CS 135I Winter 2024

# Week 1 Classwork

## Overview

Since you are all also enrolled in CS 133J and will be exploring GitHub on Thursday, we’ll use today’s class to start digging into our CSS flexbox/grid layout topic from week 2. This will also help offset our missing lecture period due to the MLK, Jr. holiday on Monday. I’d recommend using the File menu to either download a Word version of this sheet or to create a copy on your own Google drive. Fill out the notes as we go along in class, and complete the activity so that you will be able to upload your proof of completion to Moodle at the end of class.

## Flexboxes

Concept: Flexboxes are a newer technique for achieving page layouts that 1) simplify some previously complicated tasks (like creating equal height columns, or centering a block of text inside of some other element) and 2) lend themselves beautifully to responsive design

Creating flexboxes involves:

1. Creating HTML so that the items you wish to appear in rows/columns are direct children of a parent element
2. Adding \_\_\_display: flex;\_\_\_ to the styles for the \_\_parent element\_\_\_ in order to make a flexbox-container
3. Adding other flexbox-related properties as desired

Some common flexbox-related properties:

|  |  |  |  |
| --- | --- | --- | --- |
| Property Name | Sample Values | Description | Flex-container =or= flex-item =or both? |
| flex-direction | row, column, row-reverse or column reverse | Determines the main axis, horizontal or vertical, defining the flex direction | flex-container |
| flex-wrap | nowrap (default)  wrap  wrap-reverse | Allows browser to wrap items as needed. | flex-container |
| justify-content | Flex-start  Flex-end  Center  Space-between  Space-around  Space-evenly | Defines the alignment along the main/long axis. | flex-container |
| align-items | Flex-start  Flex-end  Center  Space-between  Space-around  Space-evenly | Defines the alignment along the alt/short axis. | flex-container |
| gap | gap  row-gap  column-gap | Sets the space between flex items.  Can use ems, px, etc. | flex-container |
| order | order: 3; | Changes display order in the flex container.  Must be a whole positive number. | flex-item |
| flex-grow (flex-shrink) | flex-grow: 2; | It dictates what amount of the available space inside the flex container the item should take up. | flex-items |
| align-self | Flex-start  Flex-end  Center  Space-between  Space-around  Space-evenly | This allows the default alignment (or the one specified by align-items) to be overridden for individual flex items. | flex-item |

Building Flexbox Fluency

* Play [the Flexbox Adventure](https://codingfantasy.com/games/flexboxadventure) - defeat the monsters, get the gold, and boost your understanding of Flexbox-related properties
  + I would recommend either the easy or the medium level as you get started. The content is the same (both will give you a hint about which properties are required), the easy level has a *Show Answer* feature if you get stuck.
  + The hard level is the same content, but without clues about which property to use in each encounter.
* Paste a screenshot of your successful encounter with the final level here:  
  (Displaying the “*Congratulations, you defeated the last boss!”* message)  
  **I was able to play in class.**

Applying Flexbox Concepts to a Webpage

1. Download the classwork01.zip file from Moodle
2. Extract the contents, and examine both the existing *index.html* file and the *flexbox.css* file.
3. Update the *index.html* file so that:
   1. The three <section></section> elements are all contained inside of a div with a class of *row*.
   2. The two <article></article> elements are both contained inside of a div with a class or *row*.
4. Update the *flexbox.css* file using flexbox-related properties so that:
   1. The unordered list in the nav area of the page is a flex-box container
   2. The flex-items inside of the nav ul container are aligned to the right edge of the page
   3. The items identified with the *row* class are flexbox containers
   4. The second article takes up twice as much space as the first article.   
      (Hint: the first article tag has an id of *first* and the second article tag has an id of *second* Hint 2: you’ll need to set both the *flex-grow* and *flex-basis* properties to make this work).
   5. Update the media query for smaller browsers so that
      1. The items inside the row containers are displayed in columns rather than rows
      2. The items inside the *nav ul* container are displayed in a column that takes 100% of the width of the browser window