

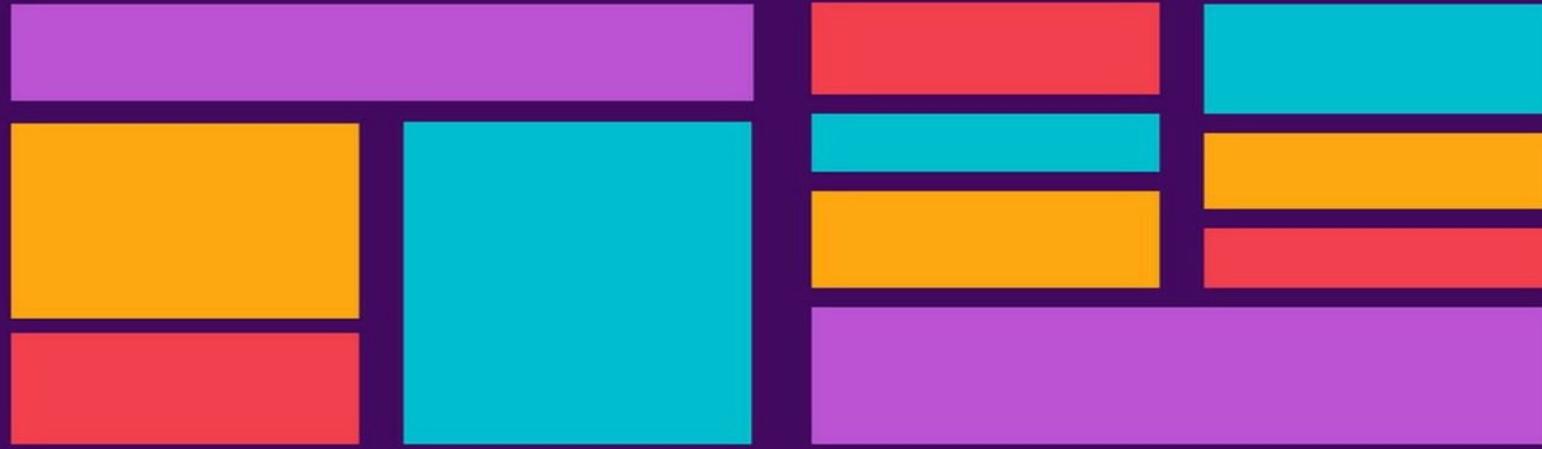
# Facilitation Guide

**Here are some quick tips for facilitating engaging classes:**

- Skip to Slide 2 and use presenter view. You can still view your notes and engage with your audience!
- The notes contain helpful lines of inquiry to lead your class with
- Additional resources(videos, articles, exercises) for this lesson are in the facilitation guide
- Review your TKH instructional/pedagogy guide for further help

**You should...**

1. Introduce yourself and share some of your experience
2. Ask questions and debrief
3. Provide opportunities for learners to collaborate
4. Collect survey data(when possible)
5. Thank your participants for their time and energy
6. Share feedback with your team
7. Have **FUN!**



