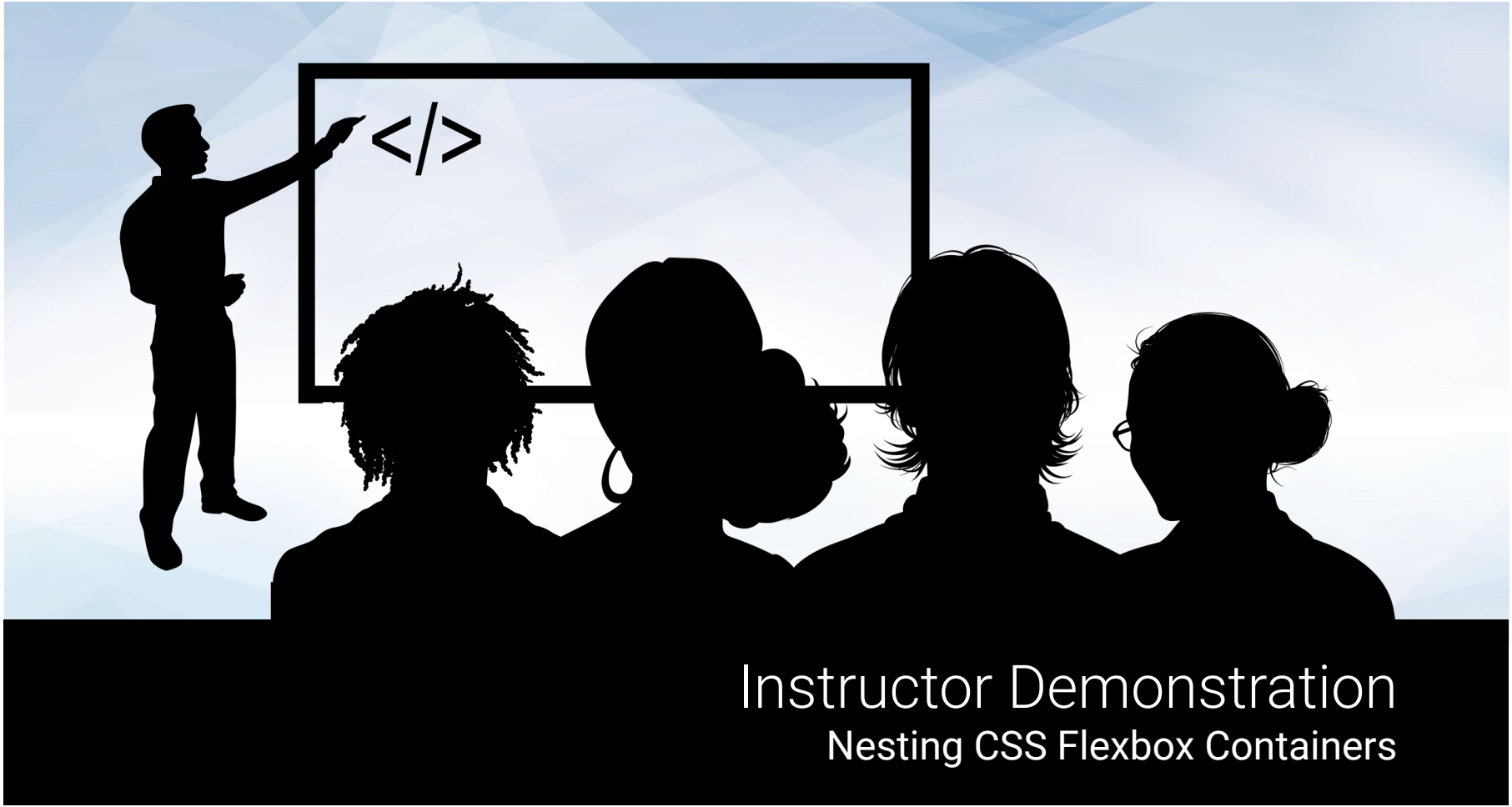
(Instructions sent via Slack)

Suggested Time:
15 minutes

Let's Review: Aligning CSS Flexbox Items

- What does the CSS property `align-items` do?
- What does the CSS property `justify-content` do?
- How are they different?

