06

Responsive Designing

WHAT YOU WILL LEARN

* What is responsive Designing?
* What is The Viewport?
* What is a Media Query and Breakpoints?
* CSS Grid Layout
* CSS Flexbox Layout Module

What is responsive Designing?

* Responsive web design makes your web page look good on all devices.
* Responsive web design uses only HTML and CSS.
* Web pages can be viewed using many different devices: desktops, tablets, and phones. Your web page should look good and be easy to use.
* when you use CSS and HTML to resize, hide, shrink, enlarge, or move the content to make it look good on any screen.

Example –

Chart, bar chart

Description automatically generated

What is Viewport?

* The viewport is the user's visible area of a web page.
* HTML5 introduced a method to let web designers take control over the viewport, through the <meta> tag.
* <meta name="viewport" content="width=device-width, initial-scale=1.0">
* This gives the browser instructions on how to control the page's dimensions and scaling.
* The width=device-width part sets the width of the page to follow the screen-width of the device

Viewport Rules:

* Do NOT use large, fixed width elements.
* Do NOT let the content rely on a particular viewport width to render well
* Use CSS media queries to apply different styling for small and large screens

What is a Media Query?

* Media query is a CSS technique introduced in CSS3.
* It uses the @media rule to include a block of CSS properties only if a certain condition is true.
* @media only screen and (max-width: 600px) {  
    body {  
      background-color: lightblue;  
    }  
  }

Add a Breakpoint

* Always Design for Mobile First - Mobile First means designing for mobile before designing for desktop or any other device (This will make the page display faster on smaller devices).
* /\* Extra small devices (phones, 600px and down) \*/  
  @media only screen and (max-width: 600px) {...}  
    
  /\* Small devices (portrait tablets and large phones, 600px and up) \*/  
  @media only screen and (min-width: 600px) {...}  
    
  /\* Medium devices (landscape tablets, 768px and up) \*/  
  @media only screen and (min-width: 768px) {...}  
    
  /\* Large devices (laptops/desktops, 992px and up) \*/  
  @media only screen and (min-width: 992px) {...}  
    
  /\* Extra large devices (large laptops and desktops, 1200px and up) \*/  
  @media only screen and (min-width: 1200px) {...}

CSS Grid Layout

* The CSS Grid Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning.
* Display Property
  + An HTML element becomes a grid container when its display property is set to grid or inline-grid.
  + display: grid;
* Grid Row
* Grid Column
  + grid-template-columns : The grid-template-columns property defines the number of columns in your grid layout, and it can define the width of each column.
  + Example - .grid-container {  
      display: grid;  
      grid-template-columns: 80px 200px auto 40px;  
    }
* Grid Gaps
  + column-gap
  + row-gap
  + gap
* Grid Lines
  + The lines between columns are called column lines.
  + The lines between rows are called row lines.
* Grid Container –
  + To make an HTML element behave as a grid container, you have to set the display property to grid or inline-grid.
  + Grid containers consist of grid items, placed inside columns and rows.
* Justify-content - The justify-content property is used to align the whole grid inside the container.
  + .grid-container {  
      display: grid;  
      justify-content: space-evenly;  
    }
  + .grid-container {  
      display: grid;  
      justify-content: space-around;  
    }
  + .grid-container {  
      display: grid;  
      justify-content: space-between;  
    }
  + .grid-container {  
      display: grid;  
      justify-content: start;  
    }
  + .grid-container {  
      display: grid;  
      justify-content: end;  
    }
* align-content - The align-content property is used to vertically align the whole grid inside the container.
  + .grid-container {  
      display: grid;  
      height: 400px;  
      align-content: center;  
    }
  + .grid-container {  
      display: grid;  
      height: 400px;  
      align-content: space-evenly;  
    }
  + .grid-container {  
      display: grid;  
      height: 400px;  
      align-content: space-around;  
    }
  + .grid-container {  
      display: grid;  
      height: 400px;  
      align-content: space-between;  
    }
  + .grid-container {  
      display: grid;  
      height: 400px;  
      align-content: start;  
    }
  + .grid-container {  
      display: grid;  
      height: 400px;  
      align-content: end;  
    }