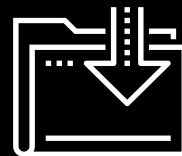
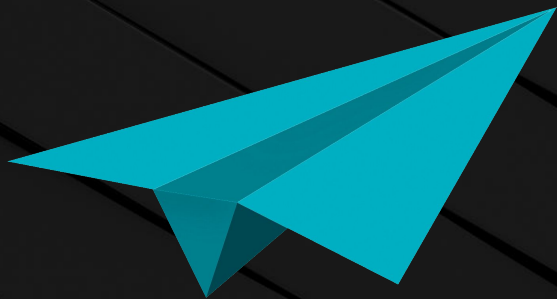


CSS Flexbox

Skills Bootcamp in Front-End Web Development

Lesson 2.1





Office Hours

30 Minutes

The background is a dark charcoal gray with a series of parallel diagonal lines running from the top-left to the bottom-right. Overlaid on this are several teal-colored geometric shapes: a large central triangle pointing right, a smaller triangle to its left, and a square to its right. Scattered around these shapes are various white line-art symbols, including a plus sign, a minus sign, a circle with a dot, a circle with a horizontal line, a circle with a vertical line, a circle with a diagonal line, a circle with a cross, a circle with a dot, a circle with a horizontal line, a circle with a vertical line, a circle with a diagonal line, a circle with a cross, a circle with a dot, a circle with a horizontal line, a circle with a vertical line, a circle with a diagonal line, and a circle with a cross.

WELCOME

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 the elements contained inside them.

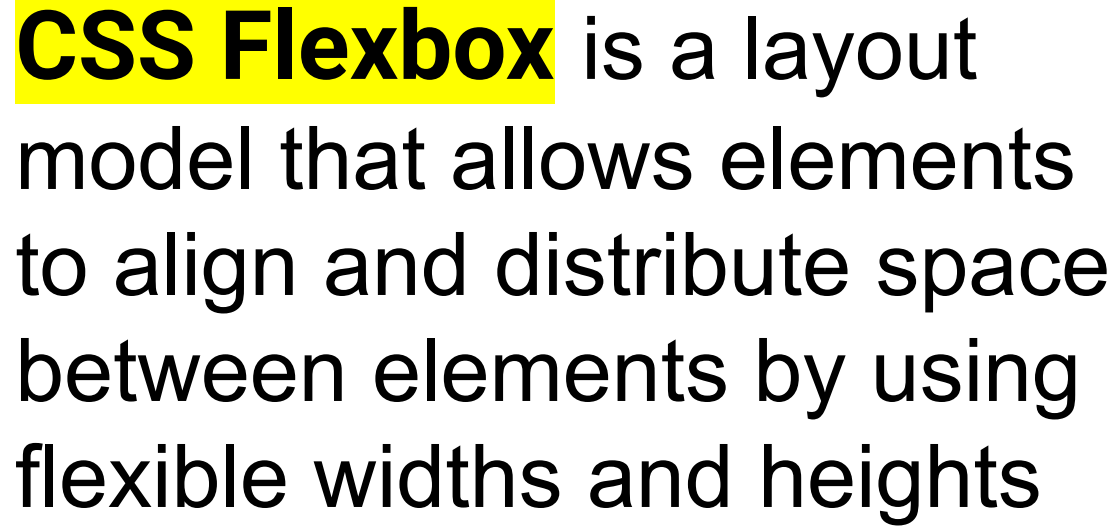


Apply CSS Flexbox skills in a coding activity called Jake's Eatery.

Introducing CSS Flexbox



What is CSS Flexbox?



CSS Flexbox is a layout model that allows elements to align and distribute space between elements by using flexible widths and heights



In short, it's a time-saver.
**Think lines of code you won't
have to write.**



**What are the three layout methods
we've already learned?**

Layout Methods

Previously, we used the following display types for layout:

01

block

Has a specified dimensions (height and width)

Takes up a whole line

02

inline

Only Uses auto height and width

Shares a line with other elements

03

inline-block

Has a specified height and width

Shares a line with other elements



Flexbox elements are flexible;
they automatically flex to fill
additional space and shrink to fit
into smaller spaces.



