UI TECHNOLOGIES

HTML 4 + HTML 5 + CSS 2 + CSS 3 + JavaScript 5 + JavaScript 6 + JavaScript 7 + Bootstrap + jQuery

by Harsha Vardhan (UI Expert)

Contents

UI Technologies

Fundamentals

Application

- It is a program that runs based on the operating system.
- Program is a collection of statements (instructions) to the system.
- For example, a statement instructs the computer to store a value. Another statement instructs the computer to do addition of numbers.
- **Examples:** Notepad, MS Word, Google, Facebook, Flipkart etc.

Client

- It is a machine or device (desktop, laptop, tablet, phone or Smart TV), which can access the data from server.
- The device which is used by the user is called as "client".

Browser

- It is software (tool) installed on the client, to see the output of the web page. User gives input in the browser and gets the output in the browser.
- The browser sends a request to server to get web page.
- Browser provides navigation among web pages.
- Browser executes the html, css, javascript programs and displays corresponding output to the user.
- Browsers are developed by different companies. For example, "Chrome" browser was developed by "Google" company.

  Important browsers:

  - Google Chrome
  - Mozilla Firefox
  - Microsoft Internet Explorer
  - Microsoft Edge
  - Apple Safari

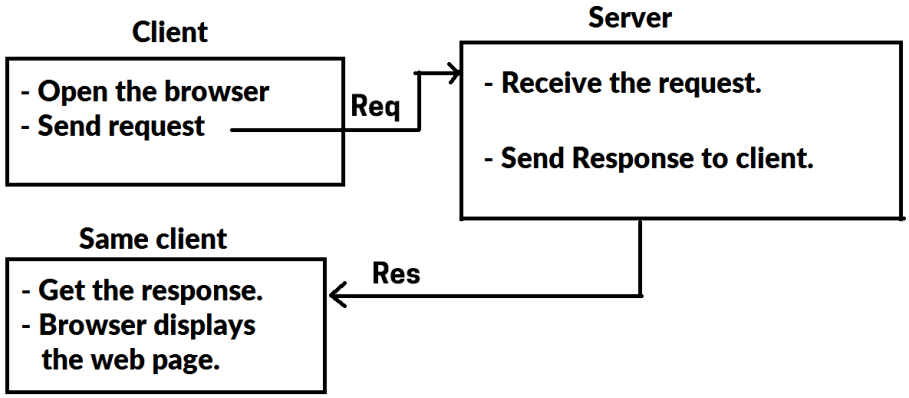
- Opera

Web application

- It is a set of programs that are running on the browser to interact with the user.
- It is responsible to store information temporarily.
- It is responsible to interact with server i.e. sending request to server and get response from server.

Http

- Http stands for “Hypertext Transfer Protocol”.
- It is a protocol, which provides a set of rules to send request to server and get response from server.
- Http protocol defines "request format" and "response format".
- Http protocol defines HTTP status codes, Http Content Types etc.



HTML 4 & 5

HTML Basics

Introduction to HTML

- HTML stands for “Hypertext Markup Language”.
- “Hypertext” means “the text that can be transferred from internet server to internet client”.
- HTML is a markup language. The markup language is a language, which syntax will be in the form of tags.
- HTML is used to design web pages. That means HTML is used to create elements (such as headings, paragraphs, icons, menus, logos, images, textboxes, buttons etc.) in the web pages.
- HTML is easy language to understand.
- HTML is “client side language”. That means the html code executes on the client (browser).
- HTML is supported by all the browsers such as Google Chrome, Mozilla Firefox, Microsoft Internet Explorer, Safari, Opera and other browsers.
- HTML is used developed by “Tim Berners-Lee” and maintained by “W3C” (World Wide Web).
- HTML is used in all real web sites today.
- The file extension should be “.html”.
- HTML is the interpreter-based language. That means the HTML code will be converted into machine language in line-by-line manner. Browser interprets HTML code.
- HTML is not a case sensitive language. That means you can write the html code in either upper case or lower case.

HTML Versions

- HTML 1.0 : Nov 1991
- HTML 2.0 : Mar 1995
- HTML 3.0 : Jan 1997
- HTML 4.0 : Dec 1997
- HTML 5.0 : Oct 2014
- HTML 5.1 : Nov 2016
- HTML 5.2 : Dec 2017

Tag

- A tag is a keyword, enclosed within “<” and “>” in HTML language.
- Tag is instruction / command to browser.
- Tag is used to display some specific output in the web page.
- **Syntax:** <tag>

Types of tags

- Tags are two types:
  1. **Paired tags:** Contains opening tag and ending tag. The opening tag specifies starting point of the output. The closing tag specifies end point of the output. Ex: <h1> hello </h1>
  2. **Unpaired tags:** Contains single tag only (no separate ending tag). Ex: <hr>

Syntax of HTML Program

- Every html program should have the following syntax:

```
<html>
  <head>
     Non Content here
  </head>
  <body>
     Content here
  </body>
</html>
```

- The <html> tag represents starting and ending point of the HTML program. The <html> tag contains two child tags, those are <head> and <body>
- The <head> tag represents non-content information of the web page. The information that doesn't appear in the web page is called as "non-content".
- The <body> tag represents content information of the web page. The information that appears in the web page is called as "content".

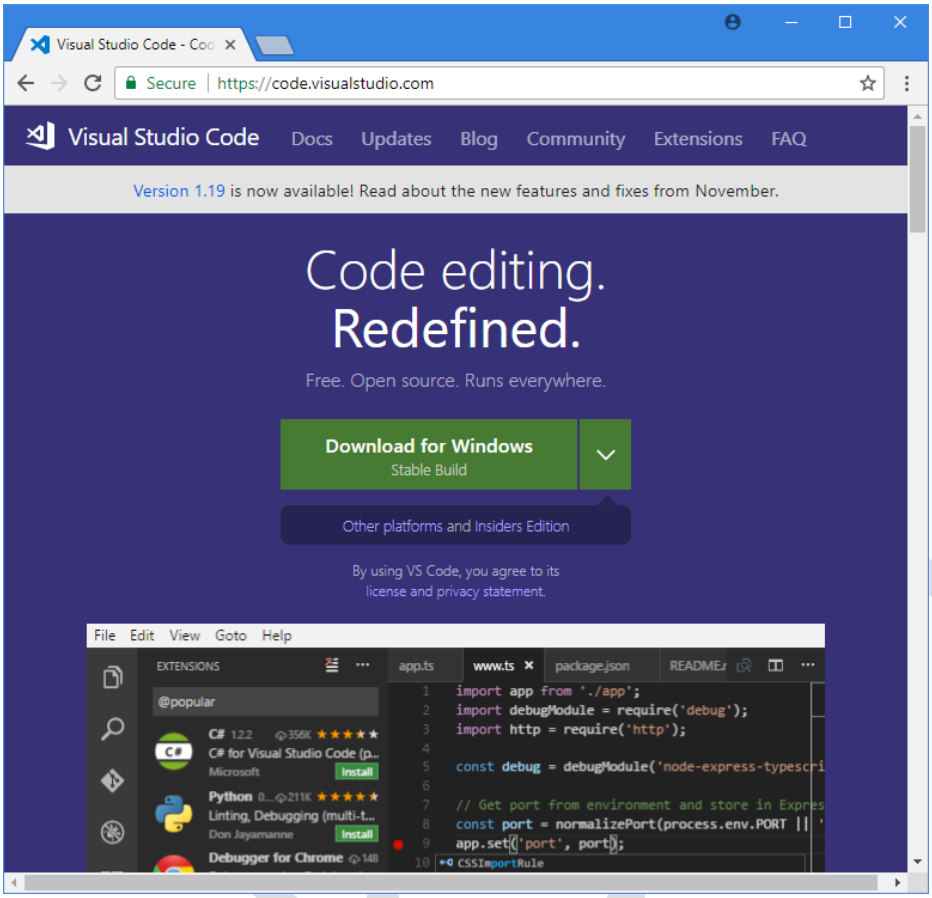
Steps to Prepare First Example in HTML

1. Installing Visual Studio Code
2. Creating HTML Program
3. Executing HTML Program

1. Installing Visual Studio Code

"Visual Studio Code" is the recommended editor for html, css, javascript and many other languages / frameworks.

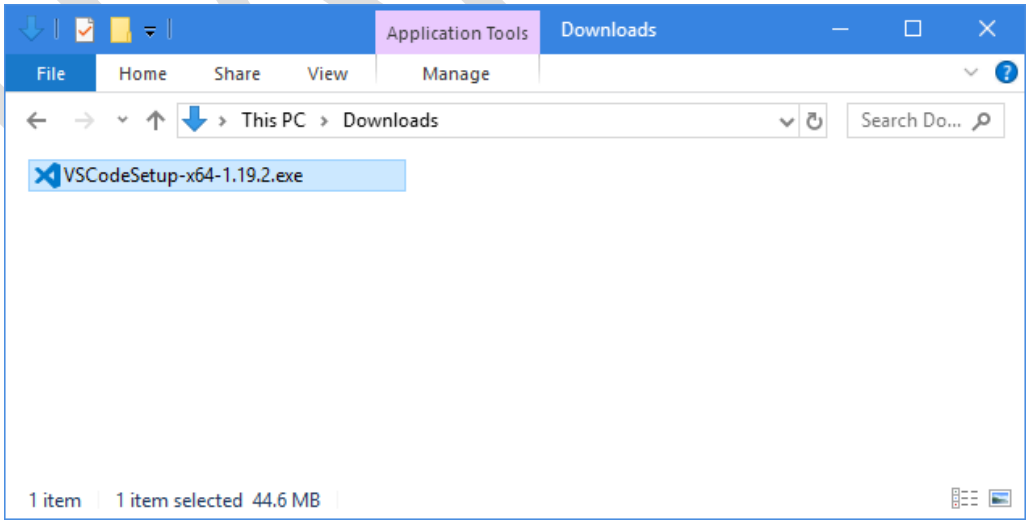
Go to https://code.visualstudio.com



Click on “Download for Windows”.

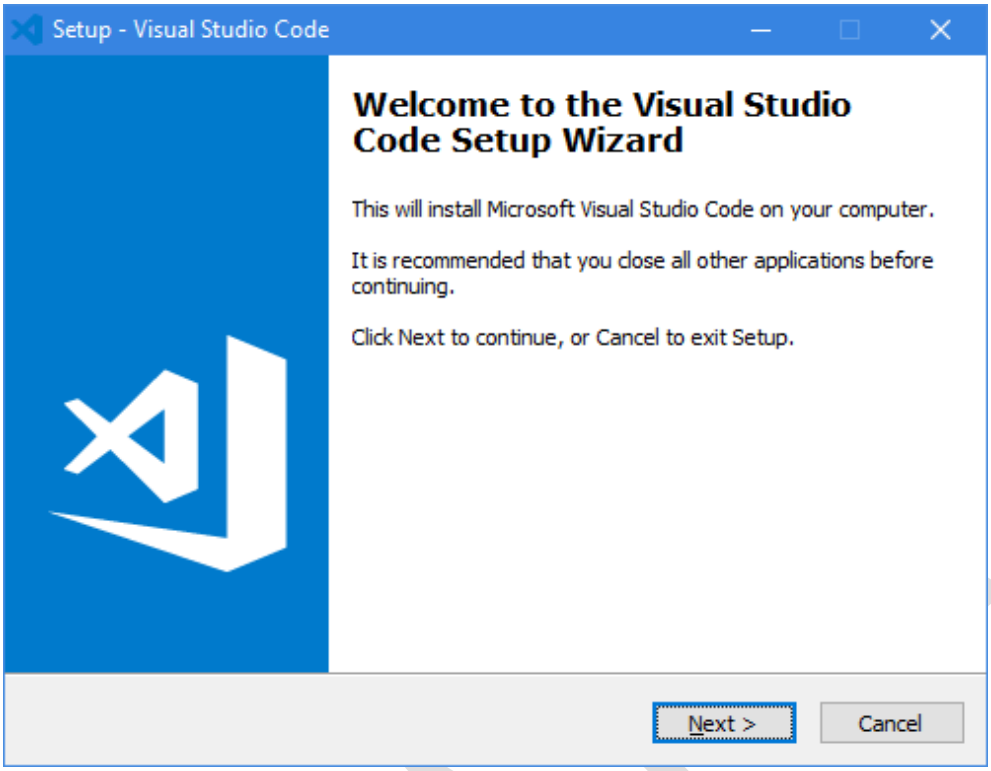
Note: The version number may be different at your practice time.

Go to “Downloads” folder; you can find “VSCodeSetup-x64-1.19.2.exe” file.

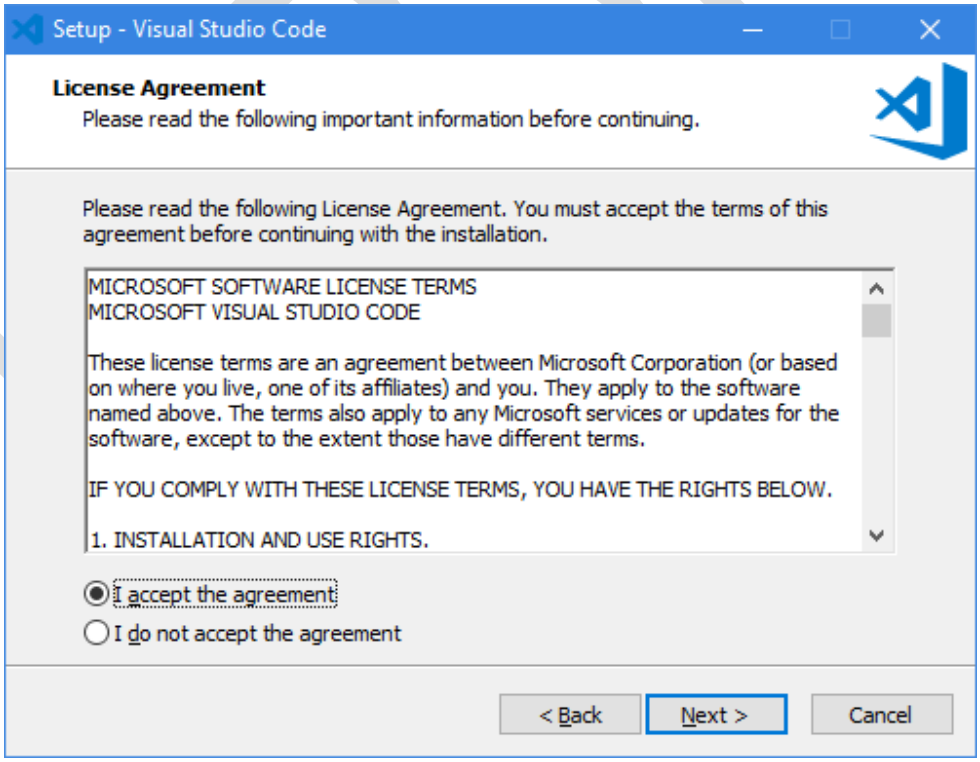


Double click on “VSCodeSetup-x64-1.19.2.exe” file.

Click on “Yes”.

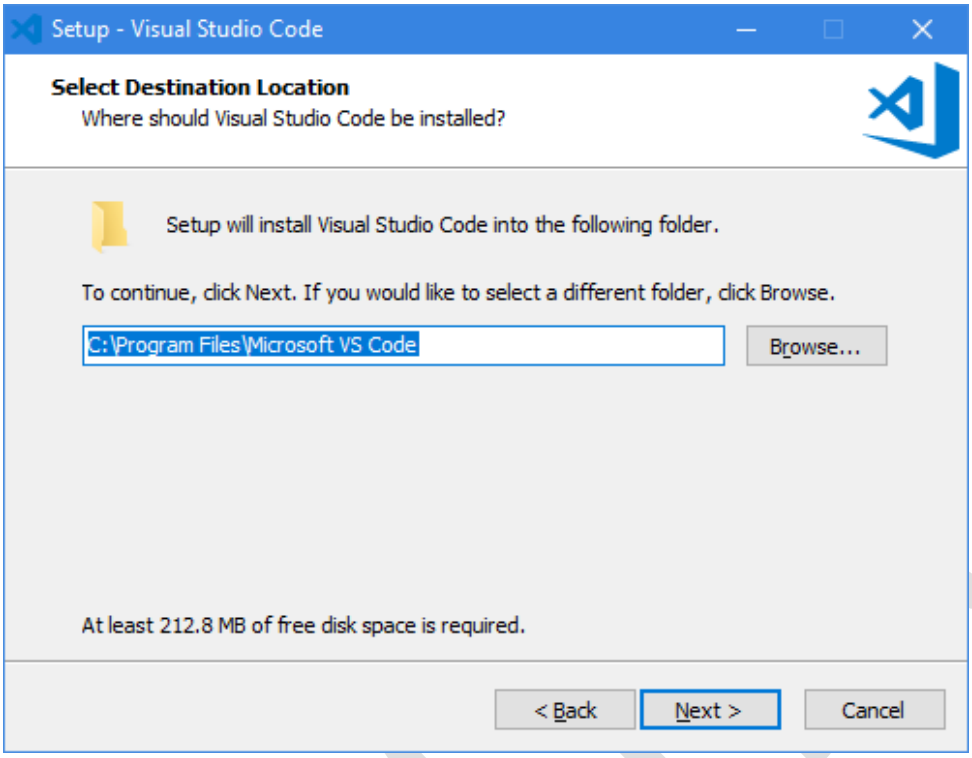


Click on “Next”.