Suggested Time: 15 minutes

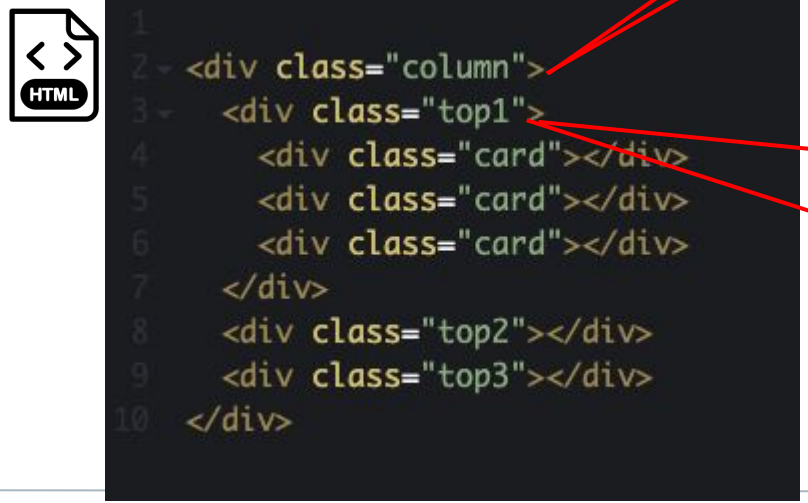


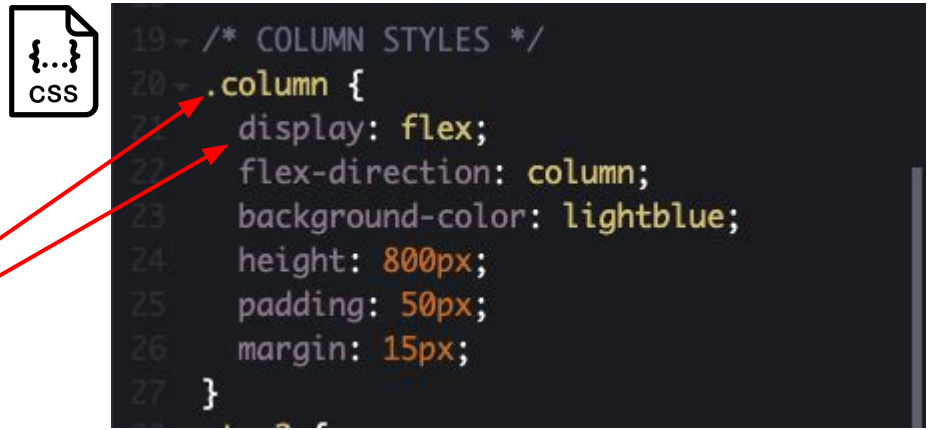
https://codepen.io/matt-boisseau/pen/zYqVaxv

Nesting CSS Flexbox Containers

Flex containers can be nested inside each other to create more complex layouts.

You can create a “nested” flex container simply by applying display flex to a flex item that is already contained inside a flex container.



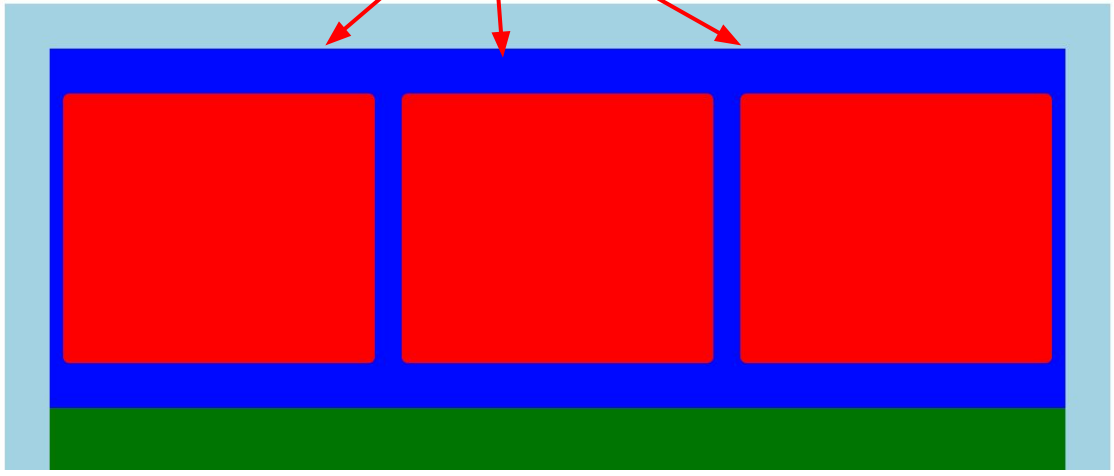


Nesting CSS Flexbox Containers

Nested flex containers are mainly used to control the contents nested inside them using `align-items` and `justify-content`.