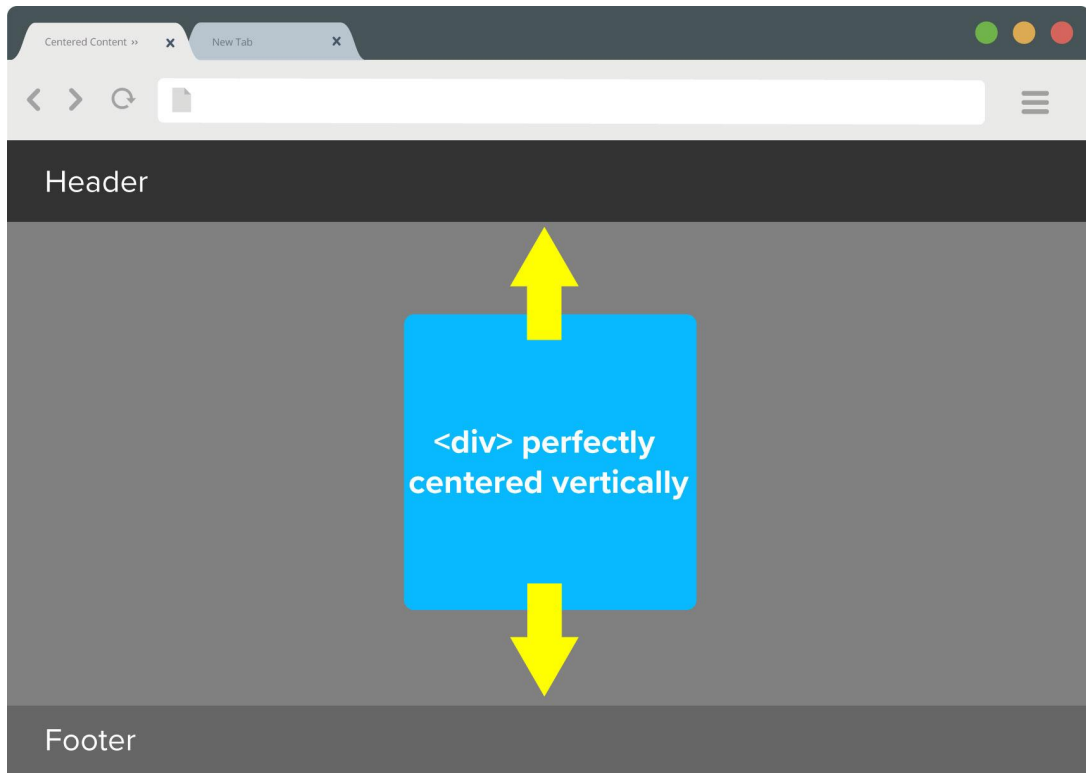
Why learn CSS Flexbox?

Simple Layout Requirements

Vertically centering elements.

