Unit-I

Web: Introduction, Internet and web, web browsers, web servers, protocols.

A web browser is a software application which enables a user to display and interact with text, images, videos, music, and other information that could be on a website. Text and images on a web page can contain hyperlinks to other web pages at the same or different website. Web browsers allow a user to quickly and easily access information provided on many web pages at many websites by traversing these links. Web browsers format HTML information for display so the appearance of a web page many differ between browsers.

Popular Browsers

Firefox

Firefox is a very popular web browser. One of the great things about Firefox is that it is supported on all different OSs. Firefox is also open source which makes its support group a very large community of open source developers. Firefox is also known for its vast range of plugins/add-ons that let the user customize in a variety of ways. Firefox is a product of the Mozilla Foundation. The latest version of Firefox is Firefox 3.

Protocols and Standards

Web browsers communicated with web servers primarily using HTTP (hypertext transfer protocol) to fetch web pages. HTTP allows web browsers to submit information to web servers as well as fetch web pages from them. Pages are identified by means of a URL (uniform resource locater), which is treated as an address, beginning with "http://" for HTTP access.

The file format for a web page is usually HTML (hyper-text markup language) and is identified in the HTTP protocol. Most web browsers also support a variety of additional formats, such as JPEG, PNG, and GIF image formats, and can be extended to support more through the use of plugins. The combination of HTTP content type and URL protocol specification allows web page designers to embed images, animations, video, sound, and streaming media into a web page, or to make them accessible through the web page.

HTML: Basics, elements, attributes, tags- list, tables, images, forms, frames, cascading style sheets.

HTML Tags

HTML tags are like keywords which define that how web browser will format and display the content. With the help of tags, a web browser can distinguish between an HTML content and a simple content. HTML tags contain three main parts: opening tag, content and closing tag. But some HTML tags are unclosed tags.

When a web browser reads an HTML document, browser reads it from top to bottom and left to right. HTML tags are used to create HTML documents and render their properties. Each HTML tags have different properties.

An HTML file must have some essential tags so that web browser can differentiate between a simple text and HTML text. You can use as many tags you want as per your code requirement.

- All HTML tags must enclosed within <> these brackets.
- Every tag in HTML perform different tasks.

HTML Elements

An HTML file is made of elements. These elements are responsible for creating web pages and define content in that webpage. An element in HTML usually consist of a start tag <tag name>, close tag </tag name> and content inserted between them. Technically, an element is a collection of start tag, attributes, end tag, content between them.

```
<!DOCTYPE html>
<html>
<head>
<title>WebPage</title>
</head>
<body>
<h1>This is my first web page</h1>
<h2> How it looks?</h2>
<p>It looks Nice!!!!!</p>
</body>
</html>
```

HTML Attribute

HTML attributes are special words which provide additional information about the elements or attributes are the modifier of the HTML element.

Each element or tag can have attributes, which defines the behavior of that element.

Attributes should always be applied with start tag.

The Attribute should always be applied with its name and value pair.

The Attributes name and values are case sensitive, and it is recommended by W3C that it should be written in Lowercase only.

You can add multiple attributes in one HTML element, but need to give space between two attributes.

Syntax

```
<element attribute name="value">content</element>
```

Java Script: Introduction to scripting, control structures, conditional statements, arrays, functions, objects.

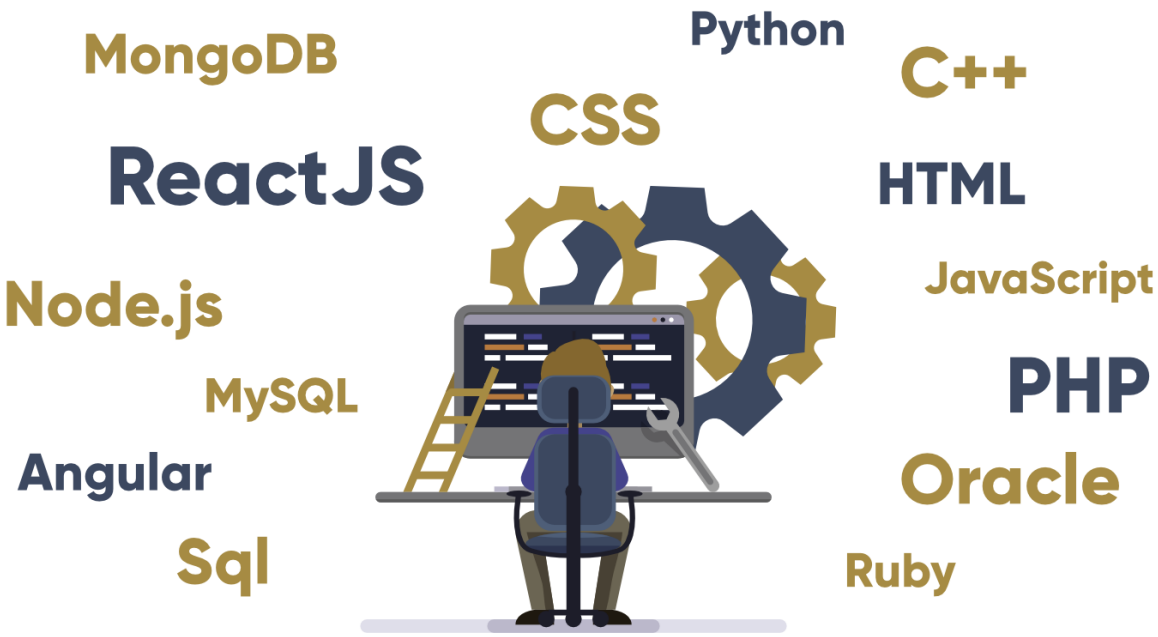
Introduction:

Full stack: In technology development, **full stack** refers to an **entire** computer system or application from the front end to the back end and the code that connects the two. The back end of a computer system encompasses "behind-the-scenes" technologies such as the database and operating system.

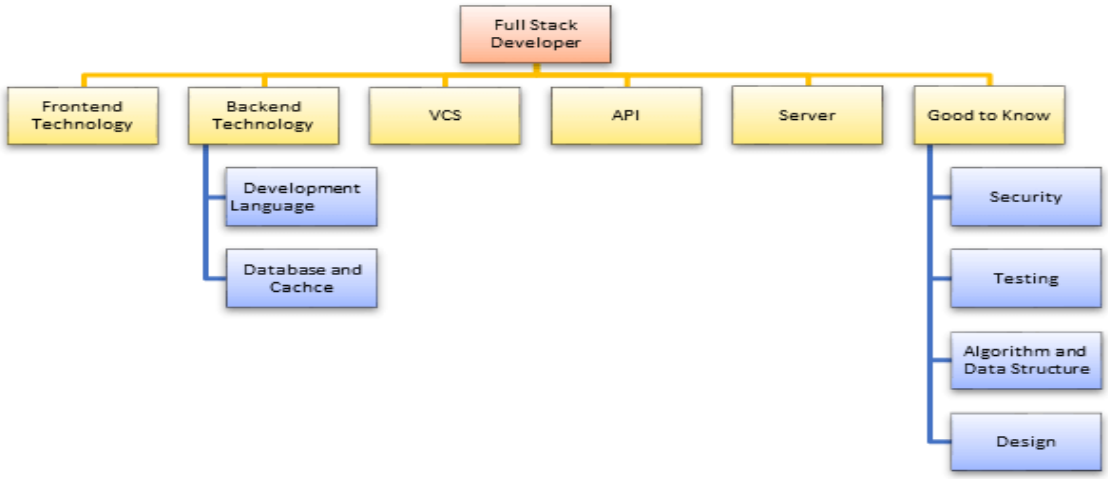
Full Stack Web Developer: A full stack web developer is a person who can develop both client and server software**.** They are proficient in both front-end and back-end languages and frameworks, as well as in server, network and hosting environments.

(or)

Full Stack Developer is an engineer who works on both client-side and server-side of the software application. This type of developer works on the Full Stack of a software application meaning Front end development, Back end development, Database, Server, API, and version controlling systems. Hence, the name "Full Stack" Developer.Full stack developer translates user requirements into the overall architecture and implement the new systems.



Full Stack Developer Skills You Need to Know:



1) Front-end technology

Full stack developer should be master of essential front-end technologies like HTML5, CSS3, JavaScript. Knowledge of third-party libraries like jQuery, LESS, Angular and ReactJS is desirable

2) Development Languages

Full stack engineer should know at least one server-side programming languages like Java, Python, Ruby, .Net etc.

3) Database and cache

Knowledge of various DBMS technology is another important need of full stack developer. MySQL, MongoDB, Oracle, SQLServer are widely used for this purpose. Knowledge of caching mechanisms like varnish, Memcached, Redis is a plus.

4) Basic design ability

In order to become a successful Full Stack web developer, the knowledge of designing is also recommended. Moreover, the person should know the principle of basic prototype design and UI /UX design.

5) Server

Exposure to handling Apache or nginx servers is desirable. A good background in Linux helps tremendously in administering servers.

6) Version control system (VCS)

A version control system allows full stack developers to keep track of all the changes made in the codebase. The knowledge of **Git** helps full stack developers to understand how to get the latest code, update parts of the code, make changes in other developer's code without breaking things.

7) Working with API (REST & SOAP):

Knowledge of web services or API is also important for full stack developers. Knowledge of creations and consumption of REST and SOAP services is desirable.

SOAP (Simple Object Access Protocol) is a standards-based web services access protocol that has been around for a long time. ... **REST** (Representational State Transfer) is another standard, made in response to **SOAP's** shortcomings. It seeks to fix the problems with **SOAP** and provide a simpler method of accessing web services.

