

HTML5 & CSS3

Class 04

Today's Agenda

1. Discuss homework
2. Flexbox
3. Practice Conditionals
4. Plan and write a program
5. QA, Testing
6. Homework

Review



Homework Example

<https://kev-n.github.io/html200-ecommerce-project/>

Cool hover effect on logo, nice interpretation of mocks

Homework Example

https://github.com/davednguyen/html200-commerce-project/blob/gh-pages/sketch/wearhouse_pages/sketch.pdf

Great sketch of intended design.

Homework Example

view-source:<https://vidyak2.github.io/html200-e-commerce-project/>

Excellent code style and organization.

Questions?



Flexbox



Intro

Allows a container element to layout, align, and distribute space among its children, even if their sizes are dynamic or unknown.

Download `flexbox-practice.zip` and open

Don't Go Overboard

Use judiciously.

Not appropriate for entire page layout.

Don't use on every section/div on a page.



Flex Container Attributes

display

flex-direction

flex-wrap

flex-flow

justify-content

align-items

display:

flex

Container is a block-level flex element.

inline-flex

Container is an inline-level flex element.

flex-direction:

row

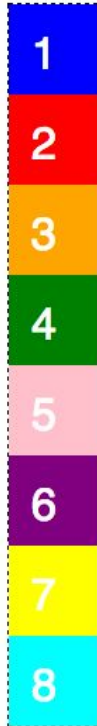


row-reverse

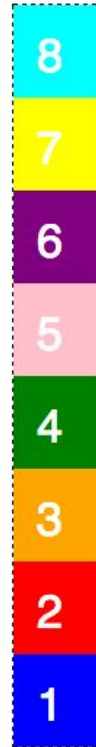


flex-direction:

column

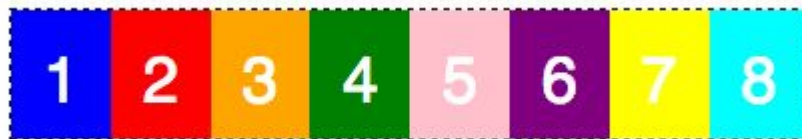


column-reverse



flex-wrap:

no-wrap



wrap



wrap-reverse



flex-flow:

flex-direction and flex-wrap in one attribute.

flex-flow: row wrap;

flex-flow: column-reverse no-wrap;

justify-content:

Aligns items along the main axis of the container. You'll get different results if your flex-direction is "column" vs "row".

There must be extra space around items for this to have any visible effect.

Values:

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

justify-content:

flex-start



flex-end



justify-content:

space-between



space-around



Practice

Write one set of style rules that produce the two layouts below based on width of browser window.



Practice

Write one set of style rules that produce the two layouts below based on width of browser window.



Practice

Set height of
.flex-container to 40rem,
then write rules to create
this layout.



align-items:

Aligns items on the “cross axis” (on vertical if direction is row, on horizontal if direction is column).

Values:

flex-start, flex-end, center, stretch, baseline

align-items:

flex-start

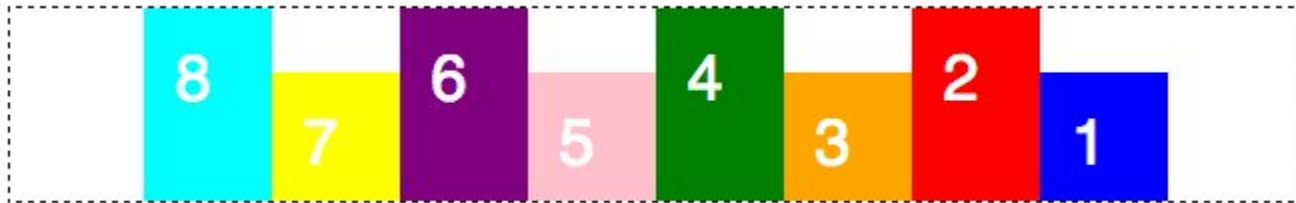


flex-end



Practice

Write style rules to change height of every other div to 6rem, then to produce the following layout.