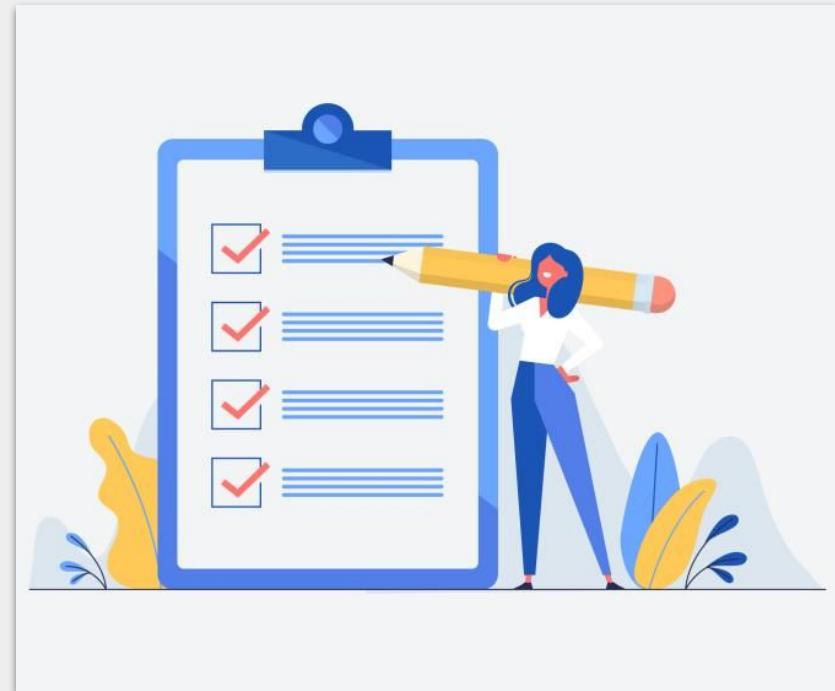
# CSS FLEXBOX

## CSS Layouts with Flex



# Learning Objectives

1. Develop familiarity with Cascading Style Sheets
2. Understand the importance of Flex in web design
3. Practice using CSS Flex to position elements on a page



# Fist-to-Five

Raise your hand with the amount of fingers that correspond to your response

- 1 = Never heard of CSS Flex before today
- 3 = Have played Flex Froggy or Zombies
- 5 = Made responsive layouts with Flex



# Warm-up

Share responses to the following amongst other members

- How do you currently position elements on a webpage?
- Have you ever struggled with positioning elements on a webpage?



# The Limits of HTML

While HTML was a huge step forward, alone it only created static web pages with very limited style and no interaction



## Welcome to Amazon.com Books!

*One million titles,  
consistently low prices.*

(If you explore just one thing, make it our personal notification service. We think it's very cool!)

### SPOTLIGHT! -- AUGUST 16TH

These are the books we love, offered at Amazon.com low prices. The spotlight moves EVERY day so please come often.

### ONE MILLION TITLES

Search Amazon.com's [million title catalog](#) by author, subject, title, keyword, and more... Or take a look at the [books we recommend](#) in over 20 categories... Check out our [customer reviews](#) and the [award winners](#) from the Hugo and Nebula to the Pulitzer and Nobel... and [bestsellers](#) are 30% off the publishers list...

### EYES & EDITORS, A PERSONAL NOTIFICATION SERVICE

Like to know when that book you want comes out in paperback or when your favorite author releases a new title? Eyes, our tireless, automated search agent, will send you mail. Meanwhile, our human editors are busy previewing galleys and reading advance reviews. They can let you know when especially wonderful works are published in particular genres or subject areas. Come in, [meet Eyes](#), and have it all explained.

### YOUR ACCOUNT

Check the status of your orders or change the email address and password you have on file with us. Please note that you **do not** need an account to use the store. The first time you place an order, you will be given the opportunity to create an account.

# CSS Brings Style to the Web

CSS was created to give webpages new structure, organization, and best of all, **STYLE!**



# Different Methods of Applying CSS

- **Inline style sheet** is defined within a tag. Applies only to that particular occurrence of that tag.
- **Internal style sheet** (also called Embedded) is defined within the head section of a page. Applies to that page only.
- **External style sheet** defined in a separate file, hence external. Applies to all pages that link to the external style sheet.

The diagram illustrates three methods of applying CSS in an HTML document:

- Link to external style sheet:** A red box highlights the line `<link rel="stylesheet" href="example.css" type="text/css">`. An arrow points from this box to the text "Link to external style sheet".
- Internal style sheet:** A red box highlights the `<style type="text/css">` block containing the rule `.auto-style1 { text-align:right; background-color:wheat; }`. An arrow points from this box to the text "Internal style sheet".
- Inline style:** A red box highlights the `<p style="text-align:left;margin-left:40px;">` line. An arrow points from this box to the text "Inline style". Below this line, a note explains: "This paragraph uses an **inline** style to align the text at the left but indents the text by 40px." Another note below it says: "This paragraph uses a class from the **internal** style sheet to align at the right and add a background color."