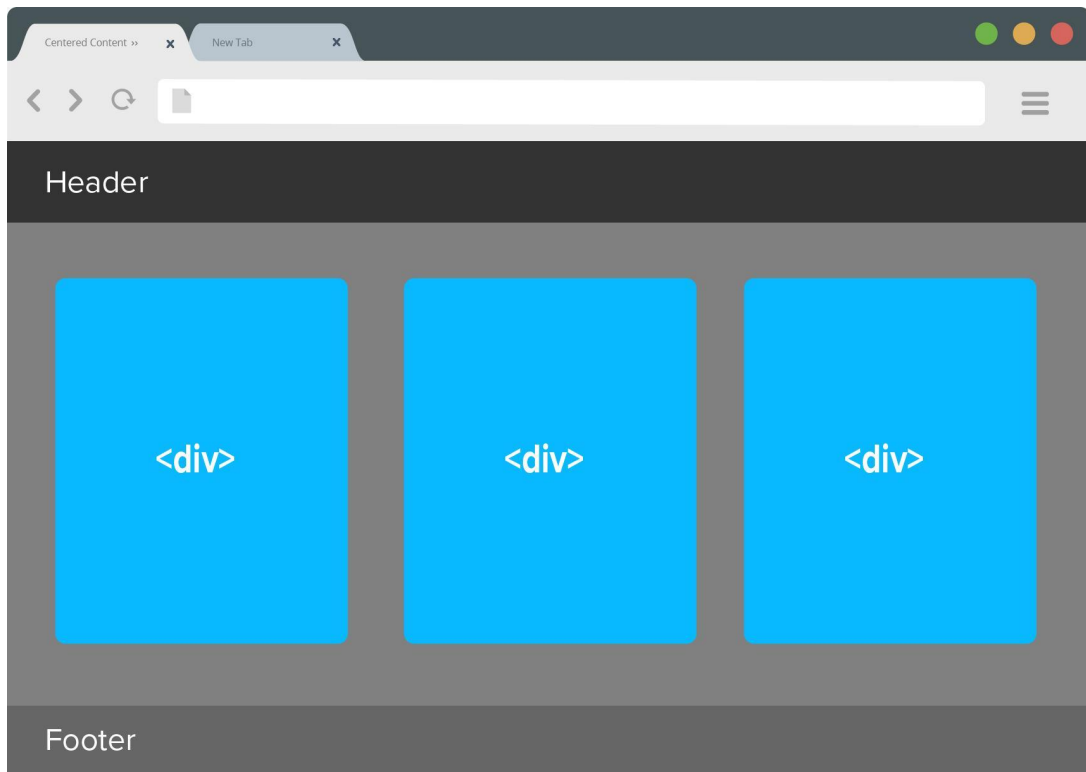
```
margin: auto
```

(Using `margin: 0 auto` only works for horizontal centering.)