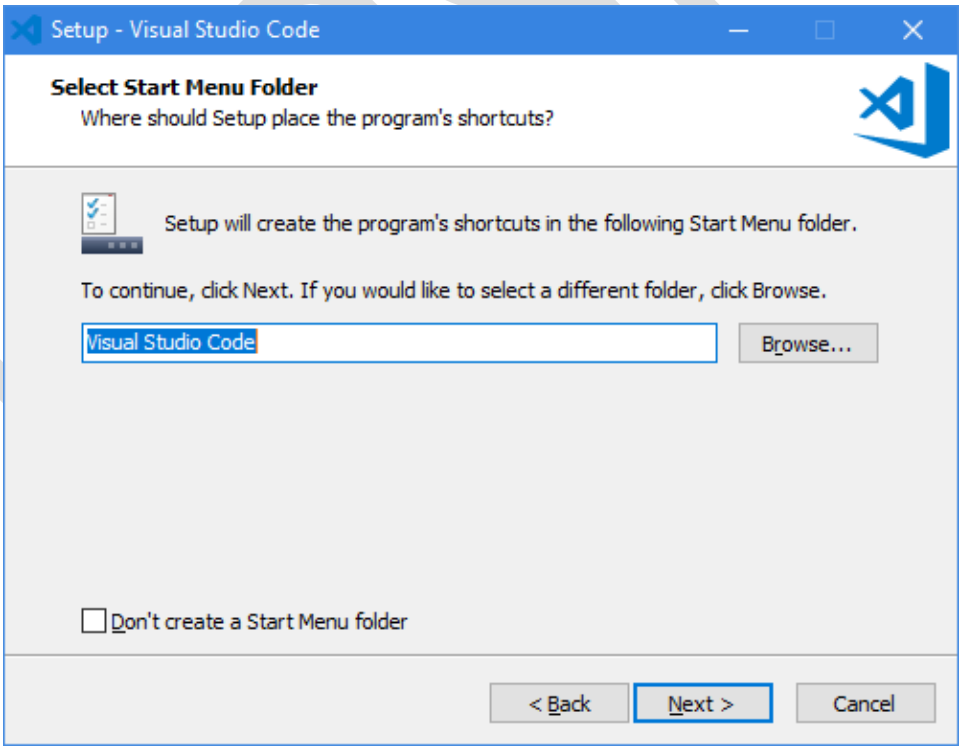


Click on “I accept the agreement”.

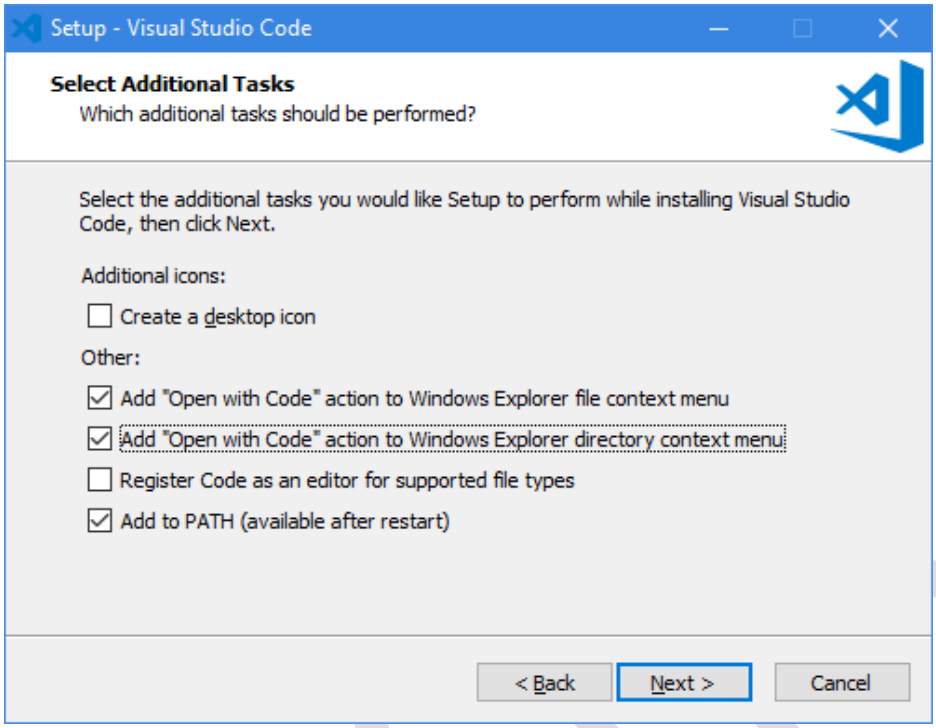
Click on “Next”.



Click on “Next”.



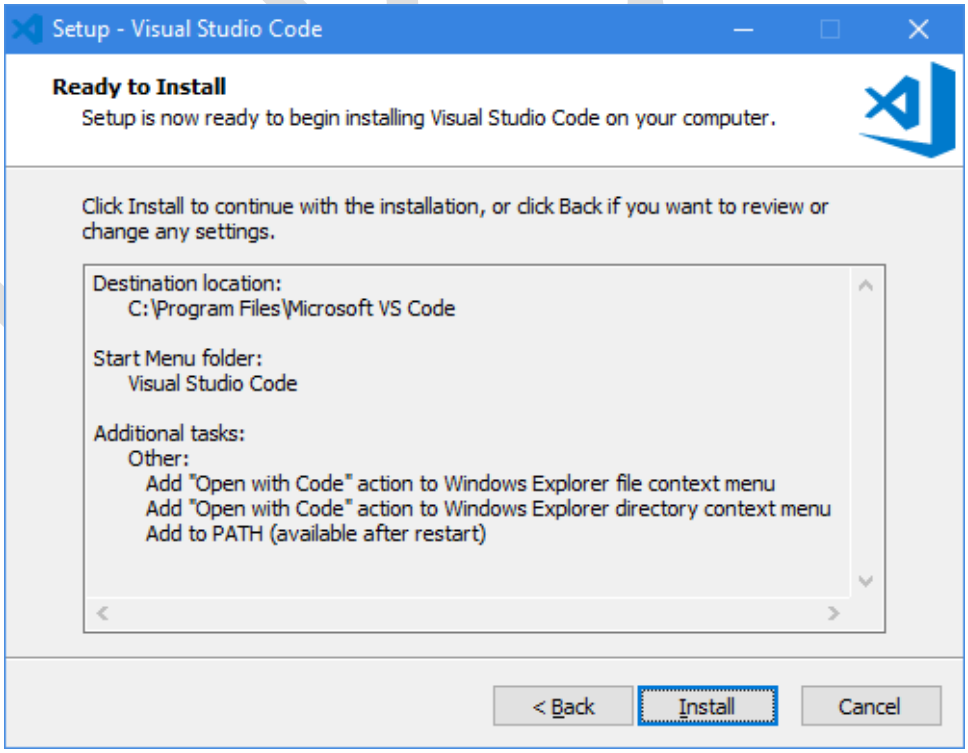
Click on “Next”.



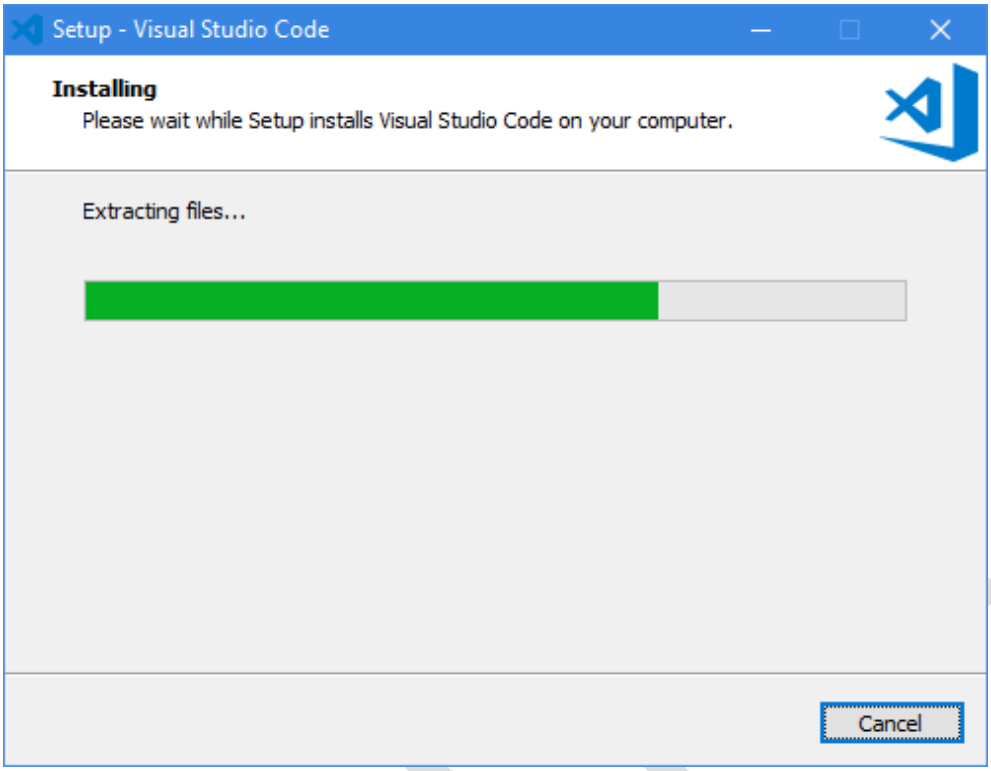
Check the checkbox “Add Open with Code action to Windows Explorer file context menu”.

Check the checkbox “Add Open with Code action to Windows Explorer directory context menu”.

Click on “Next”.



Click on “Install”.



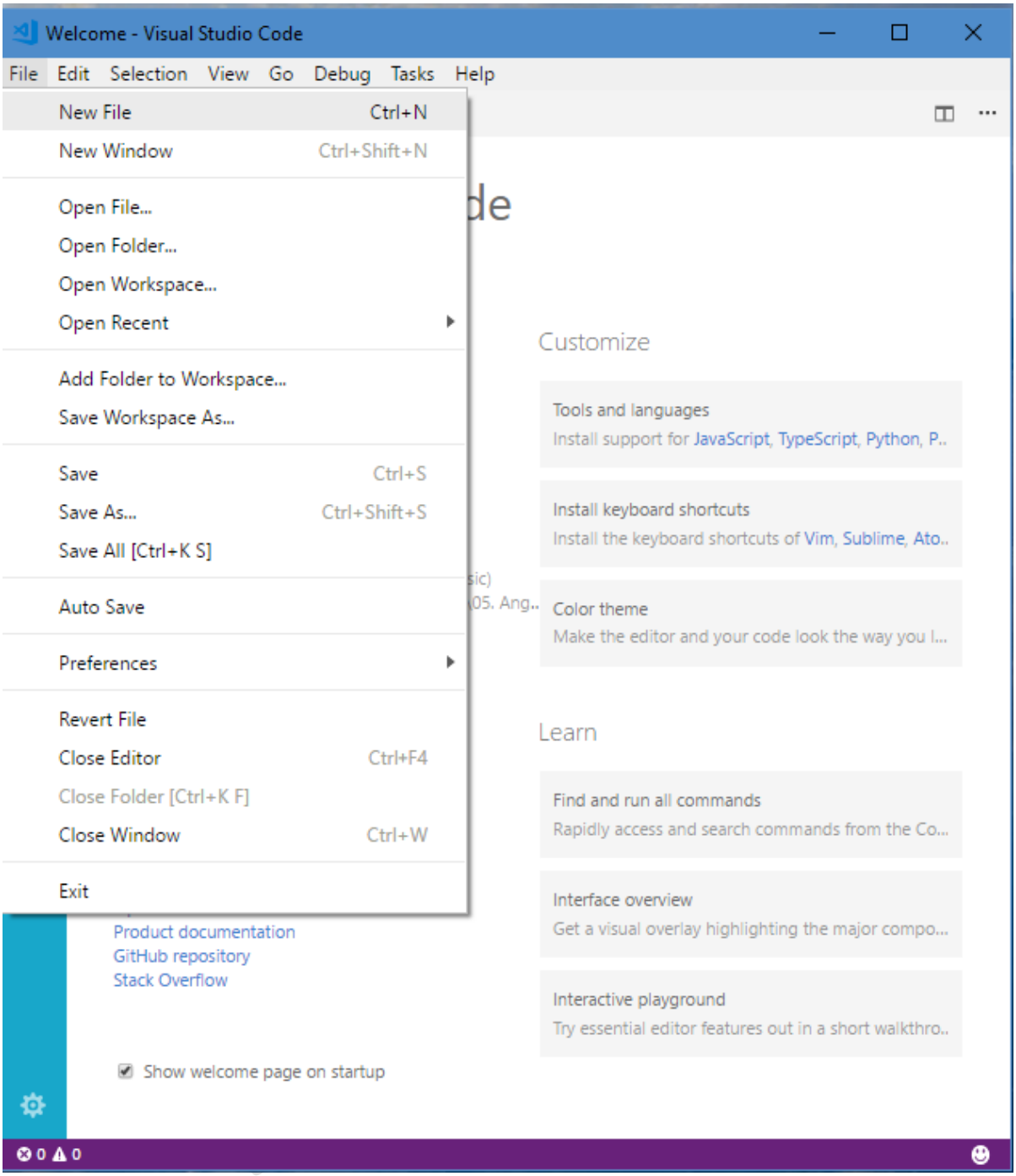
Installation is going on….

Click on “Finish”.

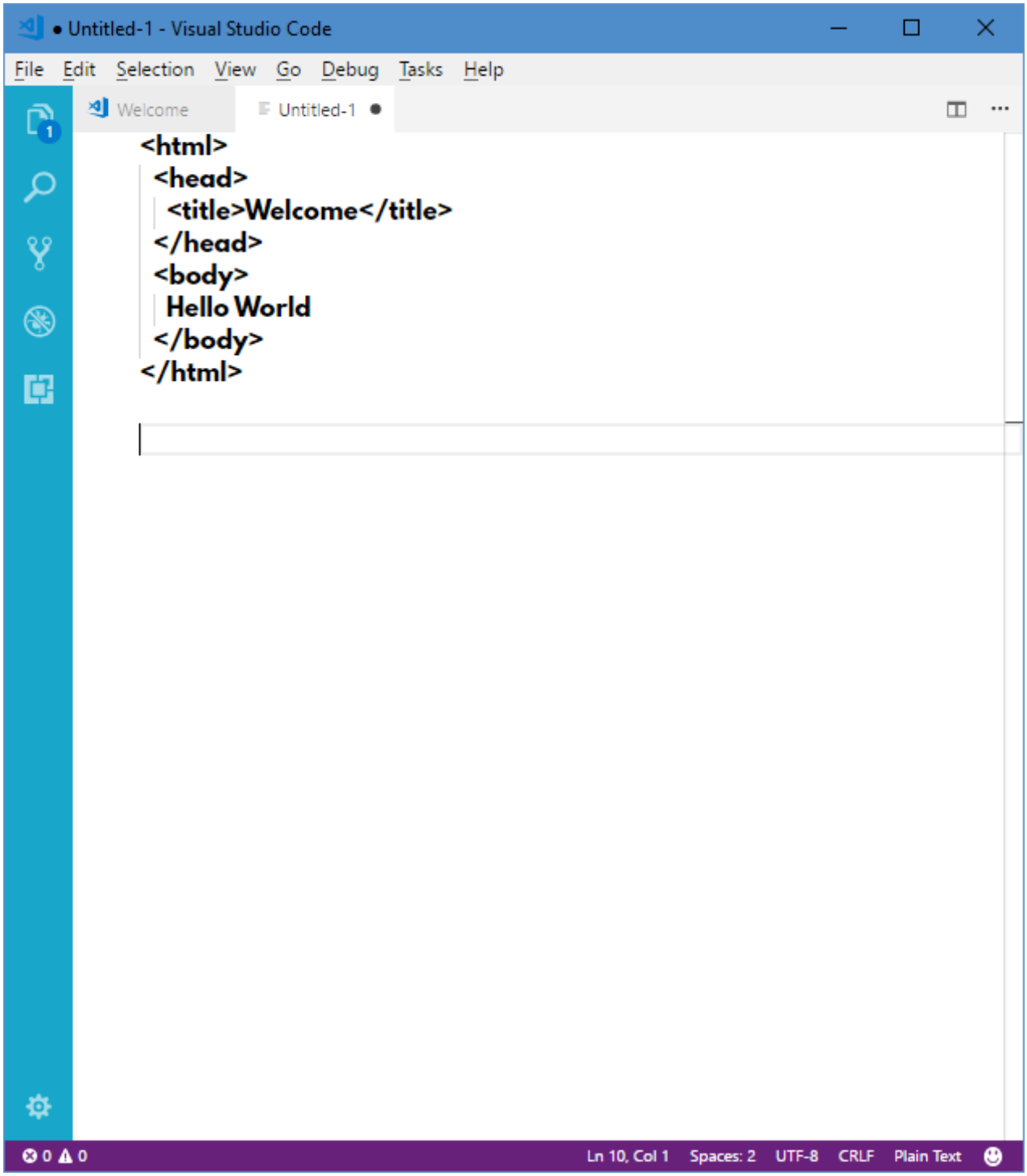
2. Create HTML Program

- Open “Visual Studio Code”, by clicking on “Start” – “Visual Studio Code”.