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2 <html>
3
4 <head>
5 <meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
6 <title>Example Page for Style Sheets</title>
7 <link rel="stylesheet" href="example.css" type="text/css"> ← Link to external
8
9 <style type="text/css"> ← Internal style sheet
10 .auto-style1 {
11   text-align:right;
12   background-color:wheat;
13 }
14 </style>
15 </head>
16
17 <body>
18 <p style="text-align:left;margin-left:40px;"> This paragraph uses an inline style to align the ← Inline style
19 text at the left but indents the text by 40px.</p>
20 <p class="auto-style1"> This paragraph uses a class from the internal style sheet to align at
21 the right and add a background color.</p>
22 <p> This paragraph uses the styling set in the attached external style sheet for all P tags. It
23 is centered.</p>
24 <p class="example1"> This paragraph uses a class from the attached external style sheet
25 to align text at the left, add a background color, and sets a width of 200px.</p>
26
27
28 </body>
29 </html>
```

# CSS Syntax

A CSS rule has two main parts: a **selector**, and one or more **properties** set to a **value**:



- The selector is normally the HTML element you want to style.
- Each declaration consists of a property and a value.
- The property is the style attribute you want to change. Each property has a value.
- **IMPORTANT:** Always end your declaration with a semicolon



# Commonly Used CSS Properties

Popular CSS Property/Values Usage Examples...			
Property*	Description	Values	Example - Selector {Property:Value }
<code>background-image</code>	Sets an image as the background	<code>url(?)</code>	<code>h2 {background-image: url(flowers.jpg) }</code>
<code>border</code>	Sets the properties for the four borders	<code>border-width</code> <code>border-style</code> <code>border-color</code>	<code>p {border-width: 3px 5px 3px 5px }</code> <code>h3 {border-style: double }</code> <code>h2 {border-color: red }</code>
<code>font-style</code> <code>font-size</code> <code>font-family</code>	Sets the properties for a font	<code>bold, italic, etc.</code> <code>px, em, pt, %</code> <code>arial, times, etc.</code>	<code>h4 {font-style: italic }</code> <code>div {font-size: 14px }</code> <code>body {font-family: verdana }</code>
<code>height</code>	Sets the height of an element	<code>px, em, pt, %</code>	<code>img {height: 200px }</code> <code>div {height: 90% }</code>
<code>list-style-type</code>	Sets the style of the list item (li) bullet	<code>disc</code> <code>circle</code> <code>square</code>	<code>ul {list-style-type: disc }</code> <code>ul {list-style-type: circle }</code> <code>ul {list-style-type: square }</code>
