**UNIT-1**

**Cascading Style Sheet**

Cascading Style Sheets (CSS) is used to format the layout of a webpage.

With CSS, you can control the color, font, the size of text, the spacing between elements, how elements are positioned and laid out, what background images or background colors are to be used, different displays for different devices and screen sizes, and much more!

The word **cascading** means that a style applied to a parent element will also apply to all children elements within the parent. So, if you set the color of the body text to "blue", all headings, paragraphs, and other text elements within the body will also get the same color (unless you specify something else)!

Using CSS

CSS can be added to HTML documents in 3 ways:

* **Inline** - by using the style attribute inside HTML elements
* **Internal** - by using a <style> element in the <head> section
* **External** - by using a <link> element to link to an external CSS file

## Inline CSS

An inline CSS is used to apply a unique style to a single HTML element.

An inline CSS uses the style attribute of an HTML element.

The following example sets the text color of the <h1> element to blue, and the text color of the <p> element to red:

**<html>**

**<body>**

**<h1 style="color:blue;">A Blue Heading</h1>**

**<p style="color:red;">A red paragraph.</p>**

**</body>**

**</html>**

## Internal CSS

An internal CSS is used to define a style for a single HTML page.

An internal CSS is defined in the <head> section of an HTML page, within a <style> element.

The following example sets the text color of ALL the <h1> elements (on that page) to blue, and the text color of ALL the <p> elements to red. In addition, the page will be displayed with a "powderblue" background color:

<!DOCTYPE html>  
<html>  
<head>  
<style>  
body {background-color: powderblue;}  
h1   {color: blue;}  
p    {color: red;}  
</style>  
</head>  
<body>  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
</body>  
</html>

## External CSS

An external style sheet is used to define the style for many HTML pages.

To use an external style sheet, add a link to it in the <head> section of each HTML page:

<!DOCTYPE html>  
<html>  
<head>  
  <link rel="stylesheet" href="styles.css">  
</head>  
<body>  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
</body>  
</html>

The external style sheet can be written in any text editor. The file must not contain any HTML code, and must be saved with a .css extension.

Here is what the "styles.css" file looks like:

body {  
  background-color: powderblue;  
}  
h1 {  
  color: blue;  
}  
p {  
  color: red;  
}

With an external style sheet, you can change the look of an entire web site, by changing one file!

## CSS Colors, Fonts and Sizes

Here, we will demonstrate some commonly used CSS properties. You will learn more about them later.

The CSS color property defines the text color to be used.

The CSS font-family property defines the font to be used.

The CSS font-size property defines the text size to be used.

### Example

Use of CSS color, font-family and font-size properties:

## **<!DOCTYPE html> <html> <head> <style> h1 {   color: blue;   font-family: verdana;   font-size: 300%; } p {   color: red;   font-family: courier;   font-size: 160%; } </style> </head> <body> <h1>This is a heading</h1> <p>This is a paragraph.</p> </body> </html>**

## CSS Border

The CSS border property defines a border around an HTML element.

**Tip:** You can define a border for nearly all HTML elements.

### Example

Use of CSS border property:

## **p {   border: 2px solid powderblue; }**

## CSS Padding

The CSS padding property defines a padding (space) between the text and the border.

Use of CSS border and padding properties:

p {  
  border: 2px solid powderblue;  
  padding: 30px;  
}

## CSS Margin

The CSS margin property defines a margin (space) outside the border.

### Example

Use of CSS border and margin properties:

p {  
  border: 2px solid powderblue;  
  margin: 50px;  
}

## Link to External CSS

External style sheets can be referenced with a full URL or with a path relative to the current web page.

