HTML stands for **H**yper**t**ext **M**arkup **L**anguage, and it is the most widely used language to write Web Pages.

- **Hypertext** refers to the way in which Web pages (HTML documents) are linked together. Thus, the link available on a webpage is called Hypertext.
- As its name suggests, HTML is a **Markup Language** which means you use HTML to simply "mark-up" a text document with tags that tell a Web browser how to structure it to display.

Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers.

Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

Basic HTML Document

In its simplest form, following is an example of an HTML document −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>This is document title</title>
   </head>

   <body>
      <h1>This is a heading</h1>
      <p>Document content goes here.....</p>
   </body>

</html>
```

HTML Tags

As told earlier, HTML is a markup language and makes use of various tags to format the content. These tags are enclosed within angle braces **<Tag Name>**. Except few tags, most of the tags have their corresponding closing tags. For example, **<html>** has its closing tag **</html>** and **<body>** tag has its closing tag **</body>** tag etc.

Above example of HTML document uses the following tags −

Sr.No	Tag & Description
1	**<!DOCTYPE...>**

	This tag defines the document type and HTML version.
2	**<html>** This tag encloses the complete HTML document and mainly comprises of document header which is represented by <head>...</head> and document body which is represented by <body>...</body> tags.
3	**<head>** This tag represents the document's header which can keep other HTML tags like <title>, <link> etc.
4	**<title>** The <title> tag is used inside the <head> tag to mention the document title.
5	**<body>** This tag represents the document's body which keeps other HTML tags like <h1>, <div>, <p> etc.
6	**<h1>** This tag represents the heading.
7	**<p>** This tag represents a paragraph.

To learn HTML, you will need to study various tags and understand how they behave, while formatting a textual document. Learning HTML is simple as users have to learn the usage of different tags in order to format the text or images to make a beautiful webpage.

World Wide Web Consortium (W3C) recommends to use lowercase tags starting from HTML 4.

HTML Document Structure

A typical HTML document will have the following structure −

```
<html>

   <head>
      Document header related tags
   </head>

   <body>
      Document body related tags
   </body>
```

```
</html>
```

We will study all the header and body tags in subsequent chapters, but for now let's see what is document declaration tag.

The <!DOCTYPE> Declaration

The <!DOCTYPE> declaration tag is used by the web browser to understand the version of the HTML used in the document. Current version of HTML is 5 and it makes use of the following declaration −

```
<!DOCTYPE html>
```

There are many other declaration types which can be used in HTML document depending on what version of HTML is being used. We will see more details on this while discussing <!DOCTYPE...> tag along with other HTML tags.

HTML - Basic Tags

Heading Tags

Any document starts with a heading. You can use different sizes for your headings. HTML also has six levels of headings, which use the elements **<h1>, <h2>, <h3>, <h4>, <h5>,** and **<h6>**. While displaying any heading, browser adds one line before and one line after that heading.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Heading Example</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>
   </body>

</html>
```

This will produce the following result −

Paragraph Tag

The **<p>** tag offers a way to structure your text into different paragraphs. Each paragraph of text should go in between an opening <p> and a closing </p> tag as shown below in the example −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Paragraph Example</title>
   </head>

   <body>
      <p>Here is a first paragraph of text.</p>
      <p>Here is a second paragraph of text.</p>
      <p>Here is a third paragraph of text.</p>
   </body>

</html>
```

This will produce the following result −

Line Break Tag

Whenever you use the **<br />** element, anything following it starts from the next line. This tag is an example of an **empty** element, where you do not need opening and closing tags, as there is nothing to go in between them.

The <br /> tag has a space between the characters **br** and the forward slash. If you omit this space, older browsers will have trouble rendering the line break, while if you miss the forward slash character and just use <br> it is not valid in XHTML.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Line Break  Example</title>
   </head>

   <body>
      <p>Hello<br />
         You delivered your assignment ontime.<br />
         Thanks<br />
         Mahnaz</p>
   </body>

</html>
```

This will produce the following result −

Centering Content

You can use **<center>** tag to put any content in the center of the page or any table cell.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Centring Content Example</title>
   </head>

   <body>
      <p>This text is not in the center.</p>

      <center>
         <p>This text is in the center.</p>
      </center>
   </body>

</html>
```

This will produce following result −

Horizontal Lines

Horizontal lines are used to visually break-up sections of a document. The **<hr>** tag creates a line from the current position in the document to the right margin and breaks the line accordingly.

For example, you may want to give a line between two paragraphs as in the given example below −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
```

```
      <title>Horizontal Line Example</title>
   </head>

   <body>
      <p>This is paragraph one and should be on top</p>
      <hr />
      <p>This is paragraph two and should be at bottom</p>
   </body>

</html>
```

This will produce the following result −

Again **<hr />** tag is an example of the **empty** element, where you do not need opening and closing tags, as there is nothing to go in between them.

The **<hr />** element has a space between the characters **hr** and the forward slash. If you omit this space, older browsers will have trouble rendering the horizontal line, while if you miss the forward slash character and just use **<hr>** it is not valid in XHTML

Preserve Formatting

Sometimes, you want your text to follow the exact format of how it is written in the HTML document. In these cases, you can use the preformatted tag **<pre>**.

Any text between the opening **<pre>** tag and the closing **</pre>** tag will preserve the formatting of the source document.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Preserve Formatting Example</title>
   </head>

   <body>
      <pre>
         function testFunction( strText ){
            alert (strText)
         }
      </pre>
   </body>

</html>
```

This will produce the following result −

Try using the same code without keeping it inside **<pre>...</pre>** tags

Nonbreaking Spaces

Suppose you want to use the phrase "12 Angry Men." Here, you would not want a browser to split the "12, Angry" and "Men" across two lines −

```
An example of this technique appears in the movie "12 Angry Men."
```

In cases, where you do not want the client browser to break text, you should use a nonbreaking space entity **&nbsp;** instead of a normal space. For example, when coding the "12 Angry Men" in a paragraph, you should use something similar to the following code −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Nonbreaking Spaces Example</title>
   </head>

   <body>
      <p>An example of this technique appears in the movie
"12&nbsp;Angry&nbsp;Men."</p>
   </body>

</html>
```

This will produce the following result −

HTML - Elements

An **HTML element** is defined by a starting tag. If the element contains other content, it ends with a closing tag, where the element name is preceded by a forward slash as shown below with few tags −

Start Tag	Content	End Tag
<p>	This is paragraph content.	</p>
<h1>	This is heading content.	</h1>
<div>	This is division content.	</div>
<br />		

So here **<p>....</p>** is an HTML element, **<h1>...</h1>** is another HTML element. There are some HTML elements which don't need to be closed, such as **<img.../>**, **<hr />** and **<br />** elements. These are known as **void elements**.

HTML documents consists of a tree of these elements and they specify how HTML documents should be built, and what kind of content should be placed in what part of an HTML document.

HTML Tag vs. Element

An HTML element is defined by a *starting tag*. If the element contains other content, it ends with a *closing tag*.

For example, **<p>** is starting tag of a paragraph and **</p>** is closing tag of the same paragraph but **<p>This is paragraph</p>** is a paragraph element.

Nested HTML Elements

It is very much allowed to keep one HTML element inside another HTML element −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Nested Elements Example</title>
   </head>

   <body>
      <h1>This is <i>italic</i> heading</h1>
      <p>This is <u>underlined</u> paragraph</p>
   </body>

</html>
```

This will display the following result −

HTML - Attributes

We have seen few HTML tags and their usage like heading tags **<h1>, <h2>,** paragraph tag **<p>** and other tags. We used them so far in their simplest form, but most of the HTML tags can also have attributes, which are extra bits of information.

An attribute is used to define the characteristics of an HTML element and is placed inside the element's opening tag. All attributes are made up of two parts − a **name** and a **value**

- The **name** is the property you want to set. For example, the paragraph **<p>** element in the example carries an attribute whose name is **align**, which you can use to indicate the alignment of paragraph on the page.

- The **value** is what you want the value of the property to be set and always put within quotations. The below example shows three possible values of align attribute: **left, center** and **right**.

Attribute names and attribute values are case-insensitive. However, the World Wide Web Consortium (W3C) recommends lowercase attributes/attribute values in their HTML 4 recommendation.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Align Attribute  Example</title>
   </head>

   <body>
      <p align = "left">This is left aligned</p>
      <p align = "center">This is center aligned</p>
      <p align = "right">This is right aligned</p>
   </body>

</html>
```

This will display the following result −

Core Attributes

The four core attributes that can be used on the majority of HTML elements (although not all) are −

- Id
- Title
- Class
- Style

The Id Attribute

The **id** attribute of an HTML tag can be used to uniquely identify any element within an HTML page. There are two primary reasons that you might want to use an id attribute on an element −

- If an element carries an id attribute as a unique identifier, it is possible to identify just that element and its content.

- If you have two elements of the same name within a Web page (or style sheet), you can use the id attribute to distinguish between elements that have the same name.

We will discuss style sheet in separate tutorial. For now, let's use the id attribute to distinguish between two paragraph elements as shown below.

Example

```
<p id = "html">This para explains what is HTML</p>
<p id = "css">This para explains what is Cascading Style
Sheet</p>
```

The title Attribute

The **title** attribute gives a suggested title for the element. They syntax for the **title** attribute is similar as explained for **id** attribute −

The behavior of this attribute will depend upon the element that carries it, although it is often displayed as a tooltip when cursor comes over the element or while the element is loading.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>The title Attribute Example</title>
   </head>

   <body>
      <h3 title = "Hello HTML!">Titled Heading Tag Example</h3>
   </body>

</html>
```

This will produce the following result −

Now try to bring your cursor over "Titled Heading Tag Example" and you will see that whatever title you used in your code is coming out as a tooltip of the cursor.

The class Attribute

The **class** attribute is used to associate an element with a style sheet, and specifies the class of element. You will learn more about the use of the class attribute when you will learn Cascading Style Sheet (CSS). So for now you can avoid it.

The value of the attribute may also be a space-separated list of class names. For example −

```
class = "className1 className2 className3"
```

The style Attribute

The style attribute allows you to specify Cascading Style Sheet (CSS) rules within the element.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>The style Attribute</title>
   </head>

   <body>
      <p style = "font-family:arial; color:#FF0000;">Some
text...</p>
   </body>

</html>
```

This will produce the following result −

At this point of time, we are not learning CSS, so just let's proceed without bothering much about CSS. Here, you need to understand what are HTML attributes and how they can be used while formatting content.

Internationalization Attributes

There are three internationalization attributes, which are available for most (although not all) XHTML elements.

- dir
- lang
- xml:lang

The dir Attribute

The **dir** attribute allows you to indicate to the browser about the direction in which the text should flow. The dir attribute can take one of two values, as you can see in the table that follows −

Value	Meaning
ltr	Left to right (the default value)
rtl	Right to left (for languages such as Hebrew or Arabic that are read right to left)

Example

Live Demo

```
<!DOCTYPE html>
<html dir = "rtl">

   <head>
      <title>Display Directions</title>
   </head>

   <body>
      This is how IE 5 renders right-to-left directed text.
   </body>

</html>
```

This will produce the following result −

When *dir* attribute is used within the <html> tag, it determines how text will be presented within the entire document. When used within another tag, it controls the text's direction for just the content of that tag.

The lang Attribute

The **lang** attribute allows you to indicate the main language used in a document, but this attribute was kept in HTML only for backwards compatibility with earlier versions of HTML. This attribute has been replaced by the **xml:lang** attribute in new XHTML documents.

The values of the *lang* attribute are ISO-639 standard two-character language codes. Check **HTML Language Codes: ISO 639** for a complete list of language codes.

Example

Live Demo

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

   <head>
      <title>English Language Page</title>
   </head>

   <body>
      This page is using English Language
   </body>

</html>
```

This will produce the following result −

The xml:lang Attribute

The *xml:lang* attribute is the XHTML replacement for the *lang* attribute. The value of the *xml:lang* attribute should be an ISO-639 country code as mentioned in previous section.

Generic Attributes

Here's a table of some other attributes that are readily usable with many of the HTML tags.

Attribute	**Options**	**Function**
align	right, left, center	Horizontally aligns tags
valign	top, middle, bottom	Vertically aligns tags within an HTML element.
bgcolor	numeric, hexidecimal, RGB values	Places a background color behind an element
background	URL	Places a background image behind an element
id	User Defined	Names an element for use with Cascading Style Sheets.
class	User Defined	Classifies an element for use with Cascading Style Sheets.
width	Numeric Value	Specifies the width of tables, images, or table cells.
height	Numeric Value	Specifies the height of tables, images, or table cells.
title	User Defined	"Pop-up" title of the elements.

We will see related examples as we will proceed to study other HTML tags. For a complete list of HTML Tags and related attributes please check reference to HTML Tags List.

HTML - Formatting

If you use a word processor, you must be familiar with the ability to make text bold, italicized, or underlined; these are just three of the ten options available to indicate how text can appear in HTML and XHTML.

Bold Text

Anything that appears within **<b>...</b>** element, is displayed in bold as shown below −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Bold Text Example</title>
   </head>

   <body>
      <p>The following word uses a <b>bold</b> typeface.</p>
   </body>

</html>
```

This will produce the following result −

Italic Text

Anything that appears within **<i>...</i>** element is displayed in italicized as shown below −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Italic Text Example</title>
   </head>

   <body>
      <p>The following word uses an <i>italicized</i>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Underlined Text

Anything that appears within **<u>...</u>** element, is displayed with underline as shown below −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Underlined Text Example</title>
   </head>

   <body>
      <p>The following word uses an <u>underlined</u>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Strike Text

Anything that appears within **<strike>...</strike>** element is displayed with strikethrough, which is a thin line through the text as shown below −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Strike Text Example</title>
   </head>

   <body>
      <p>The following word uses a <strike>strikethrough</strike>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Monospaced Font

The content of a **<tt>...</tt>** element is written in monospaced font. Most of the fonts are known as variable-width fonts because different letters are of different widths (for example, the letter 'm' is wider than the letter 'i'). In a monospaced font, however, each letter has the same width.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Monospaced Font Example</title>
   </head>

