# CSS Layout-display

CSS display is the most important property of CSS which is used to control the layout of the element. It specifies how the element is displayed. Every element has a default display value according to its nature. Every element on the webpage is a rectangular box and the CSS property defines the behavior of that rectangular box.

There are following CSS display values which are commonly used.

# 1)CSS display inline:

The inline element takes the required width only. It doesn't force the line break so the flow of text doesn't break in inline example.
The inline elements are:

- 
- <a>
- 
- <b> etc.

**Example:**

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
display: inline;
}
</style>
</head>
<body>
<p>PHP</p>
<p>Java .</p>
<p>Python.</p>
<p>HTML .</p>
<p>CSS .</p>
</body>
</html>
```

# 2)CSS display inline-block:

The CSS display inline-block element is very similar to inline element but the difference is that you are able to set the width and height.

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
display: inline-block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}
</style>
</head>
<body>
<p>JAVA</p>
<p>PYTHON</p>
<p>PHP</p>
<p>HTML</p>
<p>CSS</p>
</body>
</html>
```

---

# 3) CSS display block

The CSS display block element takes as much as horizontal space as they can. Means the block element takes the full available width. They make a line break before and after them.

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
display: block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}
</style>
</head>
```

```
<body>
<p>Java</p>
<p>JavaScript</p>
<p>SQL</p>
<p>HTML</p>
</body>
</html>
```

---

# inline-block to Create Navigation Links

```
<!DOCTYPE html>
<html>
<head>
<style>
.nav {
  background-color: yellow;
  list-style-type: none;
  text-align: center;
  margin: 0;
  padding: 0;
}
.nav li {
  display: inline-block;
  font-size: 20px;
  padding: 20px;
}
</style>
</head>
<body>
<h1>Horizontal Navigation Links</h1>
<p>By default, list items are displayed vertically.But we have used display: inline-block to display
them horizontally.</p>
<ul class="nav">
  <li><a href="home.html">Home</a></li>
  <li><a href="about.html">About Us</a></li>
  <li><a href="chapters.html">Chapters</a></li>
  <li><a href="contact.html">Contact Us</a></li>
</ul>
</body>
</html>
```

---

# CSS Layout - The position Property

The position property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).

# The position Property

The position property specifies the type of positioning method used for an element.
There are five different position values:

- static
- relative
- fixed
- absolute
- sticky

# 1)position: static;

HTML elements are positioned static by default.
Static positioned elements are not affected by the top, bottom, left, and right properties.
An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

# 2)CSS Relative Positioning

The relative positioning property is used to set the element relative to its normal position.
Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position.

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
  position: relative;
  left: 30px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>position: relative;</h2>

<p>An element with position: relative; is positioned relative to its normal position:</p>
```

```
<div class="relative">
This div element has position: relative;
</div>

</body>
</html>
```

---

## 3)CSS position: fixed;

The fixed positioning property helps to put the text fixed on the browser. This fixed test is positioned relative to the browser window, and doesn't move even you scroll the window.

```
<!DOCTYPE html>
<html>
<head>
<style>
p.fixed {
    position: fixed;
    top: 50px;
    right: 5px;
    color: blue;
}
</style>
</head>
<body>

<p>Some text...</p><p>Some text...</p><p>Some text...</p><p>........</p><p>.... ...</p
><p>........</p><p>........</p><p>........</p><p>........</p>
<p>........ </p><p>........</p><p>........</p><p>........</p><p>........</p>
<p>........</p><p>........</p><p>Some text...</p><p>Some text...</p><p>Some text...</p>
<p class="fixed">This is the fix positioned text.</p>
</body>
</html>
```

---

# 4)CSS Absolute Positioning

The absolute positioning is used to position an element relative to the first parent element that has a position other than static. However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling. With the absolute positioning, you can place an element anywhere on a page.

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
  position: relative;
  width: 400px;
  height: 200px;
  border: 3px solid #73AD21;
}

div.absolute {
  position: absolute;
  top: 80px;
  right: 0;
  width: 200px;
  height: 100px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>position: absolute;</h2>

<p>An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed):</p>

<div class="relative">This div element has position: relative;
  <div class="absolute">This div element has position: absolute;</div>
</div>

</body>
</html>
```

# 5)position: sticky;

An element with position: sticky; is positioned based on the user's scroll position.
A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed).