<code>margin</code>	Sets margin properties	<code>px, em, pt, %</code>	<code>h2 {margin: 15px }</code> <code>body {margin: 10% }</code> <code>div {margin: 1em }</code>
<code>padding</code>	Sets padding properties	<code>px, em, pt, %</code>	<code>h1 {padding: 10px }</code> <code>p {padding: 10% }</code>

\* Not all values for the properties listed are shown in this chart. As CSS evolves, new properties are emerging — refer to W3C.org.

For an extended list of Properties visit <http://www.w3schools.com/cssref/>



# Manipulating Images

Which of these look better?



The Knowledge House, a non-profit investing in the next generation of technologists! Help us achieve our goal of developing a pipeline of tech talent from low ...

1



The Knowledge House, a non-profit investing in the next generation of technologists! Help us achieve our goal of developing a pipeline of tech talent from low ...

2

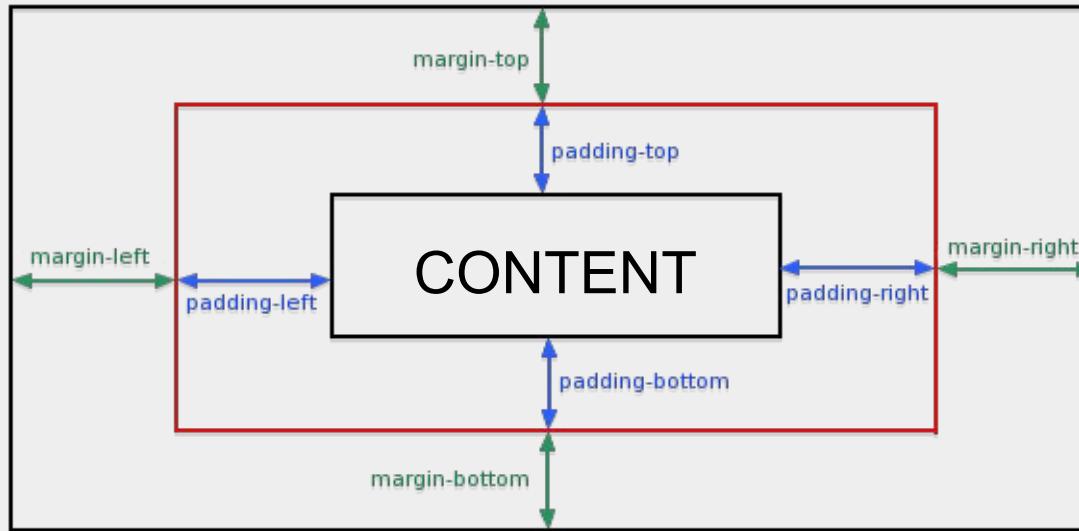


# Padding vs Margin vs Border

**Padding** - Clears an area around the content. The padding is transparent

**Border** - A border that goes around the padding and content

**Margin** - Clears an area outside the border.





# Questions? Comments? Emotional Outbursts?!



# Website Layout

What do these Websites have in common?

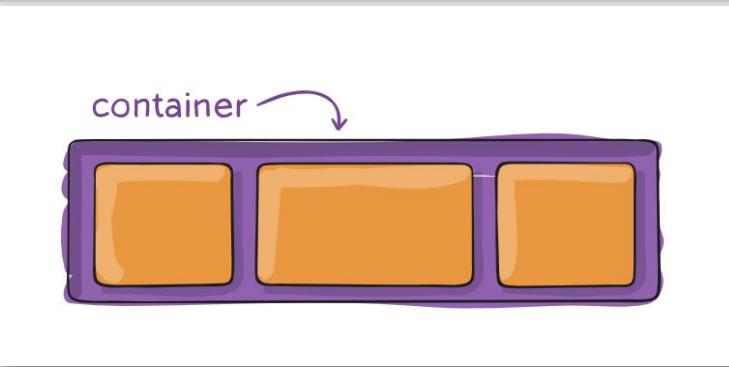
The image displays three distinct website layouts side-by-side:

- Hackleman, Olive & Judd, P.A.**: A professional law firm website featuring a large header image of Kristy Armada, a navigation bar with links like "About Us", "Attorneys", "Practice Areas", "News & Resources", and "Contact Us". Below the header is a bio for Kristy Armada, a "Practice Areas" sidebar, and a "Tweets" section.
- toby powell**: A personal website for Toby Powell. It features a large, abstract, textured background image. The top navigation includes "portfolio", "3D modeling", "contact", and "blog". Below the navigation is a bio for Toby Powell, followed by sections for "about", "HIGHLIGHTS", and "HOWS".
- Max Stalling**: A website for Max Stalling, a musician. It has a dark header with navigation links. The main content area features a large photo of Max Stalling, a bio, and a "BIOGRAPHY" section. Below this are links for "ABOUT", "HIGHLIGHTS", and "HOWS". There's also a "RECEIVE NEWSLETTER" form and a "BOOKING CONTACT INFO" button.



# CSS Flex

Since flexbox is a whole module and not a single property, it involves a lot of things including its whole set of properties. Some of them are meant to be set on the container (parent element, known as “flex container”) whereas the others are meant to be set on the children (said “flex items”).



# CSS Flex

## display

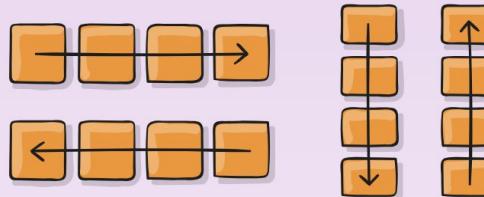
This defines a flex container; inline or block depending on the given value. It enables a flex context for all its direct children.