Other important points:

1. Ability to write quality **unit tests**
2. He or she should have a complete understanding of automated processes for building testing, document, and deploying it at scale
3. An awareness of **security** concerns is important, as each layer has its own vulnerabilities
4. Knowledge of **Algorithms** and data structures is also an essential need for professional full stack developers.

Java Full Stack Developer:

A **Java Full Stack Developer** is a developer who has expertise and deep knowledge of frameworks and tools used in Java full stack development like Core Java, servlets, APIs, database, web architecture, etc. A Full Stack Java developer can build whole Java applications including front end, back-end, database, APIs, server and version control.

Full Stack JavaScript:

JavaScript has been around for over 20 years. It is the dominant programming language in web development.

In the beginning JavaScript was a language for the web client (browser). Then came the ability to use JavaScript on the web server (with Node.js).

Today the hottest buzzword is "Full Stack JavaScript".

The idea of "Full Stack JavaScript" is that all software in a web application, both client side and server side, should be written using JavaScript only.

What is a Software Stack?

Software stack is a collection of the programs which are used together to produce a specific result. It includes an operating system and its application. For example, a smartphone software stack includes OS along with the phone app, web browsers, and default applications.

Popular Stacks:

- LAMP stack: JavaScript - Linux - Apache - MySQL - PHP
- MERN: JavaScript - MongoDB - Express - React Js - Node.js
- MEAN stack: JavaScript - MongoDB - Express - AngularJS - Node.js
- Django stack: JavaScript - Python - Django - MySQL
- Ruby on Rails: JavaScript - Ruby - SQLite - Rails

LAMP Stack:

LAMP is a widely used model for web service stacks. Its name "LAMP" is an acronym of four open-source components.

- L= Linux: An open source operating system
- A= Apache: Widely used web server software
- M= MySQL: Popular open source database
- P=PHP: Server-side open source scripting language
- Many popular websites and web applications run on LAMP stack, Example: Facebook.

MERN Stack:

MERN is a collection of JavaScript-based technologies:

- M=MongoDB: Popular nosql database
- E=Express: Light and portable web program framework
- R=React: A javascript library for building user interfaces
- N=Node.js: A server-side JavaScript run time

This stack currently in the huge demand as it is widely used to develop web applications.

MEAN Stack:

MEAN Stack Application Development is witnessing a growing trend in usage. MEAN is an abbreviation of:

- M = MongoDB: nosql Database
- E = Express: Easy to use light and portable web program framework
- A = Angular.js: Robust framework for developing HTML5 and JavaScript- web programs
- N = Node.is: a server-side JavaScript run time

What Does a Full Stack Developer Do?

As a full stack developer, you may be involved in following activities:

- Translate user requirements into the overall architecture and implementation of new systems
- Manage Project and coordinate with the Client
- Write backend code in Ruby, Python, Java, PHP languages
- Writing optimized front end code HTML and JavaScript
- Understand, create and debug database related queries
- Create test code to validate the applicaition against client requirement.
- Monitor the performance of web applications & infrastructure

- Troubleshooting web application with a fast and accurate a resolution

Advantages:

The advantage of being a full stack web developer is:

- You can master all the techniques involved in a development project
- You can make a prototype very rapidly
- You can provide help to all the team members
- You can reduce the cost of the project
- You can reduce the time used for team communication
- You can switch between front and back end development based on requirements
- You can better understand all aspects of new and upcoming technologies

Disadvantages:

- The solution chosen can be wrong for the project
- The solution chosen can be dependent on developer skills
- The solution can generate a key person risk
- Being a full stack developer is increasingly complex

Technology related to full stack development:

Front end: It is the visible part of website or web application which is responsible for user experience. The user directly interacts with the front end portion of the web application or website.

HTML, CSS, BOOTSTRAP, W3.CSS, JAVASCRIPT, ES5, HTML DOM, JSON, XML, JQUERY, ANGULAR, REACT, BACKBONE.JS, EMBER.JS, REDUX, STORYBOOK, GRAPHQL, METEOR.JS, GRUNT, GULF.

HTML: HTML stands for Hyper Text Markup Language. It is used to design the front end portion of web pages using markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. The markup language is used to define the text documentation within tag which defines the structure of web pages.

CSS: Cascading Style Sheets, fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page.

JavaScript: JavaScript is a famous scripting language used to create the magic on the sites to make the site interactive for the user. It is used to enhancing the functionality of a website to running cool games and web-based software.

Front End Frameworks and Libraries:

AngularJS**:** AngularJs is a JavaScript open source front-end framework that is mainly used to develop single page web applications(SPAs). It is a continuously growing and expanding framework which provides better ways for developing web applications. It changes the static HTML to dynamic HTML. It is an open source project which can be freely used and changed by anyone. It extends HTML attributes with Directives, and data is bound with HTML.

React.js**:** React is a declarative, efficient, and flexible JavaScript library for building user interfaces. ReactJS is an open-source, component-based front end library responsible only for the view layer of the application. It is maintained by Facebook.

Bootstrap**:** Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first web sites.

jQuery**:** jQuery is an open source JavaScript library that simplifies the interactions between an HTML/CSS document, or more precisely the Document Object Model (DOM), and JavaScript. Elaborating the terms, jQuery simplifies HTML document traversing and manipulation, browser event handling, DOM animations, Ajax interactions, and cross-browser JavaScript development.

SASS: It is the most reliable, mature and robust CSS extension language. It is used to extend the functionality of an existing CSS of a site including everything from variables, inheritance, and nesting with ease.Some other libraries and frameworks are: Semantic-UI, Foundation, Materialize, Backbone.js, Express.js, Ember.js etc.

Other Important Points:

- Work with text editors to use shortcuts and its facilities i.e. Visual studio, Atom, Sublime etc.
- Make UI responsible using grid system.
- Git and git commands like init, add, commit etc for version control and to work with team.
- Other tools like npm & yarn package managers, sass css pre-processor, browser DevTools i.e. chrome devtools.
- Understand using HTTP, JSON, GraphQL APIs to fetch data using axios or other tools.
- It also requires some design skill to make layout and look better.

Back end: It refers to the server-side development of web application or website with a primary focus on how the website works. It is responsible for managing the database through queries and APIs by client-side commands. This type of website mainly consists of three parts front end, back end, and database.

PHP,ASP,C++,C#,JAVA,PYTHON,NODE.JS,EXPRESS.JS,RUBY,REST,GO,SQL,MONGO DB,FIREBASE.COM,SASS,LESS,PARSE.COM,

PASS(AZURE,HEROKU)

The back end portion is built by using some libraries, frameworks, and languages which are discussed below:

- PHP: PHP is a server-side scripting language designed specifically for web development. Since, PHP code executed on server side so it is called server side scripting language.
- C++ It is a general purpose programming language and widely used now a days for competitive programming. It is also used as backend language.
- Java: Java is one of the most popular and widely used programming language and platform. It is highly scalable. Java components are easily available.
- Python: Python is a programming language that lets you work quickly and integrate systems more efficiently.
- JavaScript: Javascript can be used as both (front end and back end) programming languages.
    - Node.js**:** Node.js is an open source and cross-platform runtime environment for executing JavaScript code outside of a browser. You need to remember that NodeJS is not a framework and it's not a programming language. Most of the people are confused and understand it's a framework or a programming language. We often use Node.js for building back-end services like APIs like Web App or Mobile App. It's used in production by large companies such as Paypal, Uber, Netflix, Wallmart and so on.
    - **Back End Frameworks:** The list of back end frameworks are: Express, Django, Rails, Laravel, Spring etc.
    - The other back end program/scripting languages are: C#, Ruby, REST, GO etc.

Other Important Points:

- Structuring the data in efficient way.
- Handle request-response of APIs for storing and retrieve data.
- Security of data is important.

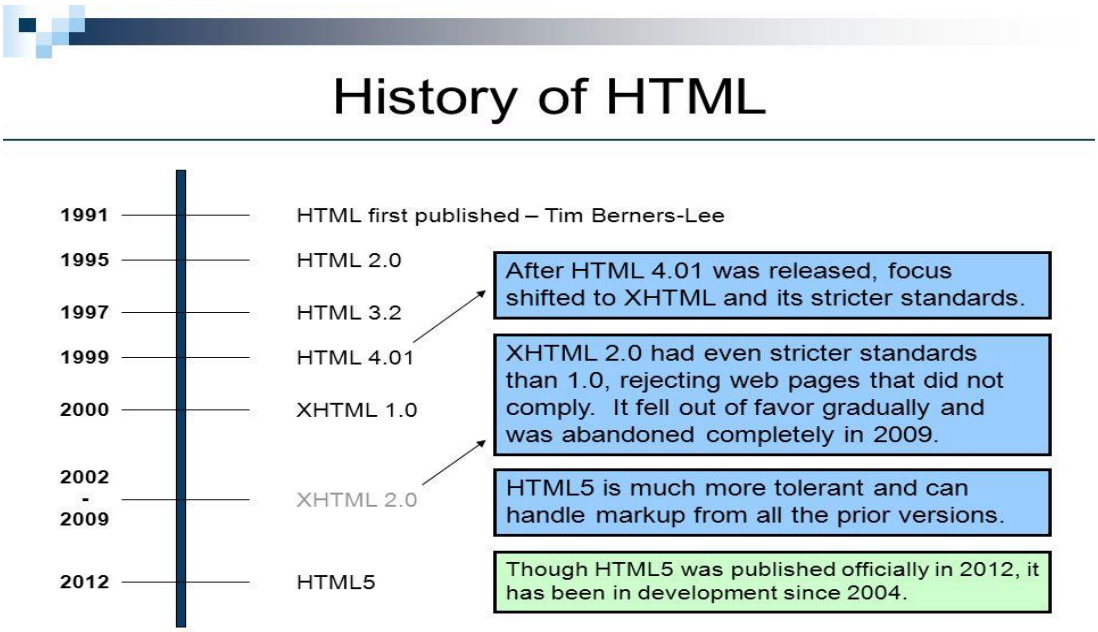
Database**:** Database is the collection of inter-related data which helps in efficient retrieval, insertion and deletion of data from database and organizes the data in the form of tables, views, schemas, reports etc.