   <body>
      <p>The following word uses a <tt>monospaced</tt>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Superscript Text

The content of a **<sup>...</sup>** element is written in superscript; the font size used is the same size as the characters surrounding it but is displayed half a character's height above the other characters.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Superscript Text Example</title>
   </head>

   <body>
      <p>The following word uses a <sup>superscript</sup>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Subscript Text

The content of a **<sub>...</sub>** element is written in subscript; the font size used is the same as the characters surrounding it, but is displayed half a character's height beneath the other characters.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Subscript Text Example</title>
   </head>

   <body>
      <p>The following word uses a <sub>subscript</sub>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Inserted Text

Anything that appears within **<ins>...</ins>** element is displayed as inserted text.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Inserted Text Example</title>
   </head>

   <body>
      <p>I want to drink <del>cola</del> <ins>wine</ins></p>
   </body>

</html>
```

This will produce the following result −

Deleted Text

Anything that appears within **<del>...</del>** element, is displayed as deleted text.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Deleted Text Example</title>
   </head>

   <body>
      <p>I want to drink <del>cola</del> <ins>wine</ins></p>
   </body>

</html>
```

This will produce the following result −

Larger Text

The content of the **<big>...</big>** element is displayed one font size larger than the rest of the text surrounding it as shown below −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Larger Text Example</title>
   </head>

   <body>
      <p>The following word uses a <big>big</big> typeface.</p>
   </body>

</html>
```

This will produce the following result −

Smaller Text

The content of the **<small>...</small>** element is displayed one font size smaller than the rest of the text surrounding it as shown below −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Smaller Text Example</title>
   </head>

   <body>
      <p>The following word uses a <small>small</small>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Grouping Content

The **<div>** and **<span>** elements allow you to group together several elements to create sections or subsections of a page.

For example, you might want to put all of the footnotes on a page within a <div> element to indicate that all of the elements within that <div> element relate to the footnotes. You might then attach a style to this <div> element so that they appear using a special set of style rules.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Div Tag Example</title>
   </head>

   <body>
      <div id = "menu" align = "middle" >
         <a href = "/index.htm">HOME</a> |
         <a href = "/about/contact_us.htm">CONTACT</a> |
         <a href = "/about/index.htm">ABOUT</a>
      </div>

      <div id = "content" align = "left" >
         <h5>Content Articles</h5>
         <p>Actual content goes here.....</p>
      </div>
   </body>

</html>
```

This will produce the following result −

The <span> element, on the other hand, can be used to group inline elements only. So, if you have a part of a sentence or paragraph which you want to group together, you could use the <span> element as follows.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Span Tag Example</title>
   </head>

   <body>
      <p>This is the example of <span style = "color:green">span
tag</span>
         and the <span style = "color:red">div tag</span>
alongwith CSS</p>
   </body>

</html>
```

This will produce the following result −

These tags are commonly used with CSS to allow you to attach a style to a section of a page.

HTML - Phrase Tags

The phrase tags have been desicolgned for specific purposes, though they are displayed in a similar way as other basic tags like **<b>, <i>, <pre>**, and **<tt>**, you have seen in previous chapter. This chapter will take you through all the important phrase tags, so let's start seeing them one by one.

Emphasized Text

Anything that appears within **<em>...</em>** element is displayed as emphasized text.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Emphasized Text Example</title>
   </head>
   <body>
      <p>The following word uses an <em>emphasized</em>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Marked Text

Anything that appears with-in **<mark>...</mark>** element, is displayed as marked with yellow ink.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Marked Text Example</title>
   </head>

   <body>
      <p>The following word has been <mark>marked</mark> with
yellow</p>
   </body>

</html>
```

This will produce the following result −

Strong Text

Anything that appears within **<strong>...</strong>** element is displayed as important text.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Strong Text Example</title>
   </head>

   <body>
```

```
      <p>The following word uses a <strong>strong</strong>
typeface.</p>
   </body>

</html>
```

This will produce the following result −

Text Abbreviation

You can abbreviate a text by putting it inside opening <abbr> and closing </abbr> tags. If present, the title attribute must contain this full description and nothing else.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Text Abbreviation</title>
   </head>

   <body>
      <p>My best friend's name is  <abbr title =
"Abhishek">Abhy</abbr>.</p>
   </body>

</html>
```

This will produce the following result −

Acronym Element

The **<acronym>** element allows you to indicate that the text between <acronym> and </acronym> tags is an acronym.

At present, the major browsers do not change the appearance of the content of the <acronym> element.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Acronym Example</title>
   </head>

   <body>
      <p>This chapter covers marking up text in
<acronym>XHTML</acronym>.</p>
   </body>

</html>
```

This will produce the following result −

Text Direction

The **<bdo>...</bdo>** element stands for Bi-Directional Override and it is used to override the current text direction.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Text Direction Example</title>
   </head>

   <body>
      <p>This text will go left to right.</p>
      <p><bdo dir = "rtl">This text will go right to
left.</bdo></p>
   </body>

</html>
```

This will produce the following result −

Special Terms

The **<dfn>...</dfn>** element (or HTML Definition Element) allows you to specify that you are introducing a special term. It's usage is similar to italic words in the midst of a paragraph.

Typically, you would use the <dfn> element the first time you introduce a key term. Most recent browsers render the content of a <dfn> element in an italic font.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
```

```
      <title>Special Terms Example</title>
   </head>

   <body>
      <p>The following word is a <dfn>special</dfn> term.</p>
   </body>

</html>
```

This will produce the following result −

Quoting Text

When you want to quote a passage from another source, you should put it in between **<blockquote>...</blockquote>** tags.

Text inside a <blockquote> element is usually indented from the left and right edges of the surrounding text, and sometimes uses an italicized font.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Blockquote Example</title>
   </head>

   <body>
      <p>The following description of XHTML is taken from the W3C
Web site:</p>

      <blockquote>XHTML 1.0 is the W3C's first Recommendation for
XHTML,following on
         from earlier work on HTML 4.01, HTML 4.0, HTML 3.2 and
HTML 2.0.</blockquote>
   </body>

</html>
```

This will produce the following result −

Short Quotations

The **<q>...</q>** element is used when you want to add a double quote within a sentence.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Double Quote Example</title>
   </head>

   <body>
      <p>Amit is in Spain, <q>I think I am wrong</q>.</p>
   </body>

</html>
```

This will produce the following result −

Text Citations

If you are quoting a text, you can indicate the source placing it between an opening **<cite>** tag and closing **</cite>** tag

As you would expect in a print publication, the content of the <cite> element is rendered in italicized text by default.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Citations Example</title>
   </head>

   <body>
      <p>This HTML tutorial is derived from <cite>W3 Standard for
HTML</cite>.</p>
   </body>

</html>
```

This will produce the following result −

Computer Code

Any programming code to appear on a Web page should be placed inside **<code>...</code>** tags. Usually the content of the <code> element is presented in a monospaced font, just like the code in most programming books.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Computer Code Example</title>
   </head>

   <body>
      <p>Regular text. <code>This is code.</code> Regular
text.</p>
   </body>

</html>
```

This will produce the following result −

Keyboard Text

When you are talking about computers, if you want to tell a reader to enter some text, you can use the **<kbd>...</kbd>** element to indicate what should be typed in, as in this example.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Keyboard Text Example</title>
   </head>

   <body>
      <p>Regular text. <kbd>This is inside kbd element</kbd>
Regular text.</p>
   </body>

</html>
```

This will produce the following result −

Programming Variables

This element is usually used in conjunction with the **<pre>** and **<code>** elements to indicate that the content of that element is a variable.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Variable Text Example</title>
   </head>

   <body>
      <p><code>document.write("<var>user-name</var>")</code></p>
   </body>

</html>
```

This will produce the following result −

Program Output

The **<samp>...</samp>** element indicates sample output from a program, and script etc. Again, it is mainly used when documenting programming or coding concepts.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Program Output Example</title>
   </head>

   <body>
      <p>Result produced by the program is <samp>Hello
World!</samp></p>
   </body>

</html>
```

This will produce the following result −

Address Text

The **<address>...</address>** element is used to contain any address.

Example

Live Demo

```
<!DOCTYPE html>
```

```
<html>

   <head>
      <title>Address Example</title>
   </head>

   <body>
      <address>388A, Road No 22, Jubilee Hills -
Hyderabad</address>
   </body>

</html>
```

This will produce the following result −

HTML - Meta Tags

HTML lets you specify metadata - additional important information about a document in a variety of ways. The META elements can be used to include name/value pairs describing properties of the HTML document, such as author, expiry date, a list of keywords, document author etc.

The **<meta>** tag is used to provide such additional information. This tag is an empty element and so does not have a closing tag but it carries information within its attributes.

You can include one or more meta tags in your document based on what information you want to keep in your document but in general, meta tags do not impact physical appearance of the document so from appearance point of view, it does not matter if you include them or not.

Adding Meta Tags to Your Documents

You can add metadata to your web pages by placing <meta> tags inside the header of the document which is represented by **<head>** and **</head>** tags. A meta tag can have following attributes in addition to core attributes −

Sr.No	Attribute & Description
1	**Name** Name for the property. Can be anything. Examples include, keywords, description, author, revised, generator etc.
2	**content** Specifies the property's value.

3	**scheme** Specifies a scheme to interpret the property's value (as declared in the content attribute).
4	**http-equiv** Used for http response message headers. For example, http-equiv can be used to refresh the page or to set a cookie. Values include content-type, expires, refresh and set-cookie.

Specifying Keywords

You can use <meta> tag to specify important keywords related to the document and later these keywords are used by the search engines while indexing your webpage for searching purpose.

Example

Following is an example, where we are adding HTML, Meta Tags, Metadata as important keywords about the document.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Meta Tags Example</title>
      <meta name = "keywords" content = "HTML, Meta Tags,
Metadata" />
   </head>

   <body>
      <p>Hello HTML5!</p>
   </body>

</html>
```

This will produce the following result −

Document Description

You can use <meta> tag to give a short description about the document. This again can be used by various search engines while indexing your webpage for searching purpose.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Meta Tags Example</title>
      <meta name = "keywords" content = "HTML, Meta Tags, 
Metadata" />
      <meta name = "description" content = "Learning about Meta 
Tags." />
   </head>

   <body>
      <p>Hello HTML5!</p>
   </body>

</html>
```

Document Revision Date

You can use <meta> tag to give information about when last time the document was updated. This information can be used by various web browsers while refreshing your webpage.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Meta Tags Example</title>
      <meta name = "keywords" content = "HTML, Meta Tags, 
Metadata" />
      <meta name = "description" content = "Learning about Meta 
Tags." />
      <meta name = "revised" content = "Tutorialspoint, 3/7/2014" 
/>
   </head>

   <body>
      <p>Hello HTML5!</p>
   </body>

</html>
```

Document Refreshing

A <meta> tag can be used to specify a duration after which your web page will keep refreshing automatically.

Example

If you want your page keep refreshing after every 5 seconds then use the following syntax.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Meta Tags Example</title>
      <meta name = "keywords" content = "HTML, Meta Tags, 
Metadata" />
      <meta name = "description" content = "Learning about Meta 
Tags." />
      <meta name = "revised" content = "Tutorialspoint, 3/7/2014" 
/>
      <meta http-equiv = "refresh" content = "5" />
   </head>

   <body>
      <p>Hello HTML5!</p>
   </body>

</html>
```

Page Redirection

You can use <meta> tag to redirect your page to any other webpage. You can also specify a duration if you want to redirect the page after a certain number of seconds.

Example

Following is an example of redirecting current page to another page after 5 seconds. If you want to redirect page immediately then do not specify *content* attribute.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Meta Tags Example</title>
      <meta name = "keywords" content = "HTML, Meta Tags, 
Metadata" />
      <meta name = "description" content = "Learning about Meta 
Tags." />
      <meta name = "revised" content = "Tutorialspoint, 3/7/2014" 
/>
```

```
      <meta http-equiv = "refresh" content = "5; url = 
http://www.tutorialspoint.com" />
   </head>
   
   <body>
      <p>Hello HTML5!</p>
   </body>
   
</html>
```

Setting Cookies

Cookies are data, stored in small text files on your computer and it is exchanged between web browser and web server to keep track of various information based on your web application need.

You can use <meta> tag to store cookies on client side and later this information can be used by the Web Server to track a site visitor.

Example

Following is an example of redirecting current page to another page after 5 seconds. If you want to redirect page immediately then do not specify *content* attribute.

Live Demo

```
<!DOCTYPE html>
<html>
   <head>
      <title>Meta Tags Example</title>
      <meta http-equiv = "cookie" content = "userid = xyz; 
expires = Wednesday, 08-Aug-15 23:59:59 GMT;" />
      
   </head>
   <body>
      <p>Hello HTML5!</p>
   </body>
</html>
```

If you do not include the expiration date and time, the cookie is considered a session cookie and will be deleted when the user exits the browser.

Note − You can check PHP and Cookies tutorial for a complete detail on Cookies.

Setting Author Name

You can set an author name in a web page using meta tag. See an example below −

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Meta Tags Example</title>
      <meta name = "keywords" content = "HTML, Meta Tags,
Metadata" />
      <meta name = "description" content = "Learning about Meta
Tags." />
      <meta name = "author" content = "Mahnaz Mohtashim" />
   </head>

   <body>
      <p>Hello HTML5!</p>
   </body>

</html>
```

Specify Character Set

You can use <meta> tag to specify character set used within the webpage.

