

Lesson:

Chrome dev tools

for CSS



Topics

- Introduction to the Chrome dev tool for CSS
- Inspecting and selecting the HTML elements
- Applying temporary CSS changes for testing
- Using box model tools for layout and positioning
- Using the Color picker and Color contrast analyzer
- Working with media Query and responsive designs
- Testing the CSS flexbox and grid layout

#pre-requisite - Chrome browser should be installed in the system.

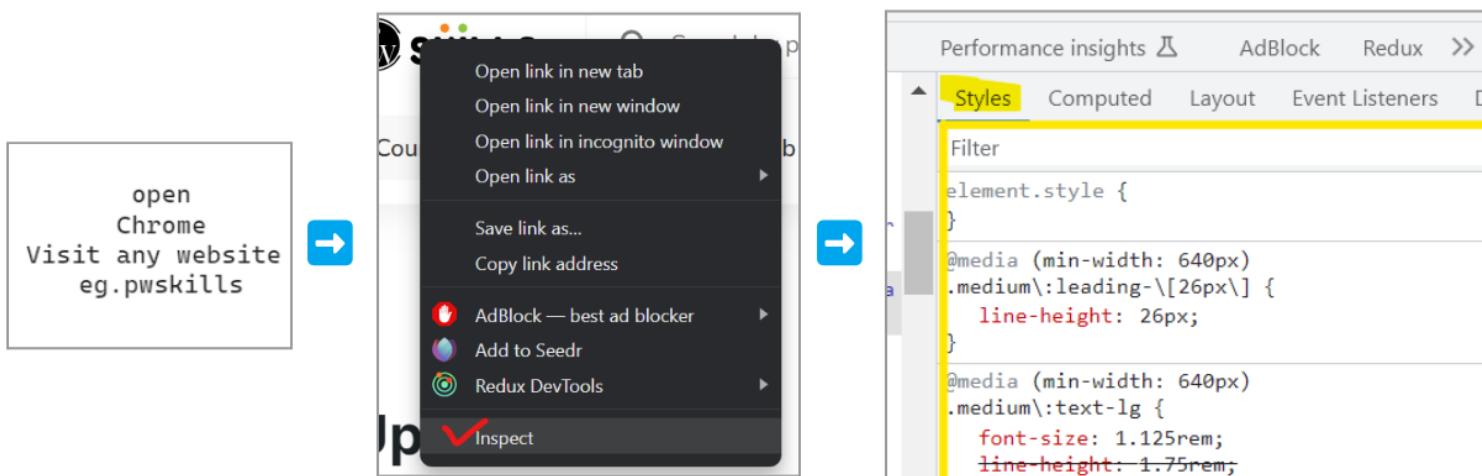
Introduction to the Chrome dev tool for CSS

The Chrome DevTools CSS panel is a powerful tool that allows developers to inspect and modify the CSS of a web page. It can be used to -

- View the CSS properties of any element on the page
- Find and fix CSS errors
- Override CSS rules
- Experiment with different CSS styles
- Preview CSS changes before they applied to the page

To open the Chrome dev tools CSS section

Go to ChromeBrowser>enter any browserPage>right click>select inspect>styles. i.e.

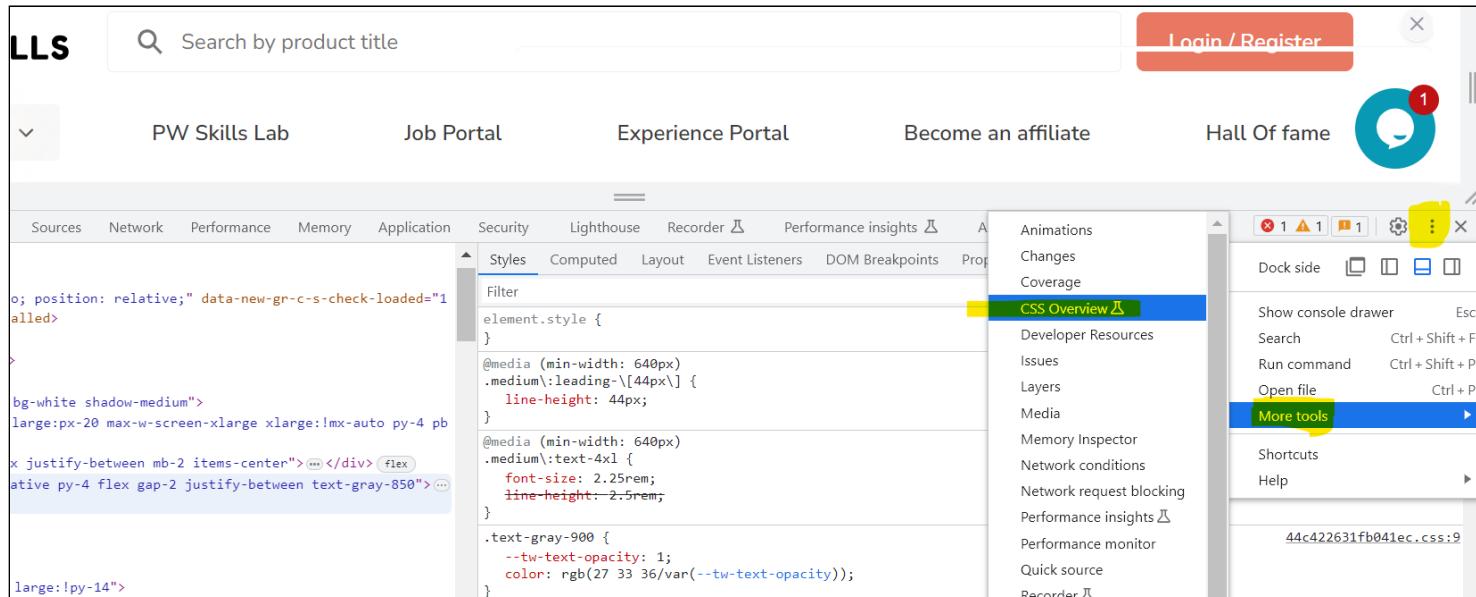


Open the CSS overview panel

The “CSS overview” tool in Chrome Devtools provides a high-level summary of the CSS on a webpage, helping developers to identify potential performance issues.