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



PERFECT CENTERING OMG

Practice: Perfect Center

Add a `<header>` that contains an `<h1>` to index.

Style the header to have background color and height, then make it a flex container and perfectly center the `h1` within it.

Celebrate.



HEADLINE

align-content:

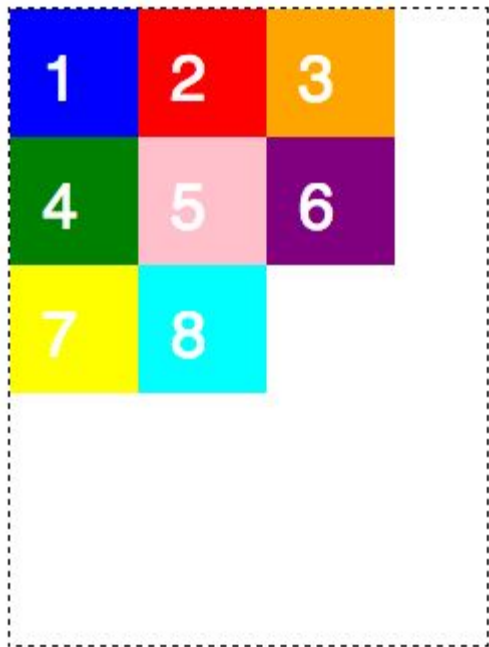
Like justify-content, but for space between wrapped lines of items. Only works if there is space available.

Values:

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

align-content

flex-start



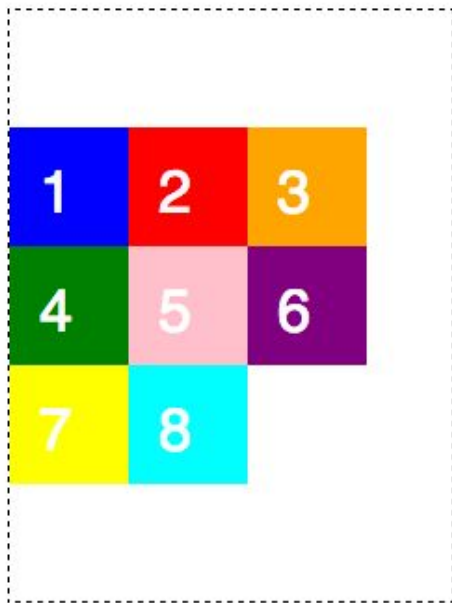
stretch

(requires no height
be set on items)