- **Oracle:** Oracle database is the collection of data which is treated as a unit. The purpose of this database is to store and retrieve information related to the query. It is a database server and used to manages information.
- MongoDB**:** MongoDB, the most popular NoSQL database, is an open source document-oriented database. The term 'NoSQL' means 'non-relational'. It means that MongoDB isn't based on the table-like relational database structure but provides an altogether different mechanism for storage and retrieval of data.
- Sql**:** Structured Query Language is a standard Database language which is used to create, maintain and retrieve the relational database.

Full Stack Developer is "jack of all trade, master on none".

Getting Started with HTML

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.
- HTML **elements** are the building blocks of HTML pages.
- HTML elements are represented by <> **tags.**
- **An HTML file is simply a text file saved with an .html or .htm extension.**



Creating First HTML Document:

Step 1: Creating the HTML file

Open up your computer's plain text editor and create a new file.

Step 2: Type some HTML code

Start with an empty window and type the following code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>A simple HTML document</title>
</head>
<body>
   <p>Hello World!<p>
</body>
</html>
```

Step 3: Saving the file

- Now save the file on your desktop as "myfirstpage.html ".
- To open the file in a browser. Navigate to your file then double click on it. It will open in your default Web browser. If it does not, open your browser and drag the file to it.
- The first line `<!DOCTYPE html>` is the document type declaration. It instructs the web browser that this document is an HTML5 document. It is case-insensitive.

- The <head> element is a container for the tags that provides information about the document, for example, <title> tag defines the title of the document.
- The <body> element contains the document's actual content (paragraphs, links, images, tables, and so on) that is rendered in the web browser and displayed to the user.

HTML Tags and Elements:

HTML is written in the form of HTML elements consisting of markup tags. These markup tags are the fundamental characteristic of HTML. Every markup tag is composed of a keyword, surrounded by angle brackets, such as `<html>`, `<head>`, `<body>`, `<title>`, `<p>`, and so on.

HTML tags normally come in pairs like `<html>` and `</html>`. The first tag in a pair is often called the opening tag (or start tag), and the second tag is called the closing tag (or end tag).

An opening tag and a closing tag are identical, except a slash (`/`) after the opening angle bracket of the closing tag, to tell the browser that the command has been completed.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>A simple HTML document</title>
</head>
<body>
   <p>This is a paragraph.</p>
   <!-- Paragraph with nested element -->
   <p>
      This is <b>another</b> paragraph.
   </p>
</body>
</html>
```

HTML TAGS:

1.	<html>	</html>	[HTML open / close tags]
2.	<h1>	</h1>	[Heading 1 open / close tags]
3.	<h2>	</h2>	[Heading 2 open / close tags]
4.	<h3>	</h3>	[Heading 3 open / close tags]
5.	<h4>	</h4>	[Heading 4 open / close tags]
6.	<h5>	</h5>	[Heading 5 open / close tags]
7.	<h6>	</h6>	[Heading 6 open / close tags]
8.	<title>	</title>	[Title open/ close tags]

9.   [Center alignment open/close tags]
10. <body> </body> [Body open/ close tags]
11. <body bgcolor=”colorname”>
12. <body background=”image.jpg”>
13. <input type=”text” > [input type tags]
14. <input type=”number”>
15. <input type=”radio”>
16. <input type=”checkbox”>
17. <input type=”password”>
18. <input type=”button”>
19. <input type=”submit”>
20. <input type=”reset”>
21. <input type=”date”>
22. <input type=”email”>
23. <input type=”file”>
24. <input type=”color”>
25. <img src=”name.jpg”>
26. <marquee> Type text </marquee>
27.   [Paragraph tag]
28. <q> </q> [Quotation tag]
29. <table> </table> [Table tag]
30. <tr> </tr> [Table row]
31. <td> </td> [Table data]
32.   [Unordered List]
33.   [List]
34. <u> </u> [Underline]
35.
 </br> [Line Break]
36. <hr> [horizontal row]
37. <del> </del> [Strikethrough]
38. [Super Script]
39. [Sub Script]

40. <select> [Drop down List tag]

<option value="Volvo">Volvo</option>

<option value="Saab">Saab</option>

<option value="Mercedes">Mercedes</option>

<option value="Audi">Audi</option>

</select>

41. Click Here [anchor tag]

42. This is some text! [Font tags]

This is some text!

This is some text!

43. <textarea rows="4" cols="50"> ……..</textarea> [Text area]

HTML 5

- HTML5 is the latest and most enhanced version of HTML. Technically, HTML is not a programming language, but rather a markup language.
- HTML5 is the next major revision of the HTML standard superseding HTML 4.01, XHTML 1.0, and XHTML 1.1. HTML5 is a standard for structuring and presenting content on the World Wide Web.
- HTML5 is a cooperation between the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG).
- The new standard incorporates features like video playback and drag-and-drop that have been previously dependent on third-party browser plug-ins such as Adobe Flash, Microsoft Silverlight, and Google Gears.

The DOCTYPE:

DOCTYPEs in older versions of HTML were longer because the HTML language was SGML based and therefore required a reference to a DTD.

DOCTYPE declaration appears at the top of a web page before all other elements. According to the HTML specification or standards, every HTML document requires a valid document type declaration to insure that your web pages are displayed the way they are intended to be displayed.

syntax:

<!DOCTYPE html>

- The above syntax is case-insensitive.

HTML Meta:

- The `<meta>` tags are typically used to provide structured metadata such as a document's *keywords*, *description*, *author name*, *character encoding*, and other metadata. Any number of meta tags can be placed inside the head section of an HTML or XHTML document.

- Metadata will not be displayed on the web page, but will be machine parsable, and can be used by the browsers, search engines like Google or other web services.

syntax:

<meta charset = "UTF-8">

The above syntax is case-insensitive.

Declaring Character Encoding in HTML:

```
<!DOCTYPE html>
<html lang="en">
 <head>
 <title>Declaring Character Encoding</title>
 <meta charset="utf-8">
 </head>
<body>
 <h1>Hello World!</h1>
 </body>
```

</html>

The <script> tag

It's common practice to add a type attribute with a value of "text/javascript" to script elements as follows −

<script type = "text/javascript" src = "scriptfile.js"></script>

HTML 5 removes extra information required and you can use simply following syntax −

<script src = "scriptfile.js"></script>

The <link> tag

So far we were writing <link> as follows −

<link rel = "stylesheet" type = "text/css" href = "stylefile.css">

HTML 5 removes extra information required and you can simply use the following syntax −

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

HTML 5 Tags:

There is a list of newly included tags in HTML 5. These HTML 5 tags (elements) provide a better document structure. This list shows all HTML 5 tags in alphabetical order with description.

List of HTML 5 Tags

Tag	Description
<article>	This element is used to define an independent piece of content in a document, that may be a blog, a magazine or a newspaper article.

<aside>	It specifies that article is slightly related to the rest of the whole page.
<audio>	It is used to play audio file in HTML.
<bdi>	The bdi stands for bi-directional isolation. It isolates a part of text that is formatted in other direction from the outside text document.
<canvas>	It is used to draw canvas.
<data>	It provides machine readable version of its data.
<datalist>	It provides auto complete feature for textfield.
<details>	It specifies the additional information or controls required by user.
<dialog>	It defines a window or a dialog box.
<figcaption>	It is used to define a caption for a <figure> element.
<figure>	It defines a self-contained content like photos, diagrams etc.
<footer>	It defines a footer for a section.
<header>	It defines a header for a section.
<main>	It defines the main content of a document.
<mark>	It specifies the marked or highlighted content.
<menuitem>	It defines a command that the user can invoke from a popup menu.

<meter>	It is used to measure the scalar value within a given range.
<nav>	It is used to define the navigation link in the document.
<progress>	It specifies the progress of the task.
<rp>	It defines what to show in browser that don't support ruby annotation.
<rt>	It defines an explanation/pronunciation of characters.
<ruby>	It defines ruby annotation along with <rp> and <rt>.

Difference between HTML and HTML5:

HTML	HTML5
HTML Doctype declaration is lengthy.	DOCTYPE declaration in HTML5 is simple.
HTML Character encoding is longer.	HTML5 Character encoding declaration is simple.
Audio and video are not HTML parts.	Audio and video are HTML5 part.
It is possible to draw a vector with the help of other technologies like Silverlight, Flash, VML, etc.	Vector graphics are a part of HTML5, e.g., canvas, SVG.

It is impossible to get the actual Geolocation of a person browsing any website.	JS Geolocation API in HTML5 enables you to identify the location of the user browsing any website.
HTML offers local storage instead of cookies.	Html5 uses cookies to store data.
In HTML, it is not possible to draw basic shapes.	In Html5, it is possible to draw basic shapes.
It allows you to run JavaScript in a browser.	It enables you to run JavaScript code in the background.
You can use HTML with all old browsers.	You can use HTML5 with all new browsers.
You can use browser cache as temporary storage.	You can use application (database and web storage) Cache as temporary storage.
Web Socket is not available.	You can establish full-duplex communication channels with a server using Web Sockets.
There is no process to handlestructurally incorrect HTML codes.	HTML5 supports persistent error handling via the improvised error handling process.
HTML is less mobile-friendly.	HTML5 is mobile friendly.
Attributes like async, charset, and ping are not present in HTML.	Attributes of async, ping, charset, and are a part of HTML5.
HTML does not allow drag and drop effects	HTML5 allows drag and drop effects.
Offer new attributes like tabinex, id, tabinex, etc.	These are certain attributes which are applied to HTML 5 elements.

