

## Task 6: Mastering Flexbox for a Responsive Profile Page Layout

### Objective:

Use CSS Flexbox to create a responsive and well-structured layout for a profile page. This task challenges you to use Flexbox to design a layout that adapts to different screen sizes, enhancing the user interface of the profile page. Focus on understanding Flexbox properties and how they control the positioning, alignment, and spacing of elements.

### Pre-requisites:

- Basic understanding of HTML elements and CSS properties
- Familiarity with a code editor like Visual Studio Code

### Concepts Covered:

- Creating a Flex Container
- Manipulating Flex Items
- Building a Responsive Navigation Bar

### Concepts:

#### 1. Creating a Flex Container:

Convert an existing section of your webpage or create a new section (like a header, main content, or footer) into a flex container using `display: flex;`. Understand how this changes the layout and behavior of the child elements (flex items).

```
.flex-container {  
  display: flex;  
  flex-direction: row;  
  justify-content: space-between;  
  align-items: center;  
}
```

#### 2. Manipulating Flex Items:

Experiment with Flexbox properties such as `flex-grow`, `flex-shrink`, and `flex-basis` to control the sizing of flex items. Use `justify-content` and `align-items` to align items horizontally and vertically within the flex container.

```
.flex-item {  
  flex-grow: 1;  
  flex-shrink: 1;  
  flex-basis: auto;  
}  
.container {  
  display: flex;  
  justify-content: center;  
  align-items: center;
```

```
height: 100vh;
}
```

### 3. Building a Responsive Navigation Bar:

Create a navigation bar using Flexbox, ensuring it is responsive and adjusts to different screen widths. Style the navigation bar and explore how Flexbox properties can be used to distribute space among the menu items or to align them to the left, right, or center.

```
.navbar {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 10px;
  background-color: #333;
}
.navbar a {
  color: #fff;
  padding: 14px 20px;
  text-decoration: none;
}
```

#### Setup:

##### 1. Install Visual Studio Code (VS Code):

Download and install VS Code from [Visual Studio Code](#).

##### 2. Web Browsers:

Use Google Chrome or Mozilla Firefox for viewing your webpage and utilizing their developer tools for debugging.

#### Tasks:

##### 1. Creating a Flex Container (15 minutes):

- Convert an existing section of your webpage or create a new section (like a header, main content, or footer) into a flex container using `display: flex;`.
- Understand how this changes the layout and behavior of the child elements (flex items).
- Example:

```
<div class="flex-container">
  <div class="flex-item">Item 1</div>
  <div class="flex-item">Item 2</div>
  <div class="flex-item">Item 3</div>
</div>
```

```
.flex-container {
  display: flex;
  flex-direction: row;
  justify-content: space-between;
  align-items: center;
}
```

## 2. Manipulating Flex Items :

- Experiment with Flexbox properties such as `flex-grow`, `flex-shrink`, and `flex-basis` to control the sizing of flex items.
- Use `justify-content` and `align-items` to align items horizontally and vertically within the flex container.
- Example:

```
<div class="container">
  <div class="flex-item">Centered Item</div>
</div>
```

```
.container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}
.flex-item {
  flex-grow: 1;
  flex-shrink: 1;
  flex-basis: auto;
}
```

## 3. Building a Responsive Navigation Bar :

- Create a navigation bar using Flexbox, ensuring it is responsive and adjusts to different screen widths.
- Style the navigation bar and explore how Flexbox properties can be used to distribute space among the menu items or to align them to the left, right, or center.
- Example:

```
<div class="navbar">
  <a href="#">Home</a>
  <a href="#">About</a>
  <a href="#">Services</a>
  <a href="#">Contact</a>
</div>
```


```
.navbar {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 10px;
  background-color: #333;
}
.navbar a {
  color: #fff;
  padding: 14px 20px;
  text-decoration: none;
}
.navbar a:hover {
  background-color: #ff5733;
  color: #fff;
}
```

### Instructions:

1. Write the required code in `index.html` and `styles.css`.
2. Open the `index.html` file in your web browser to ensure the code displays correctly.
3. Use the browser's developer tools to debug and inspect the elements.


### Resources:

- 



#### Basic concepts of flexbox - CSS: Cascading Style Sheets | MDN

The flexible box layout module (usually referred to as flexbox) is a one-dimensional layout...

 [https://developer.mozilla.org/en-US/docs/Web/CSS/CSS\\_Flexible\\_Box\\_Layout/Basic\\_Concepts\\_of\\_Flexbox](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox)

### Videos:



#### GitHub Instructions:

1. **Open in Visual Studio Code:**

After clicking on the "Open in Visual Studio Code" button from the GitHub Classroom confirmation page, VSCode will open the repository directly. If prompted, select "Open" or "Allow" to open the repository in VSCode.

2. **Open the Terminal in VSCode:**

In VSCode, open a terminal by selecting Terminal > New Terminal from the top menu.

3. **Complete the Task:**

In VSCode, write your solution in the `index.html` and `styles.css` files.

4. **Run and Test Your Code:**

Open your `index.html` file in a web browser to ensure it works correctly. Use the following command:

```
open index.html
```

5. **Commit Your Changes:**

In the VSCode terminal, add your changes to git:

```
git add index.html styles.css
```

Commit your changes with a meaningful message:

```
git commit -m "Completed task 10"
```

## 6. Push Your Changes to Your Repository:

Push your changes to your forked repository:

```
git push origin main
```

## 7. Create a Pull Request:

Go to your repository on GitHub.

Click on the "Pull Requests" tab.

Click the "New Pull Request" button.

Ensure the base repository is the original template repository and the base branch is `main`.

Ensure the head repository is your forked repository and the compare branch is `main`.

Click "Create Pull Request".

Add a title and description for your pull request and submit it.

## Summary of Commands:

```
# Open in Visual Studio Code

# Open terminal in VSCode

# Complete the task by editing index.html and styles.css

# Navigate to the directory containing index.html
cd path/to/your/index.html

# Run your code
open index.html

# Add, commit, and push your changes
git add index.html styles.css
git commit -m "Completed task 6"
git push origin main

# Create a pull request on GitHub
```