

Cascading Style Sheets

- **Cascading Style Sheets (CSS)** is a stylesheet language used to design a webpage to make it attractive.
- CSS describes how HTML elements are to be displayed on screen.
- It can control the layout of multiple web pages all at once and hence saves lot of work.
- It allows us to apply styles on web pages very easily and independently from HTML.
- There are three types of CSS which we can use in our html files/web pages:
 - **Inline styles**
 - **Internal or Embedded styles**
 - **External style sheets**

Inline Styles

- Inline style contains the CSS property in the body section of the web page attached with the element or tag.

Example-

```
<p style="color: blue; font-size: 46px;">  
    Hello Friends!  
</p>
```

- Here, inside **<p>** tag we have used style attribute to define the CSS style that directly affect the contents of **<p>** tag.
- The value of the **style** attribute will be CSS property-value pairs: "**property: value;**".
- We can write as many property value pairs as we want, by separating them with semicolon symbol.

Internal/Embedded Styles

- The Style sheet rules are written within the HTML file in the <head> section of the web page. That is why it is also called embedded style.

Example-

```
<html>
<head>
<style>
body {
    background-color: yellow;
}
h1 {
    color: red;
    margin-left: 80px;
    margin-top: 50px;
}
</style>
</head>
<body>
<h1>The internal style sheet is applied on this heading.</h1>
<p>This paragraph will not be affected.</p>
</body>
</html>
```

External Styles

- The external style sheet is generally used when you want to make changes on multiple pages.
- It facilitates you to change the look of the entire web site by creating just one CSS file.
- An External style sheet is a separate file having extension **.css** and contains only style properties with the help of tag attributes.
- It uses the **<link>** tag on every page and the **<link>** tag should be put inside the **<head>** section.

External Style- Example

web1.html-

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="main.css">
</head>
<body>
<p>CSS1.HTML</p>
<h1>Style applied on this heading.</h1>
</body>
</html>
```

web2.html-

```
<html>
<head>
<link rel="stylesheet" type="text/css" href="main.css">
</head>
<body>
<p>CSS2.HTML</p>
<h1>Style sheet is applied on this heading.</h1>
</body>
</html>
```

main.css-

```
body {
    background-color: lightblue;
}
h1 {
    color: navy;
    margin-left: 20px;
}
```

CSS Selectors

- CSS selectors are used to select the content on which you want to apply the style.
- Selectors are the part of CSS rule set.
- These selectors are used to select HTML elements according to its id, class, type, attribute, etc.
- There are mainly five different types of selectors in CSS.
 - Element Selector
 - Id Selector
 - Class Selector
 - Universal Selector
 - Group Selector

Element Selector

- The element selector selects HTML elements based on the element name (or tag name) for example p, h1, h2, body, div, etc.

Example-

```
h1 {  
    color: red;  
    font-size: 3rem;  
}
```

```
p {  
    color: white;  
    background-color: navy;  
    text-align: center;  
}
```

Id Selector

- The id selector uses the **id** attribute of an HTML element to select a specific element.
- The id selector always prefixed by the hash (#) symbol in stylesheet.

Example-

```
#hd1 {  
    color: red;  
    font-size: 25px;  
}
```


Class Selector

- The class selector selects HTML elements with a specific class attribute.
- The class selector always prefixed by the dot (.) symbol in stylesheet.
- Class selector always get the priority over element selector.

Example-

```
.p-class {  
    color:red;  
    font-family: courier;  
    background-color: yellow;  
}
```

Universal Selector

- Universal selector represents by an asterisk symbol (*) and is used to select all the elements in a HTML document or in a webpage.
- It also selects all elements which are inside under another element.

Example-

```
* {  
    color: white;  
    font-size: 20px;  
    font-family: arial;  
    background-color: black;  
}
```

Group Selector

- Group selector is used to style all comma separated elements with the same style.

Example-

```
h4, p, h5{  
    color: white;  
    background-color: purple;  
    font-family: monospace;  
}
```

- Or even we can give different type of selectors separated by comma.

Example-

```
#hd, .p-class, h1{  
    color: white;  
    background-color: purple;  
    font-family: courier;  
}
```

CSS Comments

- Comments are the statements/expressions ignored by browsers.
- Comments are used to explain the code, and may help when someone edits the source code later.

Examples-

/ A single-line comment */*

```
p {  
  color: blue;  
}
```

```
p {  
  color: blue; /* Sets text color to blue */  
}
```

```
p {  
  color: /*blue*/ purple;  
}
```

/ This is
a multi-line
comment */*

```
p {  
  color: blue;  
}
```

Background Image

- Set a background-image for HTML elements.
- Generally used to set background image for body element.
- By default, a background-image is placed at the top-left corner of an element, and repeated both vertically and horizontally.

Example-

```
body {  
  background-image: url("image1.jpg");  
  background-color: yellow;           /* will appear if image is not present */  
  background-repeat: no-repeat;  
  background-size: cover; /* Image will resize and cover the entire container */  
}
```

Example-

Sets a linear-gradient with two colors as a background image.

```
body {  
  background-image: linear-gradient(red, yellow);  
}
```

Background Position

- To set the background image position we can use the 'background-position' property.
- By default, a background image is placed at the top-left corner of an element, and repeated both vertically and horizontally.

Example-

```
body {  
  background-image: url("image1.jpg");  
  background-repeat: no-repeat;  
  background-position: top;  
}
```

Background-position (Possible values)-

left top	center top
left center	center center
left bottom	center bottom
right top	
right center	
right bottom	

Note- If we only specify one keyword/ value, the other value will be "center".

HTML <DIV> Tag

- The **<div> tag** is known as Division tag.
- It is used in HTML to make divisions of content in the web page like (text, images, header, footer, navigation bar, etc).
- <Div> tag has both opening(<div>) and closing (</div>) tag and it is mandatory to close the tag.
- <Div> tag is the most usable tag in web development because it helps us to separate out the data in a web page and we can create a particular section for particular data or function in the web pages.
- <Div> tag is block-level tag, means that it covers the entire width of browser from left to right and each <div> tag creates a new line.
- The content inside the different <div> tags will always be displayed in a new line.

Example-

```
<html>
  <body>
    <div> div tag 1 </div>
    <div> div tag 2 </div>
    <div> div tag 3 </div>
    <div> div tag 4 </div>
  </body>
</html>
```

Stylize <DIV> Tag

The stylesheet can be applied on <div> elements to make it more appealing.

Example-1:

```
<div style="border:2px solid blue;padding:25px;font-size:25px">
<p>Welcome to the front-end world, which is the combination of
HTML, JavaScript, CSS and many more technologies.</p>
<p>This is the second paragraph.</p>
</div>
```

Example-2: To make the <div> element center aligned, use **margin: auto**. To make it right align use, **margin-left: auto** and for left align use, **margin-right: auto**

```
<div style="width:500; border:5px solid green; margin:auto; background:cyan">
  <h1 align="center">
    Fron End Engineering
  </h1>
</div>
```


Exercise - 1

- Create a Red color circle of radius 100px.