Example

By default, Web servers and Web browsers use ISO-8859-1 (Latin1) encoding to process Web pages. Following is an example to set UTF-8 encoding −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Meta Tags Example</title>
      <meta name = "keywords" content = "HTML, Meta Tags,
Metadata" />
      <meta name = "description" content = "Learning about Meta
Tags." />
      <meta name = "author" content = "Mahnaz Mohtashim" />
      <meta http-equiv = "Content-Type" content = "text/html;
charset = UTF-8" />
   </head>

   <body>
      <p>Hello HTML5!</p>
   </body>

</html>
```

To serve the static page with traditional Chinese characters, the webpage must contain a <meta> tag to set Big5 encoding −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Meta Tags Example</title>
      <meta name = "keywords" content = "HTML, Meta Tags,
Metadata" />
      <meta name = "description" content = "Learning about Meta
Tags." />
      <meta name = "author" content = "Mahnaz Mohtashim" />
      <meta http-equiv = "Content-Type" content = "text/html;
charset = Big5" />
   </head>

   <body>
      <p>Hello HTML5!</p>
   </body>

</html>
```

HTML - Comments

Comment is a piece of code which is ignored by any web browser. It is a good practice to add comments into your HTML code, especially in complex documents, to indicate sections of a document, and any other notes to anyone looking at the code. Comments help you and others understand your code and increases code readability.

HTML comments are placed in between **<!-- ... -->** tags. So, any content placed with-in <!-- ... --> tags will be treated as comment and will be completely ignored by the browser.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>  <!-- Document Header Starts -->
      <title>This is document title</title>
   </head> <!-- Document Header Ends -->

   <body>
      <p>Document content goes here.....</p>
   </body>

</html>
```

This will produce the following result without displaying the content given as a part of comments −

Valid vs Invalid Comments

Comments do not nest which means a comment cannot be put inside another comment. Second the double-dash sequence "--" may not appear inside a comment except as part of the closing --> tag. You must also make sure that there are no spaces in the start-of comment string.

Example

Here, the given comment is a valid comment and will be wiped off by the browser.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Valid Comment Example</title>
   </head>

   <body>
      <!--   This is valid comment -->
      <p>Document content goes here.....</p>
   </body>

</html>
```

This will produce the following result −

But, following line is not a valid comment and will be displayed by the browser. This is because there is a space between the left angle bracket and the exclamation mark.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Invalid Comment Example</title>
   </head>

   <body>
      < !--   This is not a valid comment -->
      <p>Document content goes here.....</p>
   </body>

</html>
```

This will produce the following result −

Multiline Comments

So far we have seen single line comments, but HTML supports multi-line comments as well.

You can comment multiple lines by the special beginning tag <!-- and ending tag --> placed before the first line and end of the last line as shown in the given example below.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Multiline Comments</title>
   </head>

   <body>
      <!--
         This is a multiline comment and it can
         span through as many as lines you like.
      -->

      <p>Document content goes here.....</p>
   </body>

</html>
```

This will produce the following result −

Conditional Comments

Conditional comments only work in Internet Explorer (IE) on Windows but they are ignored by other browsers. They are supported from Explorer 5 onwards, and you can use them to give conditional instructions to different versions of IE.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Conditional Comments</title>

      <!--[if IE 6]>
         Special instructions for IE 6 here
      <![endif]-->
   </head>

   <body>
      <p>Document content goes here.....</p>
   </body>

</html>
```

You will come across a situation where you will need to apply a different style sheet based on different versions of Internet Explorer, in such situation conditional comments will be helpful.

Using Comment Tag

There are few browsers that support <comment> tag to comment a part of HTML code.

Note − The <comment> tag deprecated in HTML5. Do not use this element.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Using Comment Tag</title>
   </head>

   <body>
      <p>This is <comment>not</comment> Internet Explorer.</p>
   </body>

</html>
```

If you are using IE, then it will produce following result −

But if you are not using IE, then it will produce following result −

Commenting Script Code

Though you will learn JavaScript with HTML, in a separate tutorial, but here you must make a note that if you are using Java Script or VB Script in your HTML code then it is recommended to put that script code inside proper HTML comments so that old browsers can work properly.

Example

Live Demo

```
<!DOCTYPE html>
<html>

```

```
   <head>
      <title>Commenting Script Code</title>

      <script>
         <!--
            document.write("Hello World!")
         //-->
      </script>
   </head>

   <body>
      <p>Hello , World!</p>
   </body>

</html>
```

This will produce the following result −

Commenting Style Sheets

Though you will learn using style sheets with HTML in a separate tutorial, but here you must make a note that if you are using Cascading Style Sheet (CSS) in your HTML code then it is recommended to put that style sheet code inside proper HTML comments so that old browsers can work properly.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Commenting Style Sheets</title>

      <style>
         <!--
            .example {
               border:1px solid #4a7d49;
            }
         //-->
      </style>
   </head>

   <body>
      <div class = "example">Hello , World!</div>
   </body>

</html>
```

This will produce the following result −

HTML - Images

Images are very important to beautify as well as to depict many complex concepts in simple way on your web page. This tutorial will take you through simple steps to use images in your web pages.

Insert Image

You can insert any image in your web page by using **<img>** tag. Following is the simple syntax to use this tag.

```
<img src = "Image URL" ... attributes-list/>
```

The <img> tag is an empty tag, which means that, it can contain only list of attributes and it has no closing tag.

Example

To try following example, let's keep our HTML file test.htm and image file test.png in the same directory −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Using Image in Webpage</title>
   </head>

   <body>
      <p>Simple Image Insert</p>
      <img src = "/html/images/test.png" alt = "Test Image" />
   </body>

</html>
```

This will produce the following result −

You can use PNG, JPEG or GIF image file based on your comfort but make sure you specify correct image file name in **src** attribute. Image name is always case sensitive.

The **alt** attribute is a mandatory attribute which specifies an alternate text for an image, if the image cannot be displayed.

Set Image Location

Usually we keep all the images in a separate directory. So let's keep HTML file test.htm in our home directory and create a subdirectory **images** inside the home directory where we will keep our image test.png.

Example

Assuming our image location is "image/test.png", try the following example −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Using Image in Webpage</title>
   </head>

   <body>
      <p>Simple Image Insert</p>
      <img src = "/html/images/test.png" alt = "Test Image" />
   </body>

</html>
```

This will produce the following result −

Set Image Width/Height

You can set image width and height based on your requirement using **width** and **height** attributes. You can specify width and height of the image in terms of either pixels or percentage of its actual size.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Set Image Width and Height</title>
   </head>

   <body>
      <p>Setting image width and height</p>
      <img src = "/html/images/test.png" alt = "Test Image" width
= "150" height = "100"/>
   </body>

</html>
```

This will produce the following result −

Set Image Border

By default, image will have a border around it, you can specify border thickness in terms of pixels using border attribute. A thickness of 0 means, no border around the picture.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Set Image Border</title>
   </head>

   <body>
      <p>Setting image Border</p>
      <img src = "/html/images/test.png" alt = "Test Image"
border = "3"/>
   </body>

</html>
```

This will produce the following result −

Set Image Alignment

By default, image will align at the left side of the page, but you can use **align** attribute to set it in the center or right.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Set Image Alignment</title>
   </head>

   <body>
      <p>Setting image Alignment</p>
      <img src = "/html/images/test.png" alt = "Test Image"
border = "3" align = "right"/>
   </body>

</html>
```

This will produce the following result −

Free Web Graphics

For Free Web Graphics including patterns you can look into Free Web Graphics

HTML - Tables

The HTML tables allow web authors to arrange data like text, images, links, other tables, etc. into rows and columns of cells.

The HTML tables are created using the **<table>** tag in which the **<tr>** tag is used to create table rows and **<td>** tag is used to create data cells. The elements under <td> are regular and left aligned by default

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Tables</title>
   </head>

   <body>
      <table border = "1">
         <tr>
            <td>Row 1, Column 1</td>
            <td>Row 1, Column 2</td>
         </tr>

         <tr>
            <td>Row 2, Column 1</td>
            <td>Row 2, Column 2</td>
         </tr>
      </table>

   </body>
</html>
```

This will produce the following result −

Here, the **border** is an attribute of <table> tag and it is used to put a border across all the cells. If you do not need a border, then you can use border = "0".

Table Heading

Table heading can be defined using **<th>** tag. This tag will be put to replace <td> tag, which is used to represent actual data cell. Normally you will put your top row as table heading as shown below, otherwise you can use <th> element in any row. Headings, which are defined in <th> tag are centered and bold by default.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table Header</title>
   </head>

   <body>
      <table border = "1">
         <tr>
            <th>Name</th>
            <th>Salary</th>
         </tr>
         <tr>
            <td>Ramesh Raman</td>
            <td>5000</td>
         </tr>

         <tr>
            <td>Shabbir Hussein</td>
            <td>7000</td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result −

Cellpadding and Cellspacing Attributes

There are two attributes called *cellpadding* and *cellspacing* which you will use to adjust the white space in your table cells. The cellspacing attribute defines space between table cells, while cellpadding represents the distance between cell borders and the content within a cell.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table Cellpadding</title>
   </head>

   <body>
      <table border = "1" cellpadding = "5" cellspacing = "5">
         <tr>
            <th>Name</th>
            <th>Salary</th>
         </tr>
         <tr>
            <td>Ramesh Raman</td>
            <td>5000</td>
         </tr>
         <tr>
            <td>Shabbir Hussein</td>
            <td>7000</td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result −

Colspan and Rowspan Attributes

You will use **colspan** attribute if you want to merge two or more columns into a single column. Similar way you will use **rowspan** if you want to merge two or more rows.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table Colspan/Rowspan</title>
   </head>

   <body>
      <table border = "1">
         <tr>
            <th>Column 1</th>
            <th>Column 2</th>
            <th>Column 3</th>
         </tr>
         <tr>
            <td rowspan = "2">Row 1 Cell 1</td>
            <td>Row 1 Cell 2</td>
            <td>Row 1 Cell 3</td>
         </tr>
         <tr>
            <td>Row 2 Cell 2</td>
            <td>Row 2 Cell 3</td>
         </tr>
         <tr>
            <td colspan = "3">Row 3 Cell 1</td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result −

Tables Backgrounds

You can set table background using one of the following two ways −

- **bgcolor** attribute − You can set background color for whole table or just for one cell.
- **background** attribute − You can set background image for whole table or just for one cell.

You can also set border color also using **bordercolor** attribute.

Note − The *bgcolor*, *background*, and *bordercolor* attributes deprecated in HTML5. Do not use these attributes.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table Background</title>
   </head>

   <body>
      <table border = "1" bordercolor = "green" bgcolor =
"yellow">
         <tr>
            <th>Column 1</th>
            <th>Column 2</th>
            <th>Column 3</th>
         </tr>
         <tr>
            <td rowspan = "2">Row 1 Cell 1</td>
            <td>Row 1 Cell 2</td>
            <td>Row 1 Cell 3</td>
         </tr>
         <tr>
            <td>Row 2 Cell 2</td>
            <td>Row 2 Cell 3</td>
         </tr>
         <tr>
            <td colspan = "3">Row 3 Cell 1</td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result −

Here is an example of using **background** attribute. Here we will use an image available in /images directory.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table Background</title>
   </head>

   <body>
      <table border = "1" bordercolor = "green" background =
"/images/test.png">
         <tr>
            <th>Column 1</th>
            <th>Column 2</th>
            <th>Column 3</th>
         </tr>
         <tr>
            <td rowspan = "2">Row 1 Cell 1</td>
            <td>Row 1 Cell 2</td><td>Row 1 Cell 3</td>
         </tr>
         <tr>
            <td>Row 2 Cell 2</td>
            <td>Row 2 Cell 3</td>
         </tr>
         <tr>
            <td colspan = "3">Row 3 Cell 1</td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result. Here background image did not apply to table's header.

Table Height and Width

You can set a table width and height using **width** and **height** attributes. You can specify table width or height in terms of pixels or in terms of percentage of available screen area.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table Width/Height</title>
   </head>

   <body>
      <table border = "1" width = "400" height = "150">
         <tr>
            <td>Row 1, Column 1</td>
            <td>Row 1, Column 2</td>
         </tr>

         <tr>
            <td>Row 2, Column 1</td>
            <td>Row 2, Column 2</td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result −

Table Caption

The **caption** tag will serve as a title or explanation for the table and it shows up at the top of the table. This tag is deprecated in newer version of HTML/XHTML.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table Caption</title>
   </head>

   <body>
      <table border = "1" width = "100%">
         <caption>This is the caption</caption>

         <tr>
            <td>row 1, column 1</td><td>row 1, columnn 2</td>
         </tr>

         <tr>
            <td>row 2, column 1</td><td>row 2, columnn 2</td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result −

Table Header, Body, and Footer

Tables can be divided into three portions − a header, a body, and a foot. The head and foot are rather similar to headers and footers in a word-processed document that remain the same for every page, while the body is the main content holder of the table.

The three elements for separating the head, body, and foot of a table are −

- **<thead>** − to create a separate table header.
- **<tbody>** − to indicate the main body of the table.
- **<tfoot>** − to create a separate table footer.

A table may contain several <tbody> elements to indicate *different pages* or groups of data. But it is notable that <thead> and <tfoot> tags should appear before <tbody>

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table</title>
   </head>

   <body>
      <table border = "1" width = "100%">
         <thead>
            <tr>
               <td colspan = "4">This is the head of the
table</td>
            </tr>
         </thead>

         <tfoot>
            <tr>
               <td colspan = "4">This is the foot of the
table</td>
            </tr>
         </tfoot>
         <tbody>
            <tr>
               <td>Cell 1</td>
               <td>Cell 2</td>
               <td>Cell 3</td>
               <td>Cell 4</td>
            </tr>
         </tbody>
         
      </table>
   </body>
   
</html>
```

This will produce the following result −

Nested Tables

You can use one table inside another table. Not only tables you can use almost all the tags inside table data tag <td>.