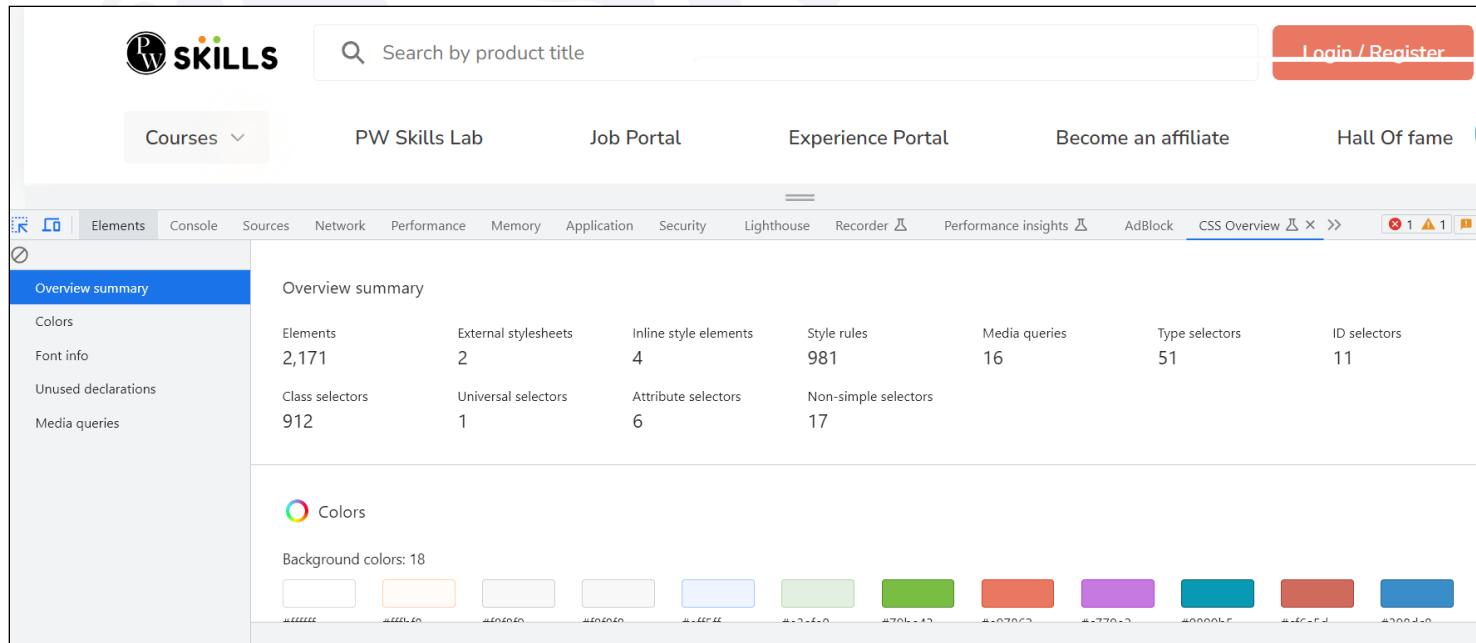
Inspect>three do(right top corner of the chrome dev tool)>more tools>CSS Overview.

i.e.



The screenshot shows the PW Skills Lab website with the Chrome DevTools sidebar open. The sidebar has a "More tools" dropdown menu where "CSS Overview" is highlighted. The main content area displays some CSS code snippets, and the bottom right corner of the sidebar shows a yellow notification badge with the number "1".

This is the CSS overview panel



The screenshot shows the PW Skills website with the Chrome DevTools sidebar open. The sidebar has a "More tools" dropdown menu where "CSS Overview" is highlighted. The main content area displays a "Overview summary" table with various CSS statistics. Below the table, there is a "Colors" section showing a color palette with 18 different colors and their corresponding hex codes.

	Elements	External stylesheets	Inline style elements	Style rules	Media queries	Type selectors	ID selectors
Colors	2,171	2	4	981	16	51	11
Font info							
Unused declarations							
Media queries	912	1	6	17			

Inspecting and selecting the HTML elements

The HTML element can be inspected by right-clicking on the particular HTML element and selecting the inspect option on the web pages.

For example -

We can change the pwskills Hero section text i.e “upscale made” to any other text

Go to pwskills>select the element to e change>right click and select inspect

Result of what we got after entering the inspection mood



```

Elements      Console      Sources      Network      Performance      Memory      Application      Security
▼ <header class="flex flex-col items-center justify-center w-full large:!w-[58%] text-ce
  flex
  ▼ <h1 class="text-2xl medium:text-4xl leading-9 medium:leading-[44px] font-bold flex f
    ow justify-center large:!justify-start flex-wrap text-gray-900 large:!self-start for
    == $0
    "Upscaling Made &nbsp;"
```

► <div class="flex large:!inline-flex text-orange-500 flex-wrap break-all justify-ce
 rt py-2 regular:!pt-0">...</div> flex
 </h1>

Now double clicking of the h1 value from the above picture. The yellow underline word can be replaced.

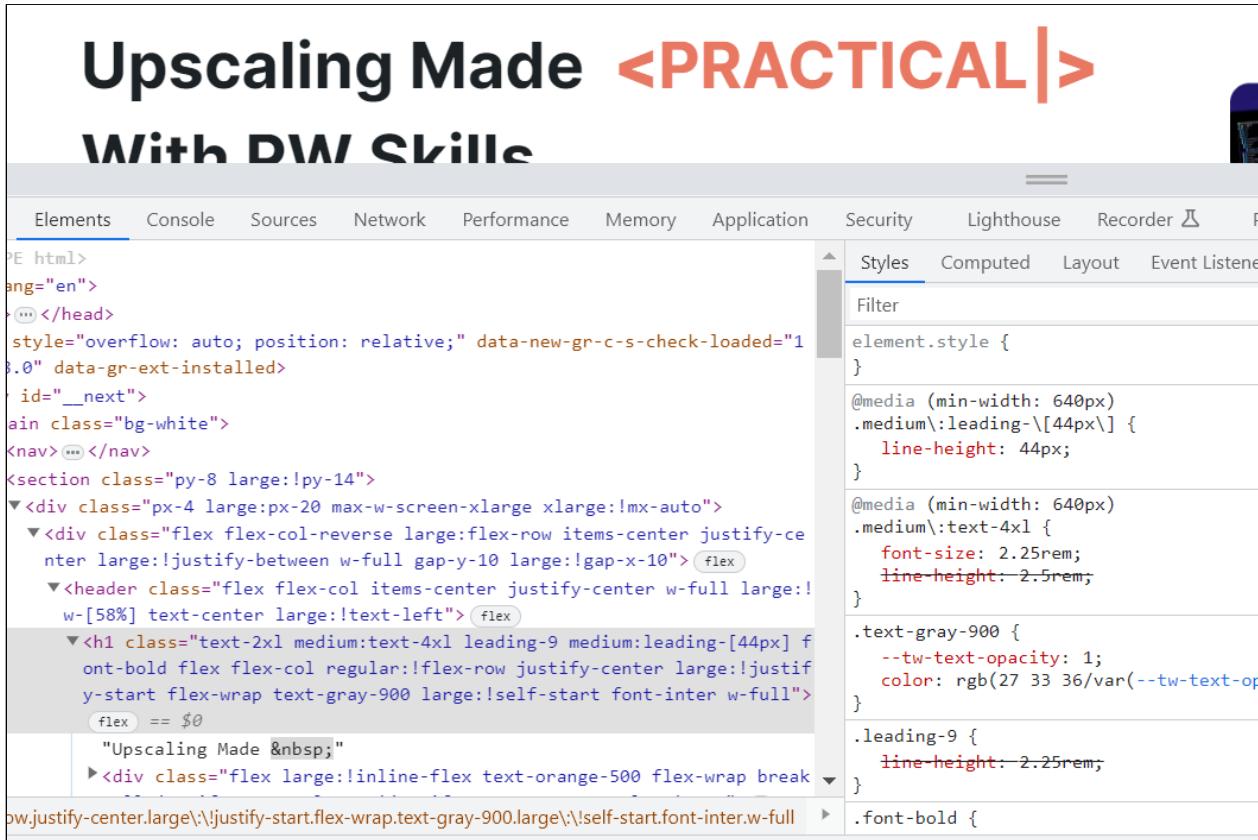
Applying temporary CSS changes for testing

Now let's see how can we change the colour of the above example temporarily for testing. It will not be saved permanently.