```
<head>
  <style>
    .square{
      width: 200px;
      height: 200px;
      margin:auto;
      background-color:  red;
      border-radius: 50%;
    }
  </style>
</head>
<body>
<div class="square"></div>
</body>
```

Text over Image- HTML



```
<div class="container">  
  <div class="bottom-left">Bottom Left</div>  
  <div class="top-left">Top Left</div>  
  <div class="top-right">Top Right</div>  
  <div class="bottom-right">Bottom Right</div>  
  <div class="centered">Centered</div>  
</div>
```

Text over Image- CSS (position: absolute)

```
.container {  
  border: 5px solid  blue;  
  color:  red;  
  background-image: url("flower.jpg");  
  background-size: cover;  
  width: 500px;  
  height: 300px;  
}
```

```
.top-left {  
  position: absolute;  
  top: 15px;  
  left: 16px;  
}
```

```
.bottom-left {  
  position: absolute;  
  bottom: 190px;  
  left: 16px;  
}
```

```
.centered {  
  position: absolute;  
  top: 28%;  
  left: 20%;  
}
```

```
.top-right {  
  position: absolute;  
  top: 15px;  
  right: 660px;  
}
```

```
.bottom-right {  
  position: absolute;  
  bottom: 190px;  
  right: 660px;  
}
```

Text over Image- CSS (position: relative)

```
.top-left {  
  position: relative;  
  top: -16px;  
  left: 4px; }
```

```
.top-right {  
  position: relative;  
  top: -36px;  
  right: -430px; }
```

```
.centered {  
  position: relative;  
  top: 23%;  
  left: 45%; }
```

```
.bottom-left {  
  position: relative;  
  bottom: -280px;  
  left: 4px; }
```

```
.bottom-right {  
  position: relative;  
  bottom: -224px;  
  right: -410px; }
```

Now, after adjusting all the text, just add **margin: auto** property to the **.container** class selector.

CSS Font Property

- Font property in CSS is used to control the look of texts on the webpage. We can change the text size, color, style and weight by using CSS font properties.

color: color property is used to change the color of the text. There are three different ways to define a color; by color name, by hexadecimal value and by RGB value.

Example-

```
h1 { color: red; }  
h2 { color: #9000A1; }  
p { color:rgb(0, 220, 98); }  
}
```

font-family: font-family property is used to change the face of the font by using font name.

Example-

```
h1 { font-family: sans-serif; }  
h2 { font-family: serif; }  
p { font-family: courier; }  
}
```

CSS Font Property

Font-style: font-style property is used to make the font bold, italic or oblique.

Example-

```
<p style="font-style: normal;">  
    Paragraph 1.  
</p>
```

```
<p style="font-style: oblique">  
    Paragraph 2.  
</p>
```

```
<p style="font-style: italic;">  
    Paragraph 3.  
</p>
```

CSS Font Property

Font-variant: font-variant property applies a small-caps effect to the text.

Example-

```
<p style="font-variant: small-caps;">This paragraph is shown with small-caps effect.</p>
```

Font-weight: It is used to increase or decrease the boldness and lightness of the font.

Example-

```
<p style="font-weight:bold;">font is bold.</p>
```

```
<p style="font-weight:bolder;">font is bolder.</p>
```

```
<p style="font-weight:lighter;">font is lighter.</p>
```

CSS Example

HTML

```
<body>
<h1>First Section Heading</h1>
<p>
Here is the first paragraph, containing
sample text that really goes on and on and repeat
itself with no meaning at all just to show how
to write long example text!
</p>
<div class="shaded">
<h1>Another Section Heading</h1>
<p>
Another paragraph.
</p>
</div>
</body>
```

CSS

```
body {
  font-family: Serif;
  font-size: 15px;
  color: black;
  background: white;
  margin: 18px;
}

h1 {
  font-size: 20px;
  margin-top: 0px;
  margin-bottom: 15px;
  border-bottom: 2px solid black
}

.shaded {
  background: lightgrey;
}
```


CSS Text Shadow

- The **text-shadow** property in CSS is used to add shadows to the text.
- This property accepts a list of shadows that are to be applied to the text, separated by the comma.

Syntax-

`text-shadow: h-shadow v-shadow blur-radius color;`

- **h-shadow**: it is required & used to specify the position of horizontal shadow of a text.
- **v-shadow**: This property is required & used to specify the position of vertical shadow of a text.
- **blur-radius**: It is used to set the blur radius. Its default value is 0 & is optional.
- **none**: It is the default value and means that no shadow applies on text.
- **color**: It is used to set the color of the shadow. It is optional.

Example-

```
<h1 style="text-shadow: 5px 3px 7px cyan;">This text is awesome!</h1>
```

This text is awesome!

CSS Pseudo Selectors

CSS Pseudo Selectors

hover - Apply rule when mouse is over element (e.g. tooltip)

```
p: hover, a: hover {  
    background-color: yellow;  
}
```

a:link, a:visited - Apply rule when link has been visited or not visited (link)

```
a: visited {  
    color: green;  
}
```

```
a: link {  
    color: blue;  
}
```

CSS Example 1

Try to create the following navigation bar by applying stylesheet.



Home About Us Contact Us LOGIN LOGOUT

CSS Example 1: Code

HTML

```
<body>
<header>
  <nav>
    <div class="n1">
      <h3 class="nav1">
        Home &nbsp; &nbsp; &nbsp; &nbsp;
      </h3>
      <h3 class="nav1">
        About Us &nbsp; &nbsp; &nbsp; &nbsp;
      </h3>
      <h3 class="nav1">
        Contact Us &nbsp; &nbsp; &nbsp; &nbsp;
      </h3>
      <h3 class="nav1">
        LOGIN &nbsp; &nbsp; &nbsp; &nbsp;
      </h3>
      <h3 class="nav1">
        LOGOUT &nbsp; &nbsp; &nbsp; &nbsp;
      </h3>
    </div>
  </nav>
</header>
</body>
```

CSS