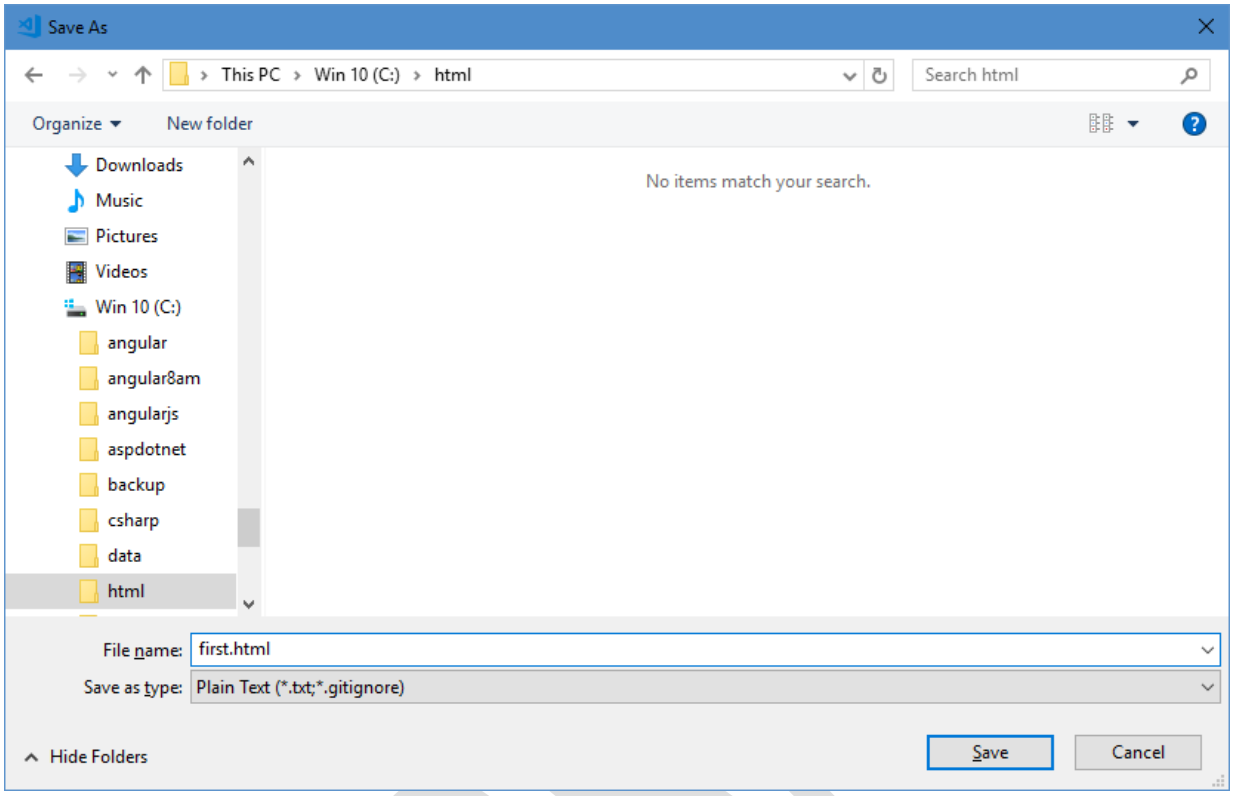
- Visual Studio Code opened.



- Go to “File” – “New File”.
- Type the program as follows:



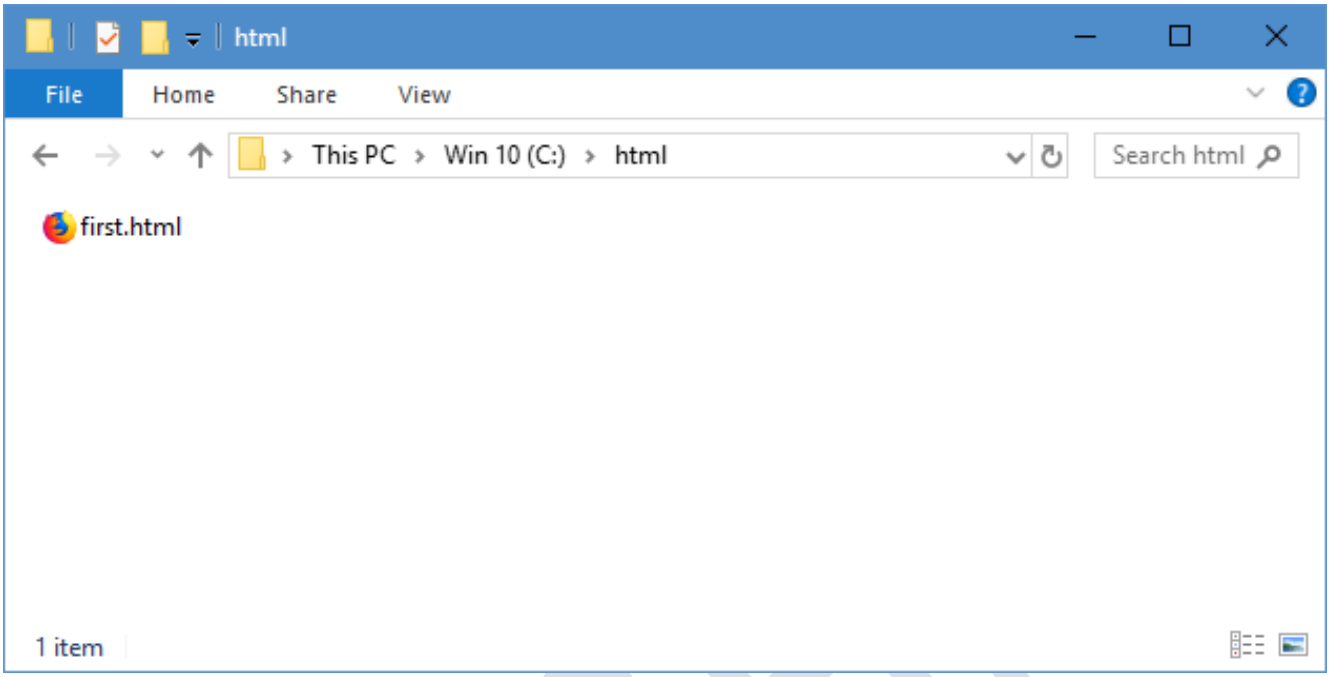
- Go to “File” menu – “Save” (or) Press Ctrl+S.



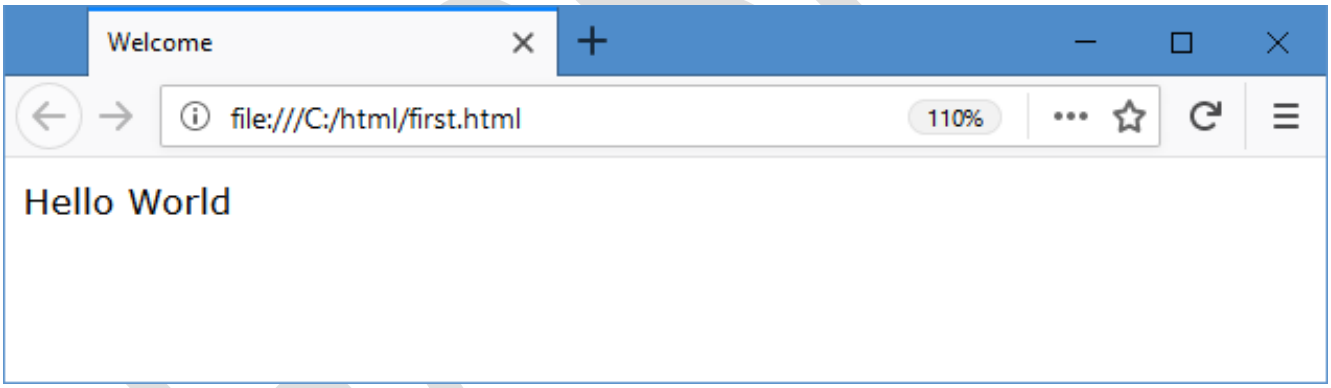
- Go to "c:\" and click on "New folder". Enter the new folder name as "html".
- Select “c:\html” folder and enter the filename as “first.html”.
- Click on “Save”.
- Now the html file (c:\html\first.html) is ready.

3. Execute the HTML Program

- Go to “Computer” or “This PC” and go to “c:\html” folder.



- Double click on “first.html” (or) Right click on “first.html” and click on “Open With” – “Google Chrome”.



Output: Hello World

Understanding HTML Syntax

<html>

- <html> tag represents starting and ending point of the html program. The <html> tag contains two child tags, those are <head> and <body>.

Syntax:

```
<html> </html>
```

Example:

```
<html> </html>
```

<head>

- <head> tag represents non content information of the page. The information that doesn't appear in the web page is called as "non content".

Syntax:

<head> </head>

Example:

<head> </head>

<body>

- <body> tag represents content information of the page. The information that appears inside the web page is called as "content".

Syntax:

<body> </body>

Example:

<body> </body>

<title>

- <title> tag is used to specify the title of the web page that appears in the browser's title bar.
- <title> tag should be used in <head> tag only.
- <title> is a paired tag.

Syntax:

<title>Title here</title>

Example:

<title>My title</title>

Attributes

- Attributes are the details about the tag (command).
- Every tag has its own set of attributes.
- Attribute contains a value; The value should be written inside the double quotes.

Syntax: <tag attribute="value"> </tag>

DOCTYPE

- DOCTYPE is a directive in HTML, which tells the browser about the version of html that you are using in the web page.
- Various versions of html have various DOCTYPE's.

1. **HTML 4**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

2. **XHTML**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

3. **HTML 5**

```
<!DOCTYPE html>
```

Basic Tags in HTML

Headings

<h1>

- It is used to create first level heading (main heading). The headings are displayed in very large size in the output. Every browser assigns a specific size for each level of heading. The size of heading is decided by the browsers, not fixed.
- It is a paired tag.

Syntax:

```
<h1>heading here</h1>
```

Example:

```
<h1>Heading 1 here</h1>
```

<h2>

- It is used to create second level heading (sub heading).
- It is a paired tag.

Syntax:

```
<h2>heading here</h2>
```

Example:

```
<h2>Heading 2 here</h2>
```

<h3>

- It is used to create third level heading (sub heading).
- It is a paired tag.

Syntax:

<h3>heading here</h3>

Example:

<h3>Heading 3 here</h3>

<h4>

- It is used to create third level heading (sub heading).
- It is a paired tag.

Syntax:

<h4>heading here</h4>

Example:

<h4>Heading 4 here</h4>

<h5>

- It is used to create third level heading (sub heading).
- It is a paired tag.

Syntax:

<h5>heading here</h5>

Example:

<h5>Heading 5 here</h5>

<h6>

- It is used to create third level heading (sub heading).
- It is a paired tag.

Syntax:

<h6>heading here</h6>

Example:

<h6>Heading 6 here</h6>

Example on Headings:

```
<html>
  <head>
    <title>Headings</title>
  </head>
  <body>
    <h1>Heading 1 here</h1>
    <h2>Heading 2 here</h2>
    <h3>Heading 3 here</h3>
    <h4>Heading 4 here</h4>
    <h5>Heading 5 here</h5>
    <h6>Heading 6 here</h6>
  </body>
</html>
```

Paragraphs

- tag is used to create a paragraph.
- Browsers display a line break, before and after each paragraph.
- Browsers display an empty line between paragraphs.
- It is a paired tag.

Syntax:

paragraph here

Example:

Hello

**Example on **

```
<html>
    <head>
        <title>Paragraphs</title>
    </head>
   <body>
        <h1>Paragraphs</h1>
        <p>Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.</p>
        <p>It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like)</p>
        <p>There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable. If you are going to use a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend to repeat predefined chunks as necessary, making this the first true generator on the Internet. It uses a dictionary of over 200 Latin words, combined with a handful of model sentence structures, to generate Lorem Ipsum which looks reasonable. The generated Lorem Ipsum is therefore always free from repetition, injected humour, or non-characteristic words etc.</p>
    </body>
</html>
```

Line Breaks

-
 tag is used to create "line breaks".
- It moves the cursor to the next line.

- The content next to the <br> will be display in the next line.
- It is an unpaired tag. That means no closing tag for <br> tag.

Syntax:

Example:

**Example on
**

```
<html>
  <head>
    <title>Br</title>
  </head>
  <body>
    <h1>Br</h1>
    One<br>Two<br>Three
  </body>
</html>
```

Text Formatting Tags

Bold

- tag is used to display the text in bold.
- The text enclosed within the tag will be display as bold.
- Use tag to display important text that you want to highlight.
- It is a paired tag.

Syntax:

bold text

Example:

Hello

**Example of **

```
<html>
  <head>
    <title>Bold</title>
  </head>
  <body>
    <h1>Bold</h1>
    This is normal text. <b>This is bold text.</b>
  </body>
</html>
```

Italic

- <i> tag is used to display the text in italic.

- The text enclosed within the tag will be display as *italic*.
- It is a paired tag.

Syntax:

italic text

Example:

Hello

**Example of **

```
<html>
  <head>
    <title>Italic</title>
   </head>
  <body>
    <h1>Italic</h1>
    This is normal text. <i>This is italic text</i>
  </body>
</html>
```

Underline

- <u> tag is used to display the text in underline.
- The text enclosed within the <u> tag will be underlined.
- Use <u> tag display important text.
- It is a paired tag.

Syntax:

<u>underline text</u>

Example:

<u>Hello</u>

Example on <u>

```
<html>
  <head>
    <title>Underline</title>
   </head>
  <body>
    <h1>Underline</h1>
    This is normal text. <u>This is underline text</u>
  </body>
</html>
```

Strikeout

- <strike> tag is used to display the text in strikeout.
- The <strike> tag to display un-important text.
- It is a paired tag.

Syntax:

<strike>strikeout text</strike>

Example:

<strike>Hello</strike>

Example on <strike>

```
<html>
  <head>
      <title>Strikeout</title>
   </head>
  <body>
      <h1>Strikeout</h1>
      This is normal text.
      <strike>This is strikeout text</strike>
  </body>
</html>
```

Strong

- tag is used to display the text in strong.
- and <b> tags are visually same.
- The strong tag content will be pronounced "strongly" (stressfully) by the screen readers for the blind people.
- It is a paired tag.

Syntax:

strong text

Example:

Hello

**Example on **

```
<html>
  <head>
      <title>Strong</title>
   </head>
  <body>
      <h1>Strong</h1>
      This is normal text.
      <b>This is bold text.</b>
      <strong>This is strong text</strong>
  </body>
</html>
```

Emphasis

- tag is used to display the text in emphasis (special status).
- and <i> tags visually same.

- The strong tag content will be pronounced stylishly by the screen readers for the blind people.
- It is a paired tag.

Syntax:

emphasis text

Example:

Hello

**Example on **

```
<html>
  <head>
      <title>Emphasis</title>
   </head>
  <body>
      <h1>Emphasis</h1>
      This is normal text.
      <i>This is italic text</i>
      <em>This is emphasis text</em>
  </body>
</html>
```

Superscript

- <sup> tag is used to display the text in superscript (The text appears a bit upper side of normal line).
- Use <sup> tag to display square or cube etc.
- It is a paired tag.

Syntax:

^{superscript text}

Example:

^{Hello}

Example on <sup>

```
<html>
  <head>
      <title>Superscript</title>
   </head>
  <body>
      <h1>Superscript</h1>
      1<sup>st</sup>
  </body>
</html>
```

Subscript

- <sub> tag is used to display the text in subscript (The text appears a bit bottom side of normal line).
- It is a paired tag.

Syntax:

_{subscript text}

Example:

_{Hello}

Example on <sub>

```
<html>
  <head>
    <title>Subscript</title>
   </head>
  <body>
    <h1>Subscript</h1>
    1<sub>st</sub>
  </body>
</html>
```

Images

Images

- <img> tag is used to display an image in the web page.
- It is strongly recommended to place the image file in the same folder, where the html file is present. For example, if the html file is present within "c:\html" folder, we have to place the image file either in the "c:\html" folder itself or in any subfolder of the "c:\html" folder.
- It is an unpaired tag.

Syntax:

<img src="image path here">

Example:

<img src="img1.jpg" width="200px" height="130px" title="this is tooltip">

Attributes:

1. **src:**
   - It is used to specify path of the image file. If the image file and html file both are in the same folder, no need to specify the full path of the image.
2. **width:**
   - It is used to specify width (horizontal size) of the image. It represents the value in the form of pixels. A pixel is a small "dot" (.) on the screen.
3. **height**
   - It is used to specify height (vertical size) of the image.
4. **title**
   - It is used to specify the tooltip (that appears when the user places mouse pointer on the image).
5. **alt**
   - It is used to specify the alternate text (that appears when the image is not loaded in the browser at run time). It is strongly recommended to use "alt" for every <img>

tag in real-time, because, if the image is not loaded, at least the "alt" text appears to the user.

Example on <img>

```
<html>
    <head>
        <title>Img</title>
    </head>
   <body>
        <h1>Img</h1>
        <img src="img1.jpg" width="200px" height="130px" title="this is tooltip" alt="This is alternate
text">
    </body>
</html>
```

Note: Place "img1.jpg" in the current folder.