```
.container {  
  display: flex; /* or inline-flex */  
}
```

css

Note that CSS columns have no effect on a flex container.

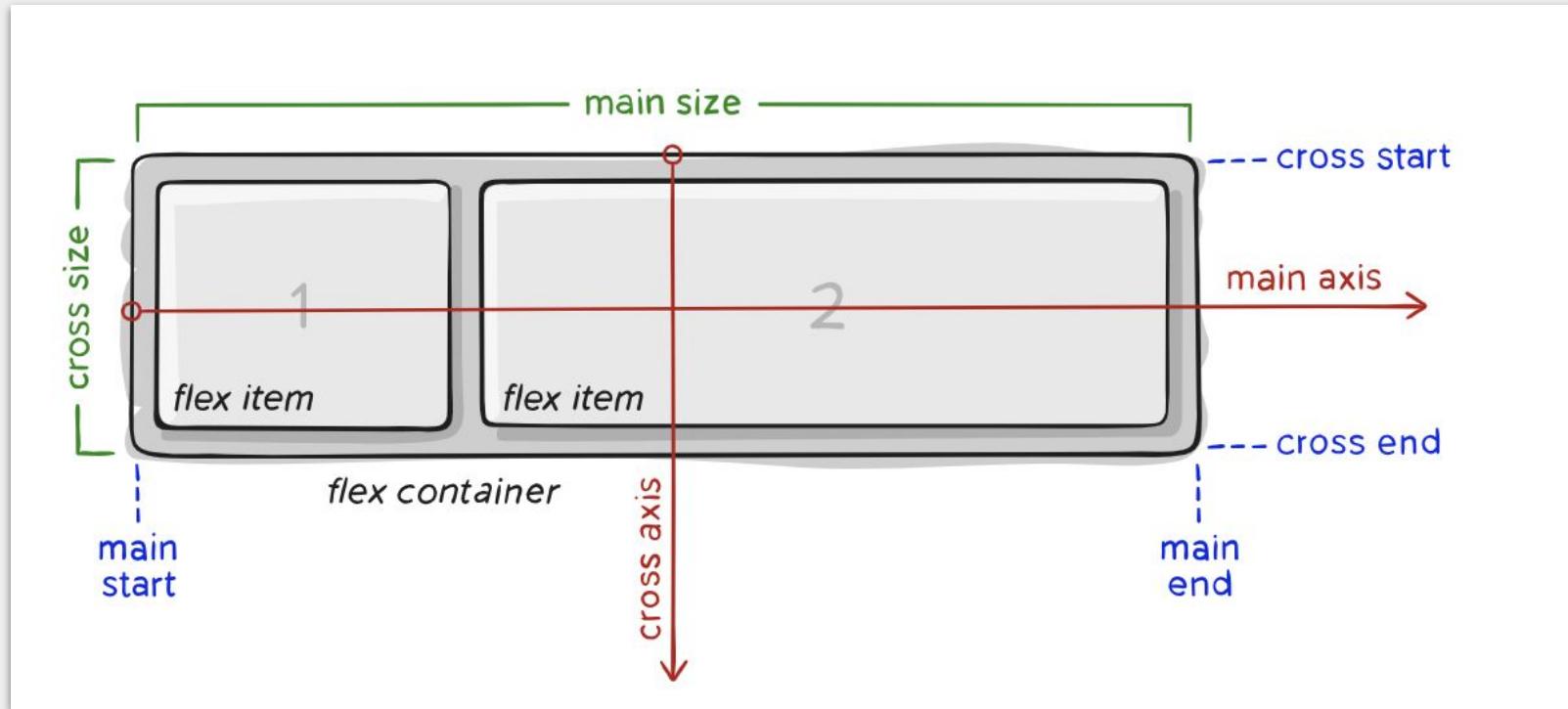
## flex-direction



This establishes the main-axis, thus defining the direction flex items are placed in the flex container. Flexbox is (aside from optional wrapping) a single-direction layout concept. Think of flex items as primarily laying out either in horizontal rows or vertical columns.



# CSS Flex



# CSS Flex

## Flexbox Cheat Sheet

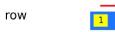
 @simonpaix

### Parent properties

**display:** enables flex context for all direct children.

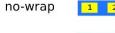
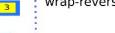
```
.container{ display: flex; // or inline-flex }
```

**flex-direction:** sets the main-axis.

row  column 

row-reverse  column-reverse 

**flex-wrap:** allows the items to wrap as needed.

no-wrap  wrap-reverse 

wrap 

### Children properties

**order:** changes the order of flex items.