```
body { background-color: purple; }

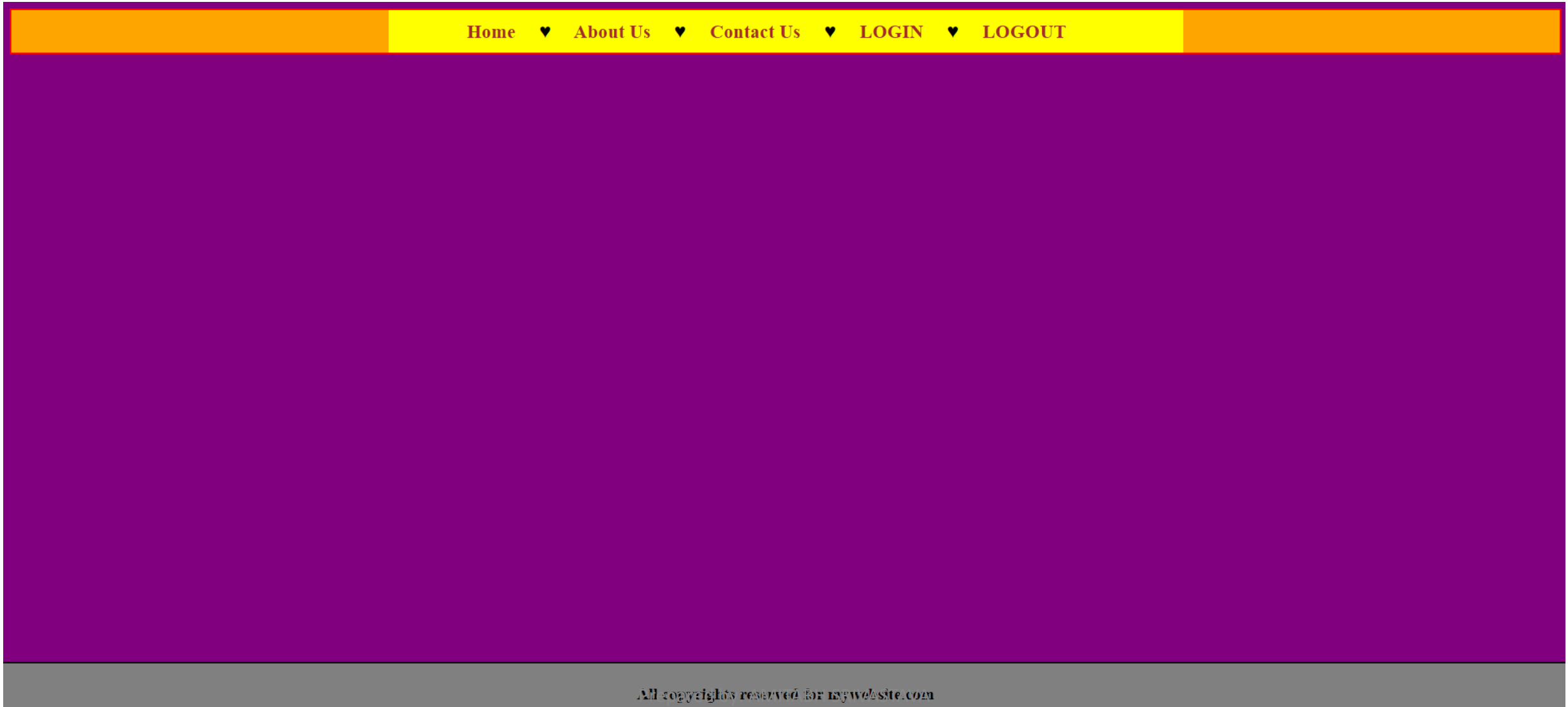
header { background-color: orange;
  border: 2px solid red }

a { color: brown; text-decoration: none }
a:hover { color: red }
.nav1 { display: inline }

.n1 { width: 50%; padding: 10px;
  margin: auto; background-color: yellow;
  text-align: center}
```

CSS Example 2

Try to create the fixed footer by applying stylesheet.



CSS Example 2: Code

HTML

```
<footer>  
  <h4>All copyrights reserved for mywebsite.com</h4>  
</footer>
```

CSS

```
footer { background-color: grey;  
  border: 2px solid black; position: fixed;  
  bottom: 0px; left: 0px; right: 0px;  
  text-align: center; height: 45px }
```

CSS Example 3

Try to create the fixed navigation bar at top by applying stylesheet.



CSS Example 3: Code

HTML

```
<body>
<div class="spandiv">
  <p>About Us</p>
  <p>Home</p>
  <p>Contact Us</p>
  <p>Login</p>
  <p>Logout</p>
</div>
<h2>About Us</h2>
<br><br><br><br><br>
<h2>Services</h2>
<br><br><br><br><br>
<h2>Offers</h2>
<br><br><br><br><br>
<h2>Clients</h2>
<br><br><br><br><br>
<h2>Technology</h2>
<br><br><br><br><br>
<h2>Mission</h2>
<br><br><br><br><br>
<h2>Contact Us</h2>
</body>
```

CSS

```
<style>
  body { background-color: lightgreen }
  .spandiv { border: 5px solid darkgreen;
              position: fixed;
              top: 0%;
              left: 0%;
              width: 98%;
              height: 30px;
              padding: 10px;
              background-color: lightgreen;}

  p {
    float: left;
    margin-top: 0px;
    border: 2px solid darkgreen;
    background-color: cyan;
    padding: 3px;
    padding-inline: 115px;
    font-size: 20px }
</style>
```


Styling Form Components

CSS can be used to create interactive form for our users / visitors.

There are many CSS properties available which can be used to create and style HTML forms to make them more interactive.

Styling the Width of Input- The **width** property is used to set the width of the input field.

Example-

```
<style>
    Input {
        width:10%; or width: 150px;
    }
</style>
```

Add padding to the Input field- padding: 12px;

Set margin between two input fields- margin: 12px;

Creating Borders: Color Property

- The **border** property is used to create a border around an element and defines how it should look.
- There are three properties of the border, **border-width**, **border-style** and **border-color**.

Border-color property-

- The **border-color** property is used to add color to the border of an element.
- The **border-color** property will only work when the **border-style** property is defined first, which is used to set the border or add the border to an element.
- There are four properties related with border-color through which we can change the color of **left border**, **right border**, **top border** and **bottom border**.
- If any of the above four border properties are not given then it inherits the color of the **border-color** property.

Styling Forms: Set Border

Set border to the Input field- It is used to the size and color of the text field border whereas border-radius property is used for adding rounded corners.

Example-

We can create an input field whose border is of 2px solid blue color with a border radius of 4px:

```
<style>
  Input {
    width:10%;
    margin: 8px;
    border: 2px solid blue;
    border-radius: 4px;
  }
</style>
```

We can also create border at any particular side and remove others or have all borders of different color.

Example-

```
border: none;

border-bottom: 3px solid blue;

border-left: 3px solid red;
```

To apply the style only the **<input type=text>**, use the input specifier as follows:

```
Input[type=text] {
  }
```

Input Field Text and Background

- 'color' property is used to change the color of the text in the input field.
- 'background-color' property is used to change the color of the background of the input field.

Example-

```
input {  
  width:10%;  
  margin: 8px;  
  border: 2px solid blue;  
  background-color: cyan;  
  color: blue;  
}
```

Focus Selector- When we click on the input field it gets an outline of red color. We can change this behavior by using: focus selector.

Example-

```
input[type=text]:focus {  
  border: 2px solid red;  
  background-color: lightblue;  
}
```

Input Field: Transition Property

- The transition property can be used over the input field to bring change in the size of the text-field by specifying the relaxed width and focused width along with the time period.
- **Relaxed width** means when there is no text entered by the user. And **focused width** means when user clicks on the text field to enter the text.

Example-

```
input {  
    transition: width 1s ease-in-out;  
    width:10%;  
    margin: 8px;  
    border: 2px solid blue;  
    background-color: cyan;  
    color: blue;  
}  
input[type=text]:focus {  
    transition: width 1s ease-in-out;  
    width: 30%;  
    background-color: yellow;  
}
```

HTML-

<form>

Firstname: <input type="text" name="fn">

Lastname: <input type="text" name="ln">

</form>

Creating Borders: Width Property

Border-width property-

- The **border-width** property is used to set the border width of all the four sides of an element.
- The border-width property also has four related properties, similarly like border-color property, to individually set the width of any side of a border.
 - **length:** It is used to set the width of the border in pixels. It does not take negative value.
 - **thin:** It is used to set the thin border at the top of element.
 - **medium:** It is used to set medium sized top border. It is the default value.
 - **thick:** It is used to set the thick top border.

Creating Borders: Style Property

Border-style property-

- It defines how the border should look like.
- It can be solid, dashed, dotted, etc.
- There are many other values which are allowed to assign to this property-

dotted: A dotted border

solid: A solid border

groove: A 3D grooved border.

inset: A 3D inset border.

none: A no border style

dashed: A dashed border

double: A double border

ridge: A 3D ridged border.

outset: A 3D outset border.

Creating Borders: Example

HTML

```
<body>
  <h1>CSS Border</h1>
</body>
```

CSS

```
h1{
  margin: auto;
  padding: 50px;
  color: purple;
  margin-top: 250px;
  font-size: 48px;
  width: fit-content;

  border-style: double;
  border-color: red;
  border-bottom-color: blue;
  border-right-color: blue;
  border-width: thick;
  border-radius: 20px;
}
```

To create a rounded border, we can add **border-radius** property with a value in px.

Box Shadow

- The CSS **box-shadow** property is used to apply shadow to an element.
- It will add shadow to the right and bottom sides of the box or an element.

Example-

```
<body>
<p>This is a paragraph element with a box-shadow!</p>
</body>
```

```
p {
  width: 250px;
  height: 65px;
  padding: 15px;
  border: 1px solid blue;
  background-color: cyan;
  box-shadow: 10px 10px cadetblue;
}
```

To add a blur effect to the shadow, we have to give the blur value in px in the same **box-shadow** property.

`box-shadow: 10px 10px 15px cadetblue;`

Use '**inset**' as the last value to give the inner shadow.

Transform Property

- The **transform property** in CSS is used to change the coordinate space of the visual element.
- This is used to add effects like skew, rotate and scale on elements.

HTML

```
<div class="main">
<h1>CSS transform Examples</h1>

<h2>transform: rotate</h2>
<div class="a">Hello Friends!</div>
<br><br>

<h2>transform: skewY</h2>
<div class="b">Hello Friends!</div>
<br><br>

<h2>transform: scaleY</h2>
<div class="c">Hello Friends!</div>
</div>
```

CSS

```
div.main {
  margin-left: 100px;
}

div.a {
  width: 150px;
  background-color: yellow;
  transform: rotate(20deg);
  border: 2px solid blue;
}

div.b {
  width: 150px;
  background-color: yellow;
  transform: skewY(20deg);
  border: 2px solid blue;
}

div.c {
  width: 150px;
  background-color: yellow;
  transform: scaleY(1.5);
  border: 2px solid blue;
}
```

CSS Transition

- **Transitions** in CSS allows us to control the way in which transition takes place between the two states of the element.
- For example, when on hovering the mouse over a button, we can change the background color of the element with the help of CSS selectors.

Example- (with no transition applied)

```
<body>
  <h1>Mouse Hover</h1>
  <div class="square">
  </div>
</body>
```

```
h1{
  color: green;
  text-align: center;
  text-decoration: underline;
}
div.square{
  height: 150px;
  width: 150px;
  border: 1px dashed black;
  margin: auto;
  background: pink;
}
div.square:hover{
  height: 300px;
  width: 300px;
  background: cyan;
}
```

CSS Transition



Example- (with transition property)

```
<body>
  <h1>Mouse Hover</h1>
  <div class="square">
  </div>
</body>
```

```
h1{
  color: green;
  text-align: center;
  text-decoration: underline;
}
div.square{
  height: 150px;
  width: 150px;
  border: 1px dashed black;
  margin: auto;
  background: pink;
  transition: height 2s, width 2s, background 2s;
}
div.square:hover{
  height: 300px;
  width: 300px;
  background: cyan;
}
```

Transition: Exercise

Create a Red color Square of sides of 200px of yellow border. On mouse over, apply the transition of 2-seconds to convert it into a Yellow color circle with red border.

