1.Bootstrap Grid System

Bootstrap Grid System is a collection of reusable code snippets to create responsive layouts. It is made up of containers, rows, and columns.

It uses a 12 column system for layouting. We can create up to 12 columns across the page.

1.1 Container

The purpose of a container is to hold rows and columns.

```
<div class="container"></div>
```

Here, the container is a div element with the Bootstrap class name container .

1.2 Row

The purpose of a row is to wrap all the columns.

```
<div class="container">
  <div class="row"></div>
</div>
```

Here, the row is a div element with the Bootstrap class name row.

1.3 Column

We should place the columns inside a row and the content inside a column.

We can specify the number of columns our content should occupy in any device. The number of columns we specify should be a number in the range of 1 to 12.

```
<div class="container">
  <div class="row">
    <div class="col-12">
        I'm your content inside the grid!
    </div>
   </div>
 </div>
```

Here, the column is a div element with the Bootstrap class name col-12.

KeyNote:

If Bootstrap class name is **col-12**, it occupies the entire width available inside the row.

The Bootstrap class names **col-*** indicates the number of columns you would like to use out of the possible **12 columns** per row. For **example**, col-1, col-5, etc.

2. Creating Multiple Column Layouts

The Layout in the below Code Playground is a **Two Column Layout**.

```
<div class="row">
  <div class="col-8">
   <h1 class="heading">SAI</h1>
    </div>
  <div class="col-4">
   <h1 class="heading">GANESH</h1>
    </div>
</div>
```

1. CSS Box Properties

1.1 Margin

We can get spacing between the two HTML elements with the CSS Box property margin.

To get space only on one particular side, we use **Margin Variants.**

- **margin-top**
- **margin-right**
- **margin-bottom**
- **margin-left**

HTML code:

```
<body>
   <div class="space orange-box"></div>
   <div class="space green-box"></div>
 </body>
```

CSS Code:

```
  . space {
     padding: 30px;
    margin-bottom: 10px;
  }
```

OUTPUT:

2. Bootstrap Spacing Utilities

2.1 Margin

CSS Margin property	Bootstrap class name
margin	m-*
margin-top	mt-*
margin-right	mr-*
margin-bottom	mb-*
margin-left	ml-*

The asterisk (*) symbol can be any number in the range of 0 to 5. For example, m-5, mr-2, mb-3, etc.

2.1.1 Margin Values

Size	Value
0	0
1	0.25 * spacer
2	0.5 * spacer
3	1 * spacer
4	1.5 * spacer
5	3 * spacer

The spacer is a variable and has a value of 16 pixels by default.

For example,

mb-3 = 1 * 16px = 16px

m-5 = 3 * 16px = 48px

WARNING: Avoid using CSS margin-left and margin-right properties for Bootstrap Grid Columns. It disturbs the Bootstrap Grid System and gives unexpected results.

2.2 Padding

CSS Padding property	Bootstrap class name
padding	p-*
padding-top	pt-*
padding-right	pr-*
padding-bottom	pb-*
padding-left	pl-*

The asterisk (*) symbol can be any number in the range of 0 to 5. For example, p-3, pr-1, pb-5, etc.

2.2.1 Padding Values

Size	Value
0	0
1	0.25 * spacer
2	0.5 * spacer
3	1* spacer
4	1.5 * spacer
5	3* spacer

The spacer is a variable and has a value of 16 pixels by default.

For **example**,

p-1 = 0.25 * 16px = 4px

pt-4 = 1.5 * 16px = 24px

HTML CODE:

```
<div class="bg-primary p-2">  // 'p' indicates padding in BootStrap Grid System
    <p>Primary</p>
  </div>
```

Developing Layouts for five Responsive Breakpoints:

Bootstrap's grid system is a powerful tool for creating responsive layouts that adapt seamlessly to different screen sizes. By understanding the core concepts and utilizing the provided breakpoints, you can design layouts that look great on a wide range of devices.

Understanding Breakpoints

Bootstrap offers five default breakpoints, each with its own media query range:

1. Extra small (xs): Devices with screen widths less than 576px
2. Small (sm): Devices with screen widths equal to or greater than 576px
3. Medium (md): Devices with screen widths equal to or greater than 768px
4. Large (lg): Devices with screen widths equal to or greater than 992px
5. Extra large (xl): Devices with screen widths equal to or greater than 1200px

HTML CODE:

```
<div class="container">
    <div class="row">
     <div class="col-12 col-md-6 col-lg-4 col-xl-3 mb-3">
      <div class="color-box bg-primary">
       <p class="color-name">Primary</p>
      </div>
  </div>
```

3. Bootstrap Components

3.1 Navbar

A Navbar is a navigation header that is placed at the top of the page. With Bootstrap, a Navbar can extend or collapse, depending on the device size.

3.1.1 HTML Nav element

The HTML nav element is a container element similar to the HTML div element. We use the HTML nav element to add a Navbar to our website.

1.Visit : https://getbootstrap.com/docs/5.0/getting-started/introduction/

2.Explore the Documentation:

On the Bootstrap website, navigate to the "Docs" section and choose "Components." Here, you'll find detailed explanations and code examples for various Bootstrap components, including the navigation bar.

3.Search for Navigation Bar Tutorial:

Within the Bootstrap documentation or using a search engine, look for tutorials specifically focused on creating a responsive navigation bar with Bootstrap.

GITHUB REPOSITORY : https://github.com/saiganesh1798/BootStrap-Navigation-Bar

3.1.2 HTML Elements

HTML elements can be thought of as building blocks for your web pages. They come in two main types:

1. Block-level Elements

Imagine these elements as tall, sturdy buildings that stand alone. They take up the entire width of their parent container and always start on a new line.

Here are some common block-level elements:

- : A general-purpose container that can hold other elements.
- <h1> to <h6>: Headings of different sizes.
- <p>: Paragraphs of text.
- and <ol>: Unordered and ordered lists.
- li>: List items.
- hr>: Horizontal rule.

EXAMPLE:

```
<h1>This is a Heading</h1>
<p>This is a paragraph.</p>
<li>Item 1</li>
```

2. Inline Elements

Unlike block-level elements, inline elements only take up as much width as necessary. They can be placed side-by-side within a line.

Here are some common inline elements:

- : A generic inline container.
- : Bold text.
- : Emphasized text (italic).
- <a>: Hyperlink.
- img>: Image.

3.2 Margin

We can align HTML Block-level elements horizontally using CSS margin property.

Apart from values that are specified in pixels, it also accepts auto keyword as a value.

margin: auto;

- Using auto as a value for the CSS margin-right property takes up all the available space, and the element gets aligned to the left.

  margin-left: auto;

- Using auto as a value for the CSS margin-left property takes up all the available space, and the element gets aligned to the right**.**

  margin-right: auto;

  Apart from the numbers 0-5, the margin also has the below Bootstrap class names.

  m-auto

  ml-auto

  mr-auto