

Handbook **HTML, CSS**

By: SachTech Solution Pvt. Ltd.

Introduction to HTML

HTML is the standard markup language for creating Web pages.

- HTML stands for Hyper Text Markup Language
- HTML describes the structure of Web pages using markup
- HTML elements are the building blocks of HTML pages
- HTML elements are represented by tags
- HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
- Browsers do not display the HTML tags, but use them to render the content of the page

Web Browsers

The purpose of a web browser (Chrome, IE, Firefox, Safari) is to read HTML documents and display them.

The browser does not display the HTML tags, but uses them to determine how to display the document.

HTML Basic Layout

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h1>My First Heading</h1>
<p>My first paragraph. </p>
</body>
</html>
```

Basic Layout Explained

The `<!DOCTYPE html>` declaration defines this document to be HTML5

The `<html>` element is the root element of an HTML page

The `<head>` element contains meta information about the document

The `<title>` element specifies a title for the document

The `<body>` element contains the visible page content

The `<h1>` element defines a large heading

The `<p>` element defines a paragraph

Inline and block elements

Inline elements flow within the text and do not start on a new line. Example includes ``, `` and `` tags.

HTML Tags

```
<h1></h1>
<h2></h2>
<h3></h3>
<h4></h4>
<h5></h5>
<h6></h6>
```

Headings are defined with the `<h1>` to `<h6>` tags.

`<h1>` defines the most important heading. `<h6>` defines the least important heading.

Syntax

`<h1>` *String needed to print* `</h1>`

.....

`<h6>` *String needed to print* `</h6>`

Example

Here is the example output for `<h1>` to `<h6>` :

```
<!DOCTYPE html>
```

```
<html lang="en" dir="ltr">
```

```
<head>
```

```
<meta charset="utf-8">
```

```
<title>HTML Headings</title>
```

```
</head>
```

```
<body>
```

```
<h1>This is Heading 1</h1>
```

```
<h2>This is Heading 2</h2>
```

```
<h3>This is Heading 3</h3>
```

```
<h4>This is Heading 4</h4>
```

```
<h5>This is Heading 5</h5>
```

```
<h6>This is Heading 6</h6>
```

```
<p>This is a paragraph. This is a paragraph.This is a paragraph.This is a paragraph.<br/>
```

```
This is a paragraph.This is a paragraph.This is a paragraph.This is a paragraph.<br/>
```

```
This is a paragraph.This is a paragraph.This is a paragraph.This is a paragraph.
```

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

Output

This is Heading 1

This is Heading 2

This is Heading 3

This is Heading 4

This is Heading 5

This is Heading 6

This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.
This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.
This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.

HTML List


There are 3 type of HTML Lists

1. Unordered HTML List
2. Ordered HTML List
3. Html definition list

HTML Unordered List

As the name suggest, unordered list is a collection of related items, which has no particular order. But further these lists are of different types. By default the type of unordered list is **disc**(small round dots). But you change its type using style. Here are different types of unordered lists:

1. Disc(default)

- 
2. Circle
 3. Square
 4. None

All the list items are enclosed in `` and ``.

Syntax

```
<ul style='list-style-type:any_type'>
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
```

Example

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>HTML Headings</title>
  </head>
  <body>
    <!-- Default list -->
    <ul>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ul>

    <!-- list type circle -->
    <ul style='list-style-type:circle'>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ul>

    <!-- list type square -->
    <ul style='list-style-type:square'>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ul>

    <!-- list type none -->
    <ul style='list-style-type:none'>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ul>
  </body>
</html>
```

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

Output

- HTML
- CSS
- Java Script

- HTML
- CSS
- Java Script

- HTML
- CSS
- Java Script

HTML
CSS
Java Script

HTML Ordered List

Unlike unordered list, ordered list contains group of similar items which retains some particular order. Further it is of different types. By default the type of ordered list is number. But you can change its type also using **type** attribute. Here are different types of ordered lists:

1. Number (For e.g. type='1')
2. Lowercase alphabets (For e.g. type='a')
3. Uppercase alphabets (For e.g. type='A')
4. Lowercase romans (For e.g. type='i')
5. Uppercase romans (For e.g. type='I')

Syntax

```
<ul style='list-style-type:any_type'>
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
```

Example

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>HTML Headings</title>
  </head>
  <body>
    <!-- Default list -->
    <ol>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ol>

    <!-- list type a -->
    <ol type='a'>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ol>
    <!-- list type A -->

    <ol type='A'>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ol>
    <!-- list type i -->

    <ol type='i'>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ol>
    <!-- list type i -->