```
<style>
  .square{
    width: 200px;
    height: 200px;
    margin:auto;
    background-color: red;
    border: 2px solid yellow;
  }
  .square:hover{
    border-radius: 50%;
    background-color: yellow;
    border: 2px solid red;
    transition: border-radius 2s, background-color 2s;
  }
</style>
```

Assignment-1

Create the following form using HTML and CSS. Green background must appear on mouse over.

SignUp Form

Please fill the below SignUp form	
Username	<input type="text" value="required"/>
First name	<input type="text" value="required"/>
Last name	<input type="text"/>
Email	<input type="text" value="required"/>
Gender	<input type="radio"/> Male <input type="radio"/> Female
<input type="button" value="SUBMIT"/> <input type="button" value="RESET"/>	

Assignment: Solution (part-1)

```
<form name="signupform" action="submit.html">
<fieldset>
  <legend>
    SignUp Form
  </legend>
  <table border="2" align="center" width="50%" bgcolor="grey">
    <tr>
      <th colspan="2">Please fill the below SignUp form</th>
    </tr>
    <tr>
      <td>Username</td>
      <td><input type="text" name="uname" required placeholder="required"></td>
    </tr>
    <tr>
      <td>First name</td>
      <td><input type="text" name="fname" required placeholder="required"></td>
    </tr>
    <tr>
      <td>Last name</td>
      <td><input type="text" name="lname"></td>
    </tr>
    <tr>
      <td>Email</td>
      <td><input type="email" name="email" required placeholder="required"></td>
    </tr>
  </table>
</fieldset>
```

Assignment: Solution (part-2)

[illegible]

Assignment: Solution (part-3)

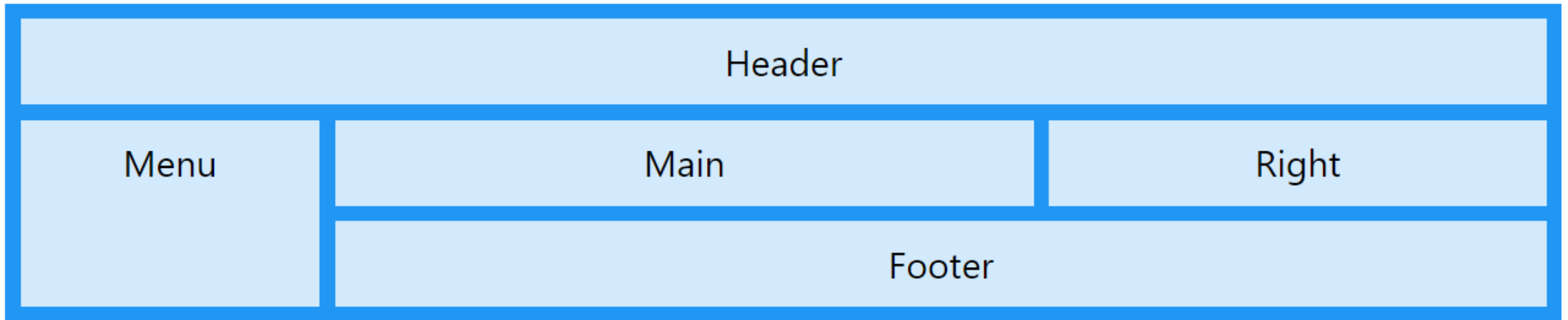
```
fieldset {  
    border: 5px double cyan;  
    box-shadow: 5px 5px 5px navy;  
    padding: 20px;  
}  
  
legend {  
    font-size: 30px;  
    color: ghostwhite;  
    font-style: italic;  
}  
  
input[type=submit], input[type=reset] {  
    background-color: cyan;  
    border-radius: 5px;  
    font-size: 15px;  
    box-shadow: 3px 3px 3px floralwhite;  
}
```

```
input[required] {  
    background-color: green;  
}  
  
input:focus {  
    background-color: white;  
}  
  
table {  
    padding: 20px;  
    border-color: navy;  
    border-width: thick;  
    box-shadow: 7px 7px 7px floralwhite;  
}
```

```
th {  
    padding: 10px;  
    box-shadow: 2px 2px 5px navy inset;  
    font-size: 18px;  
    color: navy;  
    text-decoration: underline;  
}  
  
td {  
    padding: 20px;  
    font-weight: bold;  
}
```

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.
- A grid layout consists of a parent element, with one or more child elements.
- An HTML element becomes a grid container when its **display** property is set to **grid**.
display: grid;
- All direct children of the grid container automatically become *grid items*.



CSS Grid Layout: Example

```
<div class="container">
  <div class="item1" style="background-color: red;">item 1</div>
  <div class="item2" style="background-color: green;">item 2</div>
  <div class="item3" style="background-color: blue;">item 3</div>
  <div class="item4" style="background-color: lightblue;">item 4</div>
  <div class="item5" style="background-color: lightpink;">item 5</div>
  <div class="item6" style="background-color: lightseagreen;">item 6</div>
  <div class="item7" style="background-color: darkblue;">item 7</div>
  <div class="item8" style="background-color: darkmagenta;">item 8</div>
  <div class="item9" style="background-color: darkorange;">item 9</div>
</div>
```

```
.container {
  width: 1200px;
  margin: auto;
  text-align: center;
  padding: 20px;
  display: grid;
  grid-template-columns: 400px 400px 400px;
  grid-template-rows: 200px 200px 200px;
  grid-column-gap: 20px;
  grid-row-gap: 20px;
}
```






CSS Grid Layout: Example-1

Header

Menu

Main

CSS Grid Layout: Example-1

```
<body bgcolor="black">
<div class="container">
  <div class="header" style="background-color:  grey;">Header</div>
  <div class="menu" style="background-color:  yellow;">Menu</div>
  <div class="content" style="background-color:  green;">Main</div>
</div>
</body>
```

```
.container{
  width:1140px;
  height: 485px;
  text-align: center;
  margin:auto;
  display: grid;
  grid-template-columns: 140px 500px 500px;
  grid-template-rows: 85px 400px;
}
```