HTML Video:

The HTML `<video>` element is used to show a video on a web page. HTML 5 supports <video> tag also. The HTML video tag is used for streaming video files such as a movie clip, song clip on the web page.
Currently, there are three video formats supported for HTML video tag:

1. mp4
2. webM
3. ogg

HTML Video Tag Example:

Let's see the code to play mp4 file using HTML video tag.

```
<!DOCTYPE>
<html>
<body>
<video controls>
 <source src="movie.mp4" type="video/mp4">
</video>
</body>
</html>
```

Attributes of HTML Video Tag:

Let's see the list of HTML 5 video tag attributes.

Attribute	Description
controls	It defines the video controls which is displayed with play/pause buttons.
height	It is used to set the height of the video player.
width	It is used to set the width of the video player.
poster	It specifies the image which is displayed on the screen when the video is not played.
autoplay	It specifies that the video will start playing as soon as it is ready.

loop	It specifies that the video file will start over again, every time when it is completed.
muted	It is used to mute the video output.
preload	It specifies the author view to upload video file when the page loads.
src	It specifies the source URL of the video file.

HTML Video Tag Attribute Example

Let's see the example of video tag in HTML where are using height, width, autoplay, controls and loop attributes.

```
<!DOCTYPE>
<html>
<body>
<video width="320" height="240" controls autoplay loop>
  <source src="movie.mp4" type="video/mp4">
</video>
</body>
</html>
```

Output:

HTML5 Audio Tag:

HTML audio tag is used to define sounds such as music and other audio clips. Currently there are three supported file format for HTML 5 audio tag.

1. mp3
2. wav
3. ogg

HTML5 supports <video> and <audio> controls. The Flash, Silverlight and similar technologies are used to play the multimedia items.

HTML Audio Tag Example:

```
<!DOCTYPE>

<html>

<body>

<audio controls>

  <source src="koyal.mp3" type="audio/mpeg">

</audio>

</body>

</html>
```

Attributes of HTML Audio Tag:

There is given a list of HTML audio tag.

Attribute	Description
controls	It defines the audio controls which is displayed with play/pause buttons.
autoplay	It specifies that the audio will start playing as soon as it is ready.
loop	It specifies that the audio file will start over again, every time when it is completed.
muted	It is used to mute the audio output.
preload	It specifies the author view to upload audio file when the page loads.
src	It specifies the source URL of the audio file.

HTML Audio Tag Attribute Example:

Here we are going to use controls, autoplay, loop and src attributes of HTML audio tag.

```
<!DOCTYPE>
<html>
<body>
<audio controls autoplay loop>
 <source src="koyal.mp3" type="audio/mpeg"></audio>
</body>
</html>
```

HTML Canvas Tag:

The HTML element is used to draw graphics on the fly, via scripting (usually JavaScript). The element is only a container for graphics. You must use a script to actually draw the graphics. Canvas has several methods for drawing paths, boxes, circles, text, and adding images.

How to create a HTML canvas?

A canvas is a rectangle like area on an HTML page. It is specified with canvas element. By default, the <canvas> element has no border and no content, it is like a container.

```
<canvas id = "mycanvas" width ="200" height ="100"> </canvas>
```

HTML 5 Canvas Tag Example:

```
<!DOCTYPE>
<html>
<body>
<canvas id="myCanvas" width="300" height="200" style="border:2px solid;">
</canvas>
</body>
</html>
```

Output:

HTML Canvas Tag with JavaScript

A canvas is a two dimensional grid.

Coordinates (0,0) defines the upper left corner of the canvas. The parameters (0,0,200,100) is used for fillRect() method. This parameter will fill the rectangle start with the upper-left corner (0,0) and draw a 200 * 100 rectangle.

```
<canvas id="myCanvas" width="250" height="150" style="border:1px solid #c3c3c3;">
</canvas>
<script>
var c = document.getElementById("myCanvas");
```

```
var cctx = c.getContext("2d");
ctx.fillStyle = "#FF0000";
ctx.fillRect(0,0,200,100);
</script>
```

Drawing Line on Canvas:

If you want to draw a straight line on the canvas, you can use the following two methods.

moveTo(x,y): It is used to define the starting point of the line.

lineTo(x,y): It is used to define the ending point of the line.

If you draw a line which starting point is (0,0) and the end point is (200,100), use the stroke method to draw the line.

```
<canvas id="myCanvasLine" width="200" height="100" style="border:1px solid #d3d3d3;">
</canvas>
<script>
var c = document.getElementById("myCanvasLine");
var cctx = c.getContext("2d");
ctx.moveTo(0,0);
ctx.lineTo(200,100);
ctx.stroke();
</script>
```

SVG

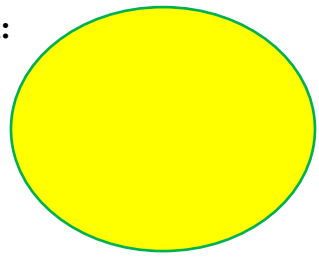
The Scalable Vector Graphics (SVG) is an XML-based image format that is used to define two-dimensional vector-based graphics for the web. Unlike raster image (Ex .jpg, .gif, .png, etc.), a vector image can be scaled up or down to any extent without losing the image quality. An SVG image is drawn out using a series of statements that follow the XML schema — that means SVG images can be created and edited with any text editor, such as Notepad. There are several other advantages of using SVG over other image formats like JPEG, GIF, PNG, etc.

- SVG images can be searched, indexed, scripted, and compressed.
- SVG images can be created and modified using JavaScript in real time.
- SVG images can be printed with high quality at any resolution.
- SVG content can be animated using the built-in animation elements.
- SVG images can contain hyperlinks to other documents.

HTML SVG Circle Example:

```
<!DOCTYPE html>
<html>
<body>
  <svg width="100" height="100">
   <circle cx="50" cy="50" r="40" stroke="yellow" stroke-width="4" fill="red" />
  </svg>
</body>
</html>
```

Output:



HTML SVG Rectangle:

```
<!DOCTYPE html>
<html>
<body>
<svg width="200" height="100">
  <rect width="200" height="100" stroke="yellow" stroke-width="4" fill="red" />
</svg>
</body>
</html>
```

output:

HTML SVG polygon Example:

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

```
<body>
  <svg height="210" width="500">
 <polygon points="100,10 40,198 190,78 10,78 160,198"
  style="fill:red;stroke:yellow;stroke-width:5;fill-rule:nonzero;" />
</svg>
</body>
</html>
```

Difference between SVG And Canvas

SVG	Canvas
Vector based (composed of shapes)	Raster based (composed of pixel)
SVG has better scalability. So it can be printed with high quality at any resolution.	Canvas has poor scalability. Hence it is not suitable for printing on higher resolution.
SVG gives better performance with smaller number of objects or larger surface.	Canvas gives better performance with smaller surface or larger number of objects.
SVG can be modified through script and CSS.	Canvas can be modified through script only.
Multiple graphical elements, which become the part of the page's DOM tree.	Single element similar to <img> in behavior. Canvas diagram can be saved to PNG or JPG format.

Web Storage

The Web Storage is one of the great features of HTML5. With the Web Storage feature, web applications can locally store data within the browser on the client side. It stores data in the form of key/value pair on the browser. Web Storage sometimes also known as DOM storage. Storing data with the help of web storage is similar to cookies, but it is better and faster than cookies storage.

In compared to cookies Web Storage has Following Advantages:

- Web Storage can use storage space upto 5MB per domain. (The browser software may prompt the user if the space limit is reached).
- It will not send data to the server side, hence it is faster than cookies storage.
- The data stored by local Storage never expires, but cookies data expires after some time or session.
- Web Storage is more secure than cookies.

Types of Web Storage:

There are two types of web storage with different scope and lifetime.

- Local Storage: Local Storages uses Windows.localStaorage object which stores data and available for every page. But data persist even if the browser is closed and reopened (Stores data with no Expiration).
- Session Storage: Session Storage uses Windows.sessionStorage object which stores data for one session and data will be lost if the window or browser tab will be closed.

1.HTML Web Storage Objects

HTML web storage provides two objects for storing data on the client:

- Window. LocalStorage - stores data with no expiration date
- Window. SessionStorage - stores data for one session (data is lost when the browser tab is closed)

Before using web storage, check browser support for localStorage and sessionStorage:

```
if (typeof(Storage) !== "undefined") {
  // Code for localStorage/sessionStorage.
} else {
  // Sorry! No Web Storage support..
}
```

1. The localStorage Object:

The localStorage object stores the data with no expiration date. The data will not be deleted when the browser is closed, and will be available the next day, week, or year.

2. The sessionStorage Object:

The sessionStorage object is equal to the localStorage object, **except** that it stores the data for only one session. The data is deleted when the user closes the specific browser tab.

2. Remove Web Storage:

As we have seen the session storage data will automatically be deleted, when you close the browser but the data saved by local storage will remain in the browser even if you close it.

Hence to delete the local storage data, you need to call two methods:

1. **localStorage.removeItem ('key'):** If you want to delete the value on a particular key, then you can use the "key," and that value will be deleted.
2. **localStorage.clear ():** If you want to delete or clear all settings with key/value pair, then you can call this method.