Hyperlinks

Hyperlinks

- <a> tag is used to create a hyperlink. When the user clicks on the hyperlink, the specified web page or web site or file will be opened.
- The web basically contains links to other pages, so it is very common to use <a> tag in real-time.
- By default, every browser provides built-in style for each hyperlink, i.e. blue color + hand symbol for mouse cursor + underline. We can customize this style by using CSS.
- It is a paired tag.

Syntax:

```
<a href="target url here">link text here</a>
```

Example:

```
<a href="http://www.google.com">Google</a>
```

Attributes:

1. **href:**
   - It is used to specify the address of web page or web site that is to be opened when the user clicks on the hyperlink. It can contain address of an internet web site, local html file, local other type of file such as pdf or word document, image file etc.
2. **target="_blank":**
   - It is used to open the target web page or web site in a separate browser tab.

Example on <a>

Page1.html

```
<html>
```

```
    <head>
        <title>Page 1</title>
    </head>
   <body>
        <h1>Page 1</h1>
        <a href="page2.html">Go to page 2</a>
    </body>
</html>
```

Page2.html

```
<html>
    <head>
        <title>Page 2</title>
    </head>
   <body>
        <h1>Page 2</h1>
        <a href="page1.html">Go to page 1</a>
    </body>
</html>
```

Example on internet links

```
<html>
    <head>
        <title>Hyperlinks</title>
    </head>
   <body>
        <h1>Hyperlinks</h1>
        <a href="http://www.google.com">Google</a>
        <a href="http://www.facebook.com">Facebook</a>
        <a href="http://www.microsoft.com">Microsoft</a>
    </body>
</html>
```

Example on target

```
<html>
    <head>
        <title>Hyperlinks</title>
    </head>
   <body>
        <h1>Hyperlinks</h1>
        <a href="http://www.google.com" target="_blank">Google</a>
    </body>
</html>
```

Example on file links

```
<html>
    <head>
        <title>Hyperlinks</title>
    </head>
   <body>
```

```
    <h1>Hyperlinks</h1>
    <p> <a href="vodafone.jpg">Click here to open image file</a> </p>
    <p> <a href="Document1.docx">Click here to open doc file</a> </p>
    <p> <a href="Sample.pdf">Click here to open pdf file</a> </p>
    <p> <a href="rain.mp3">Click here to open audio file</a> </p>
    <p> <a href="trailer.mp4">Click here to open video file</a> </p>
  </body>
</html>
```

Note: Place "vodafone.jpg", "document1.docx", "sample.pdf", "rain.mp3", "trailer.mp4" in "c:\html" folder.

Example on <a> with <img>

```
<html>
    <head>
        <title>Hyperlinks</title>
    </head>
   <body>
        <h1>Hyperlinks</h1>
        <a href="http://www.vodafone.in">
            <img src="vodafone.jpg" width="200px">
        </a>
    </body>
</html>
```

Note: Place "vodafone.jpg" in the current folder.

Example on <a> with internal links

```
<html>
    <head>
        <title>internal links</title>
    </head>
    <body>
        <a href="#first">India</a>
        <a href="#second">UK</a>
        <a href="#third">US</a>

        <h1 id="first">India</h1>
        <p>India, officially the Republic of India, is a country in South Asia. It is the seventh-largest
country by area, the second-most populous country with over 1.2 billion people, and the most populous
democracy in the world. Bounded by the Indian Ocean on the south, the Arabian Sea on the south-west,
and the Bay of Bengal on the south-east, it shares land borders with Pakistan to the west; China, Nepal,
and Bhutan to the north-east; and Burma and Bangladesh to the east. In the Indian Ocean, India is in the
vicinity of Sri Lanka and the Maldives; in addition, India's Andaman and Nicobar Islands share a
maritime border with Thailand and Indonesia.</p>
        <h1 id="second">UK</h1>
        <p>The United Kingdom of Great Britain and Northern Ireland, commonly known as the United
Kingdom (UK) and Britain, is a sovereign state located off the north-western coast of continental Europe.
The country includes the island of Great Britain, the north-eastern part of the island of Ireland, and
many smaller islands. Northern Ireland is the only part of the UK that shares a land border with another
```

state—the Republic of Ireland. Apart from this land border, the UK is surrounded by the Atlantic Ocean in the west and north, the North Sea in the east, the English Channel in the south and the Irish Sea in the west.

<h1 id="third">US</h1>

The United States of America (USA), commonly called the United States (US or U.S.) and America, is a federal constitutional republic consisting of fifty states and a federal district. The country is situated mostly in central North America, where its forty-eight contiguous states and Washington, D.C., the capital district, lie between the Pacific and Atlantic Oceans, bordered by Canada to the north and Mexico to the south. The state of Alaska is in the northwest of the continent, with Canada to the east and Russia to the west across the Bering Strait. The state of Hawaii is an archipelago in the mid-Pacific. The country also possesses several territories in the Pacific and Caribbean. At 3.79 million square miles (9.83 million km2) and with around 315 million people, the United States is the third- or fourth-largest country by total area, and the third-largest by both land area and population. It is one of the world's most ethnically diverse and multicultural nations, the product of large-scale immigration from many countries. The geography and climate of the United States is also extremely diverse and is home to a variety of species.

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

Example on <a> with page navigation

india.html

```
<html>
    <head>
        <title>India</title>
    </head>
   <body>
        <h1>India</h1>
        <a href="india.html">India</a>
        <a href="uk.html">UK</a>
        <a href="us.html">US</a>
        <p>India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, the second-most populous country with over 1.2 billion people, and the most populous democracy in the world. p>
    </body>
</html>
```

uk.html

```
<html>
    <head>
        <title>UK</title>
    </head>
   <body>
        <h1>UK</h1>
        <a href="india.html">India</a>
        <a href="uk.html">UK</a>
        <a href="us.html">US</a>
        <p>The United Kingdom of Great Britain and Northern Ireland, commonly known as the United Kingdom (UK) and Britain, is a sovereign state located off the north-western coast of continental Europe. p>
    </body>
```

```
</html>
```

us.html

```
<html>
    <head>
        <title>US</title>
    </head>
   <body>
        <h1>US</h1>
        <a href="india.html">India</a>
        <a href="uk.html">UK</a>
        <a href="us.html">US</a>
        <p>The United States of America (USA), commonly called the United States (US or U.S.) and America, is a federal constitutional republic consisting of fifty states and a federal district.</p>
    </body>
</html>
```

List tags

Unordered List

- tag is used to display the list of items with bullets.
- UL stands for “Un-Ordered List”.
- Inside <ul> tag, you should place one or more <li> tags. tag stands for "List Item", which represents a single item in the list.
- Use <ul> tag if you want to display any names such as person names, country names, city names etc.
- It is a paired tag.

Syntax:

```
<ul>
    <li>text here</li>
    <li>text here</li>
    …
</ul>
```

Example:

```
<ul>
    <li>India</li>
    <li>UK</li>
    <li>US</li>
</ul>
```

**Example on **

```
<html>
  <head>
    <title>Unordered List</title>
  </head>
 <body>
    <h1>Unordered List</h1>
    <ul>
      <li>India</li>
      <li>UK</li>
      <li>US</li>
      <li>China</li>
    </ul>
  </body>
</html>
```

Ordered List

- OL stands for "Ordered List".
- It is used to display the list of items with numbers.
- Inside <ol> tag, you should place one or more <li> tags. The <li> tag represents a single item in the list.
- Use <ol> tag if you want to display names with numbers. For example, list of academic subjects (in order).
- It is a paired tag.

Syntax:

```
<ol>
  <li>text here</li>
  <li>text here</li>
  …
</ol>
```

Example:

```
<ol>
  <li>India</li>
  <li>UK</li>
  <li>US</li>
</ol>
```

**Example on **

```
<html>
  <head>
    <title>Ordered List</title>
  </head>
 <body>
```

```
        <h1>Ordered List</h1>
        <ol>
            <li>India</li>
            <li>UK</li>
            <li>US</li>
            <li>China</li>
        </ol>
    </body>
</html>
```

Definition List

- DL stands for “Definition List”.
- <dl> tag is used to display a collection of definitions.
- Inside <dl> tag, you should place one or more <dt> and <dd> tags.
- It is a paired tag.
- <dt> and <dd> tags are also paired tags. DT stands for "Definition Title".
- <dt> is used to specify definition title.
- <dd> is used to specify actual definition. DD stands for "Definition Data".

Syntax:

```
<dl>
    <dt>title here</dt>
    <dd>definition here</dd>
    <dt>title here</dt>
    <dd>definition here</dd>
    …
</dl>
```

Example:

```
<dl>
    <dt>Computer</dt>
    <dd>Computer definition here</dd>
    <dt>Car</dt>
    <dd>Car definition here</dd>
</dl>
```

Example on <dl>

```
<html>
    <head>
        <title>Definition List</title>
    </head>
    <body>
        <h1>Definition List</h1>
        <dl>
            <dt>HTML</dt>
            <dd>HTML is used to create elements in the web page.</dd>
            <dt>CSS</dt>
```

```
        <dd>CSS is used to apply styles in the web page.</dd>
        <dt>JavaScript</dt>
        <dd>JavaScript is used to create functionality in the web page.</dd>
    </dl>
  </body>
</html>
```

Tables

Table Tags

- <table> tag is used to display table type of data in the web page.
- A table is a collection of rows. Each row is a collection of cells.
- A table is represented as <table> tag; A row is represented as <tr>; A cell is represented as <td>.
- Inside the <table> tag, we have to use <tr>; Inside the <tr> tag, we have to use <td>.
- If the cell is representing the column heading, you can use <th> tag, instead of <td> tag.
- <caption> tag is used to specify a title for the table.
- “tr” stands for “Table row”.
- “td” stands for “Table data”.
- “th” stands for “Table header”.
- <table>, <tr>, <th>, <td> and <caption> tags are paired tags.

Syntax:

```
<table>
    <tr>
        <td>data here</td>
        <td>data here</td>
        …
    </tr>
    …
</table>
```

Example:

```
<table border="1">
    <tr>
        <td>One</td>
        <td>Two</td>
    </tr>
    <tr>
        <td>Three</td>
        <td>Four</td>
```

```
    </tr>
    <tr>
        <td>Five</td>
        <td>Six</td>
    </tr>
</table>
```

Attributes of <table> tag:

1. **border**
   - **"0":** No border
   - **"1":** with border

Attributes of <td> or <th> tag:

1. **rowspan**
   - **"n":** Specifies the no. of rows to merge. The current cell occupies the space of 'n' no. of cells.
2. **colspan**
   - **"n":** Specifies the no. of columns to merge. The current cell occupies the space of 'n' no. of cells.

Example on simple table

```
<html>
    <head>
        <title>Table - Basic Example</title>
    </head>
    <body>
        <h1>Table - Basic Example</h1>
        <table border="1">
            <tr>
                <td>One</td>
                <td>Two</td>
            </tr>
            <tr>
                <td>Three</td>
                <td>Four</td>
            </tr>
            <tr>
                <td>Five</td>
                <td>Six</td>
            </tr>
        </table>
    </body>
</html>
```

Realtime Example on table

```
<html>
    <head>
```

```
        <title>Table - Students</title>
    </head>
    <body>
        <h1>Table - Students</h1>
        <table border="1">
            <caption>Students</caption>
            <tr>
                <th>Sl. No</th>
                <th>Name</th>
                <th>Marks</th>
            </tr>
            <tr>
                <td>1</td>
                <td>John</td>
                <td>89</td>
            </tr>
            <tr>
                <td>2</td>
                <td>Scott</td>
                <td>45</td>
            </tr>
            <tr>
                <td>3</td>
                <td>Allen</td>
                <td>64</td>
            </tr>
        </table>
    </body>
</html>
```

Example on colspan attribute

```
<html>
    <head>
        <title>Table - Colspan</title>
    </head>
    <body>
        <h1>Table - Colspan</h1>
        <table border="1">
            <tr>
                <th colspan="2">Phone</th>
            </tr>
            <tr>
                <td>Mobile</td>
                <td>Landline</td>
            </tr>
            <tr>
                <td>9982398231</td>
                <td>22938432</td>
            </tr>
        </table>
    </body>
```

```
</html>
```

Example on rowspan attribute:

```
<html>
    <head>
        <title>Table - Rowspan</title>
    </head>
   <body>
        <h1>Table - Rowspan</h1>
        <table border="1">
            <tr>
                <th rowspan="2">Phone</th>
                <td>Mobile</td>
                <td>9982398231</td>
            </tr>
            <tr>
                <td>Landline</td>
                <td>22938432</td>
            </tr>
        </table>
    </body>
</html>
```

Miscellaneous Tags

IFrame

- <iframe> tag is used to display another web page or web site within the current web page.
- Iframe stands for “inline frame”.
- <iframe> is a paired tag.

Syntax:

```
<iframe src="web site address here" width="n px" height="n px">
</iframe>
```

Example:

```
<iframe src="http://www.airtel.in" width="400px" height="300px">
</iframe>
```

Attributes of <iframe> tag:

1. **src**
   - **“web site path”:** Specifies the web site or web page path that is to be displayed in the iframe.
2. **width**
   - **“n px”:** Specifies the horizontal size of the iframe.
3. **height**
   - **“n px”:** Specifies the vertical size of the iframe.
4. **frameborder**

- **“n px”:** Specifies border of the iframe.

Example on <iframe>

```
<html>
    <head>
        <title>Iframe</title>
    </head>
   <body>
        <h1>Iframe</h1>
        <iframe src="http://www.lipsum.com" width="400px" height="300px">
        </iframe>
    </body>
</html>
```

Example on Youtube <iframe>

```
<html>
    <head>
        <title>Iframe - Youtube</title>
    </head>
   <body>
        <h1>Iframe - Youtube</h1>
        Enjoy the video:<br>
        <iframe width="560" height="315" src="https://www.youtube.com/embed/EJmlCNGGzdo"
frameborder="0" allowfullscreen></iframe>
    </body>
</html>
```

Example on navigation with <iframe>

Index.html

```
<html>
    <head>
        <title>Index</title>
    </head>
    <body>
        <h1>Index</h1>
        <a href="home.html" target="myiframe">Home</a>
        <a href="about.html" target="myiframe">About</a>
        <a href="contact.html" target="myiframe">Contact</a>
        <br>
        <iframe name="myiframe" width="100%" height="400px" src="home.html"></iframe>
    </body>
</html>
```

home.html

```
<html>
    <head>
        <title>Home</title>
```

```
    </head>
    <body>
        <h1>Home page</h1>
    </body>
</html>
```

about.html

```
<html>
    <head>
        <title>About</title>
    </head>
    <body>
        <h1>About page</h1>
    </body>
</html>
```

contact.html

```
<html>
    <head>
        <title>Contact</title>
    </head>
    <body>
        <h1>Contact Page</h1>
    </body>
</html>
```

HTML Entities

- HTML Entities are pre-defined codes for displaying special symbols within the web page.
- HTML Entities are case sensitive. These must be used in lower case only.

Result	Description	Entity Name	Entity Number
[space]	non-breaking space		
<	less than	&lt;	&#60;
>	greater than	&gt;	&#62;
&	ampersand	&amp;	&#38;
¢	cent	&cent;	&#162;
£	pound	&pound;	&#163;
¥	yen	&yen;	&#165;
€	euro	&euro;	&#8364;
§	section	&sect;	&#167;
©	copyright	&copy;	&#169;
®	registered trademark	&reg;	&#174;

™	trademark	&trade;	&#8482;

Example on HTML Entities

```
<html>
    <head>
        <title>Entities</title>
    </head>
   <body>
        <h1>Entities</h1>
        <h1>HTML entities</h1>
        hai          hello<br>
        &amp;nbsp;<br>
        hai&amp;nbsp;&amp;nbsp;&amp;nbsp;hello<br>
        &amp;reg;<br>
        &amp;copy;<br>
        &amp;trade;<br>
        &amp;pound;<br>
        &amp;#8377;<br>
        &amp;cent;<br>
        &amp;lt;<br>
        &amp;gt;<br>
        &amp;amp;<br>
    </body>
</html>
```

Meta

- <meta> tag is used to specify meta data (additional details) about the web page.
- <meta> tag provides information about the web page. Google like search engines will display your web page in the google search results, whenever one or more keywords are matching. You must upload your web page in the internet server to really use this.
- <meta> tag is an unpaired tag.

Example:

```
<meta name="keywords" content="Sony, Television, Price, Hyderabad">
<meta name="description" content="Sony LED BRAVIA Prices">
<meta name="author" content="Harsha">
```

Example on <meta> tag

```
<html>
    <head>
        <title>Meta</title>
        <meta name="keywords" content="ui technologies, full, free, material, html, css, javascript, jquery, angularjs, bootstrap">
        <meta name="description" content="This web page contains full UI Technologies material">
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    </head>
    <body>
        <h1>Meta</h1>
```

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

Forms

Form

- <form> tag is used to group-up the input elements; so that we can submit the entire form to the server.
- A form is a collection of form elements such as <input>, <textarea> and <select>.
- Forms are used to collect information from the user. For example, registration for collects user details such as username, email, password, mobile number etc.
- <form> is a paired tag.