    <ol type='I'>
      <li>HTML</li>
      <li>CSS</li>
      <li>Java Script</li>
    </ol>
  </body>
</html>
```


Output

1. HTML
2. CSS
3. Java Script

- a. HTML
- b. CSS
- c. Java Script

- A. HTML
- B. CSS
- C. Java Script

- i. HTML
- ii. CSS
- iii. Java Script

- I. HTML
- II. CSS
- III. Java Script

HTML Definition List

The definition list can be used where we need to define group of related items. Let's take an book index as an example. In book index there are number of related topics and their further discussion, In this case, you can use definition list more efficiently. Here is the syntax of definition list:

Syntax

```
<dl>
  <dt>definition title 1</dt>
  <dd>- definition description </dd>
  <dt> definition title 2</dt>
  <dd>- definition description </dd>
</dl>
```

Example

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>HTML Headings</title>
  </head>
  <body>
```

```
<dl>
  <dt>HTML</dt>
  <dd>HTML stands for Hyper Text Markup Language. It is used for creating web pages.</dd>
</dl>
  <dt>CSS</dt>
  <dd>CSS stands for Cascading Style Sheets. It is used for styling web pages.</dd>
</dl>
</body>
</html>
```

Output

HTML
HTML stands for Hyper Text Markup Language. It is used for creating web pages.

CSS
CSS stands for Cascading Style Sheets. It is used for styling web pages.

Nested Lists

We can put a list inside a list to create sub-lists. To do so put another ``, `` or `<dl>` tag inside one ``. Here is the code example for doing so:

Example

```
<ul>
  <li>HTML</li>
  <li>CSS
    <ol>
      <li>Inline CSS</li>
      <li>Internal CSS</li>
      <li>External CSS</li>
    </ol>
  </li>
  <li>Java Script</li>
```


Output

- HTML
- CSS
 1. Inline CSS
 2. Internal CSS
 3. External CSS
- Java Script

Inline CSS

Here are some points about inline CSS:

- Inline CSS refers to the CSS written in tag's style attribute.
- Here style is an attribute of any tag.
- It is used to style specific tag content.
- It refers to putting CSS inside same file rather than putting CSS in different file.
- Here is the syntax for inline CSS:

Syntax

```
<tag_name style='property1:value1; property2:value2'>Content Here</tag_name>
```

Here is the example for how to use inline CSS with HTML page.

Example

```
<h1 style='background-color:red; color:black;'>Hello There</h1>
```

In the above example Text color will be black and background-color will be red.

Output

Hello There

Why use Inline CSS?

Inline CSS reduces the number of files the browser has to download prior to displaying your web page. If you are using an external CSS file, your browser must first load your HTML file, then download your CSS file

If you have used Inline CSS, the browser only has to download your HTML file. Downloading one file is faster than downloading two. It may lead to optimize browser speed.

HTML Forms

HTML `<form>` tag is used to get user's input. It defines a form that is used to collect user's input. HTML form contains form elements. These are different type of input elements like text, number, dates, buttons etc. `<form>` tag is a paired tag i.e. it contains opening and closing tags. Here are some points about HTML forms:

- Whenever you want to collect information from visitors you will need a form, which lives inside a `<form>` element.
- Information from a form is sent in name/value pairs.
- Each form control is given a name, and the text the user types in or the values of the options they select are sent to the server.

Here is the syntax for `<form>` tag:

Syntax

```
<form action="" method="post">
```

```
.....
```

```
form elements
```

```
.....
```

```
</form>
```

How HTML form works?

1. A user fills data in a form and then presses a button to submit the information to the server (request)
2. The name of each form control is sent to the server along with the value the user enters or selects

3. The server processes the information using a programming language such as PHP, Python, VB.net, or Java. It may also store the information in a database
4. The server creates a new page to send back to the browser based on the information received (response)

The action and method attribute

The action attribute contains the name of the web page to which the data should be sent. By default the action sets to the current page

The method attribute specifies the way, how the data should be sent. There are two types of HTTP (Hyper Text Transfer Protocol) methods which are:

1. GET (default)
2. POST

HTTP enables communication between client and server.

GET and POST

- GET is used to request data from a specified resource
 - GET requests can be cached
 - GET requests remain in the browser history
 - GET requests can be bookmarked
 - GET requests should never be used when dealing with sensitive data
 - GET requests have length restrictions
 - GET requests is only used to request data (not modify)
 - It referred to as a less secure way of transferring data as compared to POST method
-
- POST is used to send data to a server to create/update a resource.
 - POST requests are never cached
 - POST requests do not remain in the browser history
 - POST requests cannot be bookmarked
 - POST requests have no restrictions on data length
 - It referred to as a more secure way of transferring data as compared to GET method

The <form> elements

User data can be of different type like numbers, text, dates etc. HTML form facilitates different type of elements which are described below:

The <input> element

It is a most commonly used input element. The <input> is a self-closing and inline element. It contains a type attribute which describes the type of user input whether number, range, text etc.