Example-1:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Storing Data with HTML5 Local Storage</title>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
<script>
// Check if the localStorage object exists
if(localStorage) {
$(document).ready(function() {
	$(".save").click(function() {
		// Get input name
		var firstName = $("#firstName").val();

		// Store data
		localStorage.setItem("first_name", firstName);
		alert("Your first name is saved.");
	});
	$(".access").click(function() {
		// Retrieve data
		alert("Hi, " + localStorage.getItem("first_name"));
	});
});
} else {
   alert("Sorry, your browser do not support local storage.");
}
</script>
```

```
</head>
<body>
   <form>
         <label>First Name: <input type="text" id="firstName"></label>
      <button type="button" class="save">Save Name</button>
      <button type="button" class="access">Get Name</button>
   </form>
</body>
</html>
```

HTML Drag and Drop.

HTML Drag and Drop (DnD) is a feature of HTML5. It is a powerful user interface concept which *is used to copy, reorder and delete items with the help of mouse*. You can hold the mouse button down over an element and drag it to another location. If you want to drop the element there, just release the mouse button.

Drag and drop is a very common feature. It is when you "grab" an object and drag it to a different location. HTML 5 DnD is supported by all the major browsers like Chrome, Firefox 3.5 and Safari 4.

Drag and Drop Events:

There are number of events which are fired during various stages of the drag and drop operation. These events are listed below:

Sl.No.	Events & Description
1	**dragstart** Fires when the user starts dragging of the object.
2	**drag enter** Fired when the mouse is first moved over the target element while a drag is occurring. A listener for this event should indicate whether a drop is allowed over this location. If there are no listeners, or the listeners perform no operations, then a drop is not allowed by default.
3	**drag over** This event is fired as the mouse is moved over an element when a drag is occurring. Much of the time, the operation that occurs during a listener will be the same as the drag enter event.

4	**Drag leave** This event is fired when the mouse leaves an element while a drag is occurring. Listeners should remove any highlighting or insertion markers used for drop feedback.
5	**Drag** Fires every time the mouse is moved while the object is being dragged.
6	**Drop** The drop event is fired on the element where the drop was occurred at the end of the drag operation. A listener would be responsible for retrieving the data being dragged and inserting it at the drop location.
7	**Drag end** Fires when the user releases the mouse button while dragging an object.

The DataTransfer Object

The event listener methods for all the drag and drop events accept **Event** object which has a readonly attribute called **dataTransfer**.

The **event.dataTransfer** returns **DataTransfer** object associated with the event as follows −

```
function EnterHandler(event) {
  DataTransfer dt = event.dataTransfer;
  ..............
}
```

Drag and Drop Process

Following are the steps to be carried out to implement Drag and Drop operation −

Step 1 - Making an Object Draggable:

Here are steps to be taken −

- If you want to drag an element, you need to set the **draggable** attribute to **true** for that element.
- Set an event listener for **dragstart** that stores the data being dragged.
- The event listener **dragstart** will set the allowed effects (copy, move, link, or some combination).

Following is the example to make an object draggable −

```
<!DOCTYPE HTML>
<html>
  <head>
   <style type = "text/css">
      #boxA, #boxB {
        float:left;padding:10px;margin:10px; -moz-user-select:none;
      }
      #boxA { background-color: #6633FF; width:75px; height:75px;  }
      #boxB { background-color: #FF6699; width:150px; height:150px; }
    </style>
    <script type = "text/javascript">
       function dragStart(ev) {
        ev.dataTransfer.effectAllowed = 'move';
        ev.dataTransfer.setData("Text", ev.target.getAttribute('id'));
        ev.dataTransfer.setDragImage(ev.target,0,0);
       return true;
      }
    </script>
    </head>
  <body>
     <center>
      <h2>Drag and drop HTML5 demo</h2>
      <div>Try to drag the purple box around.</div>
       <div id = "boxA" draggable = "true"
        ondragstart = "return dragStart(ev)">
        <p>Drag Me</p>
      </div>
        <div id = "boxB">Dustbin</div>
    </center>
   </body>
</html>
```

Output:

Drag and drop HTML5 demo

Try to drag the purple box around.

Drag Me

Dustbin

Step 2 - Dropping the Object:

To accept a drop, the drop target has to listen to at least three events.

- The **dragenter** event, which is used to determine whether or not the drop target is to accept the drop. If the drop is to be accepted, then this event has to be canceled.
- The **dragover** event, which is used to determine what feedback is to be shown to the user. If the event is canceled, then the feedback (typically the cursor) is updated based on the dropEffect attribute's value.
- Finally, the **drop** event, which allows the actual drop to be performed.

Following is the example to drop an object into another object −

```
<html>
  <head>
    <style type="text/css">
      #boxA, #boxB {
        float:left;padding:10px;margin:10px;-moz-user-select:none;
      }
      #boxA { background-color: #6633FF; width:75px; height:75px;  }
      #boxB { background-color: #FF6699; width:150px; height:150px; }
    </style>
    <script type="text/javascript">
      function dragStart(ev) {
        ev.dataTransfer.effectAllowed='move';
        ev.dataTransfer.setData("Text", ev.target.getAttribute('id'));
        ev.dataTransfer.setDragImage(ev.target,0,0);
        return true;
      }
      function dragEnter(ev) {
        event.preventDefault();
        return true;
      }
      function dragOver(ev) {
        return false;
      }
      function dragDrop(ev) {
        var src = ev.dataTransfer.getData("Text");
        ev.target.appendChild(document.getElementById(src));
        ev.stopPropagation();
        return false;
      }
    </script>
  </head>
```

```
  <body>
    <center>
      <h2>Drag and drop HTML5 demo</h2>
      <div>Try to move the purple box into the pink box.</div>
      <div id="boxA" draggable="true" ondragstart="return dragStart(event)">
        <p>Drag Me</p>
      </div>
      <div id="boxB" ondragenter="return dragEnter(event)" ondrop="return dragDrop(event)"
ondragover="return dragOver(event)">Dustbin</div>
    </center>
  </body>
</html>
```

Output:

Drag and drop HTML5 demo

Try to move the purple box into the pink box.

Drag Me

Dustbin

Stages during Drag and Drop operations:

1) Make an element draggable:

If you want to make an element draggable, set the draggable attribute to "true" on the element. For example:

```
<img draggable = "true">
```

2) What to drag:

Use ondragstart and setData () methods.

Specify what should happen when the element is dragged.

3) Where to Drop:

Use ondragover event.

4) Do the Drop:

Use ondrop event.

HTML5 Geo location

The Geolocation is one of the best HTML5 API which is used to identify the user's geographic location for the web application. This new feature of HTML5 allows you to navigate the latitude and longitude coordinates of the current website's visitor. These coordinates can be captured by JavaScript and send to the server which can show your current location on the website.Most of the geolocation services use Network routing addresses such as IP addresses, RFID, WIFI and MAC addresses or internal GPS devices to identify the user's location.

User privacy:

The user's location is the privacy concern, so geolocation API protects the user's privacy by taking the user's permission before getting the location. Geolocation API sends a notification prompt box which user can allow or deny, and if the user allows then only his location will be identified.

Locate the User's Position:

The HTML Geolocation API is used to get the geographical position of a user.

Since this can compromise privacy, the position is not available unless the user approves it.

Geolocation object

The Geolocation API is work with the navigation. Geolocation object. Its read-only property returns a Geolocation object which identifies the location of the user and can generate a customized result based on user location.

Syntax:

geo=navigator. geolocation;

If this object is present, then you can get the geolocation services.

Geolocation Methods:

The Geolocation API uses three methods of Geolocation interface which are given following:

Methods	Description
getCurrentPosition()	It identifies the device or the user's current location and returns a position object with data.
watchPosition()	Return a value whenever the device location changes.
clearWatch()	It cancels the previous watchPosition() call

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Geolocation API</title>
</head>
<body>
<h1>Find your Current location</h1>
<button onclick="getlocation()">Click me</button>
<div id="location"></div>
<script>
var x= document.getElementById("location");
function getlocation() {
if(navigator.geolocation){
navigator.geolocation.getCurrentPosition(showPosition)
}
else
{
alert("Sorry! your browser is not supporting")
}
}
function showPosition(position){
var x = "Your current location is (" + "Latitude: " + position.coords.latitude + ", " + "Longitude:
" +    position.coords.longitude + ")";
document.getElementById("location").innerHTML = x;
}
</script>
</body>
</html>
```

What is CSS?

Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable. CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, as well as a variety of other effects. CSS is easy to learn and understand but it provides a powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

Advantages of CSS

- **CSS saves time -** You can write CSS once and then reuse the same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many web pages as you want.
- **Pages load faster -** If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So, less code means faster download times.
- **Easy maintenance -** To make a global change, simply change the style, and all the elements in all the web pages will be updated automatically.
- **Superior styles to HTML -** CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- **Multiple Device Compatibility -** Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cellphones or for printing.
- **Global web standards –** Now HTML attributes are being deprecated and it is being recommended to use CSS. So it's a good idea to start using CSS in all the HTML pages to make them compatible with future browsers.

Basic Styling in CSS

CSS can either be attached as a separate document or embedded in the HTML document itself. There are three methods of including CSS in an HTML document:

1. Inline styles — Using the style attribute in the HTML start tag.
2. Internal Style sheets — Using the <style> element in the head section of a document.
3. External style sheets — Using the <link> element, pointing to an external CSS file.

i) Inline Styles:

Inline styles are used to apply the unique style rules to an element by putting the CSS rules directly into the start tag. It can be attached to an element using the style attribute. The style attribute includes a series of CSS property and value pairs. Each "property: value" pair is separated by a semicolon (;), just as you would write into an embedded or external style sheets. But it needs to be all in one line i.e. no line break after the semicolon, as shown here:

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
  <title>Example of CSS Inline Styles</title>
</head>
<body>
<h1 style="color:red; font-size:30px;">This is a heading</h1>
```

```
<p style="color:green; font-size:22px;">This is a paragraph.</p>
<div style="color:blue; font-size:14px;">This is some text content.</div>
</body>
</html>
```

ii) Internal Style Sheets:

Embedded or internal style sheets only affect the document they are embedded in. Embedded style sheets are defined in the <head> section of an HTML document using the <style> element. You can define any number of <style> elements in an HTML document but they must appear between the <head> and </head> tags.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
   <title>Example of CSS Embedded Style Sheet</title>
   <style>
      body { background-color: YellowGreen; }
      p { color: #fff; }
   </style>
</head>
<body>
   <h1>This is a heading</h1>
   <p>This is a paragraph of text.</p>
</body>
</html>
```

iii) External Style Sheets:

An external style sheet is ideal when the style is applied to many pages of the website. An external style sheet holds all the style rules in a separate document that you can link from any HTML file on your site. External style sheets are the most flexible because with an external style sheet, you can change the look of an entire website by changing just one file.
An external style sheet can be linked to an HTML document using the <link> tag. The <link> tag goes inside the <head> section.