After selecting the element that is the h1 HTML element tag, the colour of the text can be changed in the style panel for CSS.

i.e

Before -



Upscaling Made <PRACTICAL|>
With PW Skills

Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder P

```
PE html>
  lang="en">
    </head>
    style="overflow: auto; position: relative;" data-new-gr-c-s-check-loaded="1.0" data-gr-ext-installed>
      id="__next">
        main class="bg-white">
          <nav></nav>
          <section class="py-8 large:!py-14">
            <div class="px-4 large:px-20 max-w-screen-xlarge xl:mx-auto">
              <div class="flex flex-col-reverse large:flex-row items-center justify-center large:!justify-between w-full gap-y-10 large:!gap-x-10">
                <header class="flex flex-col items-center justify-center w-full large:w-[58%] text-center large:!text-left">
                  <h1 class="text-2xl medium:text-4xl leading-9 medium:leading-[44px] font-bold flex flex-col regular:!flex-row justify-center large:!justify-start flex-wrap text-gray-900 large:!self-start font-inter w-full">
                    flex = $0
                    "Upscaling Made &nbsp;"
                  <div class="flex large:!inline-flex text-orange-500 flex-wrap break-all justify-center large:!justify-start flex-wrap.text-gray-900.large\!self-start.font-inter.w-full">
```

Styles Computed Layout Event Listener
Filter

```
element.style {  
}  

@media (min-width: 640px) .medium\:leading-[44px] {  
  line-height: 44px;  
}  

@media (min-width: 640px) .medium\:text-4xl {  
  font-size: 2.25rem;  
  line-height: 2.5rem;  
}  

.text-gray-900 {  
  --tw-text-opacity: 1;  
  color: #2e3436; --tw-text-opacity;  
}  

.leading-9 {  
  line-height: 2.25rem;  
}  

.font-bold {  
  font-weight: 700;
```

After - temporarily changing the CSS style (color to red)



Upscaling Made <PRACTICAL|>
With PW Skills

Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder

```
header class="flex flex-col items-center justify-center w-full large:w-[58%] text-center large:!text-left">
  <h1 class="text-2xl medium:text-4xl leading-9 medium:leading-[44px] font-bold flex flex-col regular:!flex-row justify-center large:!justify-start flex-wrap text-gray-900 large:!self-start font-inter w-full">
    flex = $0
    "Upscaling Made &nbsp;"
  <div class="flex large:!inline-flex text-orange-500 flex-wrap break-all justify-center large:!justify-start py-2 regular:!pt-0">...</div>
```

Styles Computed Layout Event Listener
Filter

```
@media (min-width: 640px) .medium\:text-4xl {  
  font-size: 2.25rem;  
  line-height: 2.5rem;  
}  

.text-gray-900 {  
  --tw-text-opacity: 1;  
  color: red; --tw-text-opacity;  
}  

.leading-9 {  
  line-height: 2.25rem;  
}  

.font-bold {  
  font-weight: 700;
```

Using box model tools for layout and positioning

The box model is a CSS concept that describes the size and position of an HTML element. It consists of four parts:

- Content: This is the actual content of the element, such as text, images, or other media.
- Padding: This is the space between the content and the border of the element.
- Border: This is the line that surrounds the element.
- Margin: This is the space between the border of the element and other elements on the page.

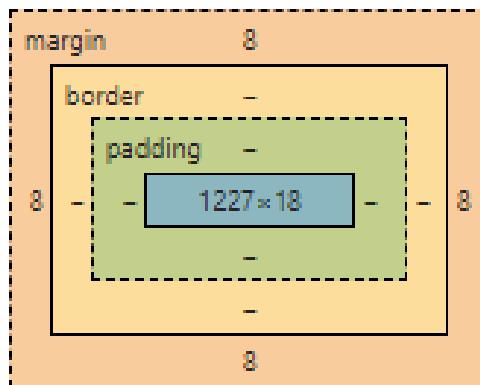


Fig. Box model figure

The box model can be used to control the layout and positioning of HTML elements in a number of ways. For example, you can use the padding property to add space around an element, the border property to create a border around an element, and the margin property to space an element away from other elements by double-clicking on the respective property.

Box model can be accessed by opening the Chrome dev tools, selecting the element panel and then clicking the style panel and scrolling down through the bottom the Box model will be available.

i.e.

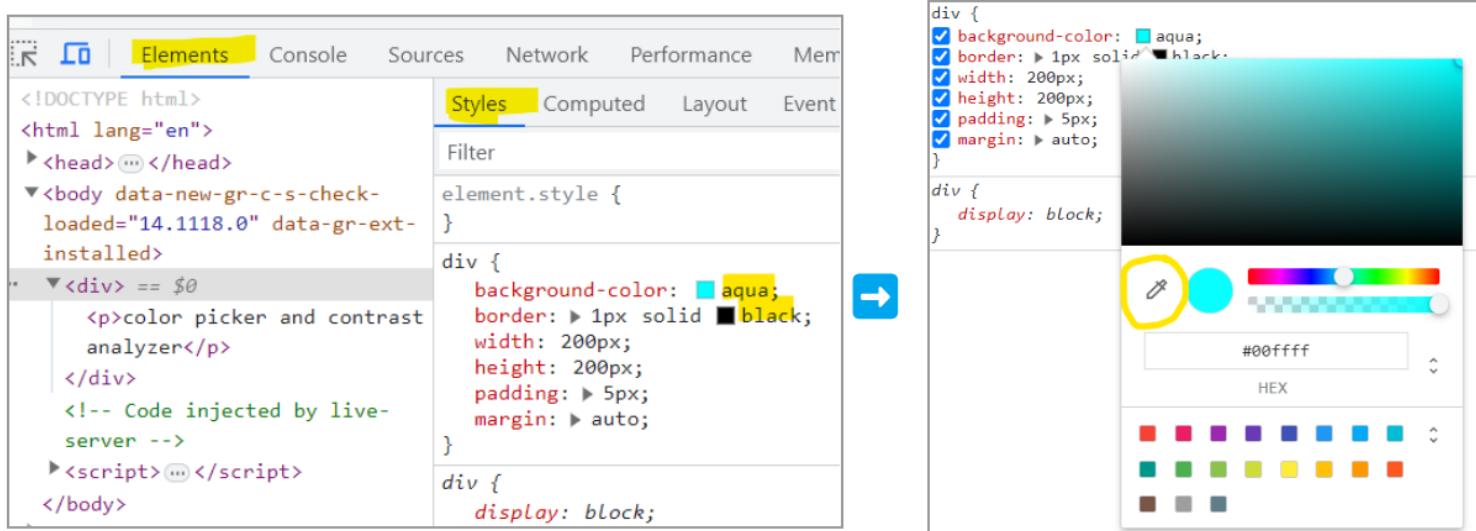
Using the Color Picker and Color contrast analyzer

Color Picker -

A color picker is a tool that allows you to select specific colors from a visual interface, typically using a graphical representation like a color wheel or a spectrum. Designers and developers use color pickers to choose colors for various elements in their projects, such as website backgrounds, text, buttons, and more. Color pickers often provide different formats for color representation, including hexadecimal (e.g., #RRGGBB), RGB (red, green, blue), and HSL (hue, saturation, lightness).