Syntax:

```
<form action="server page address here" method="get">
        form elements here
</form>
```

Example:

```
<form action="http://localhost/serverpage.aspx" method="get">
</form>
```

Attributes of <form> tag:

1. **action:** Used to specify the server page address to which the form is to be submitted.
2. **method:**
   - **get:**
     - Displays the parameter names and values in the browser's address bar.
     - Useful for searching and retrieving the data from database.
   - **post**
     - Hides the parameter names and values in the browser's address bar and allows you to pass the data in hidden format.
     - Useful for insert, update, delete, registration and login operations.
     - The "get" and "post" can be explained in server side courses like Java, .NET, PHP, NodeJS etc.

Example on <form> tag

```
<html>
    <head>
        <title>form</title>
    </head>
    <body>
        <h1>form</h1>
        <form>
            form elements here
```

```
        </form>
    </body>
</html>
```

Input Tag

- <input> tag is used to create a form control (form element).
- <input> tag can create form elements such as Textbox, Checkbox, Radio button, Browser button, submit button etc.
- <input> is an unpaired tag.

Syntax:

<input type="option here">

Example:

<input type="text">

Attributes of <input> tag:

1. **type:** text | password | checkbox | radio | file | reset | submit | image | button | hidden | color | date | time | datetime-local | month | week | week | number | range | email | url
2. maxlength
3. value
4. readonly
5. disabled
6. tabindex
7. name
8. id
9. src
10. width
11. height
12. checked
13. placeholder
14. autofocus
15. required
16. pattern
17. min

18. max
19. step
20. formaction
21. formmethod
22. formtarget
23. form
24. multiple

TextBox

- Textbox is used to accept a string value from the user.
- Textbox can accept email, mobile etc.
- The user can enter any no. of characters in the textbox.

Syntax: <input type="text">

Example on Textbox

```
<html>
    <head>
        <title>textbox</title>
    </head>
    <body>
        <h1>textbox</h1>
        <form>
            Name <input type="text"><br>
            Mobile <input type="text"><br>
            Email <input type="text">
        </form>
    </body>
</html>
```

Password TextBox

- Password textbox is used to accept a password from the user.
- Each character in the password textbox will be displayed as "." (dot).

Syntax: <input type="password">

Example on Password TextBox

```
<html>
  <head>
    <title>password</title>
  </head>
  <body>
    <h1>password</h1>
    <form>
```

```
        Password <input type="password">
      </form>
    </body>
  </html>
```

CheckBox

- Checkbox is used to display Yes/No type of option to the user.
- If the checkbox is checked, it means "yes". If the checkbox is unchecked, it means "no".

Syntax: <input type="checkbox">

Example on CheckBox

```
<html>
<head>
    <title>checkbox</title>
</head>
<body>
    <h1>checkbox</h1>
     <form>
        <input type="checkbox">I accept license agreement
     </form>
</body>
</html>
```

Checked Attribute

- The "checked" attribute of checkbox makes the checkbox by default checked, while opening the page. Of course, the user can uncheck it later, if required.
- This attribute has only one value, i.e. "checked".

Example on CheckBox - Checked

```
<html>
    <head>
        <title>checkbox - checked</title>
    </head>
    <body>
        <h1>checkbox - checked</h1>
        <form>
                <input type="checkbox" checked="checked">I accept license agreement
        </form>
    </body>
</html>
```

Radio Button

- Radio button is used to display two or more options to the user and allow the user to select any one of them. **Ex:** Gender - Male, Female

- The "name" attribute specifies common name of radio butotns. The “name” of radio buttons should be same to group-up them. Within a group of radio buttons, only one radio button can be selected by the user at-a-time.

  Syntax: <input type="radio">

Example on Radio Button

```
<html>
    <head>
        <title>radio</title>
    </head>
    <body>
        <h1>radio</h1>
        <form>
            Bank account type:
            <input type="radio" name="bankaccount">Savings account
            <input type="radio" name="bankaccount">Current account
            <input type="radio" name="bankaccount">Loan account
        </form>
    </body>
</html>
```

Example on Radio Button - Checked

```
<html>
    <head>
        <title>radio - checked</title>
    </head>
    <body>
        <h1>radio - checked</h1>
        <form>
            Bank account type:
            <input type="radio" name="bankaccount" checked="checked">Savings account
            <input type="radio" name="bankaccount">Current account
            <input type="radio" name="bankaccount">Loan account
        </form>
    </body>
</html>
```

File Browse Button

- Browse button is used for “attachment” option.
- For example, you can upload image files / documents in gmail or facebook .

  Syntax: <input type="file">

Example on File Browse Button

```
<html>
    <head>
        <title>file browse</title>
    </head>
```

```
    <body>
        <h1>file browse</h1>
        <form>
            Attachment <input type="file">
        </form>
    </body>
</html>
```

Reset Button

- Reset button clears the values of all fields (textboxes and others) within the current form.
- The reset button must be a part of the <form> tag; then only it can recognize the elements that are present inside the same form.

Syntax: <input type="reset">

Example on reset button

```
<html>
    <head>
        <title>reset</title>
    </head>
    <body>
        <h1>reset</h1>
        <form>
            Name <input type="text"><br>
            Mobile <input type="text"><br>
            Email <input type="text"><br>
            Password <input type="password"><br>
            <input type="checkbox">I accept license agreement<br>
            Gender:
            <input type="radio" name="gender">Male
            <input type="radio" name="gender">Female<br>
            Photo:
            <input type="file"><br>
            <input type="reset" value="Clear">
        </form>
    </body>
</html>
```

Submit Button

- Submit button is used to submit the form to the server page.
- While submitting the form, all the input parameter names and their values will be sent to the server program as "query string". Ex: ?param1=value&param2=value.
- The query string shown above contains "names" and "values". The names are specified by the developer by using "name" attribute; "value" will be entered by the user.
- The server program receives the submitted values and do some process such as storing the data into database.

- For every form, it is recommended to create a submit button. But the <input type="submit"> creates a normal (simple submit button)

  Syntax: <input type="submit">

Example on Submit button

```
<html>
    <head>
        <title>Submit</title>
    </head>
    <body>
        <h1>Submit</h1>
        <form action="http://localhost/someaddress">
            Firstname <input type="text" name="firstname"><br>
            Lastname <input type="text" name="lastname"><br>
            Mobile <input type="text" name="mobile"><br>
            Email <input type="text" name="email"><br>
            Password <input type="password" name="password"><br>
            <input type="checkbox" name="license">I accept license agreement<br>
            Gender:
            <input type="radio" name="gender">Male
            <input type="radio" name="gender">Female<br>
            Photo:
            <input type="file" name="attachment"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

Name Attribute

- Name attribute represents programmatic name of the input element that will be submitted to the server. Based on the “name”, we can get the value of the element in the server side program.

  Syntax: <input type="…" name="any name">

Login Form

- Login form contains username and password with a submit button.

Example on Login Form

```
<html>
    <head>
        <title>login</title>
    </head>
    <body>
        <h1>Login</h1>
        <form action="http://localhost/someaddress">
            Username: <input type="text" name="username"><br>
            Password: <input type="password" name="password"><br>
            <input type="submit" value="Login">
```

```
        </form>
    </body>
</html>
```

Registration Form

- Registration form contains username, password and other fields with a submit button.

Example on Registration Form

```
<html>
    <head>
        <title>Registration</title>
    </head>
    <body>
        <h1>Registration</h1>
        <form action="http://localhost/someaddress">
            <table>
                <tr>
                    <td>Name:</td>
                    <td><input type="text" name="personname"></td>
                </tr>
                <tr>
                    <td>Password:</td>
                    <td><input type="password" name="password"></td>
                </tr>
                <tr>
                    <td>News Letters:</td>
                    <td><input type="checkbox" name="newsletters" value="yes"></td>
                </tr>
                <tr>
                    <td>Gender:</td>
                    <td>
                        <input type="radio" name="gender" value="m">Male
                        <input type="radio" name="gender" value="f">Female
                    </td>
                </tr>
                <tr>
                    <td>Photo:</td>
                    <td><input type="file" name="photo"></td>
                </tr>
                <tr>
                    <td><input type="submit" value="Register"></td>
                    <td><input type="reset"></td>
                </tr>
            </table>
        </form>
    </body>
</html>
```

Post Submission

- Form submission is of two types: Get | POST.

- In case of "Get" submission, the parameters are visible in the browser's location bar.
- In case of "Post" submission, the parameters are sent in secret mode; and are not visible in the browser's location bar.

Example on Post Submit Button

```
<html>
    <head>
        <title>Post</title>
    </head>
    <body>
        <h1>Post</h1>
        <form action="http://localhost/someaddress" method="post">
            Username:
            <input type="text" name="username"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

Image Submit Button

- "Image Submit button" is used to submit the form to the server page.
- When the user clicks on the image submit button
- It also submits the clicked position (X and Y co-ordinates) will be submitted to the server, along with the values of other form elements.
- The "src" attribute is used to specify the path of the image in case of <input type="image">. It by default, refers to the same folder.
- It is recommended to place the image file in the same folder, where the html file is present.
- The "width" attribute is used to specify the width of the image in case of <input type="image">. It represents the value in the form of pixels.
- The "height" attribute is used to specify the height of the image in case of <input type="image">. It represents the value in the form of pixels.

  Syntax: <input type="image" src="filename.extension">

Example on Image Submit Button

```
<html>
    <head>
        <title>Image</title>
    </head>
    <body>
        <h1>Image</h1>
        <form action="http://localhost/someaddress">
            Username:
            <input type="text" name="username"><br>
            <input type="image" src="ok.png" width="20px" height="20px">
```

```
        </form>
    </body>
</html>
```

Note: Copy and paste “ok.png” into “c:\html” folder.

General Button

- When the user clicks on the general button, nothing happens by default, but you can call “JavaScript Click event” when the user clicks on the button.

  Syntax: <input type="button" value="some text">

Example on General Button

```
<html>
    <head>
        <title>Button</title>
    </head>
    <body>
        <h1>Button</h1>
        <input type="button" value="OK">
    </body>
</html>
```

Hidden Field

- Hidden field will not be appear in the web page; but will be submitted to the server.
- Hidden values are useful for the programmer to send browser / client details to server.
- Use hidden field when you want to submit some fixed value to server; that you don't want to accept from the user

  Syntax: <input type="hidden" name="some name" value="some value">

Example on hidden field

```
<html>
    <head>
        <title>Hidden</title>
    </head>
    <body>
        <h1>Hidden</h1>
        <form action="http://localhost/someaddress">
            <input type="hidden" name="x" value="100">
            <input type="submit">
        </form>
    </body>
</html>
```

Color

- Used to create a color box, where the user can select a color.

- The selected color can be applied to any element, by using JavaScript.

  Syntax: <input type="color">

Example on Color

```
<!DOCTYPE html>
<html>
<head>
  <title>color</title>
</head>
<body>
  <h1>color</h1>
  <form action="http://localhost/someaddress">
    Color: <input type="color" name="x"><br>
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

Date

- Used to create a date box (date picker / popup calendar), where the user can select a date.
- The browser by default provides a built-in date picker.
- It lacks of customization; for example, you can't block some dates in the calendar etc.

  Syntax: <input type="date">

Example on Date

```
<!DOCTYPE html>
<html>
    <head>
        <title>date</title>
    </head>
    <body>
        <h1>date</h1>
        <form action="http://localhost/someaddress">
            Date: <input type="date" name="x"><br>
            <input type="submit" value="Submit">
        </form>
    </body>
</html>
```

Time

- Used to create a time box, where the user can enter time (hours, minutes and seconds).

  Syntax: <input type="time">

Example on Time

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

```
    <head>
        <title>time</title>
    </head>
    <body>
        <h1>time</h1>
        <form action="http://localhost/someaddress">
            Time: <input type="time" name="x"><br>
            <input type="submit" value="Submit">
        </form>
    </body>
</html>
```

Datetime-Local

- Used to create a date-cum-time box, where the user can select a date and time also.

  Syntax: <input type="datetime-local">

Example on New Input Types – Datetime-local

```
<!DOCTYPE html>
<html>
  <head>
    <title>datetime-local</title>
  </head>
  <body>
    <h1>datetime-local</h1>
    <form action="http://localhost/someaddress">
      Date and Time: <input type="datetime-local" name="x"><br>
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

Month

- Used to create a month box, where the user can select a month.

  Syntax: <input type="month">

Example on Month

```
<!DOCTYPE html>
<html>
    <head>
        <title>Month</title>
    <body>
        <h1>Month</h1>
        <form action="http://localhost/someaddress">
            Month: <input type="month" name="x"><br>
            <input type="submit" value="Submit">
        </form>
    </body>
</html>
```

Week

- Used to create a week box, where the user can select a week.

  Syntax: <input type="week">

Example on Week

```
<!DOCTYPE html>
<html>
  <head>
    <title>week</title>
  </head>
  <body>
    <h1>week</h1>
    <form action="http://localhost/someaddress">
      Week: <input type="week" name="x"><br>
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

Search

- Used to create a search box, where the user can enter some search text. It also displays clear button (X) to clear the text inside the search box.

  Syntax: <input type="search">

Example on Search

```
<!DOCTYPE html>
<html>
  <head>
    <title>search</title>
  </head>
  <body>
    <h1>search</h1>
    <form action="http://localhost/someaddress">
      Search: <input type="search" name="x"><br>
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

Number

- Used to create a numerical textbox, where the user can enter some number.
- It either prevents typing alphabets and special symbols, or shows error message when alphabets / special symbols are entered.
- Some browsers also display increase / decrease buttons for the number textbox.

  Syntax: <input type="number">

Example on Number

```
<!DOCTYPE html>
<html>
  <head>
    <title>number</title>
  </head>
  <body>
    <h1>number</h1>
    <form action="http://localhost/someaddress">
      Number: <input type="number" name="x"><br>
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

Range

- Used to create a slider, based on the specific range (minimum to maximum). Ex: price range, volume.

  Syntax: <input type="range" min="minimum" max="maximum">

Example on Range

```
<!DOCTYPE html>
<html>
  <head>
    <title>range</title>
  </head>
  <body>
    <h1>range</h1>
    <form action="http://localhost/someaddress">
      Range: <input type="range" name="x" min="0" max="5"><br>
      <input type="submit" value="Submit">
    </form>
  </body>
</html>
```

Email

- Used to create an email textbox, where the user can enter a valid email address.
- It displays error message automatically, if the given email address is invalid.

  Syntax: <input type="email">

Example on Email

```
<!DOCTYPE html>
<html>
  <head>
    <title>email</title>
  </head>
  <body>
```

```
        <h1>email</h1>
        <form action="http://localhost/someaddress">
            Email: <input type="email" name="x"> <br>
            <input type="submit" value="Submit">
        </form>
    </body>
</html>
```

Url

- Used to create a url textbox, where the user can enter a valid website url. Ex: http://www.google.com

  Syntax: <input type="url">

Example on Url

```
<!DOCTYPE html>
<html>
    <head>
        <title>url</title>
    </head>
    <body>
        <h1>url</h1>
        <form>
            URL: <input type="url" name="x"><br>
            <input type="submit" value="Submit">
        </form>
    </body>
</html>
```

Maxlength Attribute

- Specifies the maximum no. of characters that can be typed in the textbox.
- By default, the user can enter unlimited no. of characters in the textbox.
- Use "Maxlength" if you want to limit to specific no. of characters.

  Syntax: <input type="text" maxlength="n">

Example on Maxlength

```
<html>
    <head>
        <title>Maxlength</title>
    </head>
    <body>
        <h1>Maxlength</h1>
        Person name (30 characters):
        <input type="text" maxlength="30">
    </body>
</html>
```

Value Attribute

- Represents the current value of the input element.
- In case of checkbox and radio button, the "value" must be set by the developer; in other type of inputs, the user enters the value.

Syntax: <input type="…" value="some value">

Example on Value

```
<html>
    <head>
        <title>Value</title>
    </head>
    <body>
        <h1>Value</h1>
        Bill amount:
        <input type="text" value="1000">
    </body>
</html>
```

Readonly Attribute

- Makes the textbox as readonly; so that the user can see the value but can't type anything the textbox.
- In the readonly textbox, the user can see, select, copy the text value.
- Cursor can be placed inside the readonly textbox.

Syntax: <input type="text" value="some value" readonly="readonly">

Example on readonly

```
<html>
    <head>
        <title>Readonly</title>
    </head>
    <body>
        <h1>Readonly</h1>
        Bill amount (readonly):
        <input type="text" value="1000" readonly="readonly">
    </body>
</html>
```

Disabled Attribute

- Used to disable the element. If the element (button, textbox, checkbox, radio button) is disabled, the user can't modify or touch the value of the element. The disabled element will be out of TAB sequence. That means cursor will not stop at the disabled element

Syntax: <input type="…" disabled="disabled">

Example on TextBox disabled

```
<html>
    <head>
        <title>TextBox Disabled</title>
    </head>
    <body>
        <h1>TextBox Disabled</h1>
        Bill amount (disabled):
        <input type="text" value="1000" disabled="disabled">
    </body>
</html>
```

Example on Button disabled

```
<html>
    <head>
        <title>Disabled</title>
    </head>
    <body>
        <h1>Disabled</h1>
        <form action="http://localhost/someaddress">
            <input type="submit" disabled="disabled">
        </form>
    </body>
</html>
```

Tabindex Attribute

- Specifies tab order. It defines tab sequence.
- When the user presses TAB key on the keyboard, the cursor jumps to the next form element which is having next higher Tabindex.

  Syntax: <input type="text" tabindex="n">

Example on Tabindex

```
<html>
    <head>
        <title>Tabindex</title>
    </head>
    <body>
        <h1>Tabindex</h1>
        Name: <input type="text" tabindex="1"><br>
        Landline: <input type="text" tabindex="3">
        Mobile: <input type="text" tabindex="2">
    </body>
</html>
```

ID Attribute

- Represents identification name of the input element that can be used in html, css, and javascript to get the element programmatically.

- ID can be used to create <label> tag in html, ID selector in CSS, getElementById( ) function JavaScript etc.

  Syntax: <input type="…" id="your id">

ID Attribute

- Represents identification name of the input element that can be used in html, css, and javascript to get the element programmatically.
- ID can be used to create <label> tag in html, ID selector in CSS, getElementById( ) function JavaScript etc.

  Syntax: <input type="…" id="your id">

Placeholder Attribute

- Used to display watermark text in the textbox.
- The watermark text explains what value should be entered in the textbox.
- The watermark text automatically disappears if any value is entered in the textbox.

  Syntax: <input type="text" placeholder="any text">

Example on Placeholder

```
<!DOCTYPE html>
<html>
    <head>
        <title>placeholder</title>
    </head>
    <body>
        <h1>placeholder</h1>
        <form action="http://localhost/someaddress">
            <input type="text" placeholder="First Name" name="firstname"><br>
            <input type="text" placeholder="Last Name" name="lastname"><br>
            <input type="submit" value="Submit">
        </form>
    </body>
</html>
```

Autofocus Attribute

- Used to display the cursor directly in the specific textbox, when the web page opened in the browser.
- You can use the "autofocus" attribute only once in the web page.

  Syntax: <input type="…" autofocus="autofocus">

Example on Autofocus

```
<!DOCTYPE html>
<html>
    <head>
        <title>autofocus</title>
```

```
    </head>
    <body>
        <h1>autofocus</h1>
        <form action="http://localhost/someaddress">
            First name:<br>
            <input type="text" name="firstname" autofocus="autofocus"><br>
            Last name:<br>
            <input type="text" name="lastname"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

Required Attribute

- Used to make the field (for example, textbox) as mandatory.
- If the textbox is empty, it shows error message automatically and the form will not be submitted to server.