Example:

```
<!DOCTYPE html>
<html  lang="en">
<head>
<meta charset="utf-8">
   <title>Example of CSS External Style Sheet</title>
```

```
   <link rel="stylesheet" href="/examples/css/style.css">
</head>
<body>
   <h1>This is a heading</h1>
   <p>This is a paragraph of text.</p>
</body>
</html>
```

Positioning And Background Images

CSS property position helps manipulate position of an element in a web page. The properties top, bottom, right, and left are used to control its exact position on the page. They specify the offsets of an element from its edges Property position can be used to create floating elements, floating sidebar, and other interactive features.

Following are the possible values:

1. Static − When you use the CSS position: static property, the element will be positioned normally in the document flow. The left, right, top, and bottom properties will not affect the element. This is the default value.

Example:

```
<html>
<head>
<style>
  .normal-box {
    display: inline-block;
    background-color: #f2c3ee;
    border: 1px solid #000000;
    padding: 10px;
    margin-bottom: 20px;
    width: 300px;
    height: 100px;
  }
  .static-box {
    display: inline-block;
    position: static;
    background-color: #bbedbb;
    border: 1px solid #000000;
    padding: 10px;
    width: 300px;
    height: 100px;
  }
</style>
</head>
```

```
<body>
  <div class="normal-box">
    <h2>Normal Box</h2>
    <p>This is a normal box.</p>
  </div>

  <div class="static-box">
    <h2>Position: static</h2>
    <p>This is a box with static position.</p>
  </div>
</body>
</html>
```

2. Relative − The CSS position: relative property positions the elements relative to their original position in the page. You can use the left, right, top, and bottom properties to move the element around, but it will still take up space in the document flow.

```
<html>
<head>
<style>
  .normal-box {
    display: inline-block;
    background-color: #f2c3ee;
    border: 1px solid #000000;
    padding: 10px;
    margin-bottom: 20px;
    width: 300px;
    height: 100px;
  }
  .relative-box {
    display: inline-block;
    position: relative;
    left: 30px;
    background-color: #bbedbb;
    border: 1px solid #000000;
    padding: 10px;
    width: 300px;
    height: 100px;
  }
</style>
</head>
<body>
  <div class="normal-box">
    <h2>Normal Box</h2>
    <p>This is a normal box.</p>
  </div>
```

```
  <div class="relative-box">
    <h2>Position: relative</h2>
    <p>This is a box with relative position.</p>
  </div>
</body>
</html>
```

3. Absolute − An element with position: absolute is taken out of the document flow and positioned relative to its nearest positioned ancestor (if any). If there is no positioned ancestor, then the element is positioned relative to the viewport. You can use top, right, bottom, and left properties to specify the position of the element relative to its containing block.

```
<html >
<head>
<style>
  .normal-box {
    background-color: #f2c3ee;
    border: 1px solid #333;
    padding: 10px;
    margin-bottom: 20px;
    width: 350px;
    height: 100px;
  }
  .absolute-box {
    background-color: #bbedbb;
    border: 1px solid #333;
    padding: 10px;
    position: relative;
    width: 300px;
    height: 100px;
    left: 20px;
    bottom: 20px;
  }
</style>
</head>
<body>
  <div class="normal-box">
    <h2>Normal Box</h2>
    <p>This is a Noraml box.</p>
    <div class="absolute-box">
      <h2>Position: Absolute</h2>
      <p>This is a box with absolute position.</p>
    </div>
  </div>
</body>
</html>
```

4. Fixed − To make an element stay in the same place on the screen even when the user scrolls, you can set the position property to fixed. You can then use the left, right, top, and bottom properties to position the element where you want it.

```
<html>
<head>
<style>
  .position_container {
    width: 400px;
    height: 200px;
    background-color: #f2c3ee;
    overflow: auto;
    padding: 5px;
  }
  .fixed-position {
    position: fixed;
    top: 15px;
    left: 60px;
    padding: 5px;
    background-color: #bbedbb;
    text-align: center;
  }
</style>
</head>
<body>
  <div class="position_container">
    <p>"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
    "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
    <p class="fixed-position">Tutorialspoint CSS Position Fixed</p>
    <p>"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
  </div>
</body>
</html>
```

5. Sticky − You can set the position property to sticky to create an element that sticks to the top of the viewport when the user scrolls through a page. The position: sticky property is a combination of the position: relative and position: fixed properties.

Example:

```
<html>
<head>
<style>
  .position_container {
    width: 400px;
    height: 200px;
    background-color: #f2c3ee;
    overflow: auto;
    padding: 5px;
  }
  .sticky-position {
    position: sticky;
    top: 15px;
    padding: 5px;
    background-color: #bbedbb;
    text-align: center;
  }
</style>
</head>
<body>
  <div class="position_container">
    <p>"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
    "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
    <p class="sticky-position">Tutorialspoint CSS Position Sticky</p>
    <p>"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
    <p>"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
    <p>"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
    <p>"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
  </div>
</body>
</html>
```

Background Images:

The CSS3 provides several new properties to manipulate the background of an element like background clipping, multiple backgrounds, and the option to adjust the background size.

i) CSS3 background-size Property:

The background-size property can be used to specify the size of the background images. Prior to CSS3, the size of the background images was determined by the actual size of the images. The background image size can be specified using the pixels or percentage values as well as the keywords auto, contain, and cover. Negative values are not allowed.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of Setting background-size of an Element</title>
<style>
.box {
        width: 250px;
        height: 150px;
        background: url("/examples/images/sky.jpg") no-repeat;
        background-size: contain;
        border: 6px solid #333;
}
</style>
</head>
<body>
   <div class="box"></div>
   <p><strong>Note:</strong> The original size of the background image is 500x300 pixels, but using the background-size CSS property we are still able to show the complete image inside the smaller box.</p>
</body>
</html>
```

ii) CSS3 background-clip Property:

The background-clip property can be used to specify whether an element's background extends into the border or not. The background-clip property can take the three values: border-box, padding-box, content-box.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of CSS3 Background Clipping</title>
<style>
.box {
        width: 250px;
```

```
        height: 150px;
        padding: 10px;
        border: 6px dashed #333;
        background: orange;
}
.clip1 {
        background-clip: border-box;
}
.clip2 {
        background-clip: padding-box;
}
.clip3 {
        background-clip: content-box;
}
</style>
</head>
<body>
   <h2>Default Background Behavior</h2>
   <div class="box"></div>
   <h2>Background Clipping Using border-box</h2>
   <div class="box clip1"></div>
   <h2>Background Clipping Using padding-box</h2>
   <div class="box clip2"></div>
   <h2>Background Clipping Using content-box</h2>
   <div class="box clip3"></div>
</body>
</html>
```

iii) CSS3 background-origin Property:

The background-origin property can be used to specify the positioning area of the background images. It can take the same values as background-clip property: border-box, padding-box, content-box.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of Setting background-origin of an Element</title>
<style>
.box {
        width: 250px;
        height: 150px;
      padding: 10px;
      border: 6px dashed #333;
        background: url("/examples/images/sky.jpg") no-repeat;
        background-size: contain;
      background-origin: content-box;
}
</style>
</head>
```

```
<body>
   <div class="box"></div>
</body>
</html>
```

iv) CSS3 Multiple Backgrounds:

CSS3 gives you ability to add multiple backgrounds to a single element. The backgrounds are layered on the top of one another. The number of layers is determined by the number of comma-separated values in the background-image or background shorthand property.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of CSS3 Multiple Backgrounds</title>
<style>
.box {
width: 100%;
height: 500px;
background: url("/examples/images/birds.png") no-repeat center,  url("/examples/images/clouds.png")  no-repeat center, url("/examples/images/sun.png")  no-repeat 10% 30%, lightblue;
}
</style>
</head>
<body>
   <div class="box">
</div>
</body>
</html>
```

Bootstrap

Bootstrap: Bootstrap 4 is the newest version of Bootstrap. **Bootstrap** can be defined as a free and open-source framework that can be used to create responsive, mobile-first, front-end web pages.

Advantages of Bootstrap:

1. The first and foremost advantage of using Bootstrap is that it is very easy to use and implement. If a person has some basic knowledge of HTML and CSS, that user can easily use Bootstrap.
2. The fact that Bootstrap can adapt to the size of any phone, tablet, desktop and so on is also very interesting feature.
3. Bootstrap 4 is also useful because it is compatible with all modern browsers which include Google Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera.
4. It also produces less cross-browser bugs.
5. It is light weighted and consequently it can be widely used as a framework for creating responsive sites.

6. Lastly, Bootstrap 4 is a very simple and yet very effective grid system.

Usages o-f Bootstrap 4 :

1. **Supported by various Browsers**: It can be supported by every browser.
2. **Simple to start and implement**: It is very easy to start and implement when the user has a fair amount of knowledge about HTML and CSS. In addition to that, the documentation is readily available on the official site.
3. **Responsive design and looks**: The web pages that are created by using the Bootstrap framework are responsive and it can adapt to any screen size like mobile, desktop, etc.
4. **Easily Customized**: It also provides some built-in components and functionalities that can be used for the purpose of easily customizing the web pages.