Example

Following is the example of using another table and other tags inside a table cell.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Table</title>
   </head>
	
   <body>
      <table border = "1" width = "100%">
         
         <tr>
            <td>
               <table border = "1" width = "100%">
                  <tr>
                     <th>Name</th>
                     <th>Salary</th>
                  </tr>
                  <tr>
                     <td>Ramesh Raman</td>
                     <td>5000</td>
                  </tr>
                  <tr>
                     <td>Shabbir Hussein</td>
                     <td>7000</td>
                  </tr>
               </table>
            </td>
         </tr>

      </table>
   </body>

</html>
```

This will produce the following result −

HTML - Lists

HTML offers web authors three ways for specifying lists of information. All lists must contain one or more list elements. Lists may contain −

- **<ul>** − An unordered list. This will list items using plain bullets.
- **<ol>** − An ordered list. This will use different schemes of numbers to list your items.
- **<dl>** − A definition list. This arranges your items in the same way as they are arranged in a dictionary.

HTML Unordered Lists

An unordered list is a collection of related items that have no special order or sequence. This list is created by using HTML **<ul>** tag. Each item in the list is marked with a bullet.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Unordered List</title>
   </head>

   <body>
      <ul>
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ul>
   </body>

</html>
```

This will produce the following result −

The type Attribute

You can use **type** attribute for <ul> tag to specify the type of bullet you like. By default, it is a disc. Following are the possible options −

```
<ul type = "square">
<ul type = "disc">
<ul type = "circle">
```

Example

Following is an example where we used <ul type = "square">

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Unordered List</title>
   </head>

   <body>
      <ul type = "square">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ul>
   </body>

</html>
```

This will produce the following result −

Example

Following is an example where we used <ul type = "disc"> −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Unordered List</title>
   </head>

   <body>
      <ul type = "disc">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ul>
   </body>

</html>
```

This will produce the following result −

Example

Following is an example where we used <ul type = "circle"> −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Unordered List</title>
   </head>

   <body>
      <ul type = "circle">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ul>
   </body>

</html>
```

This will produce the following result −

HTML Ordered Lists

If you are required to put your items in a numbered list instead of bulleted, then HTML ordered list will be used. This list is created by using **<ol>** tag. The numbering starts at one and is incremented by one for each successive ordered list element tagged with <li>.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Ordered List</title>
   </head>

   <body>
      <ol>
```

```
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ol>
   </body>

</html>
```

This will produce the following result −

The type Attribute

You can use **type** attribute for <ol> tag to specify the type of numbering you like. By default, it is a number. Following are the possible options −

```
<ol type = "1"> - Default-Case Numerals.
<ol type = "I"> - Upper-Case Numerals.
<ol type = "i"> - Lower-Case Numerals.
<ol type = "A"> - Upper-Case Letters.
<ol type = "a"> - Lower-Case Letters.
```

Example

Following is an example where we used <ol type = "1">

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Ordered List</title>
   </head>

   <body>
      <ol type = "1">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ol>
   </body>

</html>
```

This will produce the following result −

Example

Following is an example where we used <ol type = "I">

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Ordered List</title>
   </head>

   <body>
      <ol type = "I">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ol>
   </body>

</html>
```

This will produce the following result −

Example

Following is an example where we used <ol type = "i">

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Ordered List</title>
   </head>

   <body>
      <ol type = "i">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ol>
   </body>

</html>
```

This will produce the following result −

Example

Following is an example where we used <ol type = "A" >

Live Demo

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

```
   <head>
      <title>HTML Ordered List</title>
   </head>

   <body>
      <ol type = "A">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ol>
   </body>

</html>
```

This will produce the following result −

Example

Following is an example where we used <ol type = "a">

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Ordered List</title>
   </head>

   <body>
      <ol type = "a">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ol>
   </body>

</html>
```

This will produce the following result −

The start Attribute

You can use **start** attribute for <ol> tag to specify the starting point of numbering you need. Following are the possible options −

```
<ol type = "1" start = "4">    - Numerals starts with 4.
<ol type = "I" start = "4">    - Numerals starts with IV.
<ol type = "i" start = "4">    - Numerals starts with iv.
<ol type = "a" start = "4">    - Letters starts with d.
```

```
<ol type = "A" start = "4">     - Letters starts with D.
```

Example

Following is an example where we used <ol type = "i" start = "4" >

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Ordered List</title>
   </head>

   <body>
      <ol type = "i" start = "4">
         <li>Beetroot</li>
         <li>Ginger</li>
         <li>Potato</li>
         <li>Radish</li>
      </ol>
   </body>

</html>
```

This will produce the following result −

HTML Definition Lists

HTML and XHTML supports a list style which is called **definition lists** where entries are listed like in a dictionary or encyclopedia. The definition list is the ideal way to present a glossary, list of terms, or other name/value list.

Definition List makes use of following three tags.

- <dl> − Defines the start of the list
- <dt> − A term
- <dd> − Term definition
- </dl> − Defines the end of the list

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Definition List</title>
   </head>
   <body>
      <dl>
         <dt><b>HTML</b></dt>
         <dd>This stands for Hyper Text Markup Language</dd>
         <dt><b>HTTP</b></dt>
         <dd>This stands for Hyper Text Transfer Protocol</dd>
      </dl>
   </body>

</html>
```

This will produce the following result −

HTML - Text Links

A webpage can contain various links that take you directly to other pages and even specific parts of a given page. These links are known as hyperlinks.

Hyperlinks allow visitors to navigate between Web sites by clicking on words, phrases, and images. Thus you can create hyperlinks using text or images available on a webpage.

Note − I recommend you to go through a short tutorial on Understanding URL

Linking Documents

A link is specified using HTML tag <a>. This tag is called **anchor tag** and anything between the opening <a> tag and the closing </a> tag becomes part of the link and a user can click that part to reach to the linked document. Following is the simple syntax to use <a> tag.

```
<a href = "Document URL" ... attributes-list>Link Text</a>
```

Example

Let's try following example which links http://www.tutorialspoint.com at your page −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Hyperlink Example</title>
   </head>

   <body>
      <p>Click following link</p>
      <a href = "https://www.tutorialspoint.com" target =
"_self">Tutorials Point</a>
   </body>

</html>
```

This will produce the following result, where you can click on the link generated to reach to the home page of Tutorials Point (in this example).

The target Attribute

We have used **target** attribute in our previous example. This attribute is used to specify the location where linked document is opened. Following are the possible options −

Sr.No	Option & Description
1	**_blank** Opens the linked document in a new window or tab.
2	**_self** Opens the linked document in the same frame.
3	**_parent** Opens the linked document in the parent frame.
4	**_top** Opens the linked document in the full body of the window.
5	**targetframe** Opens the linked document in a named *targetframe*.

Example

Try following example to understand basic difference in few options given for target attribute.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Hyperlink Example</title>
      <base href = "https://www.tutorialspoint.com/">
   </head>

   <body>
```

```
      <p>Click any of the following links</p>
      <a href = "/html/index.htm" target = "_blank">Opens in
New</a> |
      <a href = "/html/index.htm" target = "_self">Opens in
Self</a> |
      <a href = "/html/index.htm" target = "_parent">Opens in
Parent</a> |
      <a href = "/html/index.htm" target = "_top">Opens in
Body</a>
   </body>

</html>
```

This will produce the following result, where you can click on different links to understand the difference between various options given for target attribute.

Use of Base Path

When you link HTML documents related to the same website, it is not required to give a complete URL for every link. You can get rid of it if you use **<base>** tag in your HTML document header. This tag is used to give a base path for all the links. So your browser will concatenate given relative path to this base path and will make a complete URL.

Example

Following example makes use of <base> tag to specify base URL and later we can use relative path to all the links instead of giving complete URL for every link.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Hyperlink Example</title>
      <base href = "https://www.tutorialspoint.com/">
   </head>

   <body>
      <p>Click following link</p>
      <a href = "/html/index.htm" target = "_blank">HTML
Tutorial</a>
   </body>

</html>
```

This will produce the following result, where you can click on the link generated **HTML Tutorial** to reach to the HTML tutorial.

Now given URL <a href = "/html/index.htm" is being considered as <ahref = "http://www.tutorialspoint.com/html/index.htm"

Linking to a Page Section

You can create a link to a particular section of a given webpage by using **name** attribute. This is a two-step process.

Note − The *name* attribute deprecated in HTML5. Do not use this attribute. Use *id* and *title* attribute instead.

First create a link to the place where you want to reach with-in a webpage and name it using <a...> tag as follows −

```
<h1>HTML Text Links <a name = "top"></a></h1>
```

Second step is to create a hyperlink to link the document and place where you want to reach −

```
<a href = "/html/html_text_links.htm#top">Go to the Top</a>
```

This will produce following link, where you can click on the link generated **Go to the Top** to reach to the top of the HTML Text Link tutorial.

Go to the Top

Setting Link Colors

You can set colors of your links, active links and visited links using **link**, **alink** and **vlink** attributes of <body> tag.

Example

Save the following in test.htm and open it in any web browser to see how **link**, **alink** and **vlink** attributes work.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Hyperlink Example</title>
      <base href = "https://www.tutorialspoint.com/">
   </head>

   <body alink = "#54A250" link = "#040404" vlink = "#F40633">
      <p>Click following link</p>
      <a href = "/html/index.htm" target = "_blank" >HTML
Tutorial</a>
   </body>

</html>
```

This will produce the following result. Just check color of the link before clicking on it, next check its color when you activate it and when the link has been visited.

Download Links

You can create text link to make your PDF, or DOC or ZIP files downloadable. This is very simple; you just need to give complete URL of the downloadable file as follows −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Hyperlink Example</title>
   </head>

   <body>
      <a href =
"https://www.tutorialspoint.com/page.pdf">Download PDF File</a>
   </body>

</html>
```

This will produce following link and will be used to download a file.

File Download Dialog Box

Sometimes it is desired that you want to give an option where a user will click a link and it will pop up a "File Download" box to the user instead of displaying actual content. This is very easy and can be achieved using an HTTP header in your HTTP response.

For example, if you want make a **Filename** file downloadable from a given link then its syntax will be as follows.

```
#!/usr/bin/perl

# Additional HTTP Header
print "Content-Type:application/octet-stream; name =
\"FileName\"\r\n";
print "Content-Disposition:attachment; filename =
\"FileName\"\r\n\n";

# Open the target file and list down its content as follows
open( FILE, "<FileName" );

while(read(FILE, $buffer, 100)){
   print("$buffer");
}
```

Note − For more detail on PERL CGI programs, go through tutorial PERL and CGI.

HTML - Image Links

We have seen how to create hypertext link using text and we also learnt how to use images in our webpages. Now, we will learn how to use images to create hyperlinks.

Example

It's simple to use an image as hyperlink. We just need to use an image inside hyperlink at the place of text as shown below −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Image Hyperlink Example</title>
   </head>

   <body>
      <p>Click following link</p>
      <a href = "https://www.tutorialspoint.com" target =
"_self">
         <img src = "/images/logo.png" alt = "Tutorials Point"
border = "0"/>
      </a>
   </body>

</html>
```

This will produce the following result, where you can click on the images to reach to the home page of Tutorials Point.

This was the simplest way of creating hyperlinks using images. Next we will see how we can create Mouse-Sensitive Image Links.

Mouse-Sensitive Images

The HTML and XHTML standards provides a feature that lets you embed many different links inside a single image. You can create different links on the single image based on different coordinates available on the image. Once different links are attached to different coordinates, we can click different parts of the image to open target documents. Such mouse-sensitive images are known as image maps.

There are two ways to create image maps −

- **Server-side image maps** − This is enabled by the **ismap** attribute of the <img> tag and requires access to a server and related image-map processing applications.
- **Client-side image maps** − This is created with the **usemap** attribute of the <img> tag, along with corresponding <map> and <area> tags.

Server-Side Image Maps

Here you simply put your image inside a hyper link and use **ismap** attribute which makes it special image and when the user clicks some place within the image, the browser passes the coordinates of the mouse pointer along with the URL specified in the <a> tag to the web server. The server uses the mouse-pointer coordinates to determine which document to deliver back to the browser.

When *ismap* is used, the href attribute of the containing <a> tag must contain the URL of a server application like a cgi or PHP script etc. to process the incoming request based on the passed coordinates.

The coordinates of the mouse position are screen pixels counted from the upper-left corner of the image, beginning with (0,0). The coordinates, preceded by a question mark, are added to the end of the URL.

For example, if a user clicks 20 pixels over and 30 pixels down from the upper-left corner of the following image −

Which has been generated by the following code snippet −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>ISMAP Hyperlink Example</title>
   </head>

   <body>
      <p>Click following link</p>

      <a href = "/cgi-bin/ismap.cgi" target = "_self">
         <img ismap src = "/images/logo.png" alt = "Tutorials 
Point" border = "0"/>
      </a>
   </body>

</html>
```

Then the browser sends the following search parameters to the web server which can be processed by **ismap.cgi** script or **map file** and you can link whatever documents you like to these coordinates −

```
/cgi-bin/ismap.cgi?20,30
```

This way you can assign different links to different coordinates of the image and when those coordinates are clicked, you can open corresponding linked document. To learn more about **ismap** attribute, you can check How to use Image ismap?

Note − You will learn CGI programming when you will study Perl programming. You can write your script to process these passed coordinates using PHP or any other script as well. For now, let's concentrate on learning HTML and later you can revisit this section.

Client-Side Image Maps

Client side image maps are enabled by the **usemap** attribute of the <img /> tag and defined by special <map> and <area> extension tags.

The image that is going to form the map is inserted into the page using the <img /> tag as a normal image, except it carries an extra attribute called **usemap**. The value of the usemap attribute is the value which will be used in a <map> tag to link map and image tags. The <map> along with <area> tags define all the image coordinates and corresponding links.

The <area> tag inside the map tag, specifies the shape and the coordinates to define the boundaries of each clickable hotspot available on the image. Here's an example from the image map −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>USEMAP Hyperlink Example</title>
   </head>

   <body>
      <p>Search and click the hotspot</p>
      <img src = /images/html.gif alt = "HTML Map" border = "0"
usemap = "#html"/>
      <!-- Create  Mappings -->

      <map name = "html">
         <area shape = "circle" coords = "80,80,20"
            href = "/css/index.htm" alt = "CSS Link" target =
"_self" />

         <area shape = "rect" coords = "5,5,40,40" alt = "jQuery
Link"
            href = "/jquery/index.htm" target = "_self" />
      </map>
   </body>

</html>
```

This will produce the following result −

Coordinate System

The actual value of coords is totally dependent on the shape in question. Here is a summary, to be followed by detailed examples −

- **rect = x_1 , y_1 , x_2 , y_2**

  x_1 and y_1 are the coordinates of the upper left corner of the rectangle; x_2 and y_2 are the coordinates of the lower right corner.