Color Picker can be accessed from the Chrome dev tools inside the style panel. i.e.

Open Chrome dev tools > select the element > navigate to the styles panel > double click on the color box prefix to any color name than the color panel will be shown along with the color Picker.



The screenshot shows the Chrome Dev Tools interface. On the left, the Elements panel is active, displaying the DOM structure. In the middle, the Styles panel is active, showing the CSS rules for the selected element. A color box in the 'background-color' rule is highlighted with a yellow circle. An arrow points from this color box to a larger color picker interface on the right. The color picker interface displays the CSS rule with the color set to 'aqua'. It includes a color gradient, a hex color input field ('#00ffff'), and a color palette below it. A yellow circle highlights the pen-like icon in the color picker interface.

The pen-like icon circle with yellow colour is the colour picker.

Example of Color CSS picker (to change the background color of the div container to different color i.e. aqua blue to other different colors. Using color picker inside the Chrome dev tool)

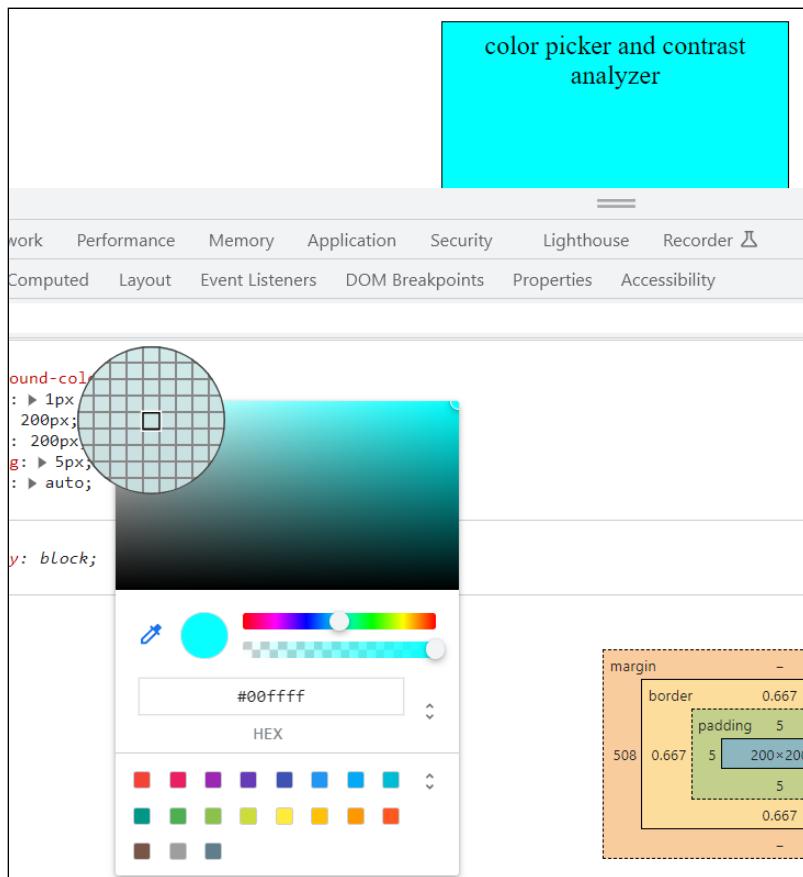
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Document</title>
    <style>
      div {
        background-color: aqua;
        border: 1px solid black;
        width: 200px;
        height: 200px;
        padding: 5px;
        margin: auto;
      }
    </style>
  </head>
  <body>
    <p>color picker and contrast analyzer</p>
  </body>

```

```


This is how it works after selecting color picker any color the can be copied and used to change the background color temporary.





The screenshot shows the Chrome DevTools interface with the "Elements" tab selected. A modal window titled "color picker and contrast analyzer" is open, displaying a gradient color picker and a color palette. The color #00ffff is selected and highlighted in the palette. On the left, the element's CSS properties are listed, including border, padding, and margin values. A detailed box model diagram on the right shows the element's dimensions and the cumulative effects of its borders, padding, and margins.



### Color contrast -



A color contrast analyzer is a tool used to evaluate the contrast between two colors, typically a foreground color (such as text) and a background color. Contrast is essential for ensuring the readability and accessibility of content, especially for individuals with visual impairments. The Web Content Accessibility Guidelines (WCAG) set forth guidelines for minimum contrast ratios between text and background colors.