### Example

This example uses a full URL to link to a style sheet:

<link rel="stylesheet" href="https://www.w3schools.com/html/styles.css">

### Example

This example links to a style sheet located in the html folder on the current web site:

<link rel="stylesheet" href="/html/styles.css">

### Example

This example links to a style sheet located in the same folder as the current page:

<link rel="stylesheet" href="styles.css">

# **CSS Selectors**

CSS selectors are used to "find" (or select) the HTML elements you want to style.We can divide CSS selectors into five categories:

**Simple selectors** (select elements based on name, id, class)

* [Combinator selectors](https://www.w3schools.com/css/css_combinators.asp) (select elements based on a specific relationship between them)
* [Pseudo-class selectors](https://www.w3schools.com/css/css_pseudo_classes.asp) (select elements based on a certain state)
* [Pseudo-elements selectors](https://www.w3schools.com/css/css_pseudo_elements.asp) (select and style a part of an element)
* [Attribute selectors](https://www.w3schools.com/css/css_attribute_selectors.asp) (select elements based on an attribute or attribute value)

**The CSS element Selector**

The element selector selects HTML elements based on the element name.

<html>

<head>

<style>

p {

text-align: center;

color: red;

}

</style>

</head>

<body>

<p>Every paragraph will be affected by the style.</p>

<p >Me too!</p>

<p>And me!</p>

</body>

</html>

**Output**

Every paragraph will be affected by the style.

Me too!

And me!

**The CSS id Selector**

The id selector uses the id attribute of an HTML element to select a specific element.

The id of an element is unique within a page, so the id selector is used to select one unique element! To select an element with a specific id, write a hash (#) character, followed by the id of the element.

<html>

<head>

<style>

#para1 {

text-align: center;

color: red;

}

</style>

</head>

<body>

<p id="para1">Hello World!</p>

<p>This paragraph is not affected by the style.</p>

</body>

</html>

**Output**

Hello World!

This paragraph is not affected by the style.

**The CSS class Selector**

The class selector selects HTML elements with a specific class attribute.

To select elements with a specific class, write a period (.) character, followed by the class name.

### **Example: In this example all HTML elements with class="center" will be red and center-aligned:**

<html>

<head>

<style>

.center {

text-align: center;

color: red;

}

</style>

</head>

<body>

<h1 class="center">Red and center-aligned heading</h1>

<p class="center">Red and center-aligned paragraph.</p>

</body>

</html>

**Output**

# **Red and center-aligned heading**

Red and center-aligned paragraph.

You can also specify that only specific HTML elements should be affected by a class.

### **Example:** In this example only <p> elements with class="center" will be red and center-aligned:

<html>

<head>

<style>

p.center {

text-align: center;

color: red;

}

</style>

</head>

<body>

<h1 class="center">This heading will not be affected</h1>

<p class="center">This paragraph will be red and center-aligned.</p>

</body>

</html>

**Output**

# **This heading will not be affected**

This paragraph will be red and center-aligned.

# **CSS Pseudo-classes**

## **What are Pseudo-classes?**

A pseudo-class is used to define a special state of an element. For example, it can be used to:

* Style an element when a user mouses over it
* Style visited and unvisited links differently
* Style an element when it gets focus

**Syntax:** The syntax of pseudo-classes:

selector:pseudo-class {  
  property: value;  
}

**Anchor Pseudo-classes**

Links can be displayed in different ways:

<html>

<head>

<style>

/\* unvisited link \*/

a:link {

color: red;

}

/\* visited link \*/

a:visited {

color: green;

}

/\* mouse over link \*/

a:hover {

color: hotpink;

}

/\* selected link \*/

a:active {

color: blue;

}

</style>

</head>

<body>

<h2>Styling a link depending on state</h2>

<p><b><a href="default.asp" target="\_blank">This is a link</a></b></p>

<p><b>Note:</b> a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.</p>

<p><b>Note:</b> a:active MUST come after a:hover in the CSS definition in order to be effective.</p>

</body>

</html>

## **How to Use Pseudo-Classes in CSS**

Pseudo-classes are selector types that allow you to select elements in a particular state. To name a few, here are some supported states:

* hover (when the mouse floats over an element)
* disabled (when an element such as an input or button is disabled)
* required (when a form element is required)

You can apply styles when elements are in these states. You select the state by using a **colon** (**:**) followed by the state. Here is an example:

<!DOCTYPE html>

<button>

Hover me

</button>

And here's the CSS:

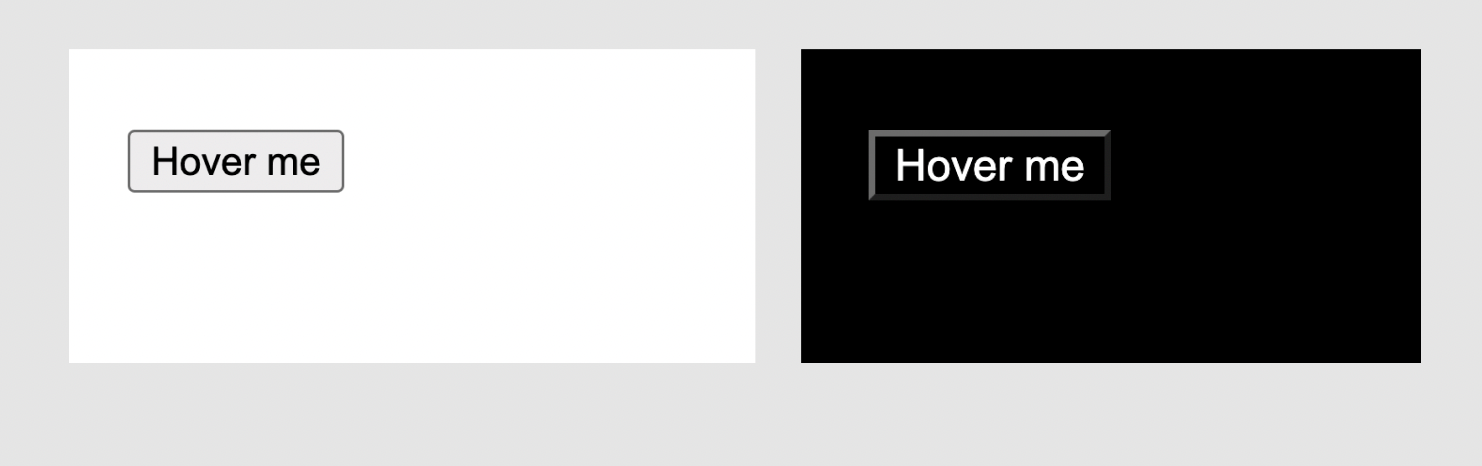
:hover {

background-color: black;

color: white;

}

This CSS would apply these styles to any element you hover over. Here's the result":



The image on the left is without the hover state. On the right, you can see the styles applied to the body and the button because we're hovering over them.

By hovering over the button, you are also hovering over the body because the button is a child of the body.

## **How to Use the Pseudo Element Selector in CSS**

Pseudo-elements (different from Pseudo Classes) are used to select a "specific part of an element". Not the whole element – just a part. And you can also use them to add pseudo (artificial) elements to an existing element.

Here are some supported pseudo-element selectors:

* selection: the highlighted part of an element
* first-line: the first line of a paragraph
* placeholder: the placeholder text of an input element

To apply styles using a pseudo-element selector, you use **double colons** (**::**) followed by the pseudo-element. Here's an example:

<!DOCTYPE html>

<input placeholder="Enter some text" />

And here's the CSS for this HTML:

input {

color: blue;

}

::placeholder {

color: red;

font-style: italic;

}



The ::placeholder pseudo-element selector styles the "placeholder part" of all form elements. As you can see in the example above, the input element itself has a color style of **blue** but the placeholder part has different styling.

## **Wrapping up**

* the universal selector, for selecting all elements
* tag names for selecting elements that match a tag name
* classes for selecting elements with a class attribute
* ids for selecting an element with an id attribute
* attributes for selecting elements that have an attribute with or without a specified value
* pseudo-classes for selecting elements in a specific state
* pseudo-elements for selecting specific parts of an element

You can also combine these selectors to be more specific about the element you want to target. You do this using **Combinators**.

Combinators allow you to use multiple selectors to target elements based on the relationship between the elements that match the selectors.