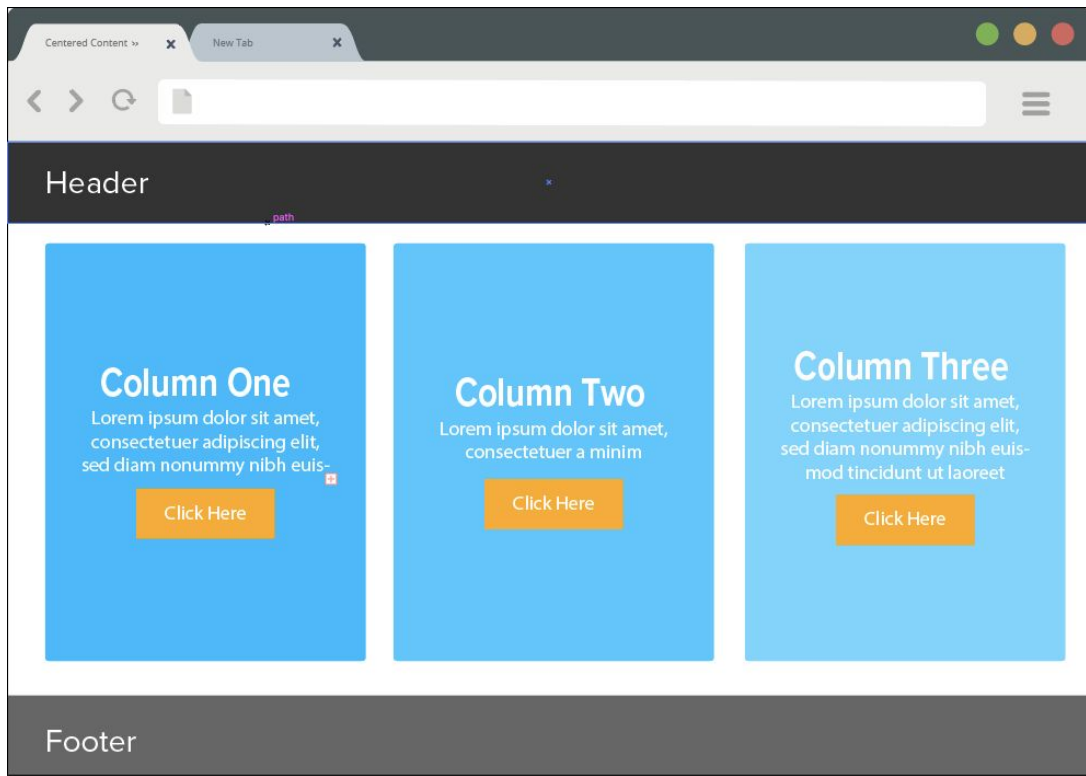
Simple Layout Requirements

Make all the children of a container take up an equal amount of the available width/height, regardless of how much width/height is available.



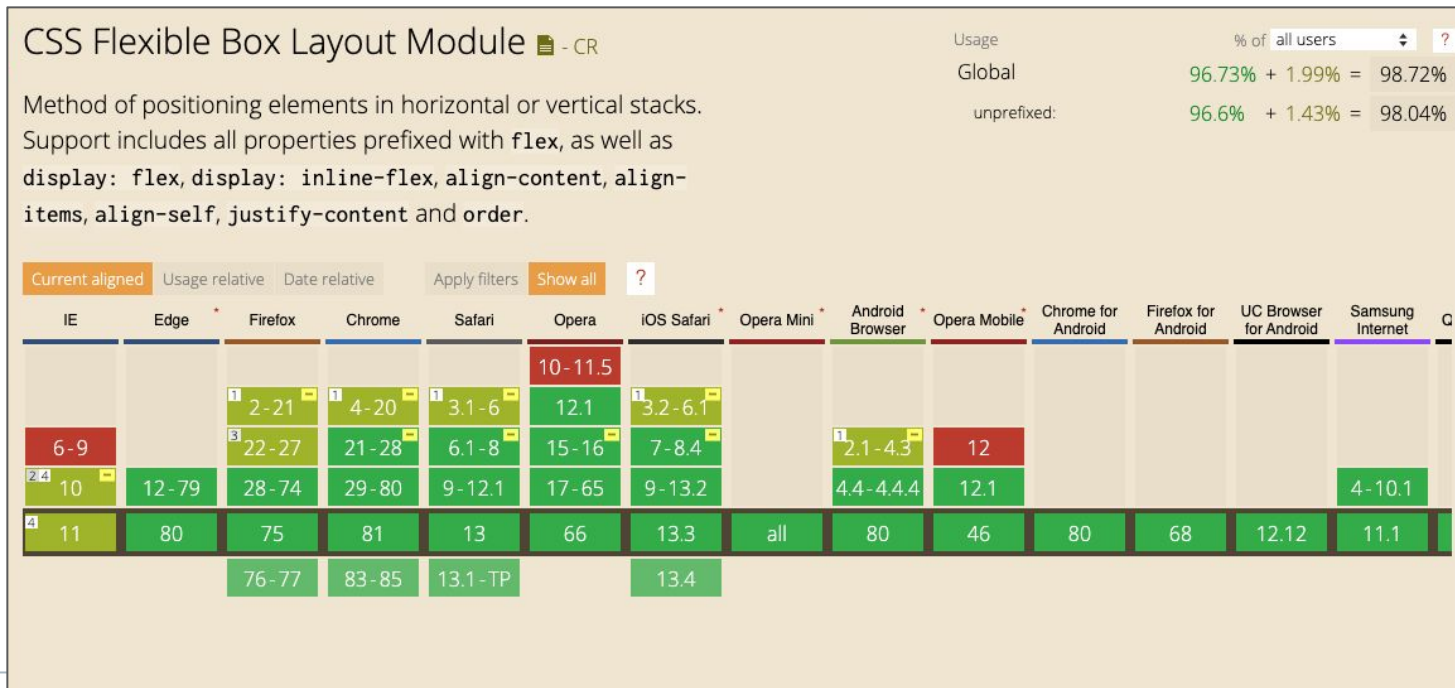
Simple Layout Requirements

Make all columns in the same container adopt the same height, even if they contain a different amount of content.