- **circle = x_c , y_c , radius**

x_c and y_c are the coordinates of the center of the circle, and radius is the circle's radius. A circle centered at 200,50 with a radius of 25 would have the attribute *coords = "200,50,25"*

- **poly = x_1 , y_1 , x_2 , y_2 , x_3 , y_3 , ... x_n , y_n**

  The various x-y pairs define vertices (points) of the polygon, with a "line" being drawn from one point to the next point. A diamond-shaped polygon with its top point at 20,20 and 40 pixels across at its widest points would have the attribute *coords = "20,20,40,40,20,60,0,40"*.

All coordinates are relative to the upper-left corner of the image (0,0). Each shape has a related URL. You can use any image software to know the coordinates of different positions.

HTML - Email Links

It is not difficult to put an HTML email link on your webpage but it can cause unnecessary spamming problem for your email account. There are people, who can run programs to harvest these types of emails and later use them for spamming in various ways.

You can have another option to facilitate people to send you emails. One option could be to use HTML forms to collect user data and then use PHP or CGI script to send an email.

A simple example, check our Contact Us Form. We take user feedback using this form and then we are using one CGI program which is collecting this information and sending us email to the one given email ID.

Note − You will learn about HTML Forms in HTML Forms and you will learn about CGI in our another tutorial **Perl CGI Programming**.

HTML Email Tag

HTML **<a>** tag provides you option to specify an email address to send an email. While using <a> tag as an email tag, you will use **mailto: email address** along with *href* attribute. Following is the syntax of using **mailto** instead of using http.

```
<a href = "mailto: abc@example.com">Send Email</a>
```

This code will generate the following link which you can use to send email.

Send Email

Now, if a user clicks this link, it launches one Email Client (like Lotus Notes, Outlook Express etc.) installed on your user's computer. There is another risk to use this option to send email because if user do not have email client installed on their computer then it would not be possible to send email.

Default Settings

You can specify a default *email subject* and *email body* along with your email address. Following is the example to use default subject and body.

```
<a href = "mailto:abc@example.com?subject = Feedback&body = 
Message">
Send Feedback
</a>
```

This code will generate the following link which you can use to send email.

Send Feedback

HTML - Frames

HTML frames are used to divide your browser window into multiple sections where each section can load a separate HTML document. A collection of frames in the browser window is known as a frameset. The window is divided into frames in a similar way the tables are organized: into rows and columns.

Disadvantages of Frames

There are few drawbacks with using frames, so it's never recommended to use frames in your webpages −

- Some smaller devices cannot cope with frames often because their screen is not big enough to be divided up.
- Sometimes your page will be displayed differently on different computers due to different screen resolution.
- The browser's *back* button might not work as the user hopes.
- There are still few browsers that do not support frame technology.

Creating Frames

To use frames on a page we use <frameset> tag instead of <body> tag. The <frameset> tag defines, how to divide the window into frames. The **rows** attribute of <frameset> tag defines horizontal frames and **cols** attribute defines vertical frames. Each frame is indicated by <frame> tag and it defines which HTML document shall open into the frame.

Note − The <frame> tag deprecated in HTML5. Do not use this element.

Example

Following is the example to create three horizontal frames −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Frames</title>
   </head>

   <frameset rows = "10%,80%,10%">
```

```
      <frame name = "top" src = "/html/top_frame.htm" />
      <frame name = "main" src = "/html/main_frame.htm" />
      <frame name = "bottom" src = "/html/bottom_frame.htm" />

      <noframes>
         <body>Your browser does not support frames.</body>
      </noframes>

   </frameset>

</html>
```

This will produce the following result −

Example

Let's put the above example as follows, here we replaced rows attribute by cols and changed their width. This will create all the three frames vertically −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Frames</title>
   </head>

   <frameset cols = "25%,50%,25%">
      <frame name = "left" src = "/html/top_frame.htm" />
      <frame name = "center" src = "/html/main_frame.htm" />
      <frame name = "right" src = "/html/bottom_frame.htm" />

      <noframes>
         <body>Your browser does not support frames.</body>
      </noframes>
   </frameset>

</html>
```

This will produce the following result −

The <frameset> Tag Attributes

Following are important attributes of the <frameset> tag −

Sr.No	Attribute & Description
1	**cols** Specifies how many columns are contained in the frameset and the size of each column.

	You can specify the width of each column in one of the four ways − Absolute values in pixels. For example, to create three vertical frames, use *cols = "100, 500, 100"*. A percentage of the browser window. For example, to create three vertical frames, use *cols = "10%, 80%, 10%"*. Using a wildcard symbol. For example, to create three vertical frames, use *cols = "10%, *, 10%"*. In this case wildcard takes remainder of the window. As relative widths of the browser window. For example, to create three vertical frames, use *cols = "3*, 2*, 1*"*. This is an alternative to percentages. You can use relative widths of the browser window. Here the window is divided into sixths: the first column takes up half of the window, the second takes one third, and the third takes one sixth.
2	**rows** This attribute works just like the cols attribute and takes the same values, but it is used to specify the rows in the frameset. For example, to create two horizontal frames, use *rows = "10%, 90%"*. You can specify the height of each row in the same way as explained above for columns.
3	**border** This attribute specifies the width of the border of each frame in pixels. For example, border = "5". A value of zero means no border.
4	**frameborder** This attribute specifies whether a three-dimensional border should be displayed between frames. This attribute takes value either 1 (yes) or 0 (no). For example frameborder = "0" specifies no border.
5	**framespacing** This attribute specifies the amount of space between frames in a frameset. This can take any integer value. For example framespacing = "10" means there should be 10 pixels spacing between each frames.

The <frame> Tag Attributes

Following are the important attributes of <frame> tag −

Sr.No	Attribute & Description

1	**src** This attribute is used to give the file name that should be loaded in the frame. Its value can be any URL. For example, src = "/html/top_frame.htm" will load an HTML file available in html directory.
2	**name** This attribute allows you to give a name to a frame. It is used to indicate which frame a document should be loaded into. This is especially important when you want to create links in one frame that load pages into an another frame, in which case the second frame needs a name to identify itself as the target of the link.
3	**frameborder** This attribute specifies whether or not the borders of that frame are shown; it overrides the value given in the frameborder attribute on the <frameset> tag if one is given, and this can take values either 1 (yes) or 0 (no).
4	**marginwidth** This attribute allows you to specify the width of the space between the left and right of the frame's borders and the frame's content. The value is given in pixels. For example marginwidth = "10".
5	**marginheight** This attribute allows you to specify the height of the space between the top and bottom of the frame's borders and its contents. The value is given in pixels. For example marginheight = "10".
6	**noresize** By default, you can resize any frame by clicking and dragging on the borders of a frame. The noresize attribute prevents a user from being able to resize the frame. For example noresize = "noresize".
7	**scrolling** This attribute controls the appearance of the scrollbars that appear on the frame. This takes values either "yes", "no" or "auto". For example scrolling = "no" means it should not have scroll bars.
8	**longdesc** This attribute allows you to provide a link to another page containing a long description of

	the contents of the frame. For example longdesc = "framedescription.htm"

Browser Support for Frames

If a user is using any old browser or any browser, which does not support frames then <noframes> element should be displayed to the user.

So you must place a <body> element inside the <noframes> element because the <frameset> element is supposed to replace the <body> element, but if a browser does not understand <frameset> element then it should understand what is inside the <body> element which is contained in a <noframes> element.

You can put some nice message for your user having old browsers. For example, *Sorry!! your browser does not support frames.* as shown in the above example.

Frame's name and target attributes

One of the most popular uses of frames is to place navigation bars in one frame and then load main pages into a separate frame.

Let's see following example where a test.htm file has following code −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Target Frames</title>
   </head>

   <frameset cols = "200, *">
      <frame src = "/html/menu.htm" name = "menu_page" />
      <frame src = "/html/main.htm" name = "main_page" />

      <noframes>
         <body>Your browser does not support frames.</body>
      </noframes>
   </frameset>

</html>
```

Here, we have created two columns to fill with two frames. The first frame is 200 pixels wide and will contain the navigation menu bar implemented by **menu.htm** file. The second column fills in remaining space and will contain the main part of the page and it is implemented by **main.htm** file. For all the three links available in menu bar, we have mentioned target frame as **main_page**, so whenever you click any of the links in menu bar, available link will open in main page.

Following is the content of menu.htm file

Live Demo

```
<!DOCTYPE html>
<html>

   <body bgcolor = "#4a7d49">
      <a href = "http://www.google.com" target =
"main_page">Google</a>
      <br />
      <br />

      <a href = "http://www.microsoft.com" target =
"main_page">Microsoft</a>
      <br />
      <br />

      <a href = "http://news.bbc.co.uk" target = "main_page">BBC
News</a>
   </body>

</html>
```

Following is the content of main.htm file −

Live Demo

```
<!DOCTYPE html>
<html>

   <body bgcolor = "#b5dcb3">
      <h3>This is main page and content from any link will be
displayed here.</h3>
      <p>So now click any link and see the result.</p>
   </body>

</html>
```

When we load **test.htm** file, it produces following result −

Now you can try to click links available in the left panel and see the result. The *targetattribute* can also take one of the following values −

Sr.No	Option & Description
1	**_self** Loads the page into the current frame.
2	**_blank** Loads a page into a new browser window. Opening a new window.

3	**_parent** Loads the page into the parent window, which in the case of a single frameset is the main browser window.
4	**_top** Loads the page into the browser window, replacing any current frames.
5	**targetframe** Loads the page into a named targetframe.

HTML - Iframes

You can define an inline frame with HTML tag **<iframe>**. The <iframe> tag is not somehow related to <frameset> tag, instead, it can appear anywhere in your document. The <iframe> tag defines a rectangular region within the document in which the browser can display a separate document, including scrollbars and borders. An inline frame is used to embed another document within the current HTML document.

The **src** attribute is used to specify the URL of the document that occupies the inline frame.

Example

Following is the example to show how to use the <iframe> −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Iframes</title>
   </head>

   <body>
      <p>Document content goes here...</p>

      <iframe src = "/html/menu.htm" width = "555" height =
"200">
         Sorry your browser does not support inline frames.
      </iframe>

      <p>Document content also go here...</p>
   </body>

</html>
```

This will produce the following result −

The <Iframe> Tag Attributes

Most of the attributes of the <iframe> tag, including *name, class, frameborder, id, longdesc, marginheight, marginwidth, name, scrolling, style,* and *title* behave exactly like the corresponding attributes for the <frame> tag.

Note − The *frameborder*, *marginwidth*, *longdesc*, *scrolling*, *marginheight* attributes deprecated in HTML5. Do not use these attributes.

Sr.No	Attribute & Description
1	**src** This attribute is used to give the file name that should be loaded in the frame. Its value can be any URL. For example, src = "/html/top_frame.htm" will load an HTML file available in html directory.
2	**name** This attribute allows you to give a name to a frame. It is used to indicate which frame a document should be loaded into. This is especially important when you want to create links in one frame that load pages into an another frame, in which case the second frame needs a name to identify itself as the target of the link.
3	**frameborder** This attribute specifies whether or not the borders of that frame are shown; it overrides the value given in the frameborder attribute on the <frameset> tag if one is given, and this can take values either 1 (yes) or 0 (no).
4	**marginwidth** This attribute allows you to specify the width of the space between the left and right of the frame's borders and the frame's content. The value is given in pixels. For example marginwidth = "10".
5	**marginheight** This attribute allows you to specify the height of the space between the top and bottom of the frame's borders and its contents. The value is given in pixels. For example marginheight = "10".
6	**height**

	This attribute specifies the height of <iframe>.
7	**scrolling** This attribute controls the appearance of the scrollbars that appear on the frame. This takes values either "yes", "no" or "auto". For example scrolling = "no" means it should not have scroll bars.
8	**longdesc** This attribute allows you to provide a link to another page containing a long description of the contents of the frame. For example longdesc = "framedescription.htm"
9	**width** This attribute specifies the width of <iframe>.

HTML - Blocks

All the HTML elements can be categorized into two categories **(a)** Block Level Elements **(b)**Inline Elements.

Block Elements

Block elements appear on the screen as if they have a line break before and after them. For example, the <p>, <h1>, <h2>, <h3>, <h4>, <h5>, <h6>, <ul>, <ol>, <dl>, <pre>, <hr />, <blockquote>, and <address> elements are all block level elements. They all start on their own new line, and anything that follows them appears on its own new line.

Inline Elements

Inline elements, on the other hand, can appear within sentences and do not have to appear on a new line of their own. The <b>, <i>, <u>, <em>, <strong>, <sup>, <sub>, <big>, <small>, <li>, <ins>, <del>, <code>, <cite>, <dfn>, <kbd>, and <var> elements are all inline elements.

Grouping HTML Elements

There are two important tags which we use very frequently to group various other HTML tags (i) <div> tag and (ii) <span> tag

The <div> tag

This is the very important block level tag which plays a big role in grouping various other HTML tags and applying CSS on group of elements. Even now <div> tag can be used to create webpage layout where we define different parts (Left, Right, Top etc.) of the page using <div> tag. This tag does not provide any visual change on the block but this has more meaning when it is used with CSS.