Here are the different types of <input> elements:

Text Box: (single-lined text)

When the type attribute has a value of text, it creates a single line text input.

```
<input type="text" name="firstname" placeholder="First Name" />
```

```
<input type="text" name="lastname" placeholder="Last Name" />
```



Password Input

When the type attribute has a value of password it creates text box that acts just like a single-line text input, except the characters are blocked out. They are hidden in this way so that if someone is looking over, they cannot see sensitive data such as passwords.

```
<input type="text" name="username" placeholder="Enter Username" />
```

```
<input type="password" name="password" placeholder="Enter Password" />
```



Radio Button:

Radio buttons allow users to pick just one of a number of options. They can appear in different ways depending upon browser.

You can add checked attribute. By adding checked attribute, particular field will be auto checked when page loads.

```
<input type="radio" name="gender" />Male
```

```
<input type="radio" name="gender" checked/>Female
```

Note: If you want to make single selection from group of radio buttons, then you need to add **name** attribute with same name, like in above example.

☐ Male ☒ Female

Checkbox:

Checkboxes are used for multiple selection. In simple words, checkboxes allow users to select (and unselect) one or more options from group of options.

You can add checked attribute. By adding checked attribute, particular field will be auto checked when page loads.

Here is the example:

```
<input type="checkbox" name="vehicle1" value="Bike" /> I have a bike<br>
```

```
<input type="checkbox" name="vehicle2" value="Car" /> I have a car<br>
```

☒ I have a bike
☒ I have a car

Date:

If you are asking the user for a date, you can use an element and give the type attribute a value of date. This will create a date input in browsers that support the new HTML5 input types. Older browsers that do not recognize these inputs will just treat them as a single line text box.

```
<input type="date" />
```

dd-mm-yyyy

October, 2018

Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

Time:

It is another input element of HTML5, which describes time. The value of the time input is always in 24-hour format: "hh:mm". It also can appear differently according to the browser.

```
<input type="time" />
```

15:00

Submit Button:

After filling data to a form, each user needs to submit the form data. For this reason, we need a submit button by which we can submit the data. Here is the code for doing so:

```
<input type='submit' name='submit' value='Register'>
```

Output

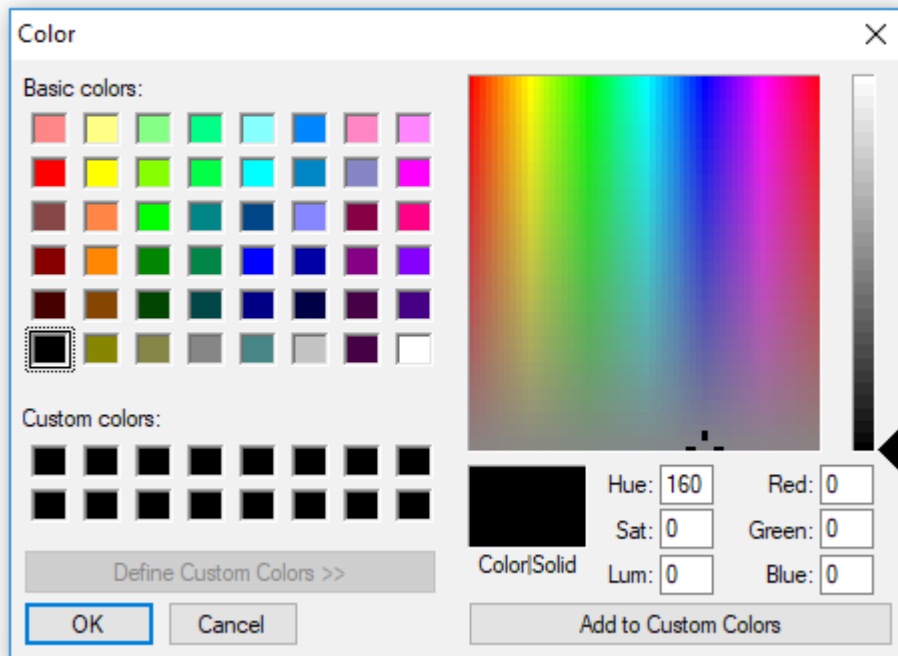
Register

Color:

If you need to take an color input from user. Then you can use input type with value color. It is an colored box, by which we can select any color(by clicking color box). Its default color value will be black. It is also an new type introduced in HTML5.