```
<!DOCTYPE html>
<html>
<head>
<style>
div.sticky {

  position: sticky;
  top: 0;
  padding: 5px;
  background-color: #cae8ca;
  border: 2px solid #4CAF50;
}
</style>
</head>
<body>

<p>Try to <b>scroll</b> inside this frame to understand how sticky positioning works.</p>

<div class="sticky">I am sticky!</div>

<div style="padding-bottom:2000px">
  <p>In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.</p>
  <p>Scroll back up to remove the stickyness.</p>
  <p>Texts are written to enable scrolling.Texts are written to enable scrolling.Texts are written to enable scrollingTexts are written to enable scrolling.Texts are written to enable scrolling.Texts are written to enable scrolling.Texts are written to enable scrolling</p>
  <p>Texts are written to enable scrolling Texts are written to enable scrollingTexts are written to enable scrolling.</p>
</div>

</body>
</html>
```

# CSS Layout - float

The CSS float property specifies how an element should float.

# The float Property

The float property is used for positioning and formatting content e.g. let an image float left to the text in a container.
The float property can have one of the following values:
left - The element floats to the left of its container
right - The element floats to the right of its container
none - The element does not float (will be displayed just where it occurs in the text). This is default.
inherit - The element inherits the float value of its parent

Example: float right

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  float: right;
}
</style>
</head>
<body>

<h2>Float Right</h2>

<p><img src="flower.jpg" style="width:170px;height:170px;margin-left:35px;">
Right image.CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.</p>
</body>
</html>
```

# CSS List:

There are various CSS properties that can be used to control lists. Lists can be classified as ordered lists and unordered lists. In ordered lists, marking of the list items is with alphabet and numbers, whereas in unordered lists, the list items are marked using bullets.

The CSS properties to style the lists are given as follows:

- **list-style-type:** This property is responsible for controlling the appearance and shape of the marker.
- **list-style-image:** It sets an image for the marker instead of the number or a bullet point.
- **list-style-position:** It specifies the position of the marker.
- **list-style:** It is the shorthand property of the above properties.
- **marker-offset:** It is used to specify the distance between the text and the marker. It is unsupported in IE6 or Netscape 7.

# The list-style-type property

It allows us to change the default list type of marker to any other type such as square, circle, roman numerals, Latin letters, and many more. By default, the ordered list items are numbered with Arabic numerals (1, 2, 3, etc.), and the items in an unordered list are marked with round bullets (•).

```
<!DOCTYPE html>
<html>
  <head>

    <style>
          .num{
            list-style-type:decimal;
          }
          .alpha{
                    list-style-type:lower-alpha;
          }
          .roman{
                    list-style-type:lower-roman;
          }
          .circle{
            list-style-type:circle;
          }
          .square{
            list-style-type:square;
          }
          .disc{
            list-style-type:disc;
          }
    </style>
```

```
  </head>
  <body>
    <h1>
      Ordered and unordered lists:
    </h1>
            <h2>
      Ordered Lists
    </h2>
    <ol class="num">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ol>
    <ol class="alpha">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ol>
            <ol class="roman">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ol>
    <h2>
      Unordered lists
    </h2>
    <ul class="disc">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ul>
    <ul class="circle">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ul>
        <ul class="square">
      <li>one</li>
      <li>two</li>
      <li>three</li>
    </ul>

  </body>
</html>
```

# 1)CSS Margins:

**The CSS `margin` properties are used to create space around elements, outside of any defined borders.**
With CSS, we can have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

# Margin - Individual Sides

CSS has properties for specifying the margin for each side of an element:

- `margin-top`
- `margin-right`
- `margin-bottom`
- `margin-left`

If the `margin` property has four values:

- **margin: 25px 50px 75px 100px;**
  - top margin is 25px
  - right margin is 50px
  - bottom margin is 75px
  - left margin is 100px

If the `margin` property has three values:

- **margin: 25px 50px 75px;**
  - top margin is 25px
  - right and left margins are 50px
  - bottom margin is 75px

If the `margin` property has two values:

- **margin: 25px 50px;**
  - top and bottom margins are 25px
  - right and left margins are 50px

If the `margin` property has one value:

- **margin: 25px;**
  - all four margins are 25px

# The auto Value

You can set the margin property to `auto` to horizontally center the element within its container.
The element will then take up the specified width, and the remaining space will be split equally between the left and right margins.

```
div {
  width: 300px;
  margin: auto;
  border: 1px solid red;
}
```

# Example:

```
<!DOCTYPE html>
<html>
<head>
<style>

p.ex {
   margin-top: 50px;
   margin-bottom: 50px;
   margin-right: 100px;
   margin-left: 100px;
}

div {
width: 300px;
margin: auto;
border: 1px solid red;
}
</style>
</head>
<body>
<p>This paragraph is not displayed with specified margin. </p>
<p class="ex">This paragraph is displayed with specified margin.<br>This paragraph is displayed with specified margin.<br>
This paragraph is displayed with specified margin.</p>

<div>

this is auto margin.....

</div>
</body>
</html>
```

# CSS menu-design:

It is the UI(user interface) element on a webpage that includes links for the other sections of the website.
A navigation bar is mostly displayed on the top of the page in the form of a horizontal list of links.
With using CSS, attractive menu can be designed. Menu can be **horizontally** arranged or **vertically** arranged and are used with various CSS properties to make menu or navigation bar.
**<li>** and **<a>** are used with various CSS properties to make menu or navigation bar.