Why Learn CSS Flexbox?

CSS Flexbox is supported across all major browsers, making it a great tool to make reliable UIs.





How can we use CSS Flexbox?

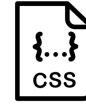
Flex Containers and Flex Items

To start using Flexbox, set `display: flex` onto any containing element to transform it into a **flex container**.

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



```
<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>
```



```
.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;  
}
```



Instructor Demonstration

Flex Containers and Flex-Direction



Activity: Your First CSS Flexbox Layout

In this activity, you'll make **two** CSS Flexbox containers using the **flex-direction** property.



CSS

Suggested Time:

15 minutes



Time's Up! Let's Review.

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?

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

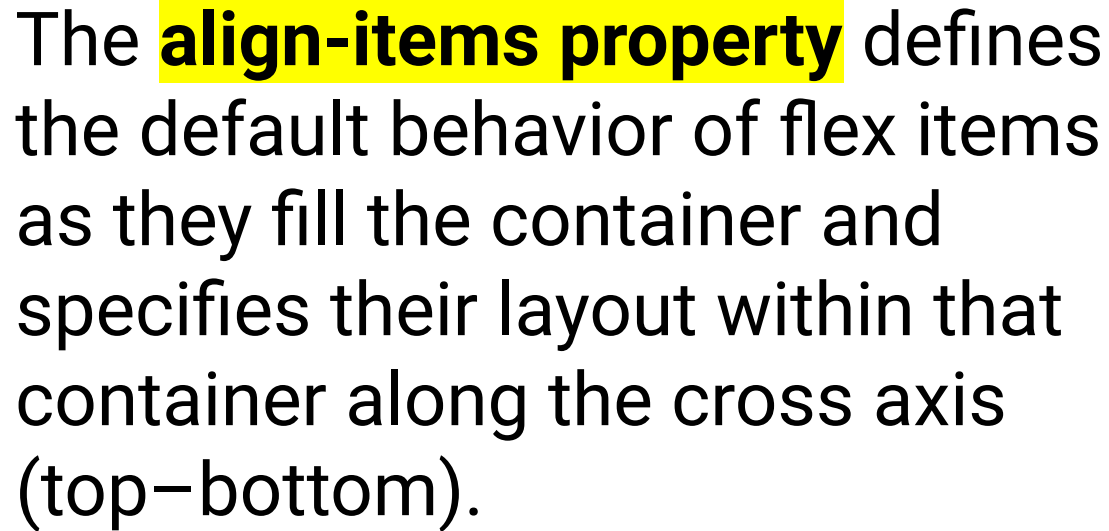
`align-items`

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.