Example

Following is a simple example of <div> tag. We will learn Cascading Style Sheet (CSS) in a separate chapter but we used it here to show the usage of <div> tag −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML div Tag</title>
   </head>

   <body>
      <!-- First group of tags -->
      <div style = "color:red">
         <h4>This is first group</h4>
         <p>Following is a list of vegetables</p>

         <ul>
            <li>Beetroot</li>
            <li>Ginger</li>
            <li>Potato</li>
            <li>Radish</li>
         </ul>
      </div>

      <!-- Second group of tags -->
      <div style = "color:green">
         <h4>This is second group</h4>
         <p>Following is a list of fruits</p>

         <ul>
            <li>Apple</li>
            <li>Banana</li>
            <li>Mango</li>
            <li>Strawberry</li>
         </ul>
      </div>
   </body>

</html>
```

This will produce the following result −

The <span> tag

The HTML <span> is an inline element and it can be used to group inline-elements in an HTML document. This tag also does not provide any visual change on the block but has more meaning when it is used with CSS.

The difference between the <span> tag and the <div> tag is that the <span> tag is used with inline elements whereas the <div> tag is used with block-level elements.

Example

Following is a simple example of <span> tag. We will learn Cascading Style Sheet (CSS) in a separate chapter but we used it here to show the usage of <span> tag −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML span Tag</title>
   </head>

   <body>
      <p>This is <span style = "color:red">red</span> and this is
         <span style = "color:green">green</span></p>
   </body>

</html>
```

This will produce the following result −

HTML - Backgrounds

By default, your webpage background is white in color. You may not like it, but no worries. HTML provides you following two good ways to decorate your webpage background.

- HTML Background with Colors
- HTML Background with Images

Now let's see both the approaches one by one using appropriate examples.

Html Background with Colors

The **bgcolor** attribute is used to control the background of an HTML element, specifically page body and table backgrounds.

Note − The *bgcolor* attribute deprecated in HTML5. Do not use this attribute.

Following is the syntax to use bgcolor attribute with any HTML tag.

```
<tagname bgcolor = "color_value"...>
```

This color_value can be given in any of the following formats −

```
<!-- Format 1 - Use color name -->
```

```
<table bgcolor = "lime" >

<!-- Format 2 - Use hex value -->
<table bgcolor = "#f1f1f1" >

<!-- Format 3 - Use color value in RGB terms -->
<table bgcolor = "rgb(0,0,120)" >
```

Example

Here are the examples to set background of an HTML tag −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Background Colors</title>
   </head>

   <body>
      <!-- Format 1 - Use color name -->
      <table bgcolor = "yellow" width = "100%">
         <tr>
            <td>
               This background is yellow
            </td>
         </tr>
      </table>

      <!-- Format 2 - Use hex value -->
      <table bgcolor = "#6666FF" width = "100%">
         <tr>
            <td>
               This background is sky blue
            </td>
         </tr>
      </table>

      <!-- Format 3 - Use color value in RGB terms -->
      <table bgcolor = "rgb(255,0,255)" width = "100%">
         <tr>
            <td>
               This background is green
            </td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result −

Html Background with Images

The **background** attribute can also be used to control the background of an HTML element, specifically page body and table backgrounds. You can specify an image to set background of your HTML page or table.

Note − The *background* attribute deprecated in HTML5. Do not use this attribute.

Following is the syntax to use background attribute with any HTML tag.

Note − The *background* attribute is deprecated and it is recommended to use Style Sheet for background setting.

```
<tagname background = "Image URL"...>
```

The most frequently used image formats are JPEG, GIF and PNG images.

Example

Here are the examples to set background images of a table.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Background Images</title>
   </head>

   <body>
      <!-- Set table background -->
      <table background = "/images/html.gif" width = "100%"
height = "100">
         <tr><td>
            This background is filled up with HTML image.
         </td></tr>
      </table>
   </body>

</html>
```

This will produce the following result −

Patterned & Transparent Backgrounds

You might have seen many pattern or transparent backgrounds on various websites. This simply can be achieved by using patterned image or transparent image in the background.

It is suggested that while creating patterns or transparent GIF or PNG images, use the smallest dimensions possible even as small as 1x1 to avoid slow loading.

Example

Here are the examples to set background pattern of a table −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Background Images</title>
   </head>

   <body>
      <!-- Set a table background using pattern -->
      <table background = "/images/pattern1.gif" width = "100%"
height = "100">
         <tr>
            <td>
               This background is filled up with a pattern image.
            </td>
         </tr>
      </table>

      <!-- Another example on table background using pattern -->
      <table background = "/images/pattern2.gif" width = "100%"
height = "100">
         <tr>
            <td>
               This background is filled up with a pattern image.
            </td>
         </tr>
      </table>
   </body>

</html>
```

This will produce the following result −

HTML - Colors

Colors are very important to give a good look and feel to your website. You can specify colors on page level using <body> tag or you can set colors for individual tags using **bgcolor** attribute.

The <body> tag has following attributes which can be used to set different colors −

- **bgcolor** − sets a color for the background of the page.
- **text** − sets a color for the body text.
- **alink** − sets a color for active links or selected links.
- **link** − sets a color for linked text.
- **vlink** − sets a color for *visited links* − that is, for linked text that you have already clicked on.

HTML Color Coding Methods

There are following three different methods to set colors in your web page −

- **Color names** − You can specify color names directly like green, blue or red.
- **Hex codes** − A six-digit code representing the amount of red, green, and blue that makes up the color.
- **Color decimal or percentage values** − This value is specified using the rgb( ) property.

Now we will see these coloring schemes one by one.

HTML Colors - Color Names

You can specify direct a color name to set text or background color. W3C has listed 16 basic color names that will validate with an HTML validator but there are over 200 different color names supported by major browsers.

Note − Check a complete list of HTML Color Name.

W3C Standard 16 Colors

Here is the list of W3C Standard 16 Colors names and it is recommended to use them.

⬛	Black	◼	Gray	◻	Silver	⬜	White
🟨	Yellow	🟩	Lime	🟦	Aqua	🟪	Fuchsia
🟥	Red	🟩	Green	🟦	Blue	🟪	Purple
🟫	Maroon	🟫	Olive	🟦	Navy	🟦	Teal

Example

Here are the examples to set background of an HTML tag by color name −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Colors by Name</title>
   </head>
   <body text = "blue" bgcolor = "green">
      <p>Use different color names for for body and table and see
the result.</p>

      <table bgcolor = "black">
         <tr>
            <td>
               <font color = "white">This text will appear white
on black background.</font>
            </td>
         </tr>
      </table>
   </body>

</html>
```

HTML Colors - Hex Codes

A hexadecimal is a 6 digit representation of a color. The first two digits(RR) represent a red value, the next two are a green value(GG), and the last are the blue value(BB).

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Paintshop Pro or MS Paint.

Each hexadecimal code will be preceded by a pound or hash sign #. Following is a list of few colors using hexadecimal notation.

Color	Color HEX
⬛	#000000
🟥	#FF0000
🟩	#00FF00
🟦	#0000FF
🟨	#FFFF00
■	#00FFFF

🟪	#FF00FF
▨	#C0C0C0
⬜	#FFFFFF

Example

Here are the examples to set background of an HTML tag by color code in hexadecimal −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Colors by Hex</title>
   </head>

   <body text = "#0000FF" bgcolor = "#00FF00">
      <p>Use different color hexa for for body and table and see
the result.</p>

      <table bgcolor = "#000000">
         <tr>
            <td>
               <font color = "#FFFFFF">This text will appear
white on black background.</font>
            </td>
         </tr>
      </table>
   </body>

</html>
```

HTML Colors - RGB Values

This color value is specified using the **rgb( )** property. This property takes three values, one each for red, green, and blue. The value can be an integer between 0 and 255 or a percentage.

Note − All the browsers does not support rgb() property of color so it is recommended not to use it.

Following is a list to show few colors using RGB values.

Color	Color RGB
⬛	rgb(0,0,0)
🟥	rgb(255,0,0)
🟩	rgb(0,255,0)
🟦	rgb(0,0,255)
🟨	rgb(255,255,0)
🟦	rgb(0,255,255)
🟪	rgb(255,0,255)
▨	rgb(192,192,192)
⬜	rgb(255,255,255)

Example

Here are the examples to set background of an HTML tag by color code using rgb() values −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Colors by RGB code</title>
   </head>

   <body text = "rgb(0,0,255)" bgcolor = "rgb(0,255,0)">
      <p>Use different color code for for body and table and see
the result.</p>

      <table bgcolor = "rgb(0,0,0)">
         <tr>
```

```
            <td>
               <font color = "rgb(255,255,255)">This text will
appear white on black background.</font>
            </td>
         </tr>
      </table>
   </body>

</html>
```

Browser Safe Colors

Here is the list of 216 colors which are supposed to be safest and computer independent colors. These colors very from hexa code 000000 to FFFFFF and they will be supported by all the computers having 256 color palette.

000000	000033	000066	000099	0000CC	0000FF
003300	003333	003366	003399	0033CC	0033FF
006600	006633	006666	006699	0066CC	0066FF
009900	009933	009966	009999	0099CC	0099FF
00CC00	00CC33	00CC66	00CC99	00CCCC	00CCFF
00FF00	00FF33	00FF66	00FF99	00FFCC	00FFFF
330000	330033	330066	330099	3300CC	3300FF
333300	333333	333366	333399	3333CC	3333FF
336600	336633	336666	336699	3366CC	3366FF
339900	339933	339966	339999	3399CC	3399FF
33CC00	33CC33	33CC66	33CC99	33CCCC	33CCFF

33FF00	33FF33	33FF66	33FF99	33FFCC	33FFFF
660000	660033	660066	660099	6600CC	6600FF
663300	663333	663366	663399	6633CC	6633FF
666600	666633	666666	666699	6666CC	6666FF
669900	669933	669966	669999	6699CC	6699FF
66CC00	66CC33	66CC66	66CC99	66CCCC	66CCFF
66FF00	66FF33	66FF66	66FF99	66FFCC	66FFFF
990000	990033	990066	990099	9900CC	9900FF
993300	993333	993366	993399	9933CC	9933FF
996600	996633	996666	996699	9966CC	9966FF
999900	999933	999966	999999	9999CC	9999FF
99CC00	99CC33	99CC66	99CC99	99CCCC	99CCFF
99FF00	99FF33	99FF66	99FF99	99FFCC	99FFFF
CC0000	CC0033	CC0066	CC0099	CC00CC	CC00FF
CC3300	CC3333	CC3366	CC3399	CC33CC	CC33FF
CC6600	CC6633	CC6666	CC6699	CC66CC	CC66FF

CC9900	CC9933	CC9966	CC9999	CC99CC	CC99FF
CCCC00	CCCC33	CCCC66	CCCC99	CCCCCC	CCCCFF
CCFF00	CCFF33	CCFF66	CCFF99	CCFFCC	CCFFFF
FF0000	FF0033	FF0066	FF0099	FF00CC	FF00FF
FF3300	FF3333	FF3366	FF3399	FF33CC	FF33FF
FF6600	FF6633	FF6666	FF6699	FF66CC	FF66FF
FF9900	FF9933	FF9966	FF9999	FF99CC	FF99FF
FFCC00	FFCC33	FFCC66	FFCC99	FFCCCC	FFCCFF
FFFF00	FFFF33	FFFF66	FFFF99	FFFFCC	FFFFFF

HTML - Fonts

Fonts play a very important role in making a website more user friendly and increasing content readability. Font face and color depends entirely on the computer and browser that is being used to view your page but you can use HTML **<font>** tag to add style, size, and color to the text on your website. You can use a **<basefont>** tag to set all of your text to the same size, face, and color.

The font tag is having three attributes called **size, color**, and **face** to customize your fonts. To change any of the font attributes at any time within your webpage, simply use the <font> tag. The text that follows will remain changed until you close with the </font> tag. You can change one or all of the font attributes within one <font> tag.

Note −The *font* and *basefont* tags are deprecated and it is supposed to be removed in a future version of HTML. So they should not be used rather, it's suggested to use CSS styles to manipulate your fonts. But still for learning purpose, this chapter will explain font and basefont tags in detail.

Set Font Size

You can set content font size using **size** attribute. The range of accepted values is from 1(smallest) to 7(largest). The default size of a font is 3.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Setting Font Size</title>
   </head>

   <body>
      <font size = "1">Font size = "1"</font><br />
      <font size = "2">Font size = "2"</font><br />
      <font size = "3">Font size = "3"</font><br />
      <font size = "4">Font size = "4"</font><br />
      <font size = "5">Font size = "5"</font><br />
      <font size = "6">Font size = "6"</font><br />
      <font size = "7">Font size = "7"</font>
   </body>

</html>
```

This will produce the following result −

Relative Font Size

You can specify how many sizes larger or how many sizes smaller than the preset font size should be. You can specify it like **<font size = "+n">** or **<font size = "−n">**

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Relative Font Size</title>
   </head>

   <body>
      <font size = "-1">Font size = "-1"</font><br />
      <font size = "+1">Font size = "+1"</font><br />
      <font size = "+2">Font size = "+2"</font><br />
      <font size = "+3">Font size = "+3"</font><br />
      <font size = "+4">Font size = "+4"</font>
   </body>

</html>
```

This will produce the following result −

Setting Font Face

You can set font face using *face* attribute but be aware that if the user viewing the page doesn't have the font installed, they will not be able to see it. Instead user will see the default font face applicable to the user's computer.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Font Face</title>
   </head>

   <body>
      <font face = "Times New Roman" size = "5">Times New
Roman</font><br />
      <font face = "Verdana" size = "5">Verdana</font><br />
      <font face = "Comic sans MS" size =" 5">Comic Sans
MS</font><br />
      <font face = "WildWest" size = "5">WildWest</font><br />
      <font face = "Bedrock" size = "5">Bedrock</font><br />
   </body>

</html>
```

This will produce the following result −

Specify alternate font faces

A visitor will only be able to see your font if they have that font installed on their computer. So, it is possible to specify two or more font face alternatives by listing the font face names, separated by a comma.

```
<font face = "arial,helvetica">
<font face = "Lucida Calligraphy,Comic Sans MS,Lucida Console">
```

When your page is loaded, their browser will display the first font face available. If none of the given fonts are installed, then it will display the default font face *Times New Roman*.

Note − Check a complete list of **HTML Standard Fonts**.

Setting Font Color

You can set any font color you like using *color* attribute. You can specify the color that you want by either the color name or hexadecimal code for that color.