Disadvantages:

1. In many cases, Bootstrap cannot be considered very practical for businesses that need a big amount of investment.

2. Moreover, Bootstrap 4 can take a lot of time to create a website. Therefore, it is not a very bright idea to use bootstrap 4 when there is no investment.

3. A person using Bootstrap 4 is not likely to earn any money even after investment.

4. And this is one of the reasons why a user can very easily end up in a lot of debt.

Bootstrap Environment Setup

Bootstrap 4 is the latest version of Bootstrap. In this section, it is shown how to download and setup Bootstrap 4. There are two ways of using Bootstrap 4 in the Website:

1. The first method that can be used is to start using Bootstrap 4 by inserting it from the CDN which stands for **Content Delivery Network**.
2. The second method is to download it from getbootstrap.com.

Using CDN: The Bootstrap 4 can be used in the website by inserting it from the CDN which is short for - Content Delivery Network. The code given below can be used to compile the Bootstrap's CDN of CSS and JS in the project.

In this process, also insert the CDN versions of **jQuery** and **Popper.js** which is used for the purpose of using the different JavaScript components, for example - modals, tooltips, popovers, and so on. This needs to be before the minified Bootstrap JavaScript, if the compiled version of JavaScript is being used. There are some components that extensively require the jQuery. Following are some of these components –

- For using the closable alerts
- For toggling the states with the help of the buttons and the checkboxes or radio buttons and using the collapse for toggling content
- For the purpose of carousel for slides, controls, and indicators
- For using Dropdowns (the Popper.js can be used for perfect positioning)
- For opening and closing the Modals
- For collapsing the Navbar
- For using the tooltips and popovers (the Popper.js can be used for perfect positioning)
- For using various other components

Downloading Bootstrap 4:

The Bootstrap 4 can be downloaded from this link given below: https://getbootstrap.com/docs/4.0/getting-started/download/.

Once the link has been opened, a window will open:

Bootstrap Templates Navbar

Navigation Bars:A navigation bar is a navigation header that is placed at the top of the page:

i) Basic Navbar:

With Bootstrap, a navigation bar can extend or collapse, depending on the screen size.

A standard navigation bar is created with the `.navbar` class, followed by a responsive collapsing class: `.navbar-expand-xl|lg|md|sm` (stacks the navbar vertically on extra large, large, medium or small screens).

To add links inside the navbar, use a `<ul>` element with `class="navbar-nav"`. Then add `<li>` elements with a `.nav-item` class followed by an `<a>` element with a `.nav-link` class:

Link 1 Link 2 Link 3

Example:

```
<!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/4.5.2/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>
<nav class="navbar navbar-expand-sm bg-light">
  <ul class="navbar-nav">
    <li class="nav-item">
      <a class="nav-link" href="#">Link 1</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link 2</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link 3</a>
    </li>
  </ul>
</nav>
<br>

<div class="container-fluid">
  <h3>Basic Navbar Example</h3>
  <p>A navigation bar is a navigation header that is placed at the top of the page.</p>
  <p>The navbar-expand-xl|lg|md|sm class determines when the navbar should stack vertically
(on extra large, large, medium or small screens).</p>
</div>
</body>
```

</html>

ii) Vertical Navbar:

Remove the `.navbar-expand-xl|lg|md|sm` class to create a vertical navigation bar:

- Link 1
- Link 2
- Link 3

Example: <!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/4.5.2/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>


<nav class="navbar bg-light">
  <ul class="navbar-nav">
    <li class="nav-item">
      <a class="nav-link" href="#">Link 1</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link 2</a>
    </li>
    <li class="nav-item">
```

```
    <a class="nav-link" href="#">Link 3</a>
   </li>
  </ul>
</nav>
<br>
<div class="container-fluid">
  <h3>Vertical Navbar Example</h3>
  <p>A navigation bar is a navigation header that is placed at the top of the page.</p>
</div>
</body>
            </html>
```

iii) Centered Navbar:

Add the `.justify-content-center` class to center the navigation bar.

The following example will center the navigation bar on medium, large and extra large screens. On small screens it will be displayed vertically and left-aligned (because of the .navbar-expand-sm class):

Link 1 Link 2 Link 3

Example:

```
<!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/4.5.2/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
```

```
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>
<nav class="navbar navbar-expand-sm bg-light justify-content-center">
  <ul class="navbar-nav">
    <li class="nav-item">
      <a class="nav-link" href="#">Link 1</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link 2</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link 3</a>
    </li>
  </ul>
</nav>
<br>

<div class="container-fluid">
  <h3>Centered Navbar</h3>
  <p>Use the .justify-content-center class to center the navigation bar.</p>
  <p>In this example, the navbar will be centered on medium, large and extra large screens. On small screens it will be displayed vertically and left-aligned (because of the .navbar-expand-sm class).</p>
</div>
</body>
                </html>
```

iv) Colored Navbar:

Use any of the `.bg-color` classes to change the background color of the navbar (`.bg-primary`, `.bg-success`, `.bg-info`, `.bg-warning`, `.bg-danger`, `.bg-secondary`, `.bg-dark` and `.bg-light`)

Tip: Add a **white** text color to all links in the navbar with the `.navbar-dark` class, or use the `.navbar-light` class to add a **black** text color.

Example:

```
<!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/4.5.2/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>
<div class="container">
  <h3>Colored Navbar</h3>
  <p>Use any of the .bg-color classes to add a background color to the navbar.</p>
  <p>Tip: Add a white text color to all links in the navbar with the .navbar-dark class, or use the
.navbar-light class to add a black text color.</p>
</div>


<nav class="navbar navbar-expand-sm bg-light navbar-light">
  <ul class="navbar-nav">
    <li class="nav-item active">
      <a class="nav-link" href="#">Active</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
```

```
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</nav>

<nav class="navbar navbar-expand-sm bg-dark navbar-dark">
  <ul class="navbar-nav">
    <li class="nav-item active">
      <a class="nav-link" href="#">Active</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</nav>

<nav class="navbar navbar-expand-sm bg-primary navbar-dark">
  <ul class="navbar-nav">
    <li class="nav-item active">
```

```
      <a class="nav-link" href="#">Active</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</nav>

<nav class="navbar navbar-expand-sm bg-success navbar-dark">
  <ul class="navbar-nav">
    <li class="nav-item active">
      <a class="nav-link" href="#">Active</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</nav>
```

```
<nav class="navbar navbar-expand-sm bg-info navbar-dark">
  <ul class="navbar-nav">
    <li class="nav-item active">
      <a class="nav-link" href="#">Active</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</nav>

<nav class="navbar navbar-expand-sm bg-warning navbar-dark">
  <ul class="navbar-nav">
    <li class="nav-item active">
      <a class="nav-link" href="#">Active</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
```

```
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</nav>

<nav class="navbar navbar-expand-sm bg-danger navbar-dark">
  <ul class="navbar-nav">
    <li class="nav-item active">
      <a class="nav-link" href="#">Active</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</nav>

<nav class="navbar navbar-expand-sm bg-secondary navbar-dark">
  <ul class="navbar-nav">
    <li class="nav-item active">
      <a class="nav-link" href="#">Active</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
```

```
    <li class="nav-item">
      <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</nav>
</body>
          </html>
```

v) Brand / Logo:

The `.navbar-brand` class is used to highlight the brand/logo/project name of your page:

Example:

```
<!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/4.5.2/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>

<nav class="navbar navbar-expand-sm bg-dark navbar-dark">
```

```
  <!-- Brand/logo -->
  <a class="navbar-brand" href="#">Logo</a>

  <!-- Links -->
  <ul class="navbar-nav">
    <li class="nav-item">
      <a class="nav-link" href="#">Link 1</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link 2</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Link 3</a>
    </li>
  </ul>
</nav>

<div class="container-fluid">
  <h3>Brand / Logo</h3>
  <p>The .navbar-brand class is used to highlight the brand/logo/project name of your page.</p>
</div>

</body>
            </html>
```

vi) Collapsing The Navigation Bar: Very often, especially on small screens, you want to hide the navigation links and replace them with a button that should reveal them when clicked on.

To create a collapsible navigation bar, use a button with `class="navbar-toggler", data-toggle="collapse" and data-target="#`*thetarget*`"`. Then wrap the navbar content (links, etc) inside a div element with `class="collapse navbar-collapse"`, followed by an id that matches the `data-target` of the button: "*thetarget*".

Example:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>

<nav class="navbar navbar-expand-md bg-dark navbar-dark">

  <a class="navbar-brand" href="#">Navbar</a>

  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#collapsibleNavbar">

    <span class="navbar-toggler-icon"></span>

  </button>

  <div class="collapse navbar-collapse" id="collapsibleNavbar">

    <ul class="navbar-nav">

      <li class="nav-item">

        <a class="nav-link" href="#">Link</a>

      </li>
```

```
      <li class="nav-item">
        <a class="nav-link" href="#">Link</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Link</a>
      </li>
    </ul>
  </div>
</nav>
<br>
<div class="container">
  <h3>Collapsible Navbar</h3>
  <p>In this example, the navigation bar is hidden on small screens and replaced by a button in the top right corner (try to re-size this window).</p>
  <p>Only when the button is clicked, the navigation bar will be displayed.</p>
  <p>Tip: You can also remove the .navbar-expand-md class to ALWAYS hide navbar links and display the toggler button.</p>
</div>
</body>
</html>
```

vii) Navbar With Dropdown:

Example:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>

<nav class="navbar navbar-expand-sm bg-dark navbar-dark">

  <!-- Brand -->

  <a class="navbar-brand" href="#">Logo</a>

  <!-- Links -->

  <ul class="navbar-nav">

    <li class="nav-item">

      <a class="nav-link" href="#">Link 1</a>

    </li>
```

```
    <li class="nav-item">
      <a class="nav-link" href="#">Link 2</a>
    </li>
    <!-- Dropdown -->
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">
        Dropdown link
      </a>
      <div class="dropdown-menu">
        <a class="dropdown-item" href="#">Link 1</a>
        <a class="dropdown-item" href="#">Link 2</a>
        <a class="dropdown-item" href="#">Link 3</a>
      </div>
    </li>
  </ul>
</nav>
<br>
<div class="container">
  <h3>Navbar With Dropdown</h3>
  <p>This example adds a dropdown menu in the navbar.</p>
</div>
</body>
</html>
```

viii) Navbar Forms and Buttons: Add a `<form>` element with `class="form-inline"` to group inputs and buttons side-by-side:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>

<nav class="navbar navbar-expand-sm bg-dark navbar-dark">

  <form class="form-inline" action="/action_page.php">

    <input class="form-control mr-sm-2" type="text" placeholder="Search">

    <button class="btn btn-success" type="submit">Search</button>

  </form>

</nav>

<br>

<div class="container">

  <h3>Navbar Forms</h3>
```

```
  <p>Use the .form-inline class to align form elements side by side inside the navbar.</p>

</div>

</body>

</html>
```

Forms And Tables

Form controls automatically receive some global styling with Bootstrap:

All textual `<input>`, `<textarea>`, and `<select>` elements with class `.form-control` have a width of 100%.