Syntax: <input type="…" required="required">

Example on Required

```
<!DOCTYPE html>
<html>
    <head>
        <title>required</title>
    </head>
    <body>
        <h1>required</h1>
        <form action="http://localhost/someaddress">
            Username:
            <input type="text" name="Username" required="required" title="Please enter your name">
            <input type="submit">
        </form>
    </body>
</html>
```

Pattern Attribute

- Used to apply a regular expression for the textbox for validation purpose.
- Regular expression represents “pattern” of the value.
- Ex: Alphabets only allowed, numbers only allowed etc.

Syntax: <input type="text" pattern="regular expression here">

Example on Pattern

```
<!DOCTYPE html>
<html>
    <head>
        <title>pattern</title>
```

```
    </head>
    <body>
        <h1>pattern</h1>
        <form action="http://localhost/someaddress">
            Person Name:<br>
            <input type="text" pattern="^[a-zA-Z ]*$" title="Only alphabets allowed"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

Min and Max Attributes

"min" attribute

- Specifies the minimum value that you want to accept.
- Applicable only for <input type=”number”>, <input type=”range”> and <input type=”date”>.

  Syntax: <input type="number | range | date" min="minimum value">

"max" attribute

- Specifies the maximum value that you want to accept.
- Applicable only for <input type=”number”>, <input type=”range”> and <input type=”date”>.

  Syntax: <input type="number | range | date" max="maximum value">

Example on Min and Max

```
<!DOCTYPE html>
<html>
    <head>
        <title>min and max</title>
    </head>
    <body>
        <h1>min and max</h1>
        <form action="http://localhost/someaddress">
            Quantity (between 1 and 50):<br>
            <input type="number" name="quantity" min="1" max="50">
            <br><br>
            DOB:<br>
            <input type="date" name="dateofbirth" min="1980-1-1" max="2017-12-31">
            <br><br>
            <input type="submit">
        </form>
    </body>
</html>
```

Step Attribute

- Specifies increment / decrement value for <input type=”number”>.
- If the user clicks on "up" button in number textbox, the "step" value will be increased.

- If the user clicks on "down" button in number textbox, the "step" value will be decreased.

  Syntax: <input type="number" step="value here">

Example on Step

```
<!DOCTYPE html>
<html>
    <head>
        <title>step</title>
    </head>
    <body>
        <h1>step</h1>
        <form action="http://localhost/someaddress">
            Amount: <input type="number" name="amount" step="10"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

Novalidate Attribute

- HTML supports built-in validations such as required, min, max, email, date, number etc.
- Disables the built-in HTML 5 validations, such as required, min, max, email, date, number etc.
- If "novalidate" attribute is applied and the user has entered invalid values in the textboxes, and click on "Submit" button, the form will be automatically submitted to the server, without any validation.

  Syntax 1: <form novalidate="novalidate">

  Syntax 2: <input type="submit" formnovalidate="novalidate">

Example on Novalidate

```
<!DOCTYPE html>
<html>
    <head>
        <title>novalidate</title>
    </head>
    <body>
        <h1>novalidate</h1>
        <form action="http://localhost/someaddress" novalidate="novalidate" method="get">
            <label for="txt1">E-mail:</label><br>
            <input type="email" name="email" id="txt1" required="required"><br>
            <label for="txt2">Age:</label><br>
            <input type="number" name="age" id="txt2" min="18" max="70"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

FormAction, FormMethod, FormTarget Attribute

FormAction Attribute

- "formaction" attribute specifies server url, to which you want to submit the form. If the user clicks on "Submit" button, the form data (parameter names and values) will be submitted to the url specified in "formaction" attribute.

  Syntax: <input type="submit" formaction="server url here">

FormMethod Attribute

- "formmethod" attribute specifies type of submission: GET or POST.
- In case of "GET" submission, the parameter names and values are visible in the browser's address bar.
- In case of "POST" submission, the parameter names and values are not visible in the browser's address bar, but will be sent in hidden mode (request body).

  Syntax: <input type="submit" formmethod="get | post">

FormTarget Attribute

- "formtarget" attribute opens the server page in separate browser tab, after submission.

  Syntax: <input type="submit" formtarget="_blank">

Example on Formaction Formmethod Formtarget

```
<!DOCTYPE html>
<html>
    <head>
        <title>formaction formmethod formtarget</title>
    </head>
    <body>
        <h1>formaction formmethod formtarget</h1>
        <form>
            First name:<br>
            <input type="text" name="FirstName"><br>
            Last name:<br>
            <input type="text" name="LastName"><br>
            <input type="submit" formaction="http://localhost/someaddress" formmethod="get"
formtarget="_blank" value="Submit">
        </form>
    </body>
</html>
```

Form Attribute

- Used to make the the element as a member of the form; so that the element also will be submitted along with other elements, while submitting the form.
- We have to specify ID of the <form> tag in "form" attribute of <input> tag.

Syntax: <input type="…" form="id of form">

Example on Form

```
<!DOCTYPE html>
<html>
    <head>
        <title>form</title>
    </head>
    <body>
        <h1>form</h1>
        <form action="http://localhost/someaddress" id="form1">
            First name:<br>
            <input type="text" name="firstname" id="txt1"><br>
            <input type="submit" value="Submit">
        </form>
        <hr>
        Last name (logically member of form1):<br>
        <input type="text" name="lastname" id="txt2" form="form1">
    </body>
</html>
```

Multiple Attribute

- By default, the user can select only one file in <input type="file">.
- The "multiple" attribute allows the user select multiple files in the File Browse Button, created using <input type="file">.

Syntax: <input type="file" form="multiple">

Example on Multiple

```
<!DOCTYPE html>
<html>
    <head>
        <title>multiple</title>
    </head>
    <body>
        <h1>multiple</h1>
        <form action="http://localhost/someaddress">
            Select images:
            <input type="file" name="myfiles" multiple="multiple"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

AutoComplete Attribute

- Browser automatically stores the history of textbox values when the form is submitted; and display the same in the textbox that has same name.
- The "autocomplete" attribute disables this feature, for security purpose.

Syntax 1: <input type="text" autocomplete="off">

Syntax 2: <form autocomplete="off"> </form>

Example on Autocomplete

```
<!DOCTYPE html>
<html>
    <head>
        <title>autocomplete</title>
    </head>
    <body>
        <h1>autocomplete</h1>
        <form action="http://localhost/someaddress">
            First name:<br>
            <input type="text" name="firstname"><br>
            Last name:<br>
            <input type="text" name="lastname" autocomplete="off"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

Button Tag

- <button> tag is used to display a submit button with image and text.
- The <button> tag is a contain, inside which you can place any content text or other html tags.
- <button> is a paired tag.

Syntax:

```
<button>
    <img src="image file path here">
    Text here
</button>
```

Example:

```
<button>
    <img src="ok.png"><br>OK
</button>
```

Example on <button> tag

```
<html>
    <head>
        <title>Button Tag</title>
    </head>
    <body>
        <h1>Button Tag</h1>
        <form action="http://localhost/someaddress">
            Username:
            <input type="text" name="username"><br>
```

```
            <button><img src="tick.png" width="25px"><br>OK</button>
        </form>
    </body>
</html>
```

Note: Place "tick.png" in the current folder.

Fieldset

- <fieldset> tag is used to display a box around a set of fields.
- <fieldset> tag is a paired tag.
- Fieldset if used to group-up the set of form elements. For example, all the personal details of the user such as Firstname, Lastname, Email, Mobile etc., can be placed inside the fieldset.
- Inside a <form> tag, we can place any no. of <fieldset> tags.

Syntax:

```
<fieldset>
    Your textboxes here
</fieldset>
```

Example:

```
<fieldset>
        <input type="text"><br>
        <input type="text">
</fieldset>
```

Example on Fieldset

```
<html>
    <head>
        <title>Fieldset</title>
    </head>
    <body>
        <h1>Fieldset</h1>
        <form action="http://localhost/someaddress">
            <fieldset>
                Username: <input type="text"><br>
                Password: <input type="password"><br>
                <input type="submit" value="Submit">
            </fieldset>
        </form>
    </body>
</html>
```

Legend

- <legend> tag is used to display a title for the fieldset.
- <legend> tag is a paired tag.
- <legend> tag can be used only inside the <fieldset>.

Syntax:

```
<fieldset>
    <legend>title here</legend>
    Your textboxes here
</fieldset>
```

Example:

```
<fieldset>
    <legend>User details</legend>
    <input type="text"><br>
    <input type="text">
</fieldset>
```

Example on Legend

```
<html>
    <head>
        <title>Fieldset and Legend</title>
    </head>
    <body>
        <h1>Fieldset and Legend</h1>
        <form action="http://localhost/someaddress">
            <fieldset>
                <legend>Login</legend>
                Username: <input type="text"><br>
                Password: <input type="password"><br>
                <input type="submit" value="Submit">
            </fieldset>
        </form>
    </body>
</html>
```

Label

- <label> tag is used to create field heading.
- The label provides description for the textbox, what value should be entered in the textbox.
- When the user clicks on the label, cursor will be appeared in the associated textbox automatically.
- <label> is a paired tag.

Syntax:

```
<label for="id of textbox here">label text here</label>
```

Example:

```
<label for="txt1">Username</label>
```

Attributes of <label> tag:

1. **for:** Used to specify the id of the textbox that is associated with the textbox.

Example on <label> tag

```
<html>
    <head>
        <title>Label</title>
    </head>
    <body>
        <h1>Label</h1>
        <form action="http://localhost/someaddress">
            <label for="txt1">First Name:</label>
            <input type="text" id="txt1"><br>
            <label for="txt2">Last Name:</label>
            <input type="text" id="txt2"><br>
            <input type="submit">
        </form>
    </body>
</html>
```

DropdownList

- <select> tag is used to create a dropdownlist or listbox.
- DropDownList is used to display few options to the user and allow the user to select any one of them.
- ListBox is used to display few options to the user and allow the user to select one or more of them.
- Inside <select> tag, you should use <option> tags. Each <option> tag represents an option in the dropdownlist.
- <select>, <option> tags are paired tags.

Syntax:

```
<select name="name here">
    <option value="short name here">Full name here</option>
        <option value="short name here">Full name here</option>
    …
</select>
```

Example:

```
<select name="PaymentMode">
    <option value="Select">Select</option>
    <option value="DC">Debit Card</option>
    <option value="CC">Credit Card</option>
    <option value="NB">Net Banking</option>
</select>
```

Attributes of <select> tag:

1. **multiple="multiple":** Used to convert the dropdownlist as listbox.

Example on Dropdownlist

```
<html>
    <head>
```

```
        <title>Dropdownlist</title>
    </head>
    <body>
        <h1>Dropdownlist</h1>
        <form action="http://localhost/someaddress">
            Country:
            <select name="country">
                <option>Please Select</option>
                <option>India</option>
                <option>China</option>
                <option>UK</option>
                <option>USA</option>
                <option>Japan</option>
            </select>
            <br>
            <input type="submit">
        </form>
    </body>
</html>
```

Option Groups

- <optgroup> tag is used to group-up the <option> tags inside the <select> tag.
- The <select> tag can contain many <optgroup> tags; the <optgroup> tag contains many <option> tags.
- Use <optgroup> tag, if you have too many no. of options in the dropdownlist.

Example on Optgroup

```
<html>
    <head>
        <title>Optgroup</title>
    </head>
    <body>
        <h1>Optgroup</h1>
        <form action="http://localhost/someaddress">
            Bank:
            <select name="bank">
                <optgroup label="Top Banks">
                    <option value="icic">ICICI Bank</option>
                    <option value="hdfc">HDFC Bank</option>
                    <option value="sbi">State Bank of India</option>
                </optgroup>
                <optgroup label="Other Banks">
                    <option value="axis">Axis Bank</option>
                    <option value="cb">Canara Bank</option>
                    <option value="boi">Bank of India</option>
                    <option value="iob">Indian Overseas Bank</option>
                </optgroup>
            </select>
            <br>
```

```
            <input type="submit">
        </form>
    </body>
</html>
```

ListBox

- ListBox is also created using <select> tag, just like DropDownList.
- But the difference is: DropDownList allows the user to select ONLY ONE element.
- ListBox allows the user select multiple options, by using Ctrl+Click or Shift+Click on the keyboard.

Example on ListBox

```
<html>
    <head>
        <title>Listbox</title>
    </head>
    <body>
        <h1>Listbox</h1>
        <form action="http://localhost/someaddress">
            Bank:<br>
            <select name="bank" multiple="multiple">
                <optgroup label="Top Banks">
                    <option value="icic">ICICI Bank</option>
                    <option value="hdfc">HDFC Bank</option>
                    <option value="sbi">State Bank of India</option>
                </optgroup>
                <optgroup label="Other Banks">
                    <option value="axis">Axis Bank</option>
                    <option value="cb">Canara Bank</option>
                    <option value="boi">Bank of India</option>
                    <option value="iob">Indian Overseas Bank</option>
                </optgroup>
            </select>
            <br>
            <input type="submit">
        </form>
    </body>
</html>
```

Selected Attribute

- The "selected" attribute of <option> tag must be set to the <option> tag, which must be by default submitted in the dropdownlist.
- If you don't specify "selected" attribute for any <option> tag, by default, the first <option> tag will be selected in the dropdownlist.
- The "selected" attribute has only one value, i.e. "selected".

Example on Dropdownlist - Selected

```
<html>
```

```
<head>
    <title>Selected</title>
</head>
<body>
    <h1>Selected</h1>
    <form action="http://localhost/someaddress">
        Country:
        <select name="country">
            <option>India</option>
            <option>China</option>
            <option selected="selected">UK</option>
            <option>USA</option>
            <option>Japan</option>
        </select>
        <br>
        <input type="submit">
    </form>
</body>
</html>
```

Textarea

- <textarea> tag is used to create a multi-line textbox.
- Ex: Comments, Description etc.
- <textarea> tag is a paired tag.
- The user can resize the textarea, at run time, in the browser.

Syntax:

```
<textarea name="name here" rows="no. of rows" cols="no. of columns">
</textarea>
```

Example:

```
<textarea name="comments" rows="5" cols="25"></textarea>
```

Example on Textarea

```
<html>
    <head>
        <title>Textarea</title>
    </head>
    <body>
        <h1>Textarea</h1>
        <form action="http://localhost/someaddress">
            Comments:<br>
            <textarea name="comments" rows="5" cols="25"></textarea><br>
            <input type="submit">
        </form>
    </body>
</html>
```

and

DIV

- is a container.
- Inside tag you can place any content like normal text or any other html tags.
- When you want to divide your web page as no. of parts, each part is represented as tag.
- is a paired tag.

Syntax:

```
<div>
    Your content here
</div>
```

Example:

```
<div>
    Hello
</div>
```

Span

- represents a small amount of text, for which you can apply some special formatting.
- tag doesn't provide any style by default, but we can apply style to the span tag, by using CSS.
- is a paired tag.

Syntax:

```
<span>Your content here</span>
```

Example:

```
<span>Hello</span>
```

Advanced Tags

Horizontal Ruler

- <hr> tag is used to display a horizontal line (horizontal ruler).
- It acts as separation between two parts of the web page
- It is an unpaired tag.

Syntax:

```
<hr>
```

Example:

```
<hr>
```

Example on <hr>

```
<html>
```

```
    <head>
        <title>Hr</title>
    </head>
   <body>
        <h1>Hr</h1>
        One<hr>Two
    </body>
</html>
```

Pre-formatted Text

- <pre> tag is used to display the text as-it-is, along with the spaces and line breaks.
- It is a paired tag.
- Generally, <pre> tag is meant for displaying computer programs (for example, "for" loop syntax) as-it-is.
- It is rare to use <pre> tag in real-time.

Syntax:

```
<pre>your text here</pre>
```

Example:

```
<pre>
one    two
three          four                five
</pre>
```

Example on <pre>

```
<html>
    <head>
        <title>Pre</title>
    </head>
   <body>
<h1>Pre</h1>
one    two
three          four                five
<hr>

<pre>
one    two

three          four                five
</pre>
    </body>
</html>
```

Abbreviations

- <abbr> tag is used to display full-form of a short-form when the user places mouse pointer on it.

- HTML / Browser doesn't provides any built-in abbreviations. We have to set both short form and full form in the code itself.
- It is a paired tag.
- The "title" tag represents the tooltip (full form), which appears when the user places mouse pointer on the content. The <title> can be applicable for any html tag (not only for <abbr> tag).

Syntax:

<abbr title="full form here">short form here</abbr>

Example:

<abbr title="as soon as possible">ASAP</abbr>

Example on <abbr>

```
<html>
  <head>
    <title>Abbr</title>
  </head>
 <body>
    <h1>Abbr</h1>
    <abbr title="as soon as possible">asap</abbr>
  </body>
</html>
```

Bi-Directional Override

- <bdo> is used to display the text in reverse (right-to-left) order.
- The default is "left-to-right" (LTR).
- The "dir" (direction) attribute decides whether the text should be "left to right" or "right to left".
- The text enclosed within the <bdo> tag will be display in the specific direction.
- It is a paired tag.

Syntax:

<bdo dir="rtl">your text here</bdo>

Example:

<bdo dir="rtl">Hai how are you</bdo>

Attributes:

1. **dir:**
   - **ltr:** It displays the text in left-to-right.
   - **rtl:** It displays the text in right-to-left.

Example on <bdo>

```
<html>
  <head>
    <title>Bdo</title>
  </head>
 <body>
```

```
    <h1>Bdo</h1>
    <p>Hai how are you</p>
    <p> <bdo dir="rtl">Hai how are you</bdo> </p>
  </body>
</html>
```

Audio

- The <audio> tag plays an audio file in the web page.
- We have to maintain the audio file in the following formats, because different browsers support different audio formats:
  - .mp3 : IE, Chrome, Firefox, Safari, Opera
  - .ogg : Chrome, Firefox, Opera
- The autoplay="autoplay" attribute starts playing the audio file as soon as web page is loaded in the browser.
- The controls="controls" attribute displays audio player skin in the web page.

Syntax:

```
<audio autoplay="autoplay" controls="controls">
    <source src="filename.mp3" type="audio/mp3">
    <source src="filename.ogg" type="audio/ogg">
</audio>
```

Note: The browsers play the first possible file.

Example on <audio>

```
<!DOCTYPE html>
<html>
  <head>
    <title>audio</title>
  </head>
  <body>
    <audio autoplay="autoplay" controls="controls">
      <source src="rain.mp3" type="audio/mp3">
      <source src="rain.ogg" type="audio/ogg">
    </audio>
  </body>
</html>
```

Note: Copy "rain.mp3" and "rain.ogg" files into current folder.

Video

- The <video> tag plays a video file in the web page.
- We have to maintain the video file in the following formats, because different browsers support different video formats:

- .mp4 : IE, Chrome, Firefox, Safari, Opera
- .webm : Chrome, Firefox, Opera
- .ogg : Chrome, Firefox, Opera

- The autoplay="autoplay" attribute starts playing the video file as soon as web page is loaded in the browser.
- The controls="controls" attribute displays video player skin in the web page.

Syntax:

```
<video autoplay=”autoplay” controls=”controls” width=”pixels” height=”pixels”>
    <source src=”filename.mp4” type=”video/mp4”>
    <source src=”filename.ogg” type=”video/ogg”>
    <source src=”filename.webm” type=”video/webm”>
</video>
```

Note: The browsers play the first possible file.

Example on <video>

```
<!DOCTYPE html>
<html>
    <head>
        <title>video</title>
    </head>
    <body>
        <h1>video</h1>
        <video autoplay="autoplay" controls="controls" width="600px" height="350px">
            <source src="trailer.mp4" type="video/mp4">
            <source src="trailer.ogg" type="video/ogg">
            <source src="trailer.webm" type="video/webm">
        </video>
    </body>
</html>
```

Note: Copy “trailer.mp4”, “trailer.ogg” and “trailer.webm” files into current folder.

Details and Summary

- <details> and <summary> tags are used to allow the user expand / collapse some information, when the user clicks on the heading.

Syntax:

```
<details>
    <summary>heading</summary>
    content
</details>
```

Example on <details> and <summary>

```
<!DOCTYPE html>
<html>
    <head>
        <title>details and summary</title>
    </head>
    <body>
        <h1>details and summary</h1>
        <details>
            <summary>iPhone 6 Plus</summary>
            <p>The original iPhone introduced the world to Multi-Touch, forever changing the way people
experience technology.</p>
        </details>
    </body>
</html>
```

Figure and Figcaption

- <figure> and <figcaption> tags are used to represent an image along with its description.
- Search engines like google can identify the image based on its description.

Syntax:

```
<figure>
    <img src=”filename.extension”>
    <figcaption>description here</figcaption>
</figure>
```

Example on <figure> and <figcaption>

```
<!DOCTYPE html>
<html>
    <head>
        <title>figure</title>
    </head>
    <body>
        <h1>figure</h1>
        <figure>
            <img src="pulpit.jpg" alt="The Pulpit Rock" width="304" height="228">
            <figcaption>Pulpit rock in Norway</figcaption>
        </figure>
    </body>
</html>
```

Note: Copy and paste “pulpit.jpg” into current folder.

DataList

- <datalist> tag is used to display search suggestions in the textbox.

- When the user types some character in the textbox, the browser automatically searches the matching options from the datalist and display them as a search suggestions below the textbox.

Syntax:

```
<datalist id=”id here”>
    <option value=”value 1”>text</option>
    <option value=”value 1”>text</option>
    …
</datalist>
<input type=”text” list=”id here”>
```

Example on <datalist>

```
<!DOCTYPE html>
<html>
  <head>
    <title>Datalist</title>
  </head>
  <body>
    <h1>datalist</h1>
    <form>
      Search:
      <input type="text" list="list1">

      <datalist id="list1">
        <option value="Abu Dhabi Commercial Bank"></option>
        <option value="Allahabad Bank"></option>
        <option value="Andhra Bank"></option>
        <option value="Axis Bank"></option>
        <option value="Bank Of America"></option>
        <option value="Bank Of Bahrain And Kuwait"></option>
        <option value="Bank Of Baroda"></option>
        <option value="Bank of India"></option>
        <option value="Bank of Maharashtra"></option>
        <option value="Canara Bank"></option>
        <option value="Central Bank of India"></option>
        <option value="City Bank"></option>
        <option value="City Union Bank"></option>
        <option value="Corporation Bank"></option>
        <option value="Dhanlaxmi Bank"></option>
        <option value="Federal Bank"></option>
        <option value="HDFC Bank"></option>
        <option value="HSBC Bank"></option>
        <option value="ICICI Bank"></option>
        <option value="IDBI Bank"></option>
        <option value="Indian Bank"></option>
        <option value="Indian Overseas Bank"></option>
        <option value="IndusInd Bank"></option>
        <option value="ING Vysya Bank"></option>
```

```
            <option value="Jammu and Kashmir Bank"></option>
            <option value="Karnataka Bank"></option>
            <option value="Kotak Mahindra Bank"></option>
            <option value="Lakshmi Vilas Bank"></option>
            <option value="Nainital Bank Limited"></option>
            <option value="Oman International Bank"></option>
            <option value="Oriental Bank of Commerce"></option>
            <option value="Punjab National Bank"></option>
            <option value="Ratnakar Bank"></option>
            <option value="Reserve Bank Of India"></option>
            <option value="Royal Bank Of Scotland"></option>
            <option value="South Indian Bank"></option>
            <option value="Standard Chartered Bank"></option>
            <option value="State Bank of India"></option>
            <option value="Syndicate Bank"></option>
            <option value="Tamilnad Mercantile Bank"></option>
            <option value="UCO Bank"></option>
            <option value="Union Bank of India"></option>
            <option value="United Bank of India"></option>
            <option value="Vijaya Bank"></option>
            <option value="Yes Bank Ltd"></option>
        </datalist>
    </form>
  </body>
</html>
```

ProgressBar

- <progress> tag is used to display the progress of a task.
- To move progressbar dynamically, we have to use JavaScript.
- **Ex:** Downloading progress, Uploading progress.

Syntax:

```
<progress min="minimum" max="maximum" value="some value">
</progress>
```

Example on <progress>

```
<!DOCTYPE html>
<html>
    <head>
        <title>progress</title>
    </head>
    <body>
        Progress:
        <progress min="0" max="100" value="80">
        </progress>
    </body>
</html>
```

Meter

- <meter> tag is used to display percentage of utilization.
- **Ex:** Disk usage.

Syntax:

<meter min="minimum" max="maximum" value="value here">

</meter>

Example on <meter>

```
<!DOCTYPE html>
<html>
    <head>
        <title>meter</title>
    </head>
    <body>
        <h1>meter</h1>
        Disk usage:
        <meter min="0" max="100" value="60">
        </meter>
    </body>
</html>
```

Output

- <output> tag is used to display the result of some calculation.
- The <output> tag is readyonly; the user can see the value; but can't modify the value.
- **Ex:** Displaying “tax” based on “amount”.

Syntax:

<output id=”id here” >

</output>

Example on <output>

```
<!DOCTYPE html>
<html>
    <head>
        <title>output</title>
    </head>
    <body>
        <h1>output</h1>
        <form>
            Product Price:<br>
            <input type="text" name="a" value="1000"><br>
            Discount:<br>
            <input type="text" name="b" value="50"><br>
            Net Price:
            <output name="c" for="a b"></output>
```

```
        </form>

        <script>
            document.getElementsByName("a")[0].addEventListener("keyup", fun1);
            document.getElementsByName("b")[0].addEventListener("keyup", fun1);
            function fun1()
            {
                var val1 = document.getElementsByName("a")[0].value;
                var val2 = document.getElementsByName("b")[0].value;
                var val3 = parseInt(val1) - parseInt(val2);
                document.getElementsByName("c")[0].value = val3;
            }
        </script>
    </body>
</html>
```

Article

- The <article> tag represents “heading” and “one or more paragraphs”.
- It is just used to group-up the headings and paragraphs, if you have too many in the web page.
- It makes no changes in the output.

Syntax:

```
<article>
  Content here
</article>
```

Example on <article>

```
<!DOCTYPE html>
<html>
    <head>
        <title>article</title>
    </head>
    <body>
        <article>
            <h3>Some heading</h3>
            <p>Some content here.</p>
        </article>
        <article>
            <h3>another heading</h3>
            <p>another paragraph.</p>
        </article>
    </body>
</html>
```

Semantic Tags

Header

- This tag represents header bar, which may include website logo, search box, main links etc.
- It don't provide any styles by default; we have to apply styles manually, using CSS.

Syntax:

```
<header>
  Content here
</header>
```

Nav

- This tag represents navigation bar, which may include top navigation menu.
- It don't provide any styles by default; we have to apply styles manually, using CSS.

Syntax:

```
<nav>
  Content here
</nav>
```

Section

- This tag represents specific section of the page (box or container), in which you can place some content.
- It is mainly meant for representation of major parts of the page.
- It don't provide any styles by default; we have to apply styles manually, using CSS.

Syntax:

```
<section>
  Content here
</section>
```

Footer

- This tag represents footer part of the web page, which may contain bottom information of the page such as copy rights information, links of other related sites etc.
- It don't provide any styles by default; we have to apply styles manually, using CSS.

Syntax:

```
<footer>
  Content here
</footer>
```

Aside

- This tag represents right side bar of the web page, which may contain ads / other promotion information.
- It don't provide any styles by default; we have to apply styles manually, using CSS.

Syntax:

```
<aside>
  Content here
</aside>
```

Example on Template Tags

```
<!DOCTYPE html>
<html>
    <head>
        <title>HTML 5 - Semantic Tags</title>
    </head>
    <body>
        <header>
            this is header area
        </header>

        <nav>
            this is navigation area
        </nav>

        <section>
            this is content area
        </section>

        <aside>
            this is side bar area
        </aside>

        <footer>
            this is footer area
        </footer>
    </body>
</html>
```

Storage

Local Storage

- “Local Storage” is used to store some data in the browser memory permanently, until you manually delete the data (or) clear the browser history.
- Local storage is used to store only string data.
- Local storage can store unlimited data.

- Local storage data is visible in the browser's developer tools (Inspect Element) option.
- If you clear the browser history, the local storage will be deleted.
- **Ex:** Google login page automatically stores your name, email and photo on successful login.

Setting data into Local Storage

localStorage.property = value;

(or)

localStorage.setItem("property", "value");

Getting data from Local Storage

localStorage.property

(or)

localStorage.getItem("property");

Example on Local Storage

```
<!DOCTYPE html>
<html>
    <head>
        <title>local Storage</title>
    </head>
    <body>
        <h1>Local Storage</h1>
        <form>
            Name:<br>
            <input type="text" id="username"><br>
            Email:<br>
            <input type="text" id="email"><br>
            <input type="button" id="button1" value="Set data into local storage">
            <input type="button" id="button2" value="Get data from local storage">
        </form>

        <script>
            document.getElementById("button1").addEventListener("click", fun1);
            function fun1()
            {
                var username = document.getElementById("username").value;
                var email = document.getElementById("email").value;
                localStorage.username = username;
                localStorage.email = email;
                document.getElementById("username").value = "";
                document.getElementById("email").value = "";
                alert("Saved");
            }

            document.getElementById("button2").addEventListener("click", fun2);
```

```
            function fun2()
            {
                var username = localStorage.username;
                var email = localStorage.email;
                document.getElementById("username").value = username;
                document.getElementById("email").value = email;
            }
        </script>
    </body>
</html>
```

Session Storage

- "Session Storage" is used to store some data in the browser memory temporarily, until the browser gets closed.
- Session storage also can store string data only, much like local storage.
- Session storage data is also visible in the browser's developer tools (Inspect Element option).

Ex: Facebook stores your email and password after login.

Setting data into Session Storage

```
sessionStorage.property = value;
```

(or)

```
sessionStorage.setItem("property", "value");
```

Getting data from Session Storage

```
sessionStorage.property
```

(or)

```
sessionStorage.getItem("property");
```

Example on Session Storage

```
<!DOCTYPE html>
<html>
    <head>
        <title>Session Storage</title>
    </head>
    <body>
        <h1>Session Storage</h1>
        <form>
            Name:<br>
            <input type="text" id="username"><br>
            Email:<br>
            <input type="text" id="email"><br>
            <input type="button" id="button1" value="Set data into session storage">
            <input type="button" id="button2" value="Get data from session storage">
        </form>
```

```
        <script>
            document.getElementById("button1").addEventListener("click", fun1);
            function fun1()
            {
                var username = document.getElementById("username").value;
                var email = document.getElementById("email").value;
                sessionStorage.username = username;
                sessionStorage.email = email;
                document.getElementById("username").value = "";
                document.getElementById("email").value = "";
                alert("Saved");
            }

            document.getElementById("button2").addEventListener("click", fun2);
            function fun2()
            {
                var username = sessionStorage.username;
                var email = sessionStorage.email;
                document.getElementById("username").value = username;
                document.getElementById("email").value = email;
            }
        </script>
    </body>
</html>
```

Geo Location

- "Geo Location" concept is used to identify the current location of the client device.
- Geo Location returns latitude, longitude values of the client device location, based on which you can display maps, driving directions, traffic, address etc.

Steps for development of Geo Location

- **Send request to server to get geo location:**

  navigator.geolocation.getCurrentPosition(successcallback, failurecallback, { timeout: milli seconds });

- **Receive the latitude, longitude values:**

```
function successcallback(event)
{
    event.coords.latitude
    event.coords.longitude
}
```

Example on Geo Location

```
<!DOCTYPE html>
<html>
    <head>
        <title>HTML 5 - Geo Location</title>
    </head>
    <body>
        Latitude: <span id="span1"></span><br>
        Longitude: <span id="span2"></span>

        <script>
            navigator.geolocation.getCurrentPosition(fun1, fun2, { timeout: 6000 } );
            function fun1(event)
            {
                document.getElementById("span1").innerHTML = event.coords.latitude;
                document.getElementById("span2").innerHTML = event.coords.longitude;
            }
            function fun2()
            {
                alert("Error");
            }
        </script>
    </body>
</html>
```

Example on Geo Location – with area name

```
<html>
    <head>
        <title>GeoLocation</title>
    </head>
    <body>
        <h1>GeoLocation</h1>
        <div id="div1">div 1</div>
        <div id="div2">div 2</div>
        <div id="div3">div 3</div>