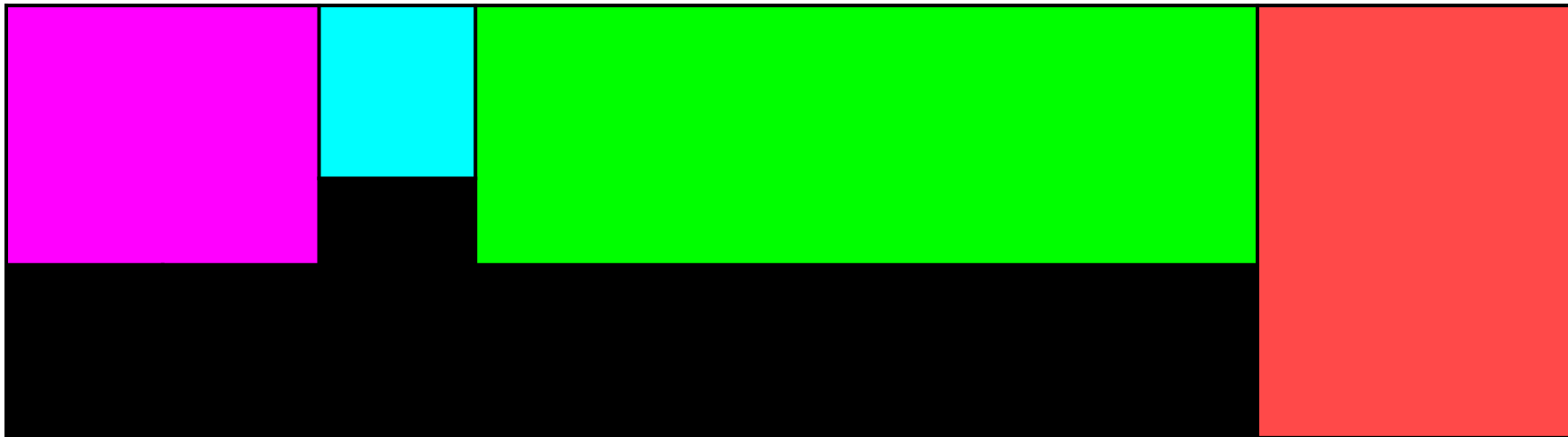
05

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

flex-start



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



flex-end



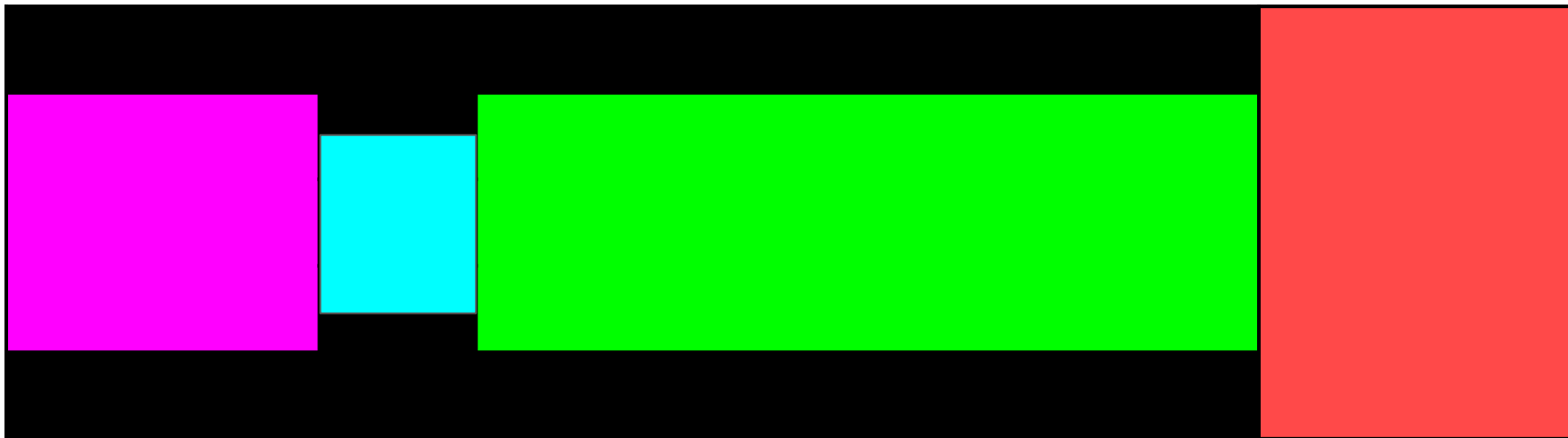
```
.flex-container {  
  align-items: flex-end;  
}
```