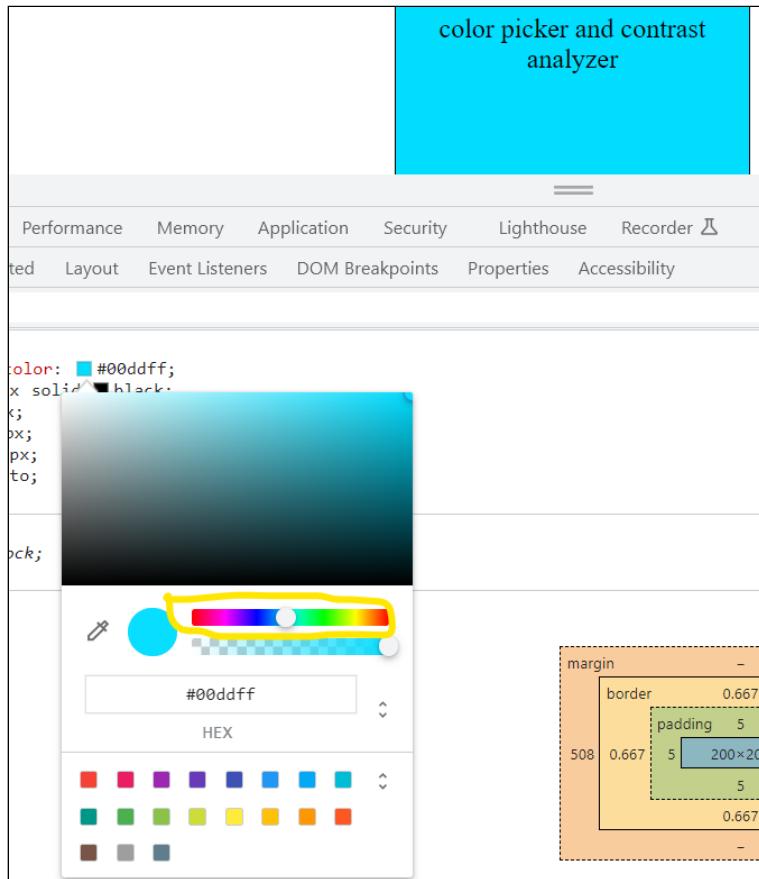


Full Stack Web Development


```

The Color contrast can be accessed the same way as the color picker.

Example - change the background color of the above example



The yellow line circle, can be scrolled horizontally to change the color contrast, resulting in a change in background color temporary.

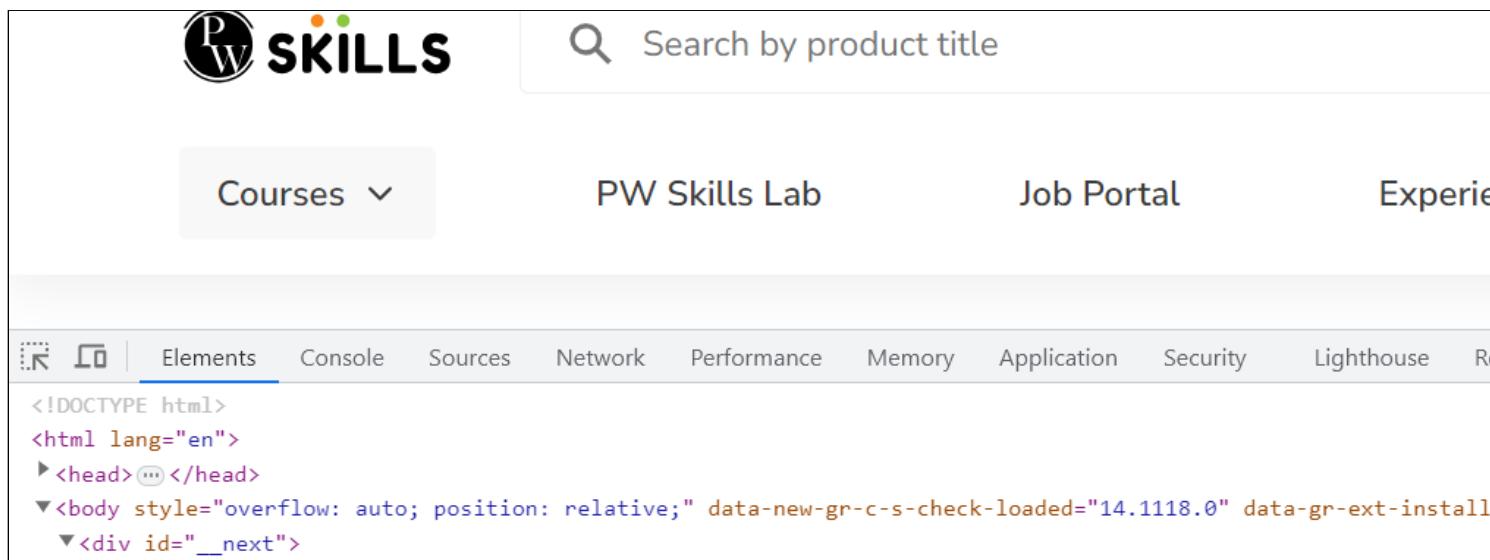
Working with media Query and responsive designs

Working with media queries and testing responsive designs in Chrome DevTools is a fundamental part of web development. Media queries allow you to apply specific CSS styles based on the characteristics of the user's device or screen, enabling you to create responsive designs that adapt to various screen sizes and orientations.

Here's how one can work with media queries and test responsive designs using Chrome DevTools. We will use the pwskills.com website to demonstrate it.

Open Chrome Devtools

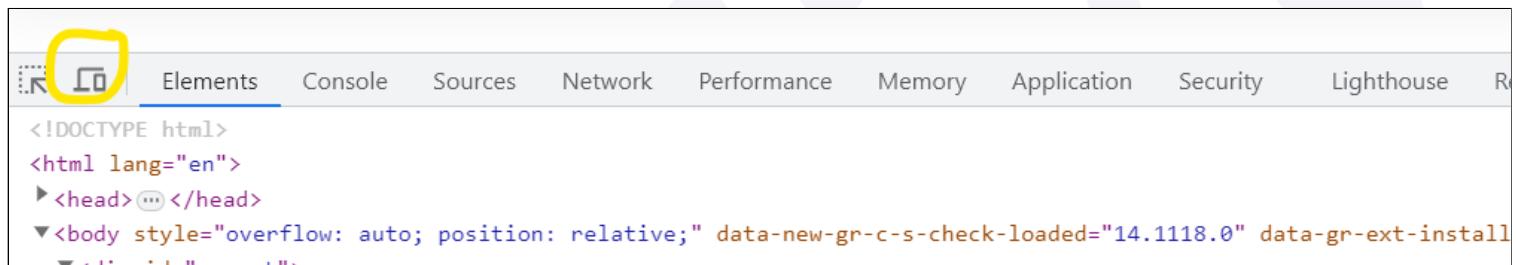
Visit the pwskills.com website, right-click and select the inspect option then chrome dev tool will open. i.e



The screenshot shows the PW Skills website with the DevTools Elements tab selected. The toggle device toolbar icon (a yellow circle with a white icon) is highlighted with a yellow circle. The DevTools interface shows the HTML structure of the page, including the head and body sections.

Toggle Device Toolbar

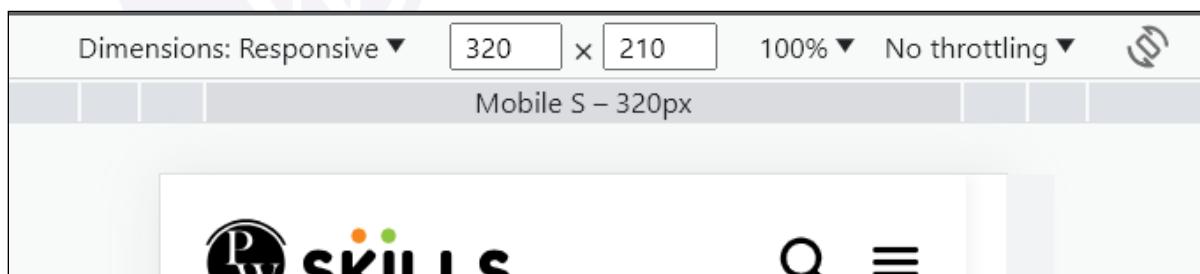
The Device Toolbar in Chrome DevTools allows you to simulate different devices and screen sizes. To toggle the Device Toolbar:



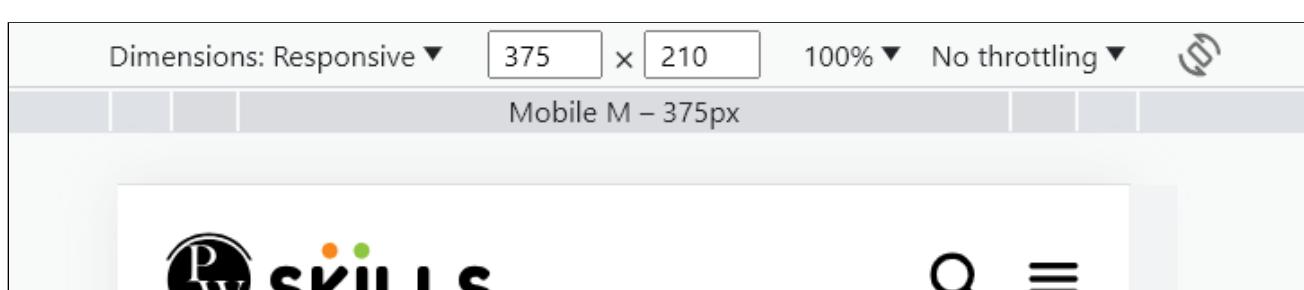