# **BootStrap Basics: Introduction to Bootstrap**

# Bootstrap is an open-source and free CSS framework that helps in directing a responsive device-friendly mobile-first front-end webpage development tool.

Bootstrap mainly includes CSS (Cascading Style Sheets) and an optional JavaScript-supported design template (plug-ins) that deals with typography, buttons, forms, and other user interface components. This Bootstrap framework helps rapid web development and supports developers in creating responsive web pages.

What does Bootstrap CSS mean?

Bootstrap is a giant collection of handy, reusable bits of code written in HTML, CSS, and JavaScript. It's also a frontend development framework that enables developers and designers to quickly build fully responsive websites.

## **What is Bootstrap?**

* Bootstrap is a free front-end framework for faster and easier web development
* Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
* Bootstrap also gives you the ability to easily create responsive designs

**Why should developers use Bootstrap?** Here are some of the essential usages of Bootstrap listed:

* **Browser supportive**: Every browser supports this Bootstrap Framework.
* **Mobile-first approach**: The Bootstrap framework has a pre-existing mobile-first style all through the library and not as separate files.
* **Simple and easy to start**: If you know HTML and CSS, you can quickly start working with Bootstrap, and its documentation is available on the official site.
* **Responsive design and looks**: Web pages designed using the Bootstrap framework has responsive CSS that can adjust to the screen size of large desktops, notebooks, tablets, and mobiles.
* **Easy customization**: It provides some built-in components and functionalities that are easy to customize.
* **Clean interface or Developers**: The bootstrap framework provides a new and consistent result for building user interfaces on web pages.
* It is an open-source framework with web-based customization.

### **Bootstrap Example**

<html lang="en">

<head>

<title>Bootstrap Example</title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>

</head>

<body>

<div class="jumbotron text-center">

<h1>My First Bootstrap Page</h1>

<p>Resize this responsive page to see the effect!</p>

</div>

<div class="container">

<div class="row">

<div class="col-sm-4">

<h3>Column 1</h3>

<p>This is column 1,Row 1</p>

<p>This is column 1,Row2</p>

</div>

<div class="col-sm-4">

<h3>Column 2</h3>

<p>This is column 2,Row 1</p>

<p>This is column 2,Row 2</p>

</div>

<div class="col-sm-4">

<h3>Column 3</h3>

<p>This is column 3,Row 1</p>

<p>This is column 3,Row 2</p>

</div>

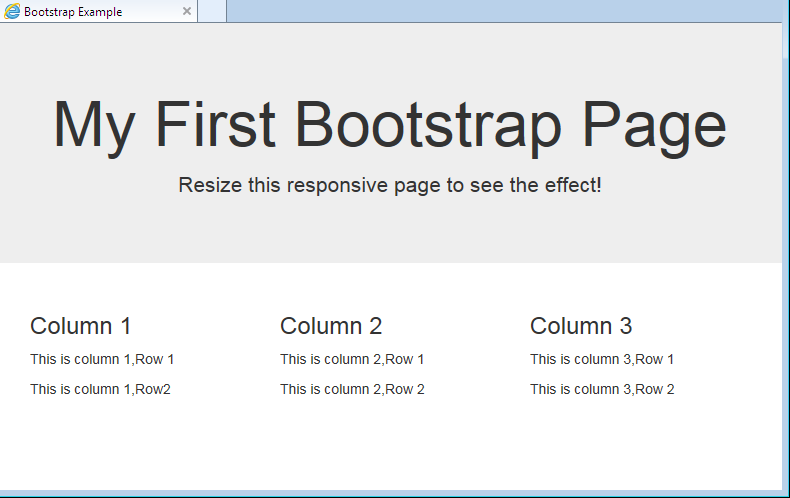
</div>

</div>

</body>

</html>

**Output**

****

\*Note:

1. <meta charset="utf-8"> **tells the browser to use the utf-8 character encoding when translating machine code into human-readable text and vice versa to be displayed in the browser**. The UTF-8 character setcovers almost all of the characters and symbols in the world.