center



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



baseline



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



stretch

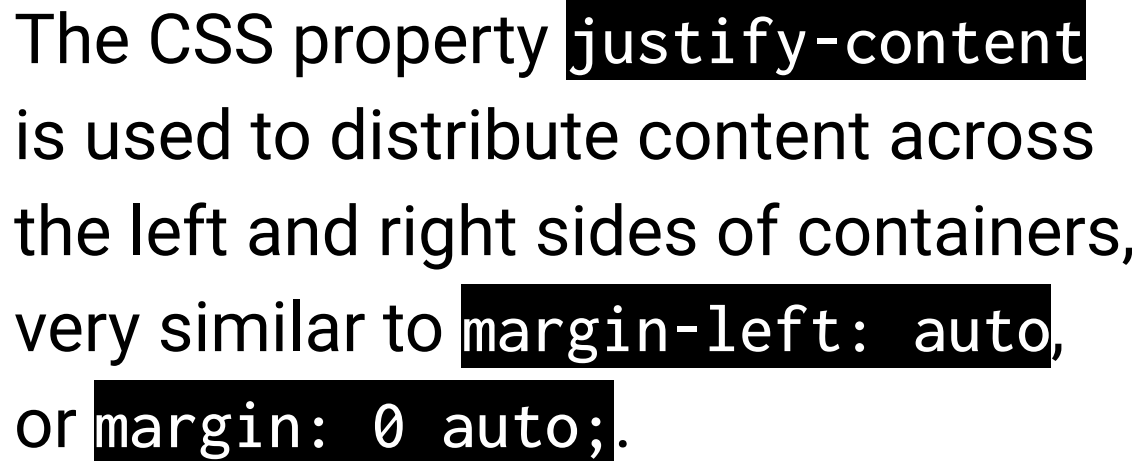


```
.flex-container {  
  align-items: stretch;  
}
```





The `justify-content` Property



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

The `justify-content` Property

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

There are six values:

- `flex-start` flex items are aligned at the start line (this is 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, last item 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.

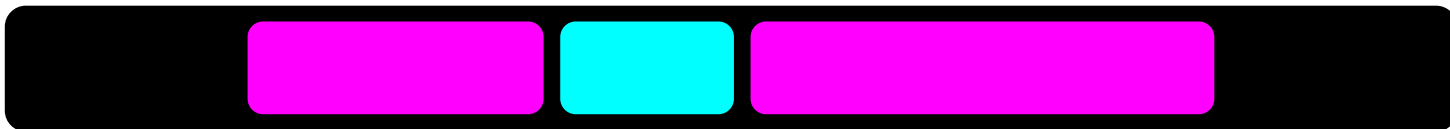
flex-start



flex-end



center



space-between



space-around



space-evenly





Activity: Aligning CSS Flexbox Items

In this activity, you'll learn how easy it is to align items inside their parent containers using CSS Flexbox.



CSS

Suggested Time:

15 minutes



Time's Up! Let's Review.

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?

Common Misconceptions and FAQs

01

`align-items` and `justify-content` are applied to the parent container and control content contained inside.

02

Using `align-items` and `justify-content`, it is possible to achieve almost any layout that requires responsive alignment.

Nesting CSS Flexbox Containers

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.



```
1
2 <div class="columnOne">
3   <div class="topRow">
4     <div class="card"></div>
5     <div class="card"></div>
6     <div class="card"></div>
7   </div>
8   <div class="middleRow"></div>
9   <div class="bottomRow"></div>
10 </div>
```



```
3  /* COLUMN STYLES */
4  .columnOne {
5    display: flex;
6    flex-direction: column;
7    background-color: lightblue;
8    height: 800px;
9    padding: 50px;
10   margin: 15px;
11 }
```



```
21 /* Nested Column Styles */
22 .topRow {
23   height: 50%;
24   background-color: blue;
25   display: flex;
26   align-items: center;
27   justify-content: center;
28 }
```