`<input type="color" />`

Output



HTML Dropdown List Box

Sometimes, we need a form that contains number of options in a dropdown menu. Usually we see this type of input when selecting some country or city. For creating a dropdown list we need to use `<select>` tag.

The `<select>` element

- It is an inline and paired tag.
- The `<select>` element is used to create a dropdown list box
- It contains two or more `<option>` elements.
- A user selects one option from a dropdown list.
- You can create an select box with multiple selection using size attribute.

Syntax

```
<select>  
  <option>Option1 </option>
```

```
<option>Option2 </option>
</select>
```

Example

```
<form class="" action="" method="post">
  Select city <select>
    <option>Sunam </option>
    <option>Sangrur </option>
    <option>Mohali </option>
    <option>Chandigarh </option>
  </select>
</form>
```

Output

Select city

- Sunam
- Sangrur
- Mohali
- Chandigarh

The <textarea> element

Here are some points about <textarea> element:

- The <textarea> element is used to create a multi-line text input
- Unlike another tags, it is an paired tag i.e. it should therefore have an opening and a closing tag
- Any text that appears between the opening <textarea> and closing </textarea> tags will appear in the text box when the page loads

Syntax

```
<textarea name="Any name" cols="Any Number" rows="Any Number">
```

Example

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
```

```
<head>

  <meta charset="utf-8">

  <title>HTML Headings</title>

</head>

<body>

  <form method="post">

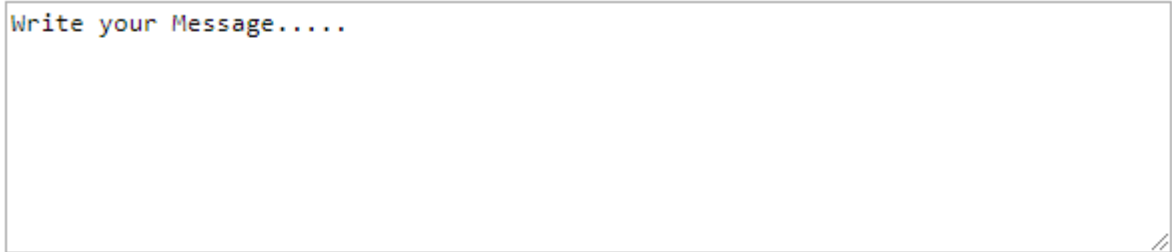
    <textarea name="name" rows="8" cols="80">Write your Message.....</textarea>

  </form>

</body>

</html>
```

Output




Write your Message.....

HTML Form validation

You have probably seen forms on the web that give users messages if the form control has not been filled in correctly, this is known as form validation. Traditionally, form validation has been performed using JavaScript. But HTML5 is introducing validation and leaving the work to the browser.

Validation helps ensure the user enters information in a form that the server will be able to understand when the form is submitted. Validating the contents of the form before it is sent to the server helps:

- Reduce the amount of work the server has to do
- Enables users to see if there are problems with the form faster than if validation were performed on the server.
- We are going to show form-validation using form's email and URL type.



HTML5 has also introduced inputs that allow visitors to enter email addresses and URLs. Browsers that do not support these input types will just treat them as text boxes. Let's see how to use them.

Email input

If you ask a user for an email address, you can use the email input. Browsers that support HTML5 validation will check that the user has provided information in the correct format of an email address. Some smart phones also optimize their keyboard to display the keys you are most likely to need when entering an email address (such as the @ symbol).

Example

```
<!DOCTYPE html>

<html lang="en" dir="ltr">

  <head>

    <meta charset="utf-8">

    <title>HTML Headings</title>

  </head>

  <body>

    <form action="" method="post">

      <input type="email" name="email" placeholder="Enter Email">

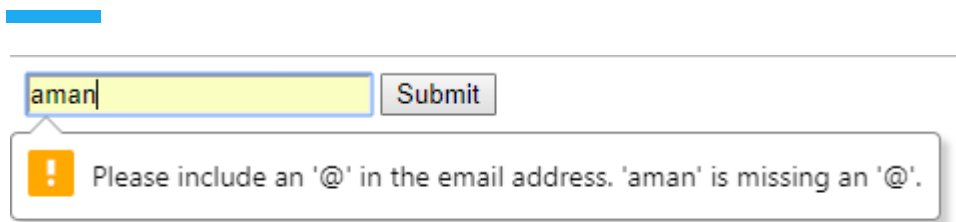
      <input type="submit" name="submit" value="Submit">