        <script src="https://maps.googleapis.com/maps/api/js">
        </script>

        <script>
            window.navigator.geolocation.getCurrentPosition(onsuccess, onerror, { timeout: 5000 });
            function onsuccess(event)
            {
                document.getElementById("div1").innerHTML = event.coords.latitude;
                document.getElementById("div2").innerHTML = event.coords.longitude;

                var geocoder = new google.maps.Geocoder();
                var latlng = new google.maps.LatLng(event.coords.latitude, event.coords.longitude);
                geocoder.geocode( {'latLng': latlng}, fun3 );
                function fun3(results, status)
                {
```

```
                if (status == google.maps.GeocoderStatus.OK)
                {
                    if (results[0])
                    {
                        var areaname= results[0].formatted_address ;
                        document.getElementById("div3").innerHTML = areaname;
                    }
                    else
                    {
                        alert("address not found");
                    }
                }
                 else
                 {
                    alert("Geocoder failed due to: " + status);
                }
            }

        }
        function onerror()
        {
        }
    </script>
  </body>
</html>
```

Web Workers

- "Web Workers" concept is used to execute a javascript file in background, without effecting the actual performance of the page.
- It is mainly used to do some background process, such as uploading the file, processing some records etc.
- The specified javascript file executes asynchronously; the code present within the javascript file can't access the DOM; but it can use postMessage( ) function to pass value from the javascript to regular script of the web page; then the script present within the web page can receive the value and do some process for it.

Steps for development of Web Workers

- **Create a javascript file (for async execution):**

filename.js

```
Any js code here

postMessage(value);
```

Note: We can't access DOM in the async javascript file. We have to use postMessage() function to pass data from the javascript file to the web page.

- **Create an object for "Worker" class:**

  var w = new Worker("filename.js");

  The "Worker" class represents an async javascript file.

- **Receive message (data) from async javascript file:**

  w.onmessage = functionname;

  function functionname(event)

  {

  event.data

  }

Example on Web Workers

webworkers.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Web Workers</title>
    </head>
    <body>
        <div id="div1"></div>

        <script>
            var worker = new Worker("script.js");
            worker.addEventListener("message", fun1);
            function fun1(event)
            {
                document.getElementById("div1").innerHTML = event.data;
            }
        </script>
    </body>
</html>
```

script.js

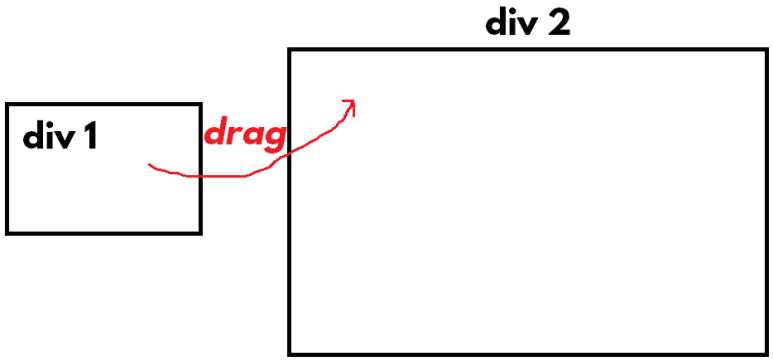
```
var i = 0;
setInterval(sample, 100);

function sample()
{
    i++;
```

```
        postMessage(i);
    }
```

Drag and Drop

- "Drag and Drop" concept allows the user to drag an element and drop into another element.
- We can drag any element and drop into another element.
- When we start dragging the source element, it executes "dragstart" event.
- When we hover on the droppable element, it executes "dragover" event.
- When we release the mouse pointer on the droppable element, it executes "drop" event.



Steps for development of Drag and Drop

- **Create a draggable element:**

```
<div id="div1" draggable="true">
</div>
```

- **Create a droppable element:**

```
<div id="div2">
</div>
```

- **Handle "dragstart" event of draggable element:**

```
document.getElementById("div1").addEventListener("dragstart", fun1);
function fun1(event)
{
    event.dataTransfer.setData("variablename", event.target.id);
}
```

Note: The “dragstart” event executes when the user starts dragging the draggable element.

- **Handle “dragover” event of droppable element:**

```
document.getElementById(“div2”).addEventListener(“dragover”, fun2);
function fun2(event)
{
    event.preventDefault( );
}
```

Note: The “dragover” event executes when the user drags the draggable element and places it on the top of droppable element.

- **Handle “drop” event of droppable element:**

```
document.getElementById(“div2”).addEventListener(“drop”, fun3);
function fun3(event)
{
    var id = event.dataTransfer.getData(“variablename”);
    document.getElementById(“div2”).appendChild(document.getElementById(id”));
}
```

Note: The “drop” event executes when the user drops the draggable element on the droppable element.

Example on Drag and Drop

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Drag and drop</title>
        <style type="text/css">
            #div1
            {
                width: 200px;
                height: 200px;
                padding: 10px;
                border: 3px solid red;
                float: left;
                background-color: skyblue;
                cursor: pointer;
            }
            #div2
```

```
            {
                width: 400px;
                height: 400px;
                padding: 10px;
                border: 6px solid darkgreen;
                float: left;
                background-color: hotpink;
                position: absolute;
                left: 280px;
                top: 5px;
                cursor: pointer;
            }
        </style>
    </head>
    <body>
        <h1>Drag and drop</h1>
        <div id="div1" draggable="true">
            <h1>div 1</h1>
        </div>
        <div id="div2">
            <h1>div 2</h1>
        </div>

        <script>
            document.getElementById("div1").addEventListener("dragstart", fun1);
            function fun1(event)
            {
                event.dataTransfer.setData("myvariable", event.target.id); //we are storing "div1" into
"myvariable"
            }

            document.getElementById("div2").addEventListener("dragover", fun2);
            function fun2(event)
            {
                event.preventDefault(); //stop the default functionality
            }

            document.getElementById("div2").addEventListener("drop", fun3);
            function fun3(event)
            {
                var id = event.dataTransfer.getData("myvariable"); //we are getting "div1" from
"myvariable"
                document.getElementById("div2").appendChild(document.getElementById(id));
            }
            //event = browser given information
        </script>
    </body>
</html>
```

Offline Apps

- "Offline Apps" concept enables the browser to load the web page from the server for the first time and store it in the browser memory; but second time onwards, it loads the page from the browser memory directly, instead of loading it from the server.
- **Advantage:** The user can view the web page, even without network connection.

Steps for development of Offline Apps

- **Create a "manifest" file:**

  filename.appcache

```
CACHE MANIFEST
#id
filename.jpg
filename.css
filename.js
```

- **Set the "manifest" file to the html file:**

```
<html manifest="filename.appcache">
```

Example on Offline Apps

c:\html\StyleSheet1.css

```
body,input
{
  font-family: Tahoma;
  font-size: 16px;
}
```

c:\html\JavaScript.js

```
function fun1()
{
    alert("hai");
}
```

c:\html\sample.appcache

```
CACHE MANIFEST
# 101
JavaScript.js
StyleSheet1.css
img1.jpg
```

c:\html\index.html

```
<!DOCTYPE html>
<html manifest="sample.appcache">
    <head>
        <title>Title here</title>
        <link href="StyleSheet1.css" rel="stylesheet">
        <script src="JavaScript.js"></script>
    </head>
    <body>
        <h1>Heading</h1>
        <p>para</p>
        <p><input type="button" value="Click me" id="btn1"></p>
        <p><img src="img1.jpg" width="250"></p>

        <script>
            document.getElementById("btn1").addEventListener("click", fun1);
        </script>
    </body>
</html>
```

Note: Place "img1.jpg" in the current folder.

Execution

- Download and install "nodejs" from "https://nodejs.org". Click here to find steps to install NodeJS.
- Open "Command Prompt" and run the following commands:

```
npm install http-server -g
cd c:\html
http-server
```

- Open browser and enter the url: http://localhost:8080/index.html

Server Sent Events

- Browser send continuous requests to server, for every "n" seconds. Ex: 5 seconds.
- This concept is used to get continuous updated information (live information) from server.
- **Ex:** Cricket score board, election results, share market etc.
- The "EventSource" constructor function is used to start sending continuous requests to server.
- Every time when we get response (update) from server, the ""message" event executes for "EventSource".
- The "server url" represents the address of server side program, to which you want to send request.
- The "event.data" property represents the actual data that is received from the browser.

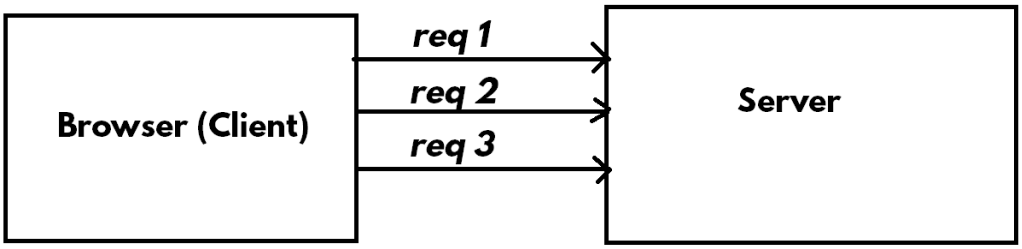
Steps for development of Offline Apps

- **Create a server side program:**

  - Set the response content type as “text/event-stream”
  - Write data to response object
  - Flush the response

- **Create a html page and call the server program:**

```
var variablename = new EventSource(“server url”);
variablename.addEventListener("message", functionname);
function functionname(event)
{
    //event.data = response from server
}
```



Example on Server Sent Events

c:\html\httpserver.js

```
var http = require("http");
var fs = require("fs");
var server = http.createServer(engine);
server.listen(8080, startup);
function startup()
{
    console.log("Server started at port 8080");
}

function engine(request, response)
{
    if (request.url == "/" || request.url == "/index.html")
    {
        fs.readFile("index.html", "utf8", fun1);
```

```
        function fun1(error, data)
        {
            if (error)
            {
                response.setHeader("content-type", "text/html");
                response.writeHead(404);
                response.write("<h1 style='color:red'>Page not found</h1>");
                response.end();
            }
            else
            {
                response.setHeader("content-type", "text/html");
                response.writeHead(200);
                response.write(data);
                response.end();
            }
        }
    }
    else if (request.url.startsWith("/getdata"))
    {
        console.log("request received at " + new Date().toLocaleTimeString());
        response.setHeader("content-type", "text/event-stream");
        response.writeHead(200);
        response.write("data: Response at " + new Date().toLocaleTimeString() + "\n\n");
        response.end();
    }
}
```

c:\html\index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Server sent events</title>
</head>
<body>
    <h1>Server sent events</h1>
    <div id="result"></div>

    <script>
        var source = new EventSource("/getdata");
        source.addEventListener("message", fun1);
        function fun1(event)
        {
            document.getElementById("result").innerHTML += event.data + "<br>";
        };
     </script>
</body>
</html>
```

Execution:

- Download and install “nodejs” from “https://nodejs.org”. Click here to find steps to install NodeJS.
- Open “Command Prompt” and run the following commands:

  cd c:\html

  node httpserver.js

- Open browser and enter the url: http://localhost:8080/index.html

Canvas

- Canvas concept is used to create graphics in the web page programmatically.
- Canvas graphics can be created using JavaScript.
- Canvas is optional & mostly not used in 95% of regular websites.
- Canvas is rare to use.

Creating Canvas Container

```
<canvas id=”canvas1” style=”border:4px solid black” width=”pixels” height=”pixels”>
</canvas>
```

Get context of canvas

Context = It is an object, which is used to write graphics on the canvas.

```
var ctx = document.getElementById(“id”).getContext(“2d”);
```

strokeStyle

It specifies color of the stroke (line).

```
ctx.strokeStyle = “color name”;
```

lineWidth

It specifies thickness of the stroke (line).

```
ctx.lineWidth = pixels;
```

strokeRect

It draws a rectangle, with a line.

```
ctx.strokeRect(x, y, width, height);
```

fillStyle

It specifies fill color.

```
ctx.fillStyle = “color name”;
```

fillRect

It draws a filled rectangle.

ctx.fillRect(x, y, width, height);

Example on Canvas

```
<!DOCTYPE html>
<html>
    <head>
        <title>Canvas - Rectangle</title>
    </head>
    <body>
        <h1>Canvas - Rectangle</h1>
        <canvas id="canvas1" style="border: 5px solid red" width="600px" height="400px">
        </canvas>

        <script>
            var c = document.getElementById("canvas1").getContext("2d");
            c.fillStyle = "pink";
            c.fillRect(50, 50, 100, 100);
            c.strokeStyle = "blue";
            c.lineWidth = 10;
            c.strokeRect(200, 50, 200, 200);
        </script>
    </body>
</html>
```

SVG

- SVG stands for Scalable Vector Graphics.
- Both Canvas and SVG are used to create graphics in the web page.
- Canvas losses its quality, if zoom is increased; but SVG is not.
- Canvas is “JavaScript-based”; SVG is “CSS-based”.

Syntax to create SVG container

<svg xmlns=”http://www.w3.org/2000/svg” version=”1.1” style=”border:4px solid black” width=”pixels” height=”pixels”>

Graphic elements here

</svg>

Example on SVG

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

```
        <title>svg</title>
    </head>
    <body>
        <h1>svg</h1>
        <svg xmlns="http://www.w3.org/2000/svg" version="1.1" style="border:solid 1px red" width="400px"
height="400px">
            <rect x="50" y="50" width="300" height="100" style="fill:#99ff99; stroke-width:5; stroke:blue" />
        </svg>
    </body>
</html>
```

Deprecated Tags / Attributes

Deprecated / Removed Tags in HTML 5

- The following tags have been removed in HTML 5.

1. **<acronym>**
   - Use <abbr> tag instead.
2. **<applet>**
   - Java applets is out-dated concept.
3. **<basefont>**
   - Use CSS to set the default font.
4. **<big>**
   - Use CSS to increase text size.
5. ** **
   - Use CSS to set center alignment.
6. **<dir>**
   - Use <ul> or <ol> to display list of items.
7. **<font>**
   - Use CSS to set font.
8. **<frame>**
   - Use <iframe>.
9. **<frameset>**
   - Use <iframe>
10. **<noframes>**
    - Use <iframe>

11. **<strike>**
    - Use CSS to set strikeout.
12. **<tt>**
    - Use CSS to change font.
13. **<bgsound>**
    - Use <audio> tag to play audio.
14. **<marquee>**
    - Use CSS / jQuery for animations.
15. **<u>**
    - Use CSS to set underline.

Deprecated / Removed Attributes in HTML 5

- The following attributes have been removed in HTML 5.

Sl. No	Tag	Attribute	Description
1	<body>	background, bgcolor, link, alink, vlink, text	Use CSS instead.
2	<h1> to <h6>	align	Use CSS instead.
3	<p>	align	Use CSS instead.
4	<hr>	align, noshade, size, width	Use CSS instead.
5	<input>	align	Use CSS instead.
6	<img>	align, border	Use CSS instead.
7	<table>	align, bgcolor, cellpadding, cellspacing, width	Use CSS instead.
8	<td>, <th>	align, valign, bgcolor, width, height	Use CSS instead.
9		align	Use CSS instead.

XHTML

Introduction to XHTML

- XHTML is “HTML” + “set of rules to maintain good quality and standards” in the html code.
- XHTML is the strict and cleaner version of HTML, recommended by W3C (World Wide Web Consortium).
- XHTML was released in 2000.

- In realtime, XHTML is recommended.

XHTML Rules

1. <!DOCTYPE> is must.

   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

2. <html>, <head>, <title>, <body> tags are must.

3. A <meta> tag with “content-type” is mandatory.

   <meta http-equiv="Content-type" content="text/html;charset=UTF-8">

   That means the html file has the content of “text/html” type; and the character encoding format is “UTF” (Unicode Transformation Format), which supports all worldwide language characters.

4. “xmlns” attribute is must for <html> tag.

   <html xmlns="http://www.w3.org/1999/xhtml">

   An “xmlns” (XML namespace) contains all the pre-defined html tags.

5. XHTML tags and attributes must be in lower case only.

   Ex: <h1>hello</h1>

6. Attribute’s values should be in double quotes.

   Ex: <img src=”img1.jpg”>

7. All the paired tags must be closed.

   Ex: Hello

8. All the unpaired tags must be end with “/”.

   Ex:

9. The inner-most tags should be closed first.

   Ex: Hello

10. Attribute minification not allowed. We must write both attribute and value.

    Ex: readonly=”readonly”

11. The following tags are only allowed in <body> tag directly. All other tags must be used only inside any of the following tags.
    -
    - <table>

-
-
-
- <h1> to <h6>

12. “alt” attribute is must for <img> tag.

XHTML – Online Validator

- Open browser: Ex: Mozilla Firefox
- Go to https://validator.w3.org.
- Click on “Validate by File Upload”.
- Click on “Browse”.
- Select the html file. Ex: xhtml.html
- Click on “Check”.
- It shows the list of errors or “Passed” message.

XHTML - Example

xhtml.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>this is title</title>
<meta http-equiv="Content-type" content="text/html;charset=UTF-8" />
    </head>
    <body>
        <h1>hai</h1>
        <p>
            <input type="text" />
            <b> <i>this is bold and italic</i> </b>
            <input type="checkbox" checked="checked" />
        </p>
        <hr />
    </body>
</html>
```