```
.header{
  grid-column-start: 1;
  grid-column-end: 4;
}
.menu{
  grid-row-start: 2;
  grid-row-end: 4;
}
.content{
  grid-column-start: 2;
  grid-column-end: 4;
  grid-row-start: 2;
  grid-row-end: 4;
  overflow: auto;
}
```

CSS Grid Layout: Example-2

Header

Content

Navigation Bar

This is a grid layout.

Grid-layout is awesome if you wanna quickly
design a webpage!

This content is overflowing and hence scrollbar

CSS Grid Layout: Code: Example-2

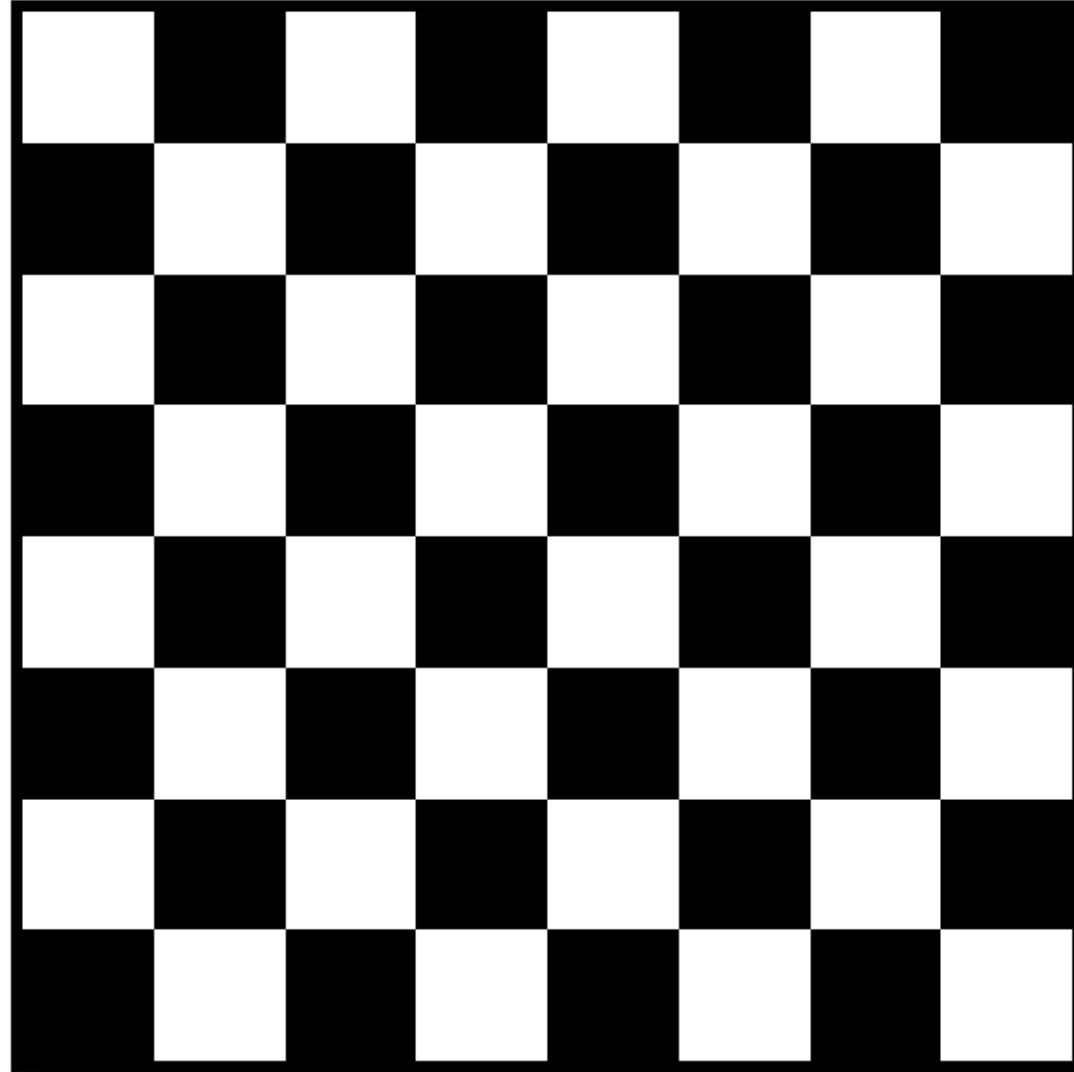
```
<div class="container">
  <div class="item1" style="background-color: blueviolet;"><h1>Header</h1></div>
  <div class="item2" style="background-color: darkviolet;">
    <h1>Content</h1>
    <p>This is a grid-layout.</p>
    <p>Grid-layout is awesome if you want to quickly design a webpage.</p>
    <p>This line is overflowing and hence scrollbar is automatically added in this item.</p>
  </div>
  <div class="item3" style="background-color: dodgerblue;"><h2>Navigation bar</h2></div>
</div>
```

```
.container{
  width:1140px;
  height: 485px;
  text-align: center;
  margin:auto;
  display: grid;
  grid-template-columns: 900px 240px;
  grid-template-rows: 85px 400px;
}
```

```
.item1{
  /* grid-column-start: 1;
  grid-column-end: 4; */
  grid-column: 1/4;
  border: 2px solid navy;
}
.item2{
  grid-row: 2/3;
  grid-column: 1/2;
  overflow:auto;
  border: 2px solid navy;
}
```

```
.item3{
  grid-column-start: 2;
  grid-column-end: 3;
  grid-row-start: 2;
  grid-row-end: 3;
  border: 2px solid navy;
}
p{
  font-size: 45px;
}
```

CSS Grid Layout: ChessBoard



Prepared by: AMITABH SRIVASTAVA

ChessBoard: HTML

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

[illegible]

Prepared by: AMITABH SRIVASTAVA

[illegible]

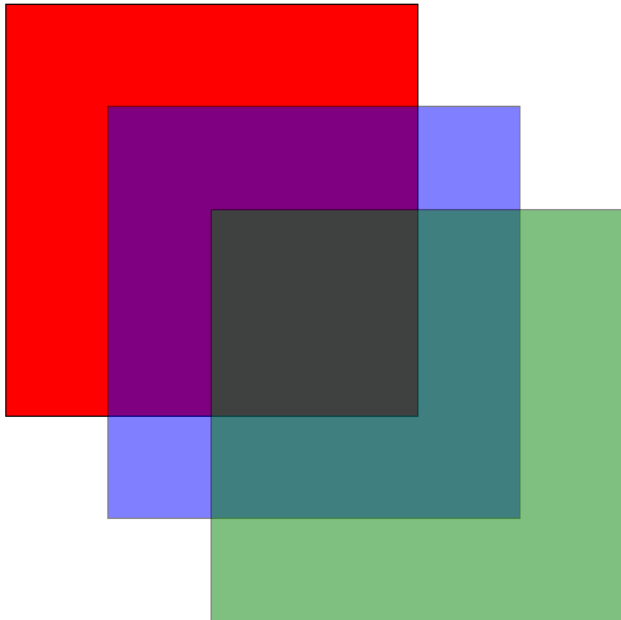
ChessBoard: CSS

```
.container {  
    border: 5px solid black;  
    box-shadow: 5px 5px 5px slategray;  
    width: 400px;  
    margin: auto;  
    display: grid;  
    grid-template-columns: 1fr 1fr 1fr 1fr 1fr 1fr 1fr 1fr;  
    grid-template-rows: 1fr 1fr 1fr 1fr 1fr 1fr 1fr 1fr;  
}  
  
.white {  
    height: 50px;  
    width: 50px;  
    background-color: white;  
}  
  
.black {  
    height: 50px;  
    width: 50px;  
    background-color: black;  
    color: white;  
}
```

Opacity

- Describes the transparency of the element.