```
.item{ order: 3 // the default is 0 }
```



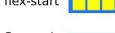
**flex-grow:** allows item to grow using remaining space.

```
.item-1 { flex-grow: 0 } //default .item-1 { flex-grow: 1 }
```

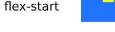
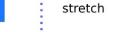
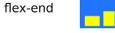


**Tip:** If all items have flex-grow: 1, the remaining space is distributed equally.

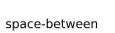
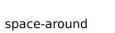
### justify-content: defines alignment along the main axis.

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

### align-items: defines alignment along the cross axis.

flex-start		stretch	
flex-end		baseline	
center			

### align-content: aligns multiple lines, like justify-content does with individual items.

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

### flex-shrink: defines the ability for a flex item to shrink.

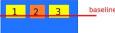
```
.one { flex-shrink: 1; } .two { flex-shrink: 2; } .three { flex-shrink: 3; } .four { flex-shrink: 4; }
```



**Tip:** Defaults to 1. The highest the value the more it shrinks compared to siblings.

### align-self: overrides default alignment (or the one specified by align-items) for a specific item.

flex-start		center	
flex-end			

**baseline** 

**stretch** 

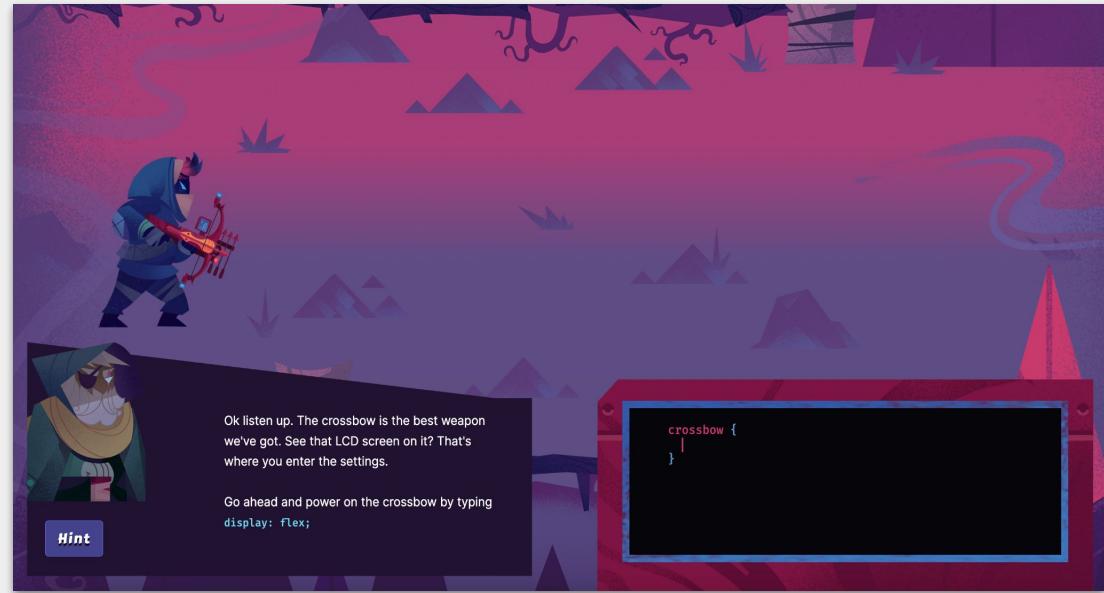
### flex-basis: sets the default size of a flex item. It accepts:

- specific values : pixels, rem, %
- auto : defaults to width or height property
- content : automatic sizing, based on its content
- global values : inherit, initial, unset

# Exercise - Coin Flip

Let's get our hands on the keyboard and learn to Flex! Navigate to the [Flexbox Froggy](#) or [Flexbox Zombie](#)

- Start with the first exercise
- Take 10 minutes to see how far you can get



# Flexbox Froggy

## FLEXBOX FROGGY

Level 1 of 24 ▶

Welcome to Flexbox Froggy, a game where you help Froggy and friends by writing CSS code! Guide this frog to the lily pad on the right by using the `justify-content` property, which aligns items horizontally and accepts the following values:

- `flex-start`: Items align to the left side of the container.
- `flex-end`: Items align to the right side of the container.
- `center`: Items align at the center of the container.
- `space-between`: Items display with equal spacing between them.
- `space-around`: Items display with equal spacing around them.

For example, `justify-content: flex-end;` will move the frog to the right.

