W6 PRACTICE

CSS Media Queries

At the end of this practice, you should be able to…

✓ Identify the best **CSS media queries approach** in different use cases
✓ From mockups to Web Responsive

How to work?

✓ Read carefully with the instruction and start planning before writing code
✓ Open the folder in VS Code

How to submit?

✓ **Push** your final code on **your GitHub repository**
✓ Then **attach the GitHub path** to the MS Team assignment and **turn it in**

Are you lost?

You can read the following documentation to be ready for this practice:

- *https://www.w3schools.com/css/css3_mediaqueries.asp*
- *https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_media_queries/Using_media_queries*

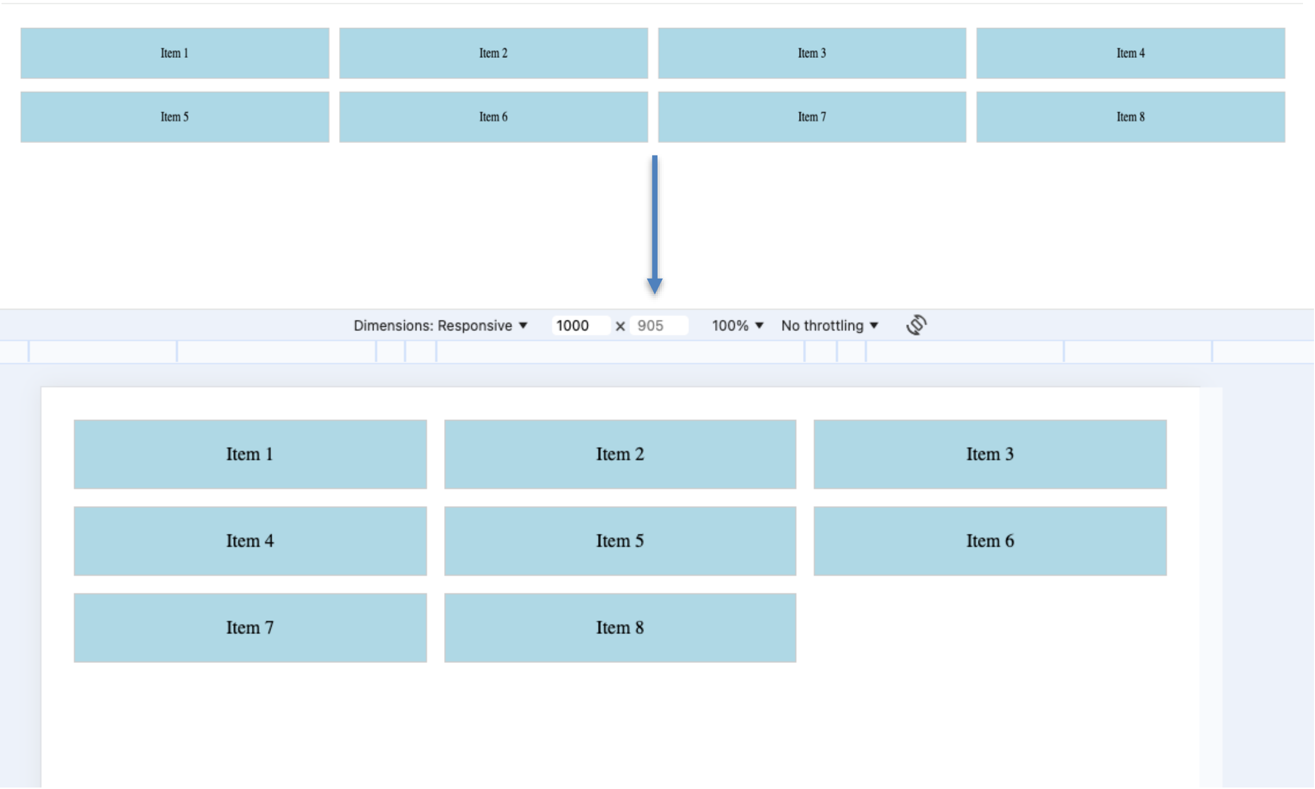
EXERCISE 1: The Basics!

Design a Responsive Grid Layout

Goal: Create a grid-based layout for a portfolio or product listing page.