    </form>

  </body>

</html>
```

Output



aman Submit

! Please include an '@' in the email address. 'aman' is missing an '@'.

As you can see in example above, it raises an validation error if the user tries to submit the form without filling an email address in email field.

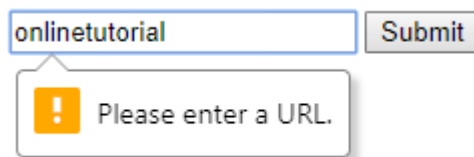
URL input

The URL input looks for an URL address, otherwise the form will not submit. Here is the example for the same:

```
<input type="url" name="website" placeholder="Enter Website URL">
```

```
<input type="submit" name="submit" value="Submit">
```

Output



onlinetutorial Submit

! Please enter a URL.

It looks for a URL(Uniform Resource Locator) like:

<http://www.onlinetutorial.co.in>

HTML File Input

If you want to allow users to upload a file (for example an image, video, mp3, or a PDF), you will need to use a file input box. Here are some points about File Input:

- This type of input creates a box that looks like a text input followed by a browse button.
- When the user clicks on the browse button, a window opens up that allows them to select a file from their computer to be uploaded to the website
- When you are allowing users to upload files, the method attribute on the `<form>` element must have a value of post. (You cannot send files using the HTTP get method.)

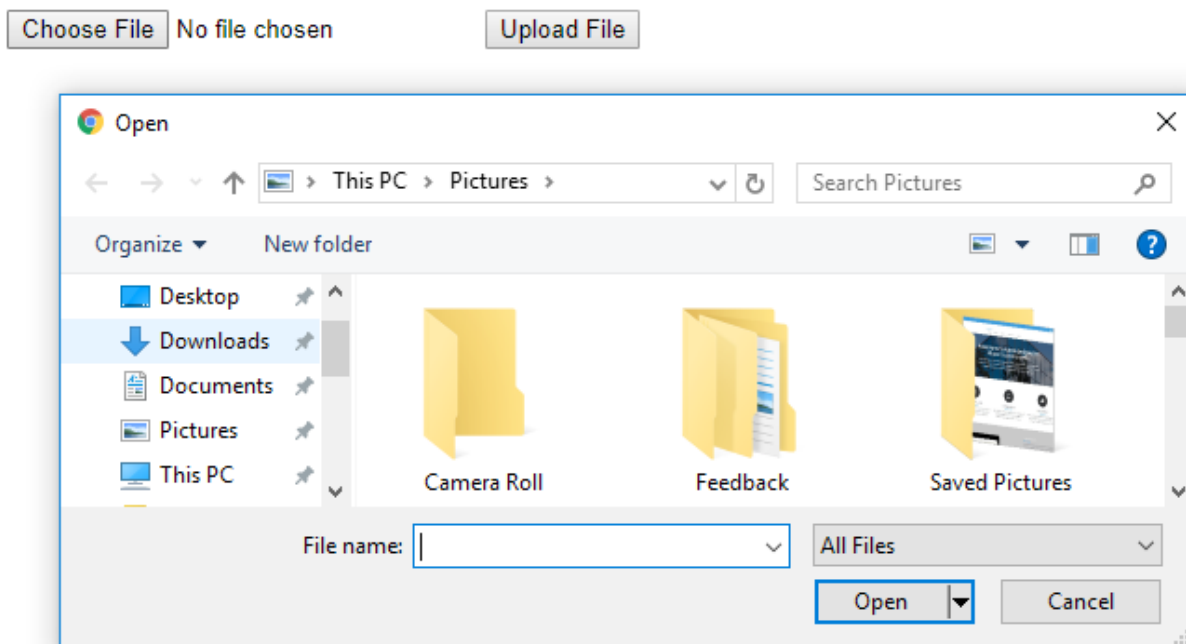
- When a user clicks on the browse button, the presentation of the window that allows them to browse for the file they want to upload will match the windows of the user's operating system. You cannot control the appearance of these windows

Example

```
<input type="file" name="file">
```

```
<input type="submit" name="submit" value="Upload File">
```

Output



HTML Audio Video Tags

HTML5 features include native audio and video support without the need for Flash. The HTML audio and video tag allows user to add multimedia to the website. It contains an **src** attribute, which defines the source of audio/ video. They also contain **controls** to control audio and videos.

Audio Tag:

HTML5 supports <audio> tag which is used to embed sound content in an HTML or XHTML

document as follows. Most commonly used audio formats are ogg, mp3 and wav.

Syntax