The screenshot shows the PW Skills website with the DevTools Elements tab selected. The toggle device toolbar icon (a yellow circle with a white icon) is highlighted with a yellow circle. The DevTools interface shows the HTML structure of the page, including the head and body sections.

Select or click the yellow circle icon to activate the toggle device toolbar

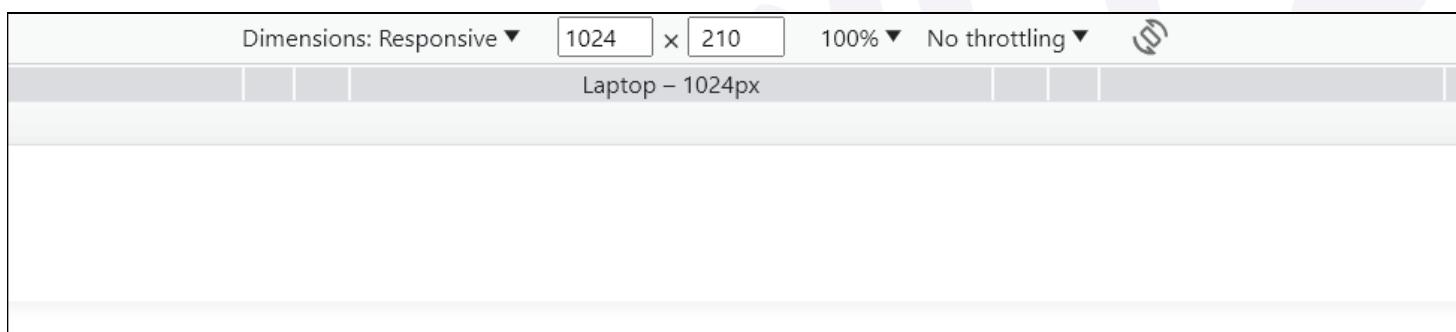
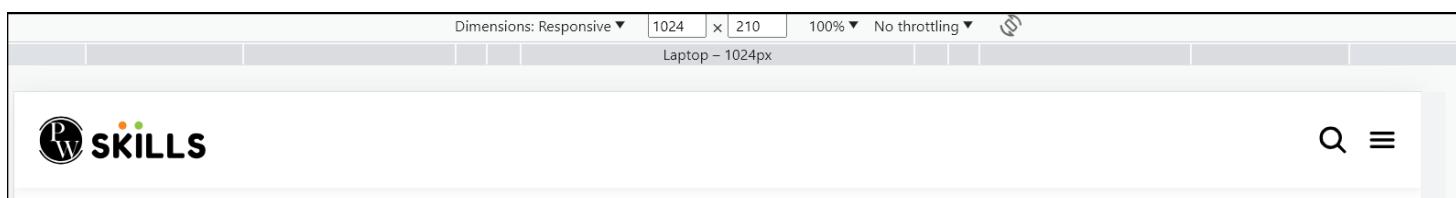
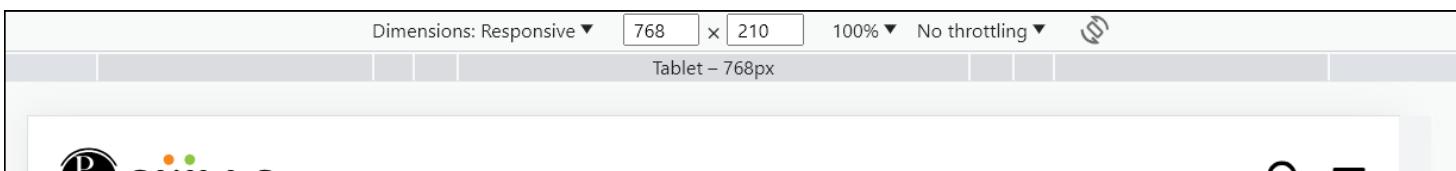
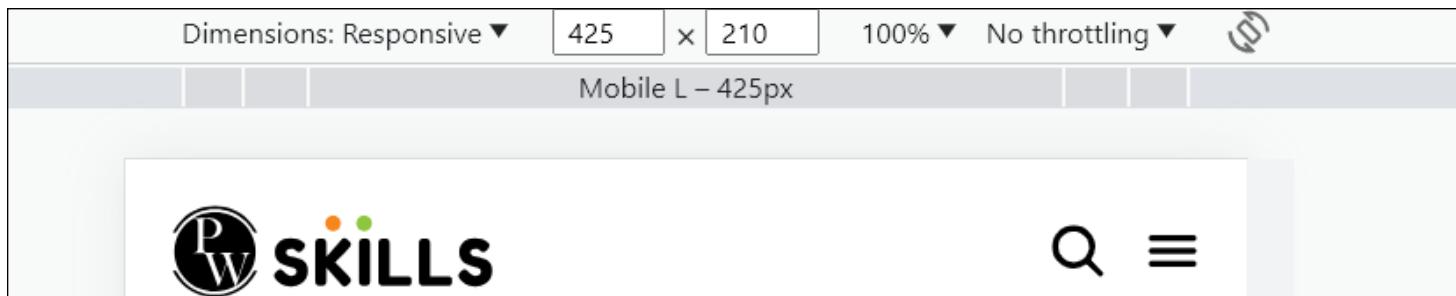
The different types of screen sizes available are -



The screenshot shows the PW Skills website in mobile S responsive mode. The DevTools interface at the top shows dimensions of 320 x 210 pixels, 100% zoom, and no throttling. The website content below is labeled "Mobile S – 320px".



The screenshot shows the PW Skills website in mobile M responsive mode. The DevTools interface at the top shows dimensions of 375 x 210 pixels, 100% zoom, and no throttling. The website content below is labeled "Mobile M – 375px".



Choose a Device or Set Custom Dimensions

The device can be set to Custom Dimension just above the predefine device size.

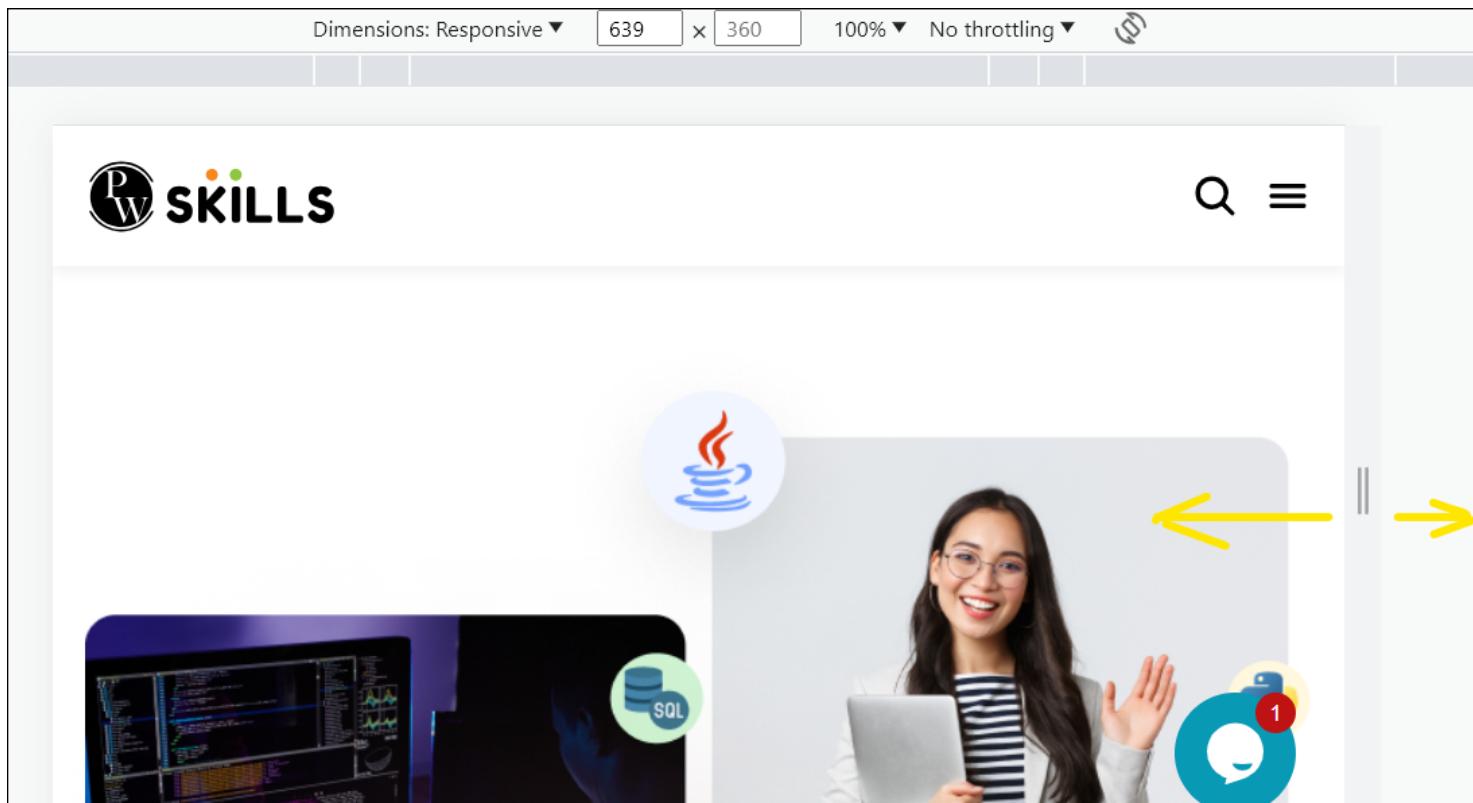
Or by dragging horizontally the width size of the page.

Set Custom dimension



In here the different types of Dimension value can be added.

Drag the screen size horizontally -



Testing the CSS flexbox and grid layout

Testing CSS Flexbox and Grid layouts in Chrome DevTools is an excellent way to fine-tune your designs and understand how your layout is behaving on different screen sizes. Both Flexbox and Grid are powerful layout techniques that enable you to create flexible and responsive designs.

Testing CSS Flexbox

index.html