2. The viewport is **the user's visible area of a web page**. It varies with the device - it will be smaller on a mobile phone than on a computer screen. The width=device-width part sets the width of the page to follow the screen-width of the device (which will vary depending on the device). The initial-scale=1.0 part **sets the initial zoom level when the page is first loaded by the browser**.

3. MaxCDN Bootstrap?

Bootstrap CDN  
  
If you don't want to download and host Bootstrap yourself, you can include it from a CDN (Content Delivery Network). MaxCDN provides CDN support for Bootstrap's CSS and JavaScript.

4. min. css has been minified. This means all the whitespace and other extra characters have been removed. This is commonly done for use in production, **to reduce the size of the file**.

5. If you don't want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network). Google is an example of someone who host jQuery:

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>

**Creating First Web Page with Bootstrap**

**1. Add the HTML5 doctype**

Bootstrap uses HTML elements and CSS properties that require the HTML5 doctype.

Always include the HTML5 doctype at the beginning of the page, along with the lang attribute and the correct character set:

<!DOCTYPE html>  
<html lang="en">  
  <head>  
    <meta charset="utf-8">   
  </head>  
</html>

**2. Bootstrap 3 is mobile-first**

Bootstrap 3 is designed to be responsive to mobile devices. Mobile-first styles are part of the core framework.

To ensure proper rendering and touch zooming, add the following <meta> tag inside the <head> element:

<meta name="viewport" content="width=device-width, initial-scale=1">

The width=device-width part sets the width of the page to follow the screen-width of the device (which will vary depending on the device).

The initial-scale=1 part sets the initial zoom level when the page is first loaded by the browser.

**3. Containers**

Bootstrap also requires a containing element to wrap site contents.

There are two container classes to choose from:

1. The .container class provides a responsive **fixed width container**
2. The .container-fluid class provides a **full width container**, spanning the entire width of the viewport

**Two Basic Bootstrap Pages**

The following example shows the code for a basic Bootstrap page (with a responsive fixed width container):

<!DOCTYPE html>  
<html lang="en">  
<head>  
  <title>Bootstrap Example</title>  
  <meta charset="utf-8">  
  <meta name="viewport" content="width=device-width, initial-scale=1">  
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">  
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>  
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>  
</head>  
<body>  
  
<div class="container">  
  <h1>My First Bootstrap Page</h1>  
  <p>This part is inside a .container class.</p>

<p> The .container class provides a responsive fixed width container.</p>  
</div>  
</body>  
</html>

**OUTPUT**

# **My First Bootstrap Page**

This part is inside a .container class.

The .container class provides a responsive fixed width container.

The following example shows the code for a basic Bootstrap page (with a full width container):

<!DOCTYPE html>

<html lang="en">

<head>

<title>Bootstrap Example</title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container-fluid">

<h1>My First Bootstrap Page</h1>

<p>This part is inside a .container-fluid class.</p>

<p>The .container-fluid class provides a full width container, spanning the entire width of the viewport.</p>

</div>

</body>

</html>

# **OUTPUT**

# **My First Bootstrap Page**

This part is inside a .container-fluid class.

The .container-fluid class provides a full width container, spanning the entire width of the viewport.

# **Responsive Web Design**

Responsive web design is about creating web pages that look good on all devices! A responsive web design will automatically adjust for different screen sizes and viewports.

Responsive Web Design is about using HTML and CSS to automatically resize, hide, shrink, or enlarge, a website, to make it look good on all devices (desktops, tablets, and phones)

**Setting The Viewport**

To create a responsive website, add the following <meta> tag to all your web pages:

### **Example**

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

</head>

<body>

<h2>Setting the Viewport</h2>

<p>This example does not really do anything, other than showing you how to add the viewport meta element.</p>

</body>

</html>

**Containers in Bootstrap**

In bootstrap, the **container**is used to set the content’s margin. It contains row elements and the row elements are containers of columns. This is known as the grid system.   
There are two container classes in bootstrap:

1. **.container**
2. **.container-fluid**

Let’s look at each of the above two classes in detail with examples:

**.container**: The .container class provides a responsive fixed width container.  
In the below example, the div with class “container” will have a fixed left and right margin and will not take the complete width of its parent or the viewport.