```
<body>
  <div class="square1"></div>
  <div class="square2"></div>
  <div class="square3"></div>
</body>
```



```
div{
  width: 200px;
  height: 200px;
  position: fixed;
}
.square1{
  background-color: red;
  border: 1px solid black;
  top: 50px;
  left: 50px;
}
.square2{
  background-color: blue;
  border: 1px solid black;
  top: 100px;
  left: 100px;
  opacity: .5;
}
.square3{
  background-color: green;
  border: 1px solid black;
  top: 150px;
  left: 150px;
  opacity: .5;
}
```

Shape Outside

- Shape-outside is a CSS trick that lets you change the shape of the items wrapped around it instead of being restricted to a rectangular box.

```
<div class="container">
  <div class="shape-circle">
  </div>
  <h2>Shape Outside CSS Property</h2>
  <p>
    Shape-outside is a CSS trick that lets
    being restricted to a rectangular box.
    takes a basic shape and applies a shap
    elements. Shape-outside is a CSS trick
    instead of being restricted to a recta
    image.
  </p>
</div>
```

```
.shape-circle {
  height: 150px;
  width: 150px;
  border-radius: 50%;
  background-color: #DC143C;
  margin: 25px 25px 5px 0;
  float: left;
}
```

Shape Outside CSS Property

Now, just add the following property in your css:

shape-outside: circle();



vvesdv sdev sdfsfdsfdfsdfsfdfsdfs fsdfs fsd fsd fsd fsdf
fsdf sdfsd fsdfsfdsf sfds fsd fsd fsd dfds ffsdfsd fsdf
fsdfsfdsfdfs fsdf sdfsfdsfdfsfsdfsd fsd fsd fsd fsdf sdf
fsdfsfds fsdfs dfsfdfs fsdfsfds fsdfsf sdfsfds. fds fs
fsdfsfdsfdfsdfs dfdsf sdfsfds fsdfsf sdfsfds fsd fsd fsd
fsdfsd sdfsd fsdf sfds fsd fsdfsfds fsdfsfds. vvesdv sdev s
fsdfsd fsdfsfds fsdfs. f sdfsfds fsdf dsfsdfsd fsdf sdfsd fsd
fsdfsf. fdfsfs fdfsfsdfs sdfsd fsdfsfdsfsdf fsdf sdfsfdsfdfsfsdfsd fsd fsd fsd fsd:

Media Queries

- CSS Media Queries are used to apply different styles on different devices based on screen width, height, device orientation, resolution etc.
- By using media queries, you can create web designs that respond perfectly to the user's device (responsive) and enhance the user experience.
- Syntax-

```
@media media-type and (media-expression) {  
/* CSS styles go here */  
}
```

- **@media** is used to represent the beginning of a media query.
- **media-type** tells the browser that for what kind of media this code is. We can use the following values for this: **all**, **print**, **screen** and **speech**.
- **media-expression** is a rule, that has to be passed for the CSS to be applied.

Media Queries: Example-1

```
@media screen and (max-width: 600px) {  
  body {  
    background-color: red;  
  }  
}
```

Here, media-type is **screen** and media-expression is **max-width: 600px** which means this will change the background color of the body when the width of the screen is 600px or less than 600px (because here the maximum width is 600px).

Media Queries: Example-2

```
@media screen and (max-width: 675px) {  
body {  
background-color: red;  
color: white;  
}  
}
```

In this example, when the screen size is 675px or less than 675px, the background color and text color of the body will be changed.

Media Queries: Example-3

```
@media screen and (orientation: landscape) {  
body {  
background-color: red;  
color: white;  
}  
}
```

In this example, when the screen orientation changed to landscape the CSS will be applied to body. Other possible value can be “**portrait**”.

Media Queries: Example-4

```
@media screen and (min-resolution: 300dpi) {  
  body {  
    background-color: red;  
    color: white;  
  }  
}
```

In this example, when the screen resolution is 300dpi, the CSS will be applied to body.

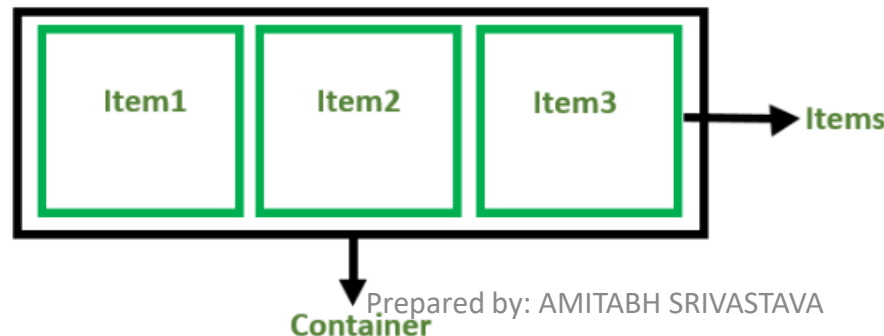
Media Queries: Example-5

```
@media (min-width: 375px) and (max-width: 758px) {  
  body {  
    background-color: red;  
    color: white;  
  }  
}
```

In this example, when the screen size is between 375px and 758px, the background color and color of the body will be changed.

Flexbox Layout

- The Flexbox layout (Flexible Box) module goal is to provide a more effective way to layout, align and distribute space among items in a container even their size is unknown/dynamic.
- The Flex layout gives the ability to the container to alter the width/height of its items to best fill the available space (mostly to accommodate to all kind of display devices and screen sizes).
- A flex container expands items to fill available free space or shrinks them to fit properly.
- We can also change the order of items present inside a Flex layout.
- It is useful for creating small-scale layouts and is responsive and mobile-friendly.
- There are two main components of a flex layout-
 - Flex container- The parent <div> which contains various divisions/sections.
 - Flex items- The items inside the <div> container.



Flexbox: display property

- The display property accepts two types of values- **flex** and **block**.
- The value '**flex**' aligns the flex items adjacent to each other.
- The '**block**' value aligns the flex items in a column.

