display Property

- The `display` property is the most important CSS property for controlling layout.
- The `display` property specifies if/how an element is displayed.
- Every HTML element has a default display value depending on what type of element it is.
- The default display value for most elements is `block` or `inline`.

Block-level Elements

- A block-level element always starts on a new line and takes up the full width available

Examples of block-level elements:

-
- <h1> - <h6>
- <p>
- <form>
- <header>
- <footer>
- <section>k

Inline Elements

- An inline element does not start on a new line
- It only takes up as much width as necessary.

Examples of inline elements:

-
- <a>
- <img>

Let's Discuss about the Display properties

Display: none;

- It use to remove the html content without deleting from html.
- It will exist in html just will be remove from browser or viewport.
- is commonly used with JavaScript to hide and show elements without deleting and recreating them.
- The `<script>` element uses `display: none;` as default.

H1{ **Display: none;** **}**	**Element h1 will be remove from viewport**

Display: block;

- It use to make a block level display to any content of HTML.
- We can use Height and width with this display.
- It get separate line for each content as block level element.

```
<a href="/html/default.asp" target="_blank">HTML</a>
<a href="/css/default.asp" target="_blank">CSS</a>
<a href="/js/default.asp" target="_blank">JavaScript</a>
```

```
a {
  display: block;
}
```

HTML
CSS
JavaScript

Display: inline;

- It use to make a inline display to any content of HTML.
- We can not use Height and width with this display.
- It get same line for each content as element.

H1 = welcome to H1 = css	

H1 {
display: inline;
}
	Welcome to css

Display: inline-block;

- It looks like inline display but.
- We can use Height and width with this display.
- It get same line for each content as element.

HTML CSS JavaScript	

H1 {
background:100px;
display: inline-block;
height:100px;
width:100px;
 	
}
	HTML CSS JavaScript

Display: list-item;

- It convert the element into the list item.
- We can use Height and width with this display.
- It get same line for each content as element.

<table>
<tr><td colspan="2">

```
<body>
<a>home<a>
<a>about<a>
<a>contact us<a>
<a>services<a>
</body>
```

</td></tr>
<tr><td>

```
a {
  display: list-item;
  list-style:inline;
  margin-left:20px;
  height:40px;
  width:100px;
}
```

</td><td>

1. home
2. about
3. contact us
4. services

</td></tr>
</table>

Visibility

The `visibility` property specifies whether or not an element is visible.

Tip: Hidden elements take up space on the page. Use the display property to both hide and remove an element from the document layout!

Property Values

Value	Description
visible	Default value. The element is visible
hidden	The element is hidden (but still takes up space)
collapse	Only for table rows (<tr>), row groups (<tbody>), columns (<col>), column groups (<colgroup>). This value removes a row or column, but it does not affect the table layout. The space taken up by the row or column will be available for other content. If collapse is used on other elements, it renders as "hidden"
initial	Sets this property to its default value. Read about *initial*
inherit	Inherits this property from its parent element. Read about *inherit*

Collaps example:

```
<table>
    <thead>
        <tr
class="collaps">
            <th>a</th>
            <th
class="b">b</th>
            <th>c</th>
            <th>d</th>
        </tr>
    </thead>

    <tbody>
        <tr>
            <td>1</td>
            <td>3</td>
            <td>5</td>
            <td>8</td>
        </tr>
    </tbody>
</table>
```

```
td, th{
    border: 1px solid black;
}
table{
    border-collapse:collapse;
}
.b{
    visibility: collapse;
}
```

CSS cursor Property

It decide the view of cursor when it will go on particular element/content.

```css
h1{
    cursor: alias;
}
```

```css
h1
{
    cursor: all-scroll;
}
```

```css
h1{
    cursor: auto;
}
```

```css
h1{
    cursor: cell;
}
```

```css
h1{
    cursor: col-resize;
}
```

```css
h1{
    cursor: context-menu;
}
```

```css
h1{
    cursor: copy;
}
```

```css
h1{
    cursor: crosshair;
}
```

```css
h1{
    cursor: default;
}
```

```css
h1{
    cursor: e-resize;
}
```

```css
h1{
    cursor: wait;
}
```

```css
h1{
    cursor: zoom-in;
}
```

```css
h1{
    cursor: ew-resize;
}
```

```css
h1{
    cursor: grab;
}
```

```css
h1{
    cursor:  grabbing;
}
```

```css
h1{
    cursor: help;
}
```

```css
h1{
    cursor: move;
}
```

```css
h1{
    cursor: n-resize;
}
```

```css
h1{
    cursor: ne-resize;
}
```

```css
h1{
    cursor: nesw-resize;
}
```

```css
h1{
    cursor: ns-resize;
}
```

```css
h1{
    cursor: nw-resize;
}
```

```css
h1{
    cursor: nwse-resize;
}
```

```css
h1{
    cursor: zoom-out;
}
```

```css
h1{
    cursor: no-drop;
}
```

```css
h1{
    cursor: none;
}
```

```css
h1{
    cursor: not-allowed;
}
```

```css
h1{
    cursor: pointer;
}
```

```css
h1{
    cursor: progress;
}
```

```css
h1{
    cursor: row-resize;
}
```

```css
h1{
    cursor: s-resize;
}
```

```css
h1{
    cursor: se-resize;
}
```

```css
h1{
    cursor: sw-resize;
}
```

```css
h1{
    cursor: text;
}
```

```css
h1{
    cursor: w-resize;
}
```

CSS position Property

Position - The CSS position property is used to set position for an element. it is also used to place an element behind another and useful for scripted animation effect

CSS Syntax

```
position: static|absolute|fixed|relative|sticky|initial|inherit;
```

Property Values

Value	Description
static	Default value. Elements render in order, as they appear in the document flow
absolute	The element is positioned relative to its first positioned (not static) ancestor element
fixed	The element is positioned relative to the browser window
relative	The element is positioned relative to its normal position, so "left:20px" adds 20 pixels to the element's LEFT position
sticky	The element 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). **Note:** Not supported in IE/Edge 15 or earlier. Supported in Safari from version 6.1 with a -webkit- prefix.

Example for Sticky position :

Take a div and para combination in html	**div.sticky {** **position: -webkit-sticky;** **position: sticky;** **top: 0;** **padding: 5px;** **background-color: #cae8ca;** **border: 2px solid #4CAF50;**

	}

z-index

- The `z-index` property specifies the stack order of an element.
- An element with greater stack order is always in front of an element with a lower stack order.
- **Note:** If two positioned elements overlap without a `z-index` specified, the element positioned last in the HTML code will be shown on top.
- **The large number of z-index value will put upside to your content and the lower number of z-index put downside/ deep to your content**