You can also use nested grids to create layouts composed of rows nested in columns or columns nested in rows.

```css
/* Nested Column Styles */
.top1 {
  height: 50%;
  background-color: blue;
  display: flex;
  align-items: center;
  justify-content: center;
}
```



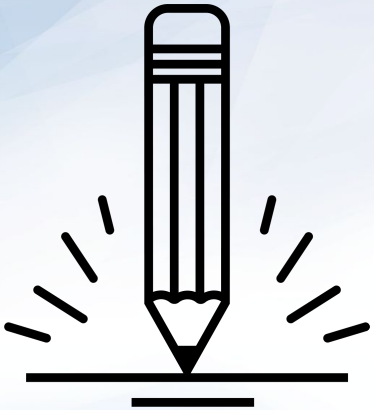

I Wanna Play a Game

Flexbox Froggy

Coding classes can be dull sometimes...

Let’s spice it up by playing a web game to help reinforce all the concepts we lectured about earlier.

https://flexboxfroggy.com/



Activity:

Play CSS Flexbox Froggy

(Instructions sent via Slack)

Suggested Time:
15 minutes

https://flexboxfroggy.com/

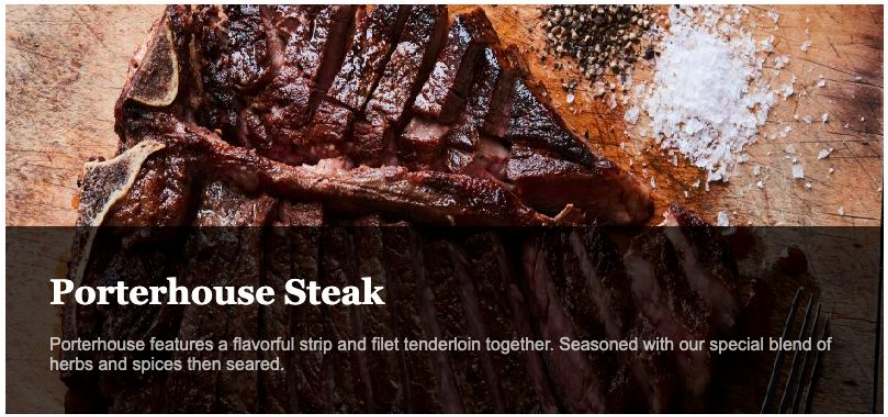
Time's Up! Let's Review.

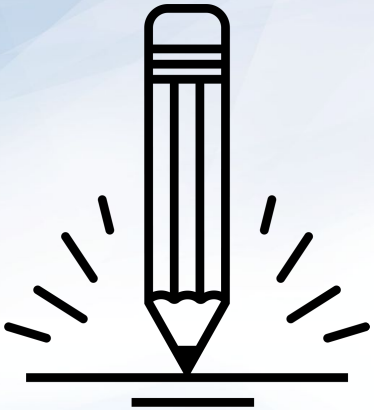

Break


Time to Code ...

JAKE'S EATERY

Order Online | Menu | Our Location | About Us | Text Us | Order Now





Activity:

Jake’s Eatery

(Instructions sent via Slack)

Suggested Time:
45 minutes

Time's Up! Let's Review.

Let's Review

01 What does it mean when we say that flex is a one-dimensional layout?

02 What does the CSS property `align-items` do?

03 What does the CSS property `justify-content` do?

04 Why would you want to nest a flex container inside another?

Congratulations! Recap

Today we learned:

01

CSS Flexbox Basics
We learned how to create flex containers and modify the children.

02

CSS Flexbox Nesting and Alignment
We learned how to nest flex containers and align the children.

03

Building Jake's Eatery
We applied our flex skills by building a full design in under an hour!

Unit 18 Homework

Unit 18 Homework

For this week's homework, you will continue to build out your portfolio webpage.

This week students will build out two more sections for their portfolio webpage using CSS Flex and CSS Grid. Students will also make their webpage responsive by writing custom media queries for mobile devices.

Design a one-page website showcasing one of your case studies.

Create the HTML structure.

Style the HTML using a CSS file.

Upload your case study page to GitHub and publish via GitHub Pages.

Questions?


The End