<!doctype html>

<html lang="en">

  <head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link href="<https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css>" rel="stylesheet">

    <link href="<https://getbootstrap.com/docs/5.2/assets/css/docs.css>" rel="stylesheet">

    <title>Bootstrap Example</title>

    <script src="<https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js>"></script>

  </head>

<body>

    <!-- Since we are using the class container, the below

        div will not take complete width of its parent. -->

    <div class="container mt-4">

      <div class="bg-dark">

        <h1 class="text-success">GeeksforGeeks</h1>

<p class="text-light">A computer science portal for geeks.</p>

      </div>

    </div>

</body>

</html>

**OUTPUT**



**.container-fluid**: The .container-fluid class provides a full-width container which spans the entire width of the viewport.  
In the below example, the div with class “container-fluid” will take up the complete width of the viewport and will expand or shrink whenever the viewport is resized.

<!doctype html>

<html lang="en">

  <head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link href="<https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css>" rel="stylesheet">

    <link href="<https://getbootstrap.com/docs/5.2/assets/css/docs.css>" rel="stylesheet">

    <title>Bootstrap Example</title>

    <script src="<https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js>"></script>

  </head>

<body>

    <!-- Since we are using the class container-fluid, the

        below div will expand whenever the viewport is resized. -->

    <div class="mt-4 container-fluid bg-dark">

        <h1 class="text-success">GeeksforGeeks</h1>

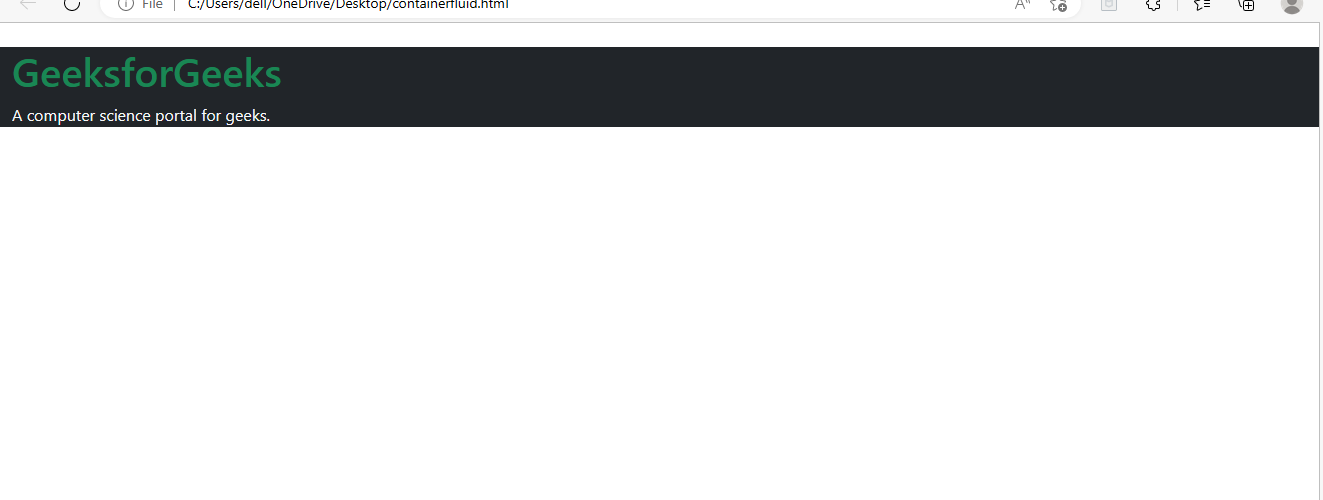
<p class="text-light">A computer science portal for geeks.</p>

     </div>

</body>

</html>

**OUTPUT**

****

**GRID in Bootstrap**

Bootstrap’s grid system uses a series of containers, rows, and columns to layout and align content. It is a very powerful tool that makes developing websites easier. It is made with flexbox hence fully responsive and also adjusts the items in the container according to the device width. The container is a wrapping element that wraps all other items and content on the web page. CSS requires this wrapping element to make the grid work properly. The .container class is the class that we usually use while making use of bootstrap in our code as it also provides some additional options like setting the alignment to the centre and horizontally padding the content.