Steps:

- Use CSS Grid or Flexbox to define a grid with 3 columns for large screens.
- Adjust the grid to 3 columns for (max-width 1000px), 2 for (max-width 768px) and 1 column for smaller devices (max-width 480px).
- Ensure grid items resize proportionally by setting flexible widths (e.g., fr or %).
- Add padding/margin adjustments between items at different screen sizes.



Dimensions: Responsive ▾ 1000 × 905 100% ▾ No throttling ▾

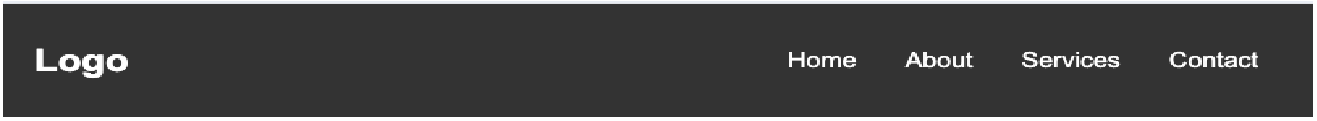
Are you lost?

TIP 1 - Here are the CSS properties involved in this exercise:

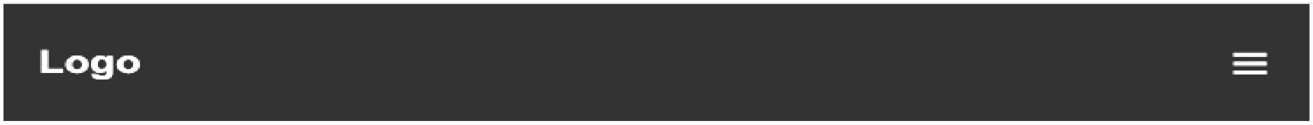
- grid-template-columns
- display
- gap

EXERCISE 2: Responsive Navigation Bar

In this exercise, you will create a **responsive navigation bar** using **HTML** and **CSS**. The navigation bar should adapt seamlessly to different screen sizes by displaying a traditional horizontal layout on larger screens and switching to a mobile-friendly dropdown menu on smaller screens.



It changes to below when screen size less than 768px

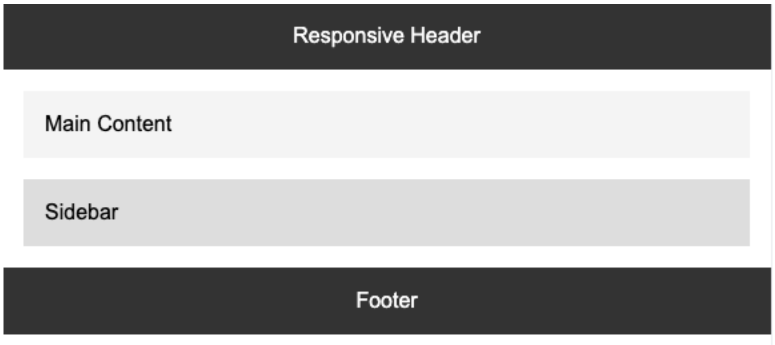


EXERCISE 3: Responsiveness with flex layout

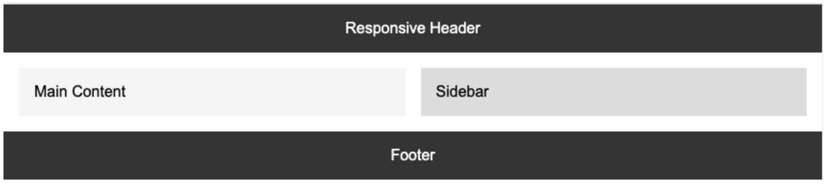
In this exercise, you will build a **responsive webpage layout** using **HTML** and **CSS** that adapts to different screen sizes, leveraging **Flexbox** for a flexible and clean design. The layout will have a **header**, a **main content area** that includes a **content section** and a **sidebar**, and a **footer**.