Note − You can check a complete list of **HTML Color Name with Codes**.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Setting Font Color</title>
   </head>

   <body>
      <font color = "#FF00FF">This text is in pink</font><br />
      <font color = "red">This text is red</font>
   </body>

</html>
```

This will produce the following result −

The <basefont> Element

The <basefont> element is supposed to set a default font size, color, and typeface for any parts of the document that are not otherwise contained within a <font> tag. You can use the <font> elements to override the <basefont> settings.

The <basefont> tag also takes color, size and face attributes and it will support relative font setting by giving size a value of +1 for a size larger or −2 for two sizes smaller.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Setting Basefont Color</title>
   </head>

   <body>
      <basefont face = "arial, verdana, sans-serif" size = "2"
color = "#ff0000">
      <p>This is the page's default font.</p>
      <h2>Example of &lt;basefont&gt; Element</h2>

      <p><font size = "+2" color = "darkgray">
            This is darkgray text with two sizes larger
         </font>
      </p>

      <p><font face = "courier" size = "-1" color = "#000000">
```

```
            It is a courier font, a size smaller and black in
color.
         </font>
      </p>
   </body>

</html>
```

This will produce the following result −

HTML - Forms

HTML Forms are required, when you want to collect some data from the site visitor. For example, during user registration you would like to collect information such as name, email address, credit card, etc.

A form will take input from the site visitor and then will post it to a back-end application such as CGI, ASP Script or PHP script etc. The back-end application will perform required processing on the passed data based on defined business logic inside the application.

There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.

The HTML **<form>** tag is used to create an HTML form and it has following syntax −

```
<form action = "Script URL" method = "GET|POST">
   form elements like input, textarea etc.
</form>
```

Form Attributes

Apart from common attributes, following is a list of the most frequently used form attributes −

Sr.No	Attribute & Description
1	**action** Backend script ready to process your passed data.
2	**method** Method to be used to upload data. The most frequently used are GET and POST methods.
3	**target** Specify the target window or frame where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.

4	**enctype** You can use the enctype attribute to specify how the browser encodes the data before it sends it to the server. Possible values are − **application/x-www-form-urlencoded** − This is the standard method most forms use in simple scenarios. **mutlipart/form-data** − This is used when you want to upload binary data in the form of files like image, word file etc.

Note − You can refer to Perl & CGI for a detail on how form data upload works.

HTML Form Controls

There are different types of form controls that you can use to collect data using HTML form −

- Text Input Controls
- Checkboxes Controls
- Radio Box Controls
- Select Box Controls
- File Select boxes
- Hidden Controls
- Clickable Buttons
- Submit and Reset Button

Text Input Controls

There are three types of text input used on forms −

- **Single-line text input controls** − This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML **<input>** tag.
- **Password input controls** − This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTMl <input> tag.
- **Multi-line text input controls** − This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML **<textarea>** tag.

Single-line text input controls

This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML <input> tag.

Example

Here is a basic example of a single-line text input used to take first name and last name −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Text Input Control</title>
   </head>

   <body>
      <form >
         First name: <input type = "text" name = "first_name" />
         <br>
         Last name: <input type = "text" name = "last_name" />
      </form>
   </body>

</html>
```

This will produce the following result −

Attributes

Following is the list of attributes for <input> tag for creating text field.

Sr.No	Attribute & Description
1	**type** Indicates the type of input control and for text input control it will be set to **text**.
2	**name** Used to give a name to the control which is sent to the server to be recognized and get the value.
3	**value** This can be used to provide an initial value inside the control.
4	**size** Allows to specify the width of the text-input control in terms of characters.

5	**maxlength** Allows to specify the maximum number of characters a user can enter into the text box.

Password input controls

This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML <input>tag but type attribute is set to **password**.

Example

Here is a basic example of a single-line password input used to take user password −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Password Input Control</title>
   </head>

   <body>
      <form >
         User ID : <input type = "text" name = "user_id" />
         <br>
         Password: <input type = "password" name = "password" />
      </form>
   </body>

</html>
```

This will produce the following result −

Attributes

Following is the list of attributes for <input> tag for creating password field.

Sr.No	Attribute & Description
1	**type** Indicates the type of input control and for password input control it will be set to **password**.

2	**name** Used to give a name to the control which is sent to the server to be recognized and get the value.
3	**value** This can be used to provide an initial value inside the control.
4	**size** Allows to specify the width of the text-input control in terms of characters.
5	**maxlength** Allows to specify the maximum number of characters a user can enter into the text box.

Multiple-Line Text Input Controls

This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML <textarea> tag.

Example

Here is a basic example of a multi-line text input used to take item description −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Multiple-Line Input Control</title>
   </head>

   <body>
      <form>
         Description : <br />
         <textarea rows = "5" cols = "50" name = "description">
            Enter description here...
         </textarea>
      </form>
   </body>

</html>
```

This will produce the following result −

Attributes

Following is the list of attributes for <textarea> tag.

Sr.No	Attribute & Description
1	**name** Used to give a name to the control which is sent to the server to be recognized and get the value.
2	**rows** Indicates the number of rows of text area box.
3	**cols** Indicates the number of columns of text area box

Checkbox Control

Checkboxes are used when more than one option is required to be selected. They are also created using HTML <input> tag but type attribute is set to **checkbox.**.

Example

Here is an example HTML code for a form with two checkboxes −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Checkbox Control</title>
   </head>

   <body>
      <form>
         <input type = "checkbox" name = "maths" value = "on">
Maths
         <input type = "checkbox" name = "physics" value = "on">
Physics
      </form>
   </body>

</html>
```

This will produce the following result −

Attributes

Following is the list of attributes for <checkbox> tag.

Sr.No	Attribute & Description
1	**type** Indicates the type of input control and for checkbox input control it will be set to **checkbox.**.
2	**name** Used to give a name to the control which is sent to the server to be recognized and get the value.
3	**value** The value that will be used if the checkbox is selected.
4	**checked** Set to *checked* if you want to select it by default.

Radio Button Control

Radio buttons are used when out of many options, just one option is required to be selected. They are also created using HTML <input> tag but type attribute is set to **radio**.

Example

Here is example HTML code for a form with two radio buttons −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Radio Box Control</title>
   </head>

   <body>
      <form>
```

```
         <input type = "radio" name = "subject" value = "maths">
Maths
         <input type = "radio" name = "subject" value =
"physics"> Physics
      </form>
   </body>

</html>
```

This will produce the following result −

Attributes

Following is the list of attributes for radio button.

Sr.No	Attribute & Description
1	**type** Indicates the type of input control and for checkbox input control it will be set to radio.
2	**name** Used to give a name to the control which is sent to the server to be recognized and get the value.
3	**value** The value that will be used if the radio box is selected.
4	**checked** Set to *checked* if you want to select it by default.

Select Box Control

A select box, also called drop down box which provides option to list down various options in the form of drop down list, from where a user can select one or more options.

Example

Here is example HTML code for a form with one drop down box

Live Demo

```
<!DOCTYPE html>
```

```
<html>

   <head>
      <title>Select Box Control</title>
   </head>

   <body>
      <form>
         <select name = "dropdown">
            <option value = "Maths" selected>Maths</option>
            <option value = "Physics">Physics</option>
         </select>
      </form>
   </body>

</html>
```

This will produce the following result −

Attributes

Following is the list of important attributes of <select> tag −

Sr.No	Attribute & Description
1	**name** Used to give a name to the control which is sent to the server to be recognized and get the value.
2	**size** This can be used to present a scrolling list box.
3	**multiple** If set to "multiple" then allows a user to select multiple items from the menu.

Following is the list of important attributes of <option> tag −

Sr.No	Attribute & Description
1	**value** The value that will be used if an option in the select box box is selected.

2	**selected** Specifies that this option should be the initially selected value when the page loads.
3	**label** An alternative way of labeling options

File Upload Box

If you want to allow a user to upload a file to your web site, you will need to use a file upload box, also known as a file select box. This is also created using the <input> element but type attribute is set to **file**.

Example

Here is example HTML code for a form with one file upload box −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>File Upload Box</title>
   </head>

   <body>
      <form>
         <input type = "file" name = "fileupload" accept =
"image/*" />
      </form>
   </body>

</html>
```

This will produce the following result −

Attributes

Following is the list of important attributes of file upload box −

Sr.No	Attribute & Description
1	**name** Used to give a name to the control which is sent to the server to be recognized and get

	the value.
2	**accept** Specifies the types of files that the server accepts.

Button Controls

There are various ways in HTML to create clickable buttons. You can also create a clickable button using <input>tag by setting its type attribute to **button**. The type attribute can take the following values −

Sr.No	Type & Description
1	**submit** This creates a button that automatically submits a form.
2	**reset** This creates a button that automatically resets form controls to their initial values.
3	**button** This creates a button that is used to trigger a client-side script when the user clicks that button.
4	**image** This creates a clickable button but we can use an image as background of the button.

Example

Here is example HTML code for a form with three types of buttons −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>File Upload Box</title>
   </head>

   <body>
      <form>
```

```
         <input type = "submit" name = "submit" value = "Submit"
/>
         <input type = "reset" name = "reset"  value = "Reset" />
         <input type = "button" name = "ok" value = "OK" />
         <input type = "image" name = "imagebutton" src =
"/html/images/logo.png" />
      </form>
   </body>

</html>
```

This will produce the following result −

Hidden Form Controls

Hidden form controls are used to hide data inside the page which later on can be pushed to the server. This control hides inside the code and does not appear on the actual page. For example, following hidden form is being used to keep current page number. When a user will click next page then the value of hidden control will be sent to the web server and there it will decide which page will be displayed next based on the passed current page.

Example

Here is example HTML code to show the usage of hidden control −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>File Upload Box</title>
   </head>

   <body>
      <form>
         <p>This is page 10</p>
         <input type = "hidden" name = "pagename" value = "10" />
         <input type = "submit" name = "submit" value = "Submit"
/>
         <input type = "reset" name = "reset"  value = "Reset" />
      </form>
   </body>

</html>
```

This will produce the following result −

HTML - Embed Multimedia

Sometimes you need to add music or video into your web page. The easiest way to add video or sound to your web site is to include the special HTML tag called **<embed>**. This tag causes the browser itself to include controls for the multimedia automatically provided browser supports <embed> tag and given media type.

You can also include a **<noembed>** tag for the browsers which don't recognize the <embed> tag. You could, for example, use <embed> to display a movie of your choice, and **<noembed>** to display a single JPG image if browser does not support <embed> tag.

Example

Here is a simple example to play an embedded midi file −

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML embed Tag</title>
   </head>

   <body>
      <embed src = "/html/yourfile.mid" width = "100%" height =
"60" >
         <noembed><img src = "yourimage.gif" alt = "Alternative
Media" ></noembed>
      </embed>
   </body>

</html>
```

The <embed> Tag Attributes

Following is the list of important attributes which can be used with <embed> tag.

Note −The *align* and *autostart* attributes deprecated in HTML5. Do not use these attributes.

Sr.No	Attribute & Description
1	**align** Determines how to align the object. It can be set to either center, *left or right*.
2	**autostart** This boolean attribute indicates if the media should start automatically. You can set it either true or false.

3	**loop** Specifies if the sound should be played continuously (set loop to true), a certain number of times (a positive value) or not at all (false)
4	**playcount** Specifies the number of times to play the sound. This is alternate option for *loop* if you are usiong IE.
5	**hidden** Specifies if the multimedia object should be shown on the page. A false value means no and true values means yes.
6	**width** Width of the object in pixels
7	**height** Height of the object in pixels
8	**name** A name used to reference the object.
9	**src** URL of the object to be embedded.
10	**volume** Controls volume of the sound. Can be from 0 (off) to 100 (full volume).

Supported Video Types

You can use various media types like Flash movies (.swf), AVI's (.avi), and MOV's (.mov) file types inside embed tag.

- **.swf files** − are the file types created by Macromedia's Flash program.
- **.wmv files** − are Microsoft's Window's Media Video file types.
- **.mov files** − are Apple's Quick Time Movie format.
- **.mpeg files** − are movie files created by the Moving Pictures Expert Group.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML embed Tag</title>
   </head>

   <body>
      <embed src = "/html/yourfile.swf" width = "200" height =
"200" >
         <noembed><img src = "yourimage.gif" alt = "Alternative
Media" ></noembed>
      </embed>
   </body>

</html>
```

This will produce the following result −

Background Audio

You can use HTML **<bgsound>** tag to play a soundtrack in the background of your webpage. This tag is supported by Internet Explorer only and most of the other browsers ignore this tag. It downloads and plays an audio file when the host document is first downloaded by the user and displayed. The background sound file also will replay whenever the user refreshes the browser.

Note − The bgsound tag is deprecated and it is supposed to be removed in a future version of HTML. So they should not be used rather, it's suggested to use HTML5 tag audio for adding sound. But still for learning purpose, this chapter will explain bgsound tag in detail.

This tag is having only two attributes *loop* and *src*. Both these attributes have same meaning as explained above.