Form Layouts:

Bootstrap provides two types of form layouts:

- Stacked (full-width) form
- Inline form

i) Stacked Form:

Example:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
```

```
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>


<div class="container">

  <h2>Stacked form</h2>

  <form action="/action_page.php">

    <div class="form-group">

      <label for="email">Email:</label>

      <input type="email" class="form-control" id="email" placeholder="Enter email" name="email">

    </div>

    <div class="form-group">

      <label for="pwd">Password:</label>

      <input type="password" class="form-control" id="pwd" placeholder="Enter password" name="pswd">

    </div>

    <div class="form-group form-check">

      <label class="form-check-label">

        <input class="form-check-input" type="checkbox" name="remember"> Remember me

      </label>

    </div>

    <button type="submit" class="btn btn-primary">Submit</button>

  </form>
```

```
</div>

</body>

</html>
```

ii)Inline Form:

Example:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

  <h2>Inline form</h2>

  <p>Make the viewport larger than 576px wide to see that all of the form elements are inline and
left-aligned. On small screens, the form groups will stack horizontally.</p>

  <form class="form-inline" action="/action_page.php">
```

```
    <label for="email">Email:</label>

    <input type="email" class="form-control" id="email" placeholder="Enter email"
name="email">

    <label for="pwd">Password:</label>

    <input type="password" class="form-control" id="pwd" placeholder="Enter password"
name="pswd">

    <div class="form-check">

      <label class="form-check-label">

        <input class="form-check-input" type="checkbox" name="remember"> Remember me

      </label>

    </div>

    <button type="submit" class="btn btn-primary">Submit</button>

  </form>

</div>

</body>

</html>
```

iii) Form Row/Grid:

You can also use columns (`.col`) to control the width and alignment of form inputs without using spacing utilities. Just remember to put them inside a `.row` container.

Example:

```
<!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/4.5.2/css/bootstrap.min.css">
```

```
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>
<div class="container">
  <h2>Form Grid</h2>
  <p>Create two form elements that appear side by side with .row and .col:</p>
  <form action="/action_page.php">
    <div class="row">
      <div class="col">
        <input type="text" class="form-control" id="email" placeholder="Enter email"
name="email">
      </div>
      <div class="col">
        <input type="password" class="form-control" placeholder="Enter password"
name="pswd">
      </div>
    </div>
    <button type="submit" class="btn btn-primary mt-3">Submit</button>
  </form>
</div>
</body>
</html>
```

iv) Form Validation: You can use different validation classes to provide valuable feedback to users. Add either .was-validated or .needs-validation to the <form> element, depending on whether you want to provide validation feedback before or after submitting the form. The input fields will have a green (valid) or red (invalid) border to indicate what's missing in the form. You can also add a .valid-feedback or .invalid-feedback message to tell the user explicitly what's missing, or needs to be done before submitting the form.

Example:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

  <h2>Form Validation</h2>

  <p>In this example, we use <code>.was-validated</code> to indicate what's missing before
submitting the form:</p>

  <form action="/action_page.php" class="was-validated">

    <div class="form-group">

      <label for="uname">Username:</label>

      <input type="text" class="form-control" id="uname" placeholder="Enter username"
name="uname" required>

      <div class="valid-feedback">Valid.</div>

      <div class="invalid-feedback">Please fill out this field.</div>

    </div>

    <div class="form-group">

      <label for="pwd">Password:</label>

      <input type="password" class="form-control" id="pwd" placeholder="Enter password"
name="pswd" required>
```

```
    <div class="valid-feedback">Valid.</div>
    <div class="invalid-feedback">Please fill out this field.</div>
  </div>
  <div class="form-group form-check">
    <label class="form-check-label">
      <input class="form-check-input" type="checkbox" name="remember" required> I agree on blabla.
      <div class="valid-feedback">Valid.</div>
      <div class="invalid-feedback">Check this checkbox to continue.</div>
    </label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
 </form>
</div>
</body>
</html>
```

Tables: A basic Bootstrap 4 table has a light padding and horizontal dividers.

The `.table` class adds basic styling to a table:

i) Striped Rows:

The `.table-striped` class adds zebra-stripes to a table:

Example:

```
<!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/4.5.2/css/bootstrap.min.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
</head>
<body>
<div class="container">
  <h2>Striped Rows</h2>
  <p>The .table-striped class adds zebra-stripes to a table:</p>
  <table class="table table-striped">
    <thead>
      <tr>
        <th>Firstname</th>
        <th>Lastname</th>
        <th>Email</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>John</td>
```

```
      <td>Doe</td>
      <td>john@example.com</td>
    </tr>
    <tr>
      <td>Mary</td>
      <td>Moe</td>
      <td>mary@example.com</td>
    </tr>
    <tr>
      <td>July</td>
      <td>Dooley</td>
      <td>july@example.com</td>
    </tr>
  </tbody>
 </table>
</div>
</body>
</html>
```

ii) Bordered Table:

The `.table-bordered` class adds borders on all sides of the table and cells:

Example:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

  <h2>Bordered Table</h2>

  <p>The .table-bordered class adds borders on all sides of the table and the cells:</p>

  <table class="table table-bordered">

    <thead>

      <tr>

        <th>Firstname</th>

        <th>Lastname</th>

        <th>Email</th>

      </tr>

    </thead>

    <tbody>

      <tr>
```

```
      <td>John</td>
      <td>Doe</td>
      <td>john@example.com</td>
    </tr>
    <tr>
      <td>Mary</td>
      <td>Moe</td>
      <td>mary@example.com</td>
    </tr>
    <tr>
      <td>July</td>
      <td>Dooley</td>
      <td>july@example.com</td>
    </tr>
  </tbody>
 </table>
</div>
</body>
</html>
```

iv) Hover Rows

The `.table-hover` class adds a hover effect (grey background color) on table rows:

Example:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

  <h2>Hover Rows</h2>

  <p>The .table-hover class enables a hover state (grey background on mouse over) on table
rows:</p>

  <table class="table table-hover">

    <thead>

      <tr>

        <th>Firstname</th>

        <th>Lastname</th>

        <th>Email</th>

      </tr>

    </thead>
```

```
    <tbody>
      <tr>
        <td>John</td>
        <td>Doe</td>
        <td>john@example.com</td>
      </tr>
      <tr>
        <td>Mary</td>
        <td>Moe</td>
        <td>mary@example.com</td>
      </tr>
      <tr>
        <td>July</td>
        <td>Dooley</td>
        <td>july@example.com</td>
      </tr>
    </tbody>
  </table>
</div>
</body>
</html>
```

v) Black/Dark Table:

The `.table-dark` class adds a black background to the table:

```
<!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/4.5.2/css/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

  <h2>Black/Dark Table</h2>

  <p>The .table-dark class adds a black background to the table:</p>

  <table class="table table-dark">

    <thead>

      <tr>

        <th>Firstname</th>

        <th>Lastname</th>

        <th>Email</th>

      </tr>
```

```
    </thead>
    <tbody>
      <tr>
        <td>John</td>
        <td>Doe</td>
        <td>john@example.com</td>
      </tr>
      <tr>
        <td>Mary</td>
        <td>Moe</td>
        <td>mary@example.com</td>
      </tr>
      <tr>
        <td>July</td>
        <td>Dooley</td>
        <td>july@example.com</td>
      </tr>
    </tbody>
  </table>
</div>
</body>
</html>
```

Bootstrap Forms

Bootstrap forms are input-based components which are designed to collect users' data. Used as a login, subscribe or contact form, all of them can be easily customized. Bootstrap forms in Material Design are simple and eye-pleasant. While creating MDB, we were aware of their importance in almost every project, so we have put in a lot of effort to get them right. At your disposal are such constructions as predefined Form logins, Form registers, Form subscriptions or Form contacts and various other layout forms. Each of the forms offers a different type of functionality.

Example:

```
<!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/3.7.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>
<body>
  <div class="container">
  <h2>Vertical (basic) form</h2>
  <form action="/action_page.php">
    <div class="form-group">
      <label for="email">Email:</label>
      <input type="email" class="form-control" id="email" placeholder="Enter email" name="email">
    </div>
```

```
    <div class="form-group">
      <label for="pwd">Password:</label>
      <input type="password" class="form-control" id="pwd" placeholder="Enter password" name="pwd">
    </div>
    <div class="checkbox">
      <label><input type="checkbox" name="remember"> Remember me</label>
    </div>
    <button type="submit" class="btn btn-default">Submit</button>
  </form>
</div>
</body>
</html>
```