Key Features to Implement:

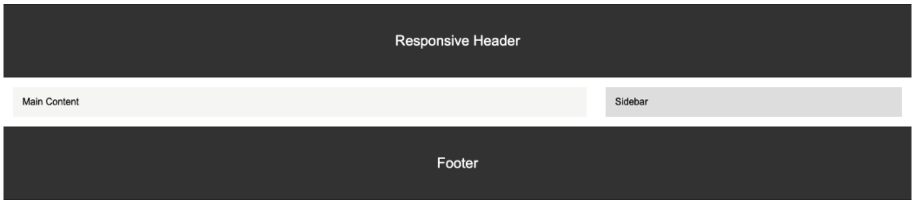
1. **Mobile Layout (Default - Below 600px):**
   - *The header, content, and sidebar are stacked vertically (one column).*
   - *The content and sidebar sections should take equal width, with padding for spacing.*



2. **Medium Screens (600px to 900px):**
   - *The layout will switch to a two-column format (content and sidebar side-by-side) using Flexbox.*
   - *The header and footer will be slightly enlarged for better readability on medium-sized devices.*

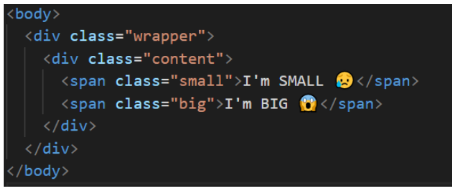


3. **Large Screens (900px and wider):**
   - *The layout will remain in two columns, but the header and footer will have more padding and larger text.*
   - *The content area will take more space, with the sidebar shrinking a little*

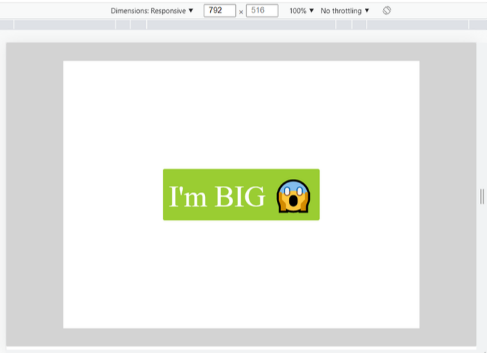


EXERCISE 4: media and Clamp function

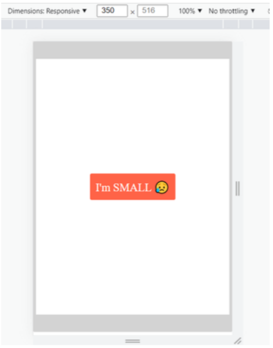
1. with “@media only screen” property



```
<body>
  <div class="wrapper">
    <div class="content">
      <span class="small">I'm SMALL 😥</span>
      <span class="big">I'm BIG 😱</span>
    </div>
  </div>
</body>
```



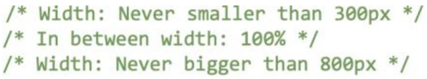
Dimensions: Responsive ▾ 792 × 516 100% ▾ No throttling ▾



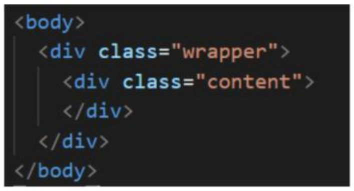
Dimensions: Responsive ▾ 350 × 516 100% ▾ No throttling ▾

2. with “clamp” property

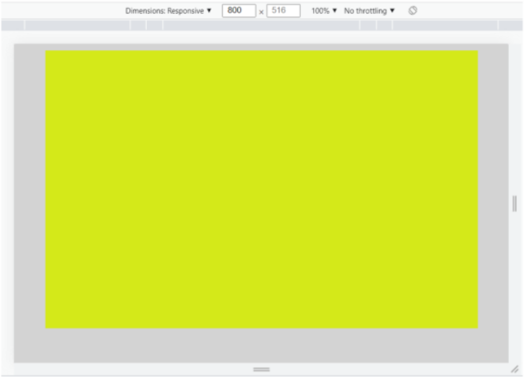
Please make the green box size restricted to the following conditions with using Clamp function



```
/* Width: Never smaller than 300px */
/* In between width: 100% */
/* Width: Never bigger than 800px */
```



```
<body>
  <div class="wrapper">
    <div class="content">
    </div>
  </div>
</body>
```



Dimensions: Responsive ▾ 800 × 516 100% ▾ No throttling ▾