**Lab 3 - Flexbox**

The goal of this lab is to help familiarise yourself with **flexbox**. You should not need to modify the index.html page, only index.css.

In this lab we will be using **functional** CSS. This focuses on using naming for CSS classes that reflects the functionality of the class, e.g. m-20 class will correspond to a 4-sided margin with a value of 20 units, and mt-20 will correspond to a margin-top value of 20 units.

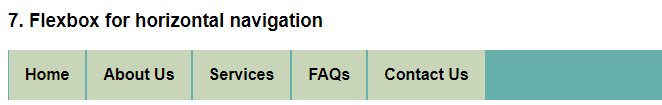
1. Familiarize yourself with the provided CSS and HTML files for Exercise 3, paying attention to the class names and element structure.
2. Basic flexbox :
   1. Add a CSS rule for the flex-container class and set it to use a flexbox display model
   2. Add rules for the flex-item-1, -2, -3, and -4 classes and set flex: <number> to them accordingly
   3. Compare how the HTML file looks in a browser before and after your changes. How does the value for the flex affect the size of the item? Answer: the new changes collapses the flex container’s width (no full page length) into single line

Before

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

1. Centering with flexbox :
   1. Add a CSS rule for the flex-center class and set values for justify-content and align-items
   2. Try out different values for both of the above CSS properties. Your final goal is to center an item within its container both horizontally and vertically
2. Assigning leftover space :
   1. Add a CSS rule for the flex-space-between class and experiment with different values for the justify-content property. Observe how the alignment of the flex items changes
   2. Set an appropriate value for justify-content to push the elements to the left and right sides with leftover space in the middle
3. Wrapping items :
   1. Add rules for the flex-basis-item-50 and flex-wrap classes so that the container has flex-wrap: wrap; and flex items have flex-basis set to 50%
   2. Observe what happens when changing the value for flex-wrap to wrap-reverse
   3. Choose a new value for flex-basis that makes all 4 flex items fit on the one line
4. Changing direction :
   1. Add a flex-direction property to the flex-container class and observe the behaviour when using different values. Use the link and reference below to find some of the commonly used values.
5. Flexbox for horizontal navigation :
   1. Add CSS for the flex-menu class and its children to make it look like the screenshot:  
      
   2. Use the justify-content property to align the menu items to the right
   3. Use the justify-content property to align the menu items to the center of the row
   4. Use the order property to move the FAQs menu item to last
   5. Use the flex-grow property to divide the total space evenly between each menu item, then center the links.

**Links:**

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

<https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_flexible_box_layout>

**Reference:**

Refer to these collected CSS values for flexbox here and try out various combinations of parameters:

flexbox container:

display: flex | inline-flex;

flex-direction: row | row-reverse | column | column-reverse;

flex-wrap: nowrap | wrap | wrap-reverse;

justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly;

align-items: stretch | flex-start | flex-end | center | baseline;

align-content: flex-start | flex-end | center | space-between | space-around | space-evenly | stretch;

flex-flow: <'flex-direction'> <'flex-wrap'>; // shorthand property

flexbox items:

flex-grow: <number>;

flex-shrink: <number>;

flex-basis: auto | <width>;

order: <number>;

align-self: auto | flex-start | flex-end | center | baseline | stretch;

flex: none | [ <'flex-grow'> <'flex-shrink'>? || <'flex-basis'> ]; // shorthand property