# Vertical Navigation Bar

```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}

li a {
  display: block;
  width: 60px;
  background-color: #dddddd;
}
</style>
</head>
<body>

<ul>
  <li><a href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>

<p>A background color is added to the links to show the link area.</p>
<p>Notice that the whole link area is clickable, not just the text.</p>

</body>
</html>
```

# Vertical Navigation Bar(link color change on hover)

```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  width: 200px;
  background-color: lightblue;
}

li a {
  display: block;
  color: blue;
  font-size:20px;
  padding: 8px 16px;
  text-decoration: none;
}
.active{
  background-color: orange;
  color: white;
}
li a:hover {
  background-color: pink;
  color: white;
}
</style>
</head>
<body>

<h2>Vertical Navigation Bar</h2>

<ul>
  <li><a href="#" class = "active">Home</a></li>
  <li><a href = "#">Java</a></li>
  <li><a href = "#">CSS</a></li>
  <li><a href = "#">HTML</a></li>
  <li><a href = "#">Bootstrap</a></li>
</ul>

</body>
</html>
```

# Full-height Fixed Vertical Navbar

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  margin: 0;
}
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  width: 25%;
  background-color: #f1f1f1;
  position: fixed;
  height: 100%;
  overflow: auto;
}
li a {
  display: block;
  color: #000;
  padding: 8px 16px;
  text-decoration: none;
}
li a.active {
  background-color: #4CAF50;
  color: white;
}
li a:hover {
  background-color: red;
  color: blue;
}
</style>
</head>
<body>
<ul>
  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>
<div style="margin-left:25%;padding:1px 16px;height:1000px;">
  <h2>Fixed Full-height Side Nav</h2>
  <h3>This is scrollable </h3>
```

**This will add scrollbar if side navigation is too long**

```
  <p> we have set overflow:auto to sidenav. To enable scrolling.</p>
  <p>Some text..</p>
</div>
</body></html>
```

---

# Horizontal Navigation Bar(Example-1)

```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #333;
}
li {
  float: left;
}
li a {
  display: block;
  color: white;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
}
.active {
  background-color: cyan;
}
li a:hover {
  background-color: red;
}
</style>
</head>
<body>
<ul>
  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>
</body></html>
```

# Fixed Navigation Bar:

Make the navigation bar stay at the top or the bottom of the page, even when the user scrolls the page.

```
<!DOCTYPE html>
<html>
<head>
<style>
body {margin:0;}
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #333;
  position: fixed;
  top: 0;
  width: 100%;
}
li {
  float: left;
}
li a {
  display: block;
  color: white;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
}
li a:hover{
  background-color: #111;
}
.active {
  background-color: #4CAF50;
}
</style>
</head>
<body>
<ul>
  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>
<div style="padding:20px;margin-top:30px;background-color:#1abc9c;height:1500px;">
```

```
  <h1>Fixed Top Navigation Bar</h1>
  <h2>Scroll this page to see the effect</h2>
  <h2>The navigation bar will stay at the top of the page while scrolling</h2>
<p>Some text some text some text some text..</p>
  <p>Some text some text some text some text..</p>
  <p>Some text some text some text some text..</p>
 <p>Some text some text some text some text..</p>
</div>
</body>
</html>
```

---

# Sticky Navbar

Add position: sticky; to <ul> to create a sticky navbar.
A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the **viewport** - then it "sticks" in place.

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  font-size: 28px;
}

ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #333;
  position: sticky;
  top: 0;
}

li {
  float: left;
}

li a {
  display: block;
  color: white;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
```

```
}

li a:hover {
  background-color: red;
}

.active {
  background-color: #4CAF50;
}
</style>
</head>
<body>

<div class="header">
  <h2>Scroll Down</h2>
  <p>Scroll down to see the sticky effect.</p>
</div>

<ul>
  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
</ul>

<h3>Sticky Navigation Bar Example</h3>
<p>The navbar will <strong>stick</strong> to the top when you reach its scroll position.</p>

<p>Cascading Style Sheets (CSS) provide a flexible way to add style to the pages of your website. This collection of formatting rules governs the appearance of your pages, and lets you define the fonts, colors, layout, and other presentation features.</p>


<p>By using CSS to control your fonts, you can ensure greater consistency in the appearance and layout of your pages in multiple browsers. Some of the many text properties that CSS lets you control include font family and size, text and background color, text formatting, and link color.</p>

Using CSS, you can also position, add color to, float text around, and set margins and borders for block-level elements. A block-level element is a standalone piece of content that's visually formatted as a block. For example, content blocks (which are equivalent to p tags) and panels (which are the same as div tags) are both block-level elements
p>Cascading Style Sheets (CSS) provide a flexible way to add style to the pages of your website. This collection of formatting rules governs the appearance of your pages, and lets you define the fonts, colors, layout, and other presentation features.</p>
```

By using CSS to control your fonts, you can ensure greater consistency in the appearance and layout of your pages in multiple browsers. Some of the many text properties that CSS lets you control include font family and size, text and background color, text formatting, and link color.

Using CSS, you can also position, add color to, float text around, and set margins and borders for block-level elements. A block-level element is a standalone piece of content that's visually formatted as a block. For example, content blocks (which are equivalent to p tags) and panels (which are the same as div tags) are both block-level elements
</body>
</html>