The bootstrap grid has 12 columns present it, although it is not necessary to make use of all the columns, the sum must not go beyond 12. They can also be merged to make wider columns as per the preference.

Bootstrap Grid System allows up to 12 columns across the page. You can use each of them individually or merge them together for wider columns. You can use all combinations of values summing up to 12. You can use 12 columns each of width 1, or use 4 columns each of width 3 or any other combination.

**syntax:**

.class {

display:grid;

}

**Note:** An HTML element becomes a grid if that element sets **display:** **grid;** in style section or **inline-grid. Example:**

| <!DOCTYPE html>  <html>  <head>      <style>          /\* Designing all grid \*/      .grid-container {          display: grid;          grid-template-columns: auto auto auto;          background-color: gray;          padding: 5px;      }      /\* Designing all grid-items \*/      .grid-item {          background-color: rgba(255, 255, 255, 0.8);          border: 1px solid black;          padding: 20px;          font-size: 30px;          text-align: center;      }      /\* Designing h1 element \*/      h1 {          color: green;          text-align: center;      </style>  </head>  <body>      <h1>GeeksforGeeks</h1>      <!-- Creating grid -->      <div class="grid-container">          <div class="grid-item">1</div>          <div class="grid-item">2</div>          <div class="grid-item">3</div>          <div class="grid-item">4</div>          <div class="grid-item">5</div>          <div class="grid-item">6</div>          <div class="grid-item">7</div>          <div class="grid-item">8</div>          <div class="grid-item">9</div>      </div>  </body>  </html>  **Output:**  Description: https://media.geeksforgeeks.org/wp-content/uploads/20190708210306/Screenshot-466.png **Bootstrap Tables** **Bootstrap**provides a series of classes that can be used to apply various styling to the tables such as changing the heading appearance, making the rows stripped, adding or removing borders, making rows hoverable, etc. Bootstrap also provides classes for making tables responsive.  **Simple Table:** The *.table* class is used to create a simple Bootstrap table. This class name is used with the <table> tag to create a table.  **Syntax:**  <table class="table"> Table Contents... <table>   | <!DOCTYPE html> <html lang="en">  <head>      <title>Bootstrap Tables</title>      <meta charset="utf-8">      <meta name="viewport"          content="width=device-width, initial-scale=1">      <link rel="stylesheet" href=  "<https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css>">      <script src="<https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js>">      </script>      <script src="<https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js>">      </script>      <script src="<https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js>">      </script>  </head>  <body>      <div class="container">          <h1 style="text-align:center;color:green;">              GeeksforGeeks          </h1>          <!-- Bootstrap table class -->          <table class="table">              <thead>                  <tr>                      <th scope="col">S. No.</th>                      <th scope="col">Name</th>                      <th scope="col">City</th>                      <th scope="col">Age</th>                  </tr>              </thead>              <tbody>                  <tr>                      <th scope="row">1</th>                      <td>Ajay</td>                      <td>Patna</td>                      <td>20</td>                  </tr>                  <tr>                      <th scope="row">2</th>                      <td>Rahul</td>                      <td>Chandigarh</td>                      <td>17</td>                  </tr>                  <tr>                      <th scope="row">3</th>                      <td>Parush</td>                      <td>Kolkata</td>                      <td>22</td>                  </tr>              </tbody>          </table>      </div>  </body>  </html> | | --- |   Description: https://media.geeksforgeeks.org/wp-content/uploads/20190509114815/table110.png  **Stripped rows:** The *.table-stripped* class is used to create an alternate dark and light rows. Use the combination of table and table-stripped classes within the <table> tag to create a striped table.  **Syntax:**  <table class="table table-stripped"> Table Contents... <table> |
| --- | --- |