```
<body>
  <h2> Flexbox Example </h2>
  <div class="flex-container">
    <div>Item-A</div>
    <div>Item-B</div>
    <div>Item-C</div>
  </div>
</body>
```

```
.flex-container {
  display: flex;
  background-color: blue;
}

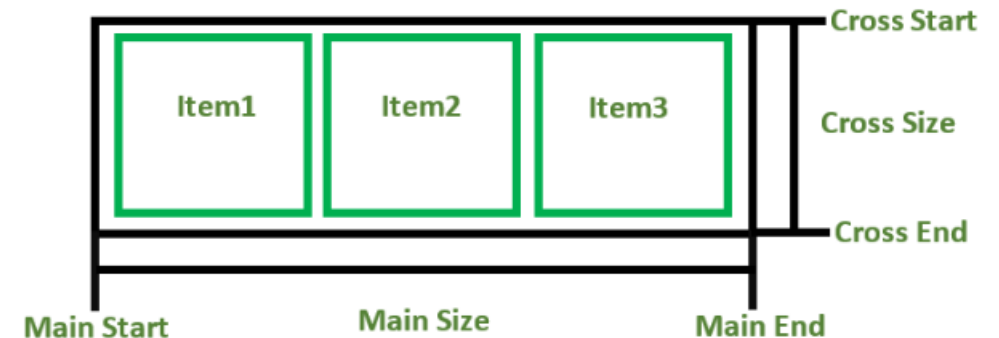
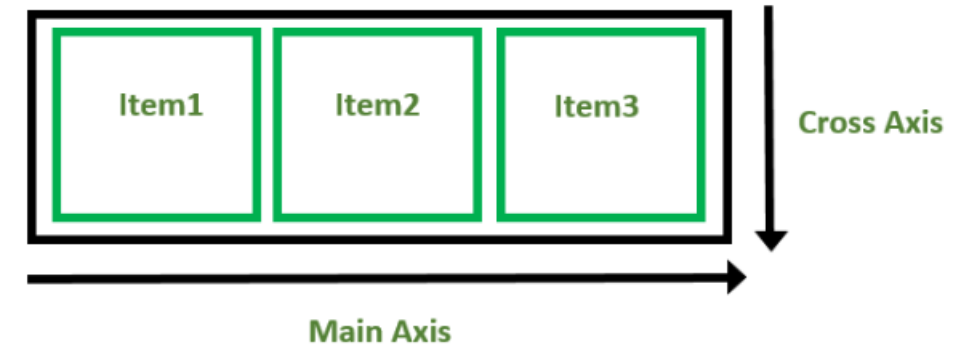
div{
  background-color: cyan;
  margin: 10px;
  padding: 15px;
  border: 2px dashed blue;
}
```

- Try to change the value of display from '**flex**' to '**block**' and '**inline-flex**'.

Flexbox Axis

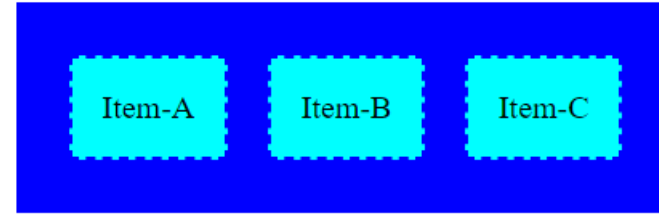
Every Flexbox layout contains 2 axes: **Main Axis**, **Cross Axis**.

- **Main Axis:** By default, it runs from left to right.
- **Main Start:** The start of the main axis is called Main Start.
- **Main Size:** The length between Main Start and Main End.
- **Main End:** The endpoint is called Main End.
- **Cross Axis:** It will be perpendicular to the main axis, i.e. from top to bottom.
- **Cross Start:** The start of the Cross axis is called Cross Start.
- **Cross Size:** The length between Cross Start and Cross End.
- **Cross End:** The endpoint is called Cross End.

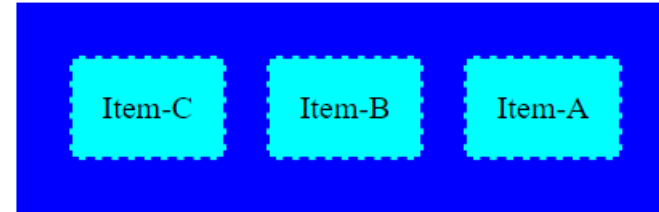


Flexbox: direction property

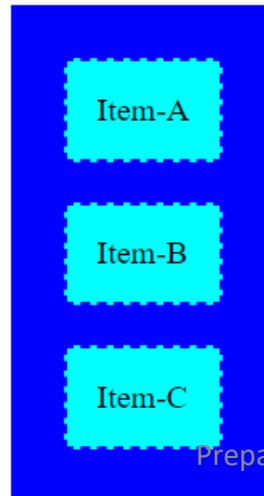
To align flex items left to right-
`flex-direction: row;`



To align flex items right to left-
`flex-direction: row-reverse;`

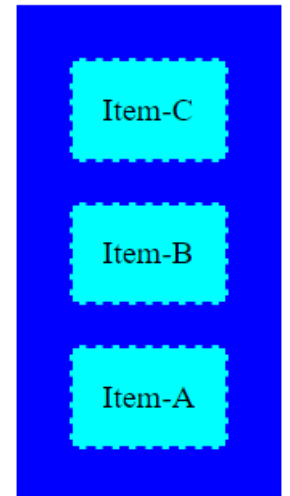


To align flex items top to bottom-
`flex-direction: column;`



Prepared by: AMITABH SRIVASTAVA

To align flex items bottom to top-
`flex-direction: column-reverse;`



Flexbox: justify-content

```
<body>
<h1>Flexbox Example</h1>
<div class="flex-container">
  <div>Item-A</div>
  <div>Item-B</div>
  <div>Item-C</div>
</div>
</body>
```

```
<style>
  .flex-container {
    display: flex;
    flex-direction: row;
    background-color: blue;
    justify-content: center;
  }
  div {
    background-color: cyan;
    margin: 10px;
    padding: 15px;
    border: 2px dashed blue;
  }
</style>
```

flex-start



flex-end



center



space-between



space-around



space-evenly