```
<audio controls>
  <source src=" file.extension " type="audio/mpeg">
  Your browser does not support the audio tag.
</audio>
```

Controls attribute used to show controls for the audio file.

Video Tag:

HTML5 supports <video> tag which is used to embed sound content in an HTML or XHTML document as follows

Syntax

```
<video width="320" height="240" controls>
  <source src ="file.extension" type="video/mp4">
  Your browser does not support the video tag.
</video>
```

Example

```
<!DOCTYPE html>

<html>

<head>

  <title>Page Title</title>

</head>

<body>

  <audio controls>

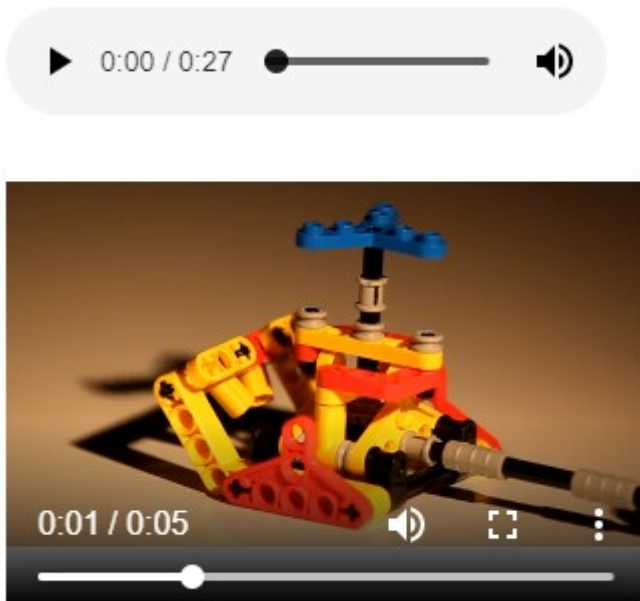
    <source src="SampleAudio_0.4mb.mp3" type="audio/mpeg">

    Your browser does not support the audio tag.

  </audio> <br/>
```

```
<video width="320" height="240" controls autoplay>
  <source src="http://techslides.com/demos/sample-videos/small.mp4"
type="video/mp4">
  Your browser does not support the video tag.
</video>
</body>
</html>
```

Output



By adding **autoplay** attribute, the video plays automatically on page loading.

HTML Img and anchor tag

A picture can say a thousand words, and great images help make the difference between an average-looking site and a really engaging one. HTML allows us to add images in a web page.

Syntax

```

```

src

This tells the browser where it can find the image file. This will usually be a relative URL pointing to an image on your own site. Its value is the full path of an image with extension.

alt

Attribute alt stands for alternative text. This provides a text description of the image which describes the image if you cannot see it due to wrong path or any other reason.

Example

```
<!DOCTYPE html>  
  
<html>  
  
<head>  
  
  <title>Page Title</title>  
  
</head>  
  
<body>  
  
    
  
</body>  
  
</html>
```

Output



Here Office Meeting is a title of the image.

Anchor Tag :

The most important part of any website is linking web pages together, to do this we use the anchor tag. It is also an inline and paired tag. The text or image on which we need a link, placed inside opening and closing tag.

To open a link in another window, you can set target to `_blank`.

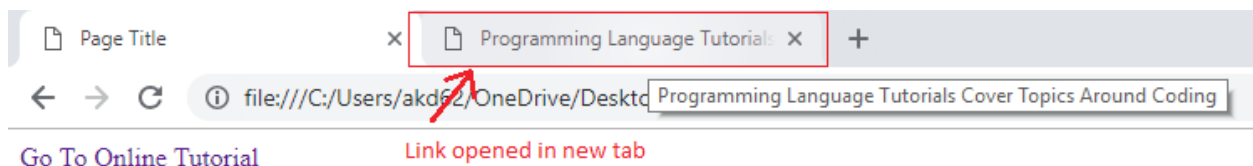
Syntax

```
<a href= "<!-- link to file -->" target="_blank">Anchor Link</a>
```

Example

```
<a href="http://www.onlinetutorial.co.in" target="_blank">Go To Online Tutorial</a>
```

Output



CSS:

CSS stands for Cascading Style Sheets. It is used to styling web pages. A simple website with no color and alignment looks very ugly. CSS3 contains various animation effects also. Here are different types of CSS:

Types of CSS:

1. Inline
2. Internal
3. External

In Inline CSS we use style as an attribute i.e. inside a single tag. You can see the description of inline CSS on page 10 of the same book.

In internal CSS, the CSS placed inside a <style> tag in page <head> tag. <style> is a paired tag. Here we use different selectors to select different elements for styling. The benefit of using

Internal CSS are:

- we can style group of elements at once.
- It leads a clean and less code.
- It avoids the repetition of code.
- The code written in <style> is easy to read as compared to code written in each tag.
- It refers to putting CSS inside same file rather than putting CSS in different file.
- Like inline CSS, it also reduces the number of files, a browser need to download prior to show your web page.
- So it also optimizes the browsing speed.

Id and Class Selector:

To select anything using ID we need to use # and to select class we use .(dot)

Example :

ID :