```
1 #pond {  
2   display: flex;  
3   |  
4 }  
5  
6  
7  
8  
9  
10
```

[Next](#)

Flexbox Froggy is created by CodePen • [GitHub](#) • [Twitter](#) • [Settings](#)

Want to learn CSS grid? Play [Grid Garden](#).





# Flexbox Zombies

## Flexbox Zombies

Chapter 1

### The Crossbow

flex-direction

BEGIN

Chapter 2

### The Justify Laser

justify-content

Chapter 3

### The Alignment Lasers

align-items

Chapter 4

### One of These Things is Not Like the Other

align-self

Chapter 5

### Where the Wild Things Grow

flex-grow

Chapter 6

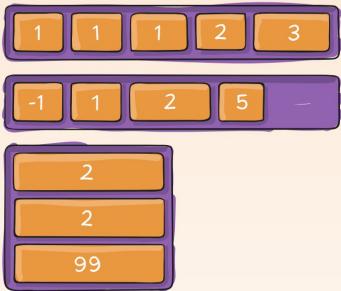
### That Shrinking Feeling

flex-shrink



# CSS Flex

## order



By default, flex items are laid out in the source order. However, the `order` property controls the order in which they appear in the flex container.

```
.item {  
  order: 5; /* default is 0 */  
}
```

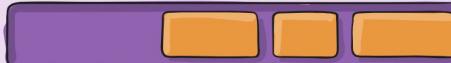
Items with the same `order` revert to source order.

## justify-content

### flex-start



### flex-end



### center



### space-between



### space-around

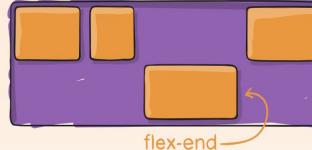


### space-evenly



## align-self

### flex-start



This allows the default alignment (or the one specified by `align-items`) to be overridden for individual flex items.

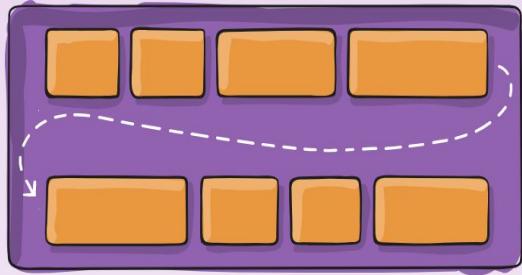
Please see the `align-items` explanation to understand the available values.

```
.item {  
  align-self: auto | flex-start | flex-end | center | ba  
}
```

Note that `float`, `clear` and `vertical-align` have no effect on a flex item.

# CSS Flex

## flex-wrap



By default, flex items will all try to fit onto one line. You can change that and allow the items to wrap as needed with this property.

css

```
.container {  
  flex-wrap: nowrap | wrap | wrap-reverse;  
}
```

- nowrap (default): all flex items will be on one line
- wrap: flex items will wrap onto multiple lines, from top to bottom.
- wrap-reverse: flex items will wrap onto multiple lines from bottom to top.



# Exercise

- **Objective:** Use your newfound knowledge of CSS Flex to execute at least one of the following 3 layouts
  - Each layout should be its own **Repl** with an `index.html` and `styles.css` page
  - Your CSS should be linked externally to the `styles.css`
  - Style it to make it easier to read
- **How to Submit Assignment:**
  - Place in a **Replit** project to submit for homework later
- **Share!**



# CSS Layouts

1

2

3

4

5

Layout 1

Layout 2

1

2

3

Layout 2



# CSS Layouts

Layout 3

1

2

3

4



# Debrief

1. What are your initial reactions to using CSS Flex?
2. How could you have used CSS Flex in past projects? What are some other ways to position elements on a page



# Thank You

Thanks so much for your time and energy! Would you mind filling out this brief survey to tell us about your experience? Or consider leaving a Google Review!

Please leave any questions and comments here for us to respond to via email

- What more would you like to learn in this lesson?



# Next Steps

# Revisions