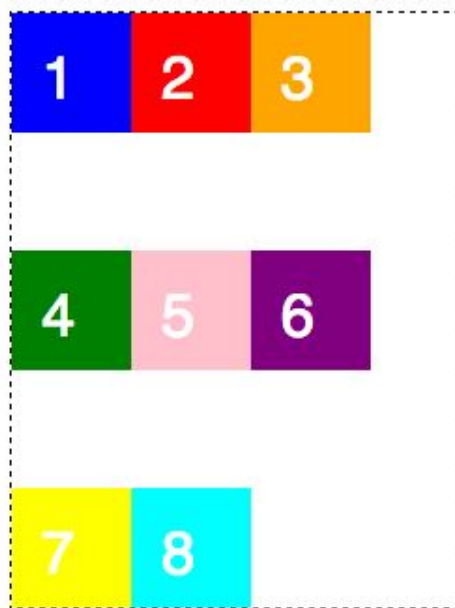


align-content

center

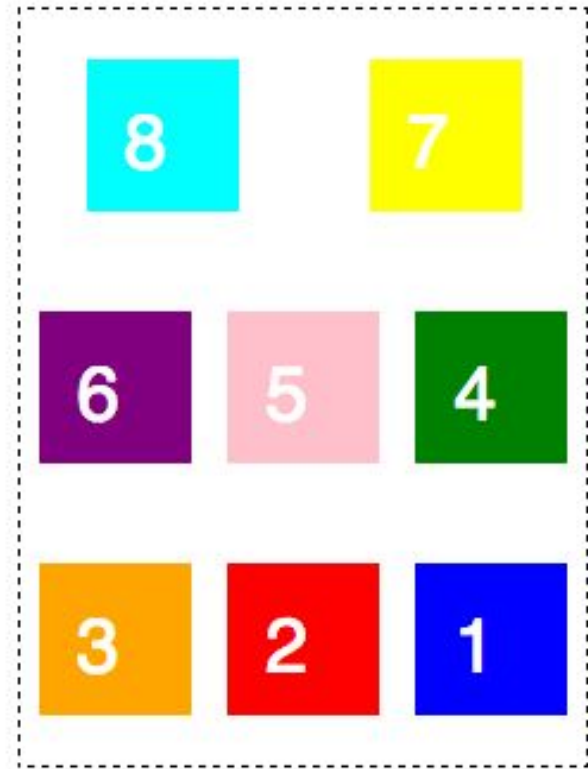


space-between



Practice

Set `.flex-container`'s height to 20rem and width to 15rem.
Then recreate this layout.



Practice

Set `.flex-container`'s height to 20rem and width to 15rem.
Then recreate this layout.



Also,

Heck yeah you can nest flexbox layouts!

Be aware that margins applied to flex items can complicate things.

Practice Conditionals



```
var loggedIn = false;
var secretNumber = "3";
var guess = 0;

while(!loggedIn){
    if (guess === secretNumber){
        console.log("You guessed it!")
        loggedIn = true;
    }
}
```

Value of guess	Value of loggedIn at end of loop
0	
9	
6	
3	

```

var loggedIn = false;
var secretNumber = "3";
var guess = 0;

while(!loggedIn){
    if (secretNumber % guess == 0){
        console.log("You guessed it!")
        loggedIn = true;
    }
}

```

Value of guess	Value of loggedIn at end of loop
0	
42	
6	
3	
10	
1	
9	

```

var loggedIn = false;
var secretNumber = "3";
var guess = 0;

while(!loggedIn){
  var superSecret = secretNumber * 5;
  var encryptedGuess = guess/2;

  if (encryptedGuess % superSecret == 1){
    console.log("You guessed it!")
    loggedIn = true;
  }
}

```

guess	super Secret	encrypted Guess	loggedIn at end of loop
0			
42			
6			
3			
10			
1			
9			

A Program!



Setup

Create a new directory containing an HTML and JS file (adventure.js). HTML file should include minimum content and a headline.

Anatomy of a <script>

JS gets into a page in one of two ways, inline or external. We will prefer external scripts (just like with CSS).

```
<script type="text/javascript">console.log('hello!');</script>
```

```
<script type="text/javascript" src="js/my_file.js"></script>
```


Including JS

The script tag that imports a JS file should be at the end of the HTML file, the last thing before the closing `</body>` tag.

```
<script src="scripts/game.js"></script>  
</body>  
</html>
```

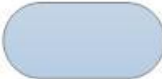




Text Based Adventure

User will seek a treasure using text interface. They can move in different directions (“north”, “east”) by entering them in prompt. This will move them around game play area until they find the hidden treasure.

Plan: Flowchart

Work with your Slack group to sketch up a flowchart for the program.

Be prepared to discuss with the class.

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

Flowchart Tools

Options:

1. Draw on paper
2. <https://www.draw.io>
3. Google Drive “drawing” file
4. Type it as an outline in Slack

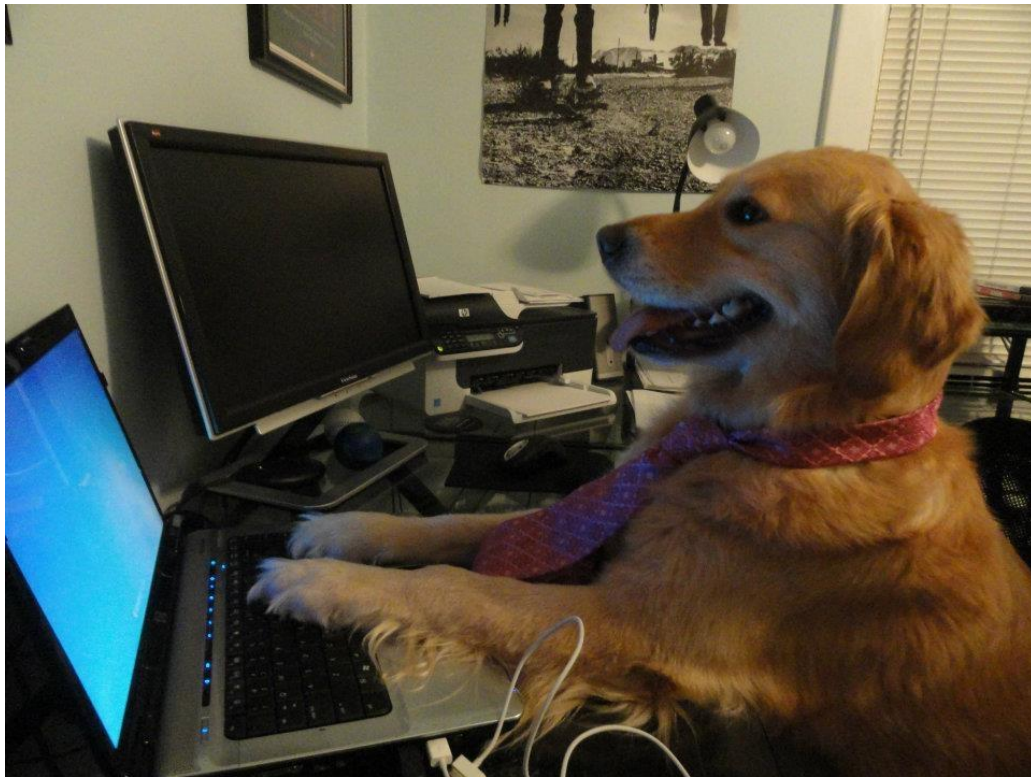
Plan: Psuedocode

We'll do this together.

Logic Check

We'll work through our psuedocode, tracking the value of variables given different inputs. The goal is to find errors or gaps in our logic.

Write Code



Test

Run the program repeatedly, enter weird inputs, see if anything breaks or there are any unexpected or undesirable outputs.

Refactor

Fix any bugs you found during testing.

Improve any undesirable outcomes found during testing.

Clean up code style, readability.

QA, Testing



Manual vs Automated

Automated tests repeat predefined actions, comparing program's expected and actual outcomes.

Manual tests are done by a human viewing program actual outcomes, comparing them to expected.

There are some tools for automated front end testing, but they are not comprehensive, and outside the scope of this class.

Business Logic

What are users supposed to be able to accomplish on this site? Can they?

Usability

- Ease of use- instructions, navigation
- General appearance
- Content is free of typos, grammar errors
- Subjective user satisfaction

Functionality

- HTML and CSS is valid
- All links work
- All pages load correctly
- All forms work correctly

Performance

How long does it take to load in different environments, browsers, and devices? Could that be improved?

<https://www.webpagetest.org/>

<https://tools.pingdom.com/>

Accessibility

Does site meet a11y standards?

<http://a11yproject.com/checklist.html>

<https://www.w3.org/WAI/intro/accessibility.php>

Without Javascript

Does the site still work???

<http://www.wikihow.com/Disable-JavaScript>

Browser/OS Compatibility

Use a tool or service to let you see how site works on other browsers and operating systems.

<https://www.browserling.com>

Chrome dev console device options

Homework

Note: Surveys are Coming



If you have opinions,
keep an eye on your
email for the link!

Homework

Improve the adventure game. Some suggestions are available in Canvas, but you're welcome to do any improvement you'll be proud of.

(optional) Read Chapter 3 of Responsive Web Design