```
#abc{  
  CSS Prop Here  
}
```

Class :

```
.abc{  
  CSS Prop Here  
}
```

CSS Properties :

> **Height:**

Height Property use to define height of the element.

> Width :

Width Property use to define width of the element.

> text-decoration:

Possible values in text-decoration:

Underline | overline | line-through | none

Ex: text-decoration:underline;

> text-indent:

Text indent property is used to define wheres from our first line of the paragraph will be started.

> letter-spacing:

To give space between the letters we use letter spacing property.

> word-spacing:

Word-spacing property used to define gap between the words.

> text-transform:

Possible values are

Uppercase | lowercase | Capitalize

>line-height:

To define how much space line can acquire on the page.

> text-shadow:

Text shadow property used to apply shadow on the text.

Ex: text-shadow: 10px 10px 10px blue;

text-shadow: h-shadow, v-shadow, blur-rate, color;

> box-shadow:

Box shadow property used to apply shadow on box.

Ex: box-shadow: 10px 10px 10px blue;

box-shadow: h-shadow, v-shadow, blur-rate, color;

CSS Background Properties:

> background-color: to apply background color.

> background-image: use image on element background

Ex : background-image: url('<!-- img link here -->')

> background-size: background size property will be used with the background image to define the size of the background.

Ex : background-size:100% 100%; || background-size:h-size v-size;

> background-repeat:

Possible values are :
No-repeat | repeat-x | repeat-y | repeat.

> **background-attachment:**

Possible values are :
scroll | fixed

Margin & Padding

Margin and padding is used to define the space outside or inside the element, margin used to define the outside space and padding used to define the inside space in the html content.

> **margin** : use any value in pixels to define margin on all the sides.

> **margin-left**: to give margin from the left side.

> **margin-right**: to give margin from the right side.

> **margin-top**: to give margin from the top side.

> **margin-bottom**: to give margin from the bottom side.

> **padding** : use any value in pixels to define padding on all the sides.

> **padding-left**: to give padding from the left side.

> **padding-right**: to give padding from the right side.

> **padding-top**: to give padding from the top side.

> **padding-bottom**: to give padding from the bottom side.

Border Property:

To define the border on the element we have used border property. Example for the same as follow:

> **border** :

Border:1px solid red;

Define : border : border-size border-style border-color;

> **border-radius** :

To define radius on the edge of any element.

> **overflow** :

Overflow property is used to define what to do when something overflowing from the element.

Possible values are:

Scroll | hidden | auto

Text Property:

>**font-size**: to define the size of the text.

- > **font-family:** to change the text style.
- > **color:** to change the color of the text we use text color property.
- > **text-align:** possible properties are (left | right | center).
- > **font-weight:bold;** to make text bolder

CSS floating elements :

CSS floating elements used to laying out the page on the web for example if we have 2 box with the 50% in the size we can use floating to align them in the single row:

```
<div style="width:50%;height:400px; background:red; float:left;">Left</div>  
<div style="width:50%;height:400px; background:green; float:right;">Right</div>
```

CSS media query :

Css media queries are used to make our webpage responsive on all the devices:

Example :

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

The above code will change the color of the background on the page of lower than 600 in pixel.

CSS Positions:

Css positions are used to align items on the page according to user needs.

Ex : position: absolute;

Possible values are :

Static : Default value. Elements render in order, as they appear in the document flow

Absolute :The element is positioned relative to its first positioned (not static) ancestor element

Fixed :The element is positioned relative to the browser window

relative :The element is positioned relative to its normal position, so "left:20px" adds 20 pixels to the element's LEFT position

Sticky: The element is positioned based on the user's scroll position

CSS pseudo classes:

Pseudo classes are used for the special functions like animation:

1: hover : to change anything on mouse over the box.

2: focus: focus pseudo class will be used for the forms if you need to change some kind of the functionality on input focus.

Bootstrap

Bootstrap Structure:

1. Container
2. Row
3. Column

Example Code :