```
<body>
  <div class="container">
    <div class="box">1</div>
    <div class="box">2</div>
    <div class="box">3</div></div>
</body>
```

style.css

```

body {
    margin: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    min-height: 100vh;
    background-color: #f0f0f0;
}

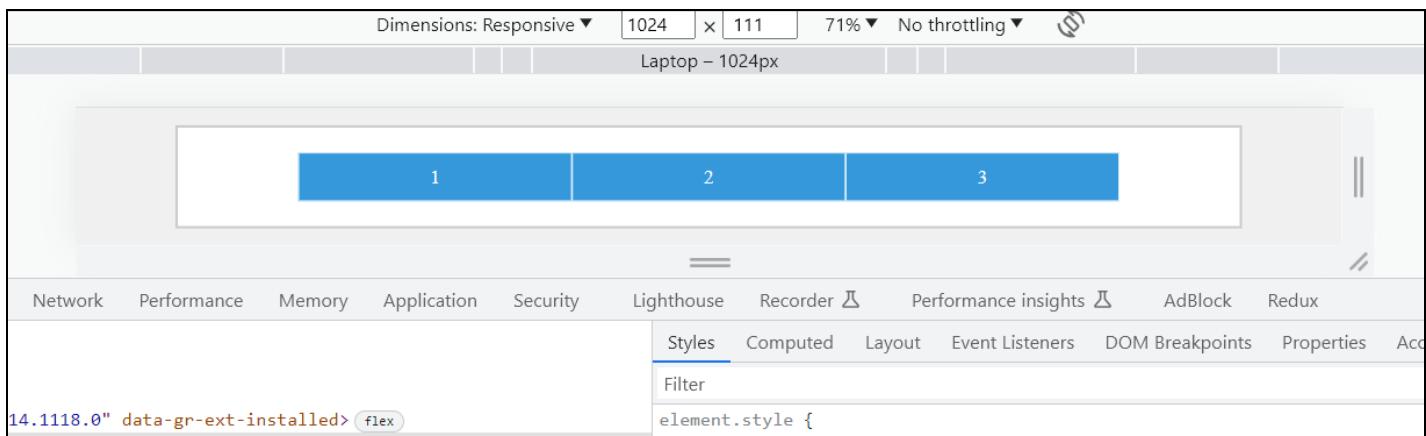
.container {
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    width: 80%;
    background-color: #fff;
    padding: 20px;
    border: 2px solid #ccc;
}