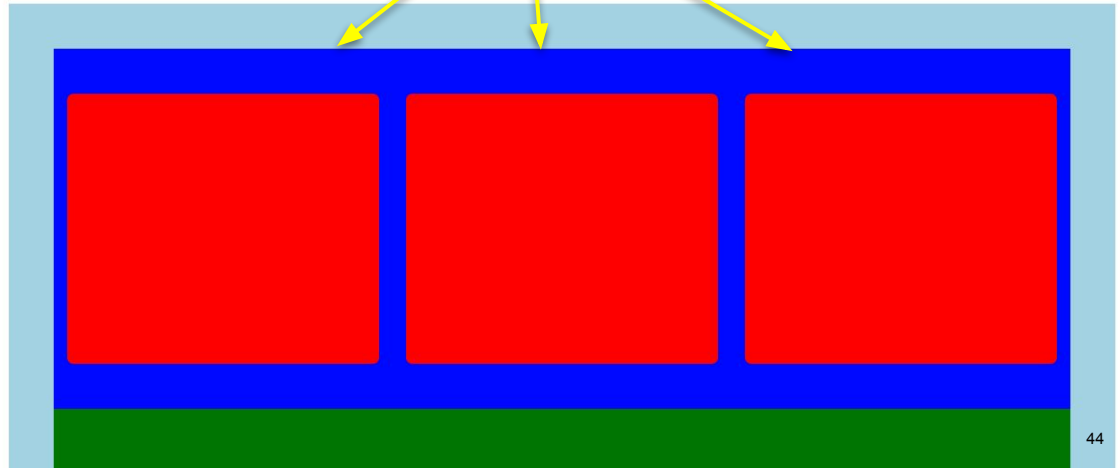
Nesting CSS Flexbox Containers

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



```
21  /* Nested Column Styles */
22  ▼ .topRow {
23      height: 50%;
24      background-color: blue;
25      display: flex;
26      align-items: center;
27      justify-content: center;
28  }
```

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



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/>





Time's Up! Let's Review.



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

Let's Review: CSS Flexbox Froggy

You nest flex containers inside each other in one of two scenarios:

01

If you want to control the children of a flex container with `justify-content` or `align-items`.

02

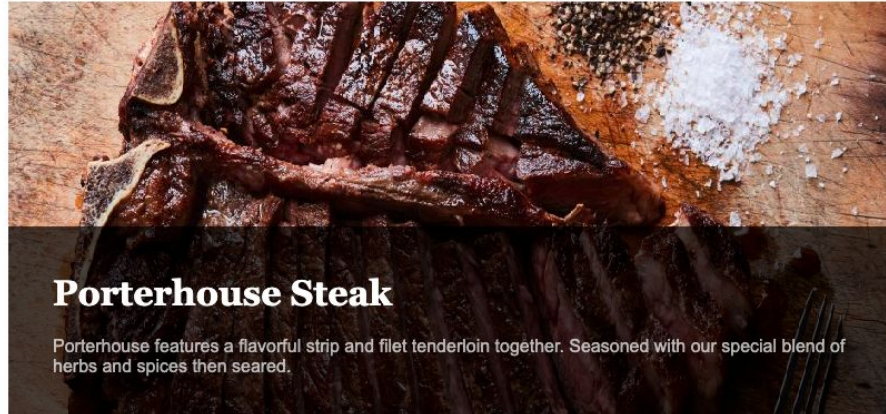
When you need a complex flex structure like a column with rows or vice versa.



Break

Introduce Jake's Eatery Activity

JAKE'S EATERY

[Order Online](#)[Menu](#) [Our Location](#) [About Us](#) [Text Us](#)[Order Now](#)



Activity: Jake's Eatery

In this activity, you'll build Jake's Eatery from scratch, starting from the HTML up.

Suggested Time:

30 minutes



Time's Up! Let's Review.

Let's Review: Jake's Eatery

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 nesting and alignment

We learned how to nest flex containers and align the children.



02

CSS Flexbox basics

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



03

Building Jake's Eatery

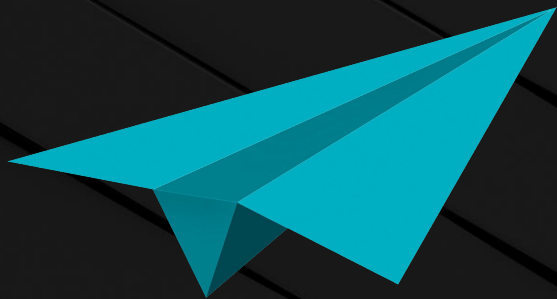
We applied our flex skills by building a full design in under an hour!



Questions?



*The
End*



Office Hours

30 Minutes