```
<html>
  <head>
    <title>BS Layout</title>
    <link rel="stylesheet" href="css/bootstrap.css">
  </head>
  <body>
    <div class="container">
      <div class="row">
        <div class="col-md-12"></div>
      </div>
    </div>
  </body>
</html>
```

Screen Size in Bootstrap:

There are 4 type of screens supported by bootstrap 4 :

1. Lg - large screen (LCD)
2. Md - medium screen (Computer)
3. Sm - small screen (Tablet)
4. Xs - extra small screen (Mobile)

Bootstrap Grid System:

Some Bootstrap 4 grid system rules:

- > Rows must be placed within a .container (fixed-width) or .container-fluid (full-width) for proper alignment and padding
- > Use rows to create horizontal groups of columns
- > Content should be placed within columns, and only columns may be immediate children of rows

- > Predefined classes like `.row` and `.col-sm-4` are available for quickly making grid layouts
- > Columns create gutters (gaps between column content) via padding. That padding is offset in rows for the first and last column via negative margin on `.rows`
- > Grid columns are created by specifying the number of 12 available columns you wish to span. For example, three equal columns would use three `.col-sm-4`
- > Column widths are in percentage, so they are always fluid and sized relative to their parent element

Bootstrap Typography Classes :

Text-center: to center align text within the contained div.

Text-left : to left align text.

Text-right : to right align text.

Text-success | info | danger | dark | light | warning : classes for changing the text color.

Bootstrap Headings:

.display-1: to .display-4 : display-1 is the largest heading in bootstrap

Bootstrap Table Classes:

.table : to create simple bootstrap table.

.table-striped : for striped bootstrap table table class must be there for this to work.

.table-bordered: for bordered table using bootstrap.

.table-hover : for hover table

.table-<colorname> : use success info warning danger etc to change color of the table.

Bootstrap Images Classes:

.rounded : to make image rounded from the edges.

.rounded-circle : for complete circle.

.img-thumbnail : to show images as a gallery.

Bootstrap Buttons :

.btn: for make nice bootstrap button.

.btn-<color name> : to change color of the button

.btn-lg, .btn-sm, btn-md : 3 size of the button

Bootstrap Forms:

.form-control : form-control class is used to make bootstrap input field looks nice.

Bootstrap List:

Here is the code for bootstrap Lists :

```
<ul class="list-group">
  <li class="list-group-item">1</li>
  <li class="list-group-item">2</li>
</ul>
```

Bootstrap Slider:

Requirement for bootstrap slider

Jquery.js and bootstrap.js file should be included on header.

Here is the code for the bootstrap slider:

```
<div id="demo" class="carousel slide" data-ride="carousel">
  <!-- The slideshow -->
  <div class="carousel-inner">
    <div class="carousel-item active">
      
    </div>
    <div class="carousel-item">
      
    </div>
    <div class="carousel-item">
      
    </div>
  </div>
  <!-- Left and right controls -->
  <a class="carousel-control-prev" href="#demo" data-slide="prev">
    <span class="carousel-control-prev-icon"></span>
  </a>
  <a class="carousel-control-next" href="#demo" data-slide="next">
    <span class="carousel-control-next-icon"></span>
  </a>
```

```
</a>
</div>
```

Bootstrap Modal :

Bootstrap modal are used as a popup windows to show any alert or the forms on the page.

Code for the bootstrap modal.

```
<!-- Trigger the modal with a button -->
```

```
<button type="button" class="btn btn-info btn-lg" data-toggle="modal" data-
target="#myModal">Open Modal</button>
```

```
<!-- Modal -->
```

```
<div id="myModal" class="modal fade" role="dialog">
<div class="modal-dialog">
```

```
<!-- Modal content-->
```

```
<div class="modal-content">
```

```
<div class="modal-header">
```

```
<button type="button" class="close" data-dismiss="modal">&times;</button>
```

```
<h4 class="modal-title">Modal Header</h4>
```

```
</div>
```

```
<div class="modal-body">
```

```
<p>Some text in the modal.</p>
```

```
</div>
```

```
<div class="modal-footer">
```

```
<button type="button" class="btn btn-default" data-
dismiss="modal">Close</button>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

Bootstrap Navbar With dropdown menu:

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

```
<!-- Brand -->
```

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

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

  <!-- Dropdown -->
  <li class="nav-item dropdown">
    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-
toggle="dropdown">
      Dropdown link
    </a>
    <div class="dropdown-menu">
      <a class="dropdown-item" href="#">Link 1</a>
      <a class="dropdown-item" href="#">Link 2</a>
      <a class="dropdown-item" href="#">Link 3</a>
    </div>
  </li>
</ul>
</nav>
```