.box {
    width: 200px;
    background-color: #3498db;
    color: #fff;
    text-align: center;
    border: 1px solid;
    padding: 10px; }

```

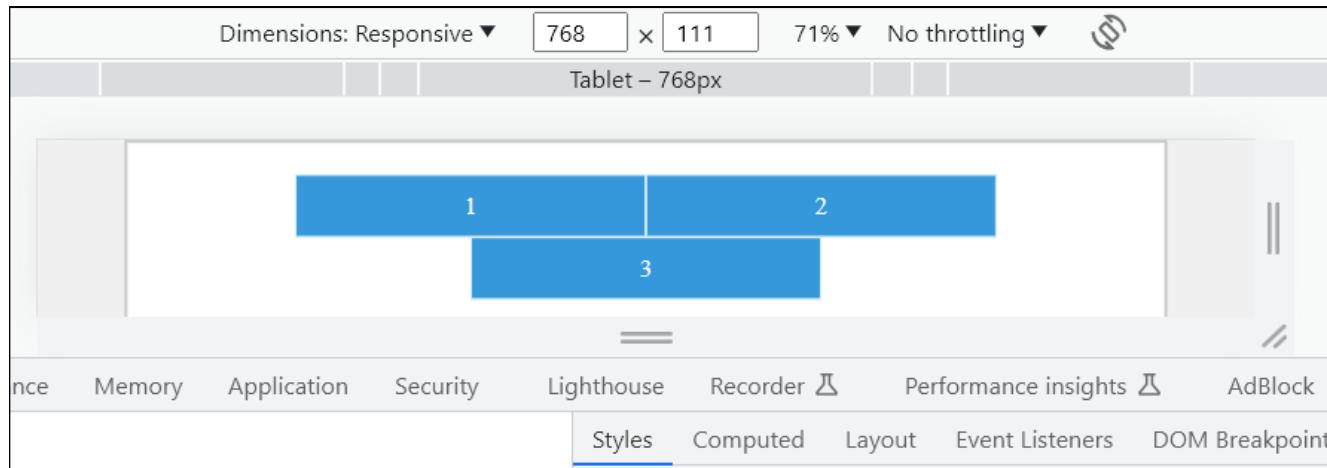
Browser output (Testing in flexbox in chrome dev tool) -

Screen size - laptop - 1024px, Here the flex items are in a single row

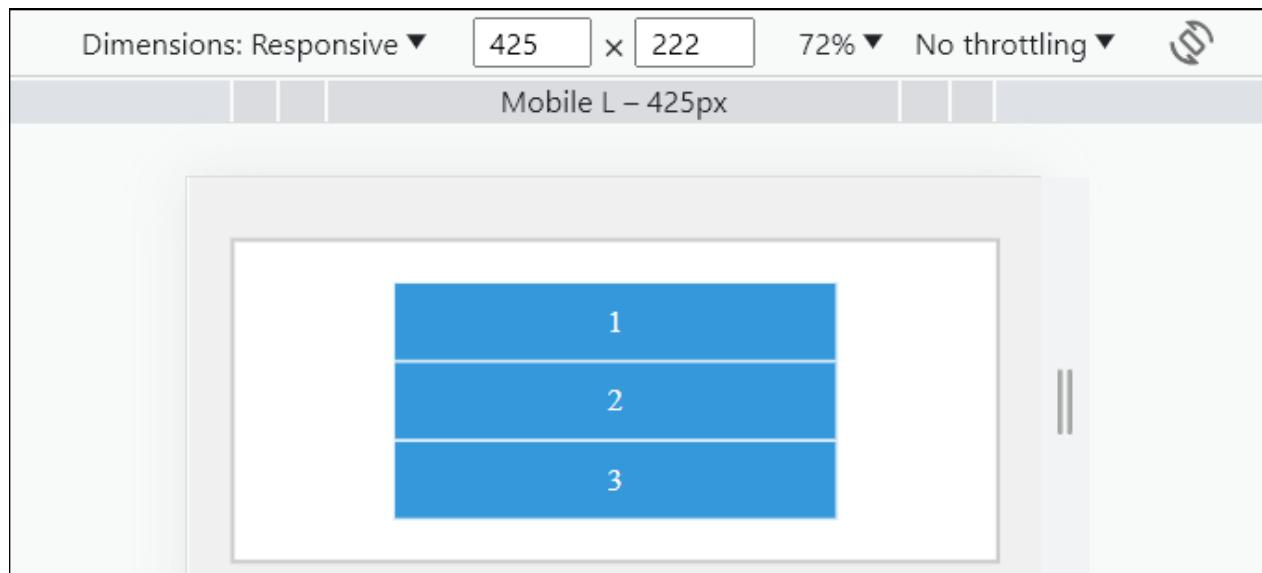


The screenshot shows the Chrome DevTools interface with the "Styles" tab selected in the bottom navigation bar. The main view displays a flex container with three items labeled 1, 2, and 3, arranged horizontally. The browser dimensions are set to "Responsive" with a width of 1024px and a height of 111px. The "Network", "Performance", "Memory", "Application", "Security", "Lighthouse", "Recorder", "Performance insights", "AdBlock", and "Redux" tabs are also visible at the top.

Tablet-size -768px, Here are the flex items in a two-row which stack on each other since the screen does not fit it in one row.



Mobile - 425px, Here the flex items are in one single column



The above example shows the responsiveness of the HTML elements using CSS flexbox.

Testing CSS Grid layout

index.html

```
<body>
  <div class="grid-container">
    <div class="item">1</div>
    <div class="item">2</div>
    <div class="item">3</div>
    <div class="item">4</div>
  </div>
</body>
```

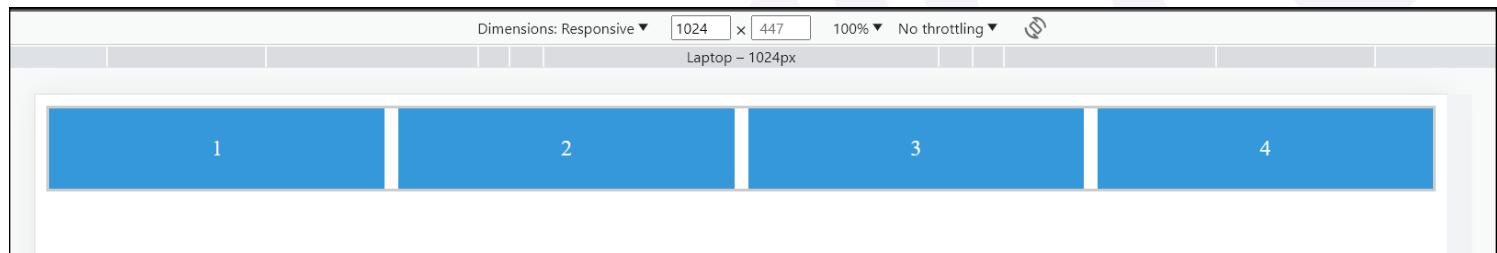
style.css

```
.grid-container {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: 10px;
    background-color: #fff;
    border: 2px solid #ccc;
}

.item {
    background-color: #3498db;
    color: #fff;
    text-align: center;
    padding: 20px;
}
```

Browser output (Testing in Grid layout in chrome dev tool) -

Screen size - laptop - 1024px, Here the Grid items are in a single row



Screen size - tablet-768px, Here the Grid items are in a two-row which stack on each other since the screen does not fit it in on row.



Mobile - 425px, Here the flex items are in one single column



The above example shows the responsiveness of the HTML elements using CSS Grid layout.