Here is a simple example to play a small midi file −

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML embed Tag</title>
   </head>

   <body>
      <bgsound src = "/html/yourfile.mid">
         <noembed><img src = "yourimage.gif" ></noembed>
      </bgsound>
   </body>

</html>
```

This will produce the blank screen. This tag does not display any component and remains hidden.

Internet Explorer can also handle only three different sound format files − wav, the native format for PCs; au, the native format for most Unix workstations; and MIDI, a universal music-encoding scheme.

HTML Object tag

HTML 4 introduces the **<object>** element, which offers an all-purpose solution to generic object inclusion. The **<object>** element allows HTML authors to specify everything required by an object for its presentation by a user agent.

Here are a few examples −

Example - 1

You can embed an HTML document in an HTML document itself as follows −

```
<object data = "data/test.htm" type = "text/html" width = "300"
height = "200">
   alt : <a href = "data/test.htm">test.htm</a>
</object>
```

Here *alt* attribute will come into picture if browser does not support *object* tag.

Example - 2

You can embed a PDF document in an HTML document as follows −

```
<object data = "data/test.pdf" type = "application/pdf" width =
"300" height = "200">
   alt : <a href = "data/test.pdf">test.htm</a>
</object>
```

Example - 3

You can specify some parameters related to the document with the **<param>** tag. Here is an example to embed a wav file −

```
<object data = "data/test.wav" type = "audio/x-wav" width = "200"
height = "20">
   <param name = "src" value = "data/test.wav">
   <param name = "autoplay" value = "false">
   <param name = "autoStart" value = "0">
   alt : <a href = "data/test.wav">test.wav</a>
</object>
```

Example - 4

You can add a flash document as follows −

```
<object classid = "clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" id
= "penguin"
   codebase = "someplace/swflash.cab" width = "200" height =
"300">

   <param name = "movie" value = "flash/penguin.swf" />
   <param name = "quality" value = "high" />
   <img src = "penguin.jpg" width = "200" height = "300" alt =
"Penguin" />
</object>
```

Example - 5

You can add a java applet into HTML document as follows −

```
<object classid = "clsid:8ad9c840-044e-11d1-b3e9-00805f499d93"
   width = "200" height = "200">
   <param name = "code" value = "applet.class">
</object>
```

The **classid** attribute identifies which version of Java Plug-in to use. You can use the optional *codebase* attribute to specify if and how to download the JRE.

HTML - Marquees

An HTML marquee is a scrolling piece of text displayed either horizontally across or vertically down your webpage depending on the settings. This is created by using HTML <marquees> tag.

Note − The <marquee> tag deprecated in HTML5. Do not use this element, instead you can use JavaScript and CSS to create such effects.

Syntax

A simple syntax to use HTML <marquee> tag is as follows −

```
<marquee attribute_name = "attribute_value"....more attributes>
   One or more lines or text message or image
</marquee>
```

The <marquee> Tag Attributes

Following is the list of important attributes which can be used with <marquee> tag.

Sr.No	Attribute & Description
1	**width** This specifies the width of the marquee. This can be a value like 10 or 20% etc.

2	**height** This specifies the height of the marquee. This can be a value like 10 or 20% etc.
3	**direction** This specifies the direction in which marquee should scroll. This can be a value like *up, down, left* or *right*.
4	**behavior** This specifies the type of scrolling of the marquee. This can have a value like *scroll, slide* and *alternate*.
5	**scrolldelay** This specifies how long to delay between each jump. This will have a value like 10 etc.
6	**scrollamount** This specifies the speed of marquee text. This can have a value like 10 etc.
7	**loop** This specifies how many times to loop. The default value is INFINITE, which means that the marquee loops endlessly.
8	**bgcolor** This specifies background color in terms of color name or color hex value.
9	**hspace** This specifies horizontal space around the marquee. This can be a value like 10 or 20% etc.
10	**vspace** This specifies vertical space around the marquee. This can be a value like 10 or 20% etc.

Below are few examples to demonstrate the usage of marquee tag.

Examples - 1

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML marquee Tag</title>
   </head>

   <body>
      <marquee>This is basic example of marquee</marquee>
   </body>

</html>
```

This will produce the following result −

Examples - 2

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML marquee Tag</title>
   </head>

   <body>
      <marquee width = "50%">This example will take only 50%
width</marquee>
   </body>

</html>
```

This will produce the following result −

Examples - 3

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML marquee Tag</title>
   </head>

   <body>
      <marquee direction = "right">This text will scroll from
left to right</marquee>
   </body>

</html>
```

This will produce the following result −

Examples - 4

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML marquee Tag</title>
   </head>

   <body>
      <marquee direction = "up">This text will scroll from bottom
to up</marquee>
   </body>

</html>
```

This will produce the following result −

HTML - Header

We have learnt that a typical HTML document will have following structure −

```
Document declaration tag
<html>

   <head>
      Document header related tags
   </head>

   <body>
      Document body related tags
   </body>

</html>
```

This chapter will give a little more detail about header part which is represented by HTML <head> tag. The <head> tag is a container of various important tags like <title>, <meta>, <link>, <base>, <style>, <script>, and <noscript> tags.

The HTML <title> Tag

The HTML <title> tag is used for specifying the title of the HTML document. Following is an example to give a title to an HTML document −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
```

```
      <title>HTML Title Tag Example</title>
   </head>

   <body>
      <p>Hello, World!</p>
   </body>

</html>
```

This will produce the following result −

The HTML <meta> Tag

The HTML <meta> tag is used to provide metadata about the HTML document which includes information about page expiry, page author, list of keywords, page description etc.

Following are few of the important usages of <meta> tag inside an HTML document −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Meta Tag Example</title>

      <!-- Provide list of keywords -->
      <meta name = "keywords" content = "C, C++, Java, PHP, Perl,
Python">

      <!-- Provide description of the page -->
      <meta name = "description" content = "Simply Easy Learning
by Tutorials Point">

      <!-- Author information -->
      <meta name = "author" content = "Tutorials Point">

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

      <!-- Page refreshing delay -->
      <meta http-equiv = "refresh" content = "30">

      <!-- Page expiry -->
      <meta http-equiv = "expires" content = "Wed, 21 June 2006
14:25:27 GMT">

      <!-- Tag to tell robots not to index the content of a page
-->
      <meta name = "robots" content = "noindex, nofollow">
   </head>

   <body>
      <p>Hello, World!</p>
   </body>

</html>
```

This will produce the following result −

The HTML <base> Tag

The HTML <base> tag is used for specifying the base URL for all relative URLs in a page, which means all the other URLs will be concatenated into base URL while locating for the given item.

For example, all the given pages and images will be searched after prefixing the given URLs with base URL http://www.tutorialspoint.com/ directory −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Base Tag Example</title>
      <base href = "https://www.tutorialspoint.com/" />
   </head>

   <body>
      <img src = "/images/logo.png" alt = "Logo Image"/>
      <a href = "/html/index.htm" title = "HTML Tutorial"/>HTML
Tutorial</a>
   </body>

</html>
```

This will produce the following result −

But if you change base URL to something else, for example, if base URL is http://www.tutorialspoint.com/home then image and other given links will become like http://www.tutorialspoint.com/home/images/logo.png and http://www.tutorialspoint.com/html/index.htm

The HTML <link> Tag

The HTML <link> tag is used to specify relationships between the current document and external resource. Following is an example to link an external style sheet file available in **css** sub-directory within web root −

Live Demo

```
<!DOCTYPE html>
```

```html
<html>

   <head>
      <title>HTML link Tag Example</title>
      <base href = "https://www.tutorialspoint.com/" />
      <link rel = "stylesheet" type = "text/css" href =
"/css/style.css">
   </head>

   <body>
      <p>Hello, World!</p>
   </body>

</html>
```

This will produce the following result −

The HTML <style> Tag

The HTML <style> tag is used to specify style sheet for the current HTML document. Following is an example to define few style sheet rules inside <style> tag −

Live Demo

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

   <head>
      <title>HTML style Tag Example</title>
      <base href = "https://www.tutorialspoint.com/" />

      <style type = "text/css">
         .myclass {
            background-color: #aaa;
            padding: 10px;
         }
      </style>
   </head>

   <body>
      <p class = "myclass">Hello, World!</p>
   </body>

</html>
```

This will produce the following result −

Note − To learn about how Cascading Style Sheet works, kindly check a separate tutorial available at css

The HTML <script> Tag

The HTML <script> tag is used to include either external script file or to define internal script for the HTML document. Following is an example where we are using JavaScript to define a simple JavaScript function −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML script Tag Example</title>
      <base href = "http://www.tutorialspoint.com/" />

      <script type = "text/JavaScript">
         function Hello() {
            alert("Hello, World");
         }
      </script>
   </head>

   <body>
      <input type = "button" onclick = "Hello();" name = "ok"
value = "OK"  />
   </body>

</html>
```

This will produce the following result, where you can try to click on the given button −

Note − To learn about how JavaScript works, kindly check a separate tutorial available at javascript

HTML - Style Sheet

Cascading Style Sheets (CSS) describe how documents are presented on screens, in print, or perhaps how they are pronounced. W3C has actively promoted the use of style sheets on the Web since the consortium was founded in 1994.

Cascading Style Sheets (CSS) provide easy and effective alternatives to specify various attributes for the HTML tags. Using CSS, you can specify a number of style properties for a given HTML element. Each property has a name and a value, separated by a colon (:). Each property declaration is separated by a semi-colon (;).

Example

First let's consider an example of HTML document which makes use of <font> tag and associated attributes to specify text color and font size −

Note − The *font* tag deprecated and it is supposed to be removed in a future version of HTML. So they should not be used rather, it's suggested to use CSS styles to manipulate your fonts. But still for learning purpose, this chapter will work with an example using the font tag.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML CSS</title>
   </head>

   <body>
      <p><font color = "green" size = "5">Hello,
World!</font></p>
   </body>

</html>
```

We can re-write above example with the help of Style Sheet as follows −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML CSS</title>
   </head>

   <body>
      <p style = "color:green; font-size:24px;" >Hello,
World!</p>
   </body>

</html>
```

This will produce the following result −

You can use CSS in three ways in your HTML document −

- **External Style Sheet** − Define style sheet rules in a separate .css file and then include that file in your HTML document using HTML <link> tag.
- **Internal Style Sheet** − Define style sheet rules in header section of the HTML document using <style> tag.
- **Inline Style Sheet** − Define style sheet rules directly along-with the HTML elements using **style** attribute.

Let's see all the three cases one by one with the help of suitable examples.

External Style Sheet

If you need to use your style sheet to various pages, then its always recommended to define a common style sheet in a separate file. A cascading style sheet file will have extension as **.css** and it will be included in HTML files using <link> tag.

Example

Consider we define a style sheet file **style.css** which has following rules −

```
.red {
   color: red;
}
.thick {
   font-size:20px;
}
.green {
   color:green;
}
```

Here we defined three CSS rules which will be applicable to three different classes defined for the HTML tags. I suggest you should not bother about how these rules are being defined because you will learn them while studying CSS. Now let's make use of the above external CSS file in our following HTML document −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML External CSS</title>
      <link rel = "stylesheet" type = "text/css" href =
"/html/style.css">
   </head>

   <body>
      <p class = "red">This is red</p>
      <p class = "thick">This is thick</p>
      <p class = "green">This is green</p>
      <p class = "thick green">This is thick and green</p>
   </body>

</html>
```

This will produce the following result −

Internal Style Sheet

If you want to apply Style Sheet rules to a single document only, then you can include those rules in header section of the HTML document using <style> tag.

Rules defined in internal style sheet overrides the rules defined in an external CSS file.

Example

Let's re-write above example once again, but here we will write style sheet rules in the same HTML document using <style> tag −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Internal CSS</title>

      <style type = "text/css">
         .red {
            color: red;
         }
         .thick{
            font-size:20px;
         }
         .green {
            color:green;
         }
      </style>
   </head>

   <body>
      <p class = "red">This is red</p>
      <p class = "thick">This is thick</p>
      <p class = "green">This is green</p>
      <p class = "thick green">This is thick and green</p>
   </body>

</html>
```

This will produce the following result −

Inline Style Sheet

You can apply style sheet rules directly to any HTML element using **style** attribute of the relevant tag. This should be done only when you are interested to make a particular change in any HTML element only.

Rules defined inline with the element overrides the rules defined in an external CSS file as well as the rules defined in <style> element.

Example

Let's re-write above example once again, but here we will write style sheet rules along with the HTML elements using **style** attribute of those elements.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Inline CSS</title>
   </head>

   <body>
      <p style = "color:red;">This is red</p>
      <p style = "font-size:20px;">This is thick</p>
      <p style = "color:green;">This is green</p>
      <p style = "color:green;font-size:20px;">This is thick and 
green</p>
   </body>

</html>
```

This will produce the following result −

HTML - JavaScript

A **script** is a small piece of program that can add interactivity to your website. For example, a script could generate a pop-up alert box message, or provide a dropdown menu. This script could be written using JavaScript or VBScript.

You can write various small functions, called event handlers using any of the scripting language and then you can trigger those functions using HTML attributes.

Now-a-days, only **JavaScript** and associated frameworks are being used by most of the web developers, VBScript is not even supported by various major browsers.

You can keep JavaScript code in a separate file and then include it wherever it's needed, or you can define functionality inside HTML document itself. Let's see both the cases one by one with suitable examples.

External JavaScript

If you are going to define a functionality which will be used in various HTML documents then it's better to keep that functionality in a separate JavaScript file and then include that file in your HTML documents. A JavaScript file will have extension as **.js** and it will be included in HTML files using <script> tag.

Example

Consider we define a small function using JavaScript in **script.js** which has following code −

```
function Hello() {
   alert("Hello, World");
}
```

Now let's make use of the above external JavaScript file in our following HTML document −

Live Demo

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

```
   <head>
      <title>Javascript External Script</title>
      <script src = "/html/script.js" type = 
"text/javascript"/></script>
   </head>

   <body>
      <input type = "button" onclick = "Hello();" name = "ok" 
value = "Click Me" />
   </body>

</html>
```

This will produce the following result, where you can try to click on the given button –

Internal Script

You can write your script code directly into your HTML document. Usually we keep script code in header of the document using <script> tag, otherwise there is no restriction and you can put your source code anywhere in the document but inside <script> tag.

Example

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>JavaScript Internal Script</title>
      <base href = "https://www.tutorialspoint.com/" />

      <script type = "text/JavaScript">
         function Hello() {
            alert("Hello, World");
         }
      </script>
   </head>

   <body>
      <input type = "button" onclick = "Hello();" name = "ok" 
value = "Click Me" />
   </body>

</html>
```

This will produce the following result, where you can try to click on the given button –

Event Handlers

Event handlers are nothing but simply defined functions which can be called against any mouse or keyboard event. You can define your business logic inside your event handler which can vary from a single to 1000s of line code.

Following example explains how to write an event handler. Let's write one simple function *EventHandler()* in the header of the document. We will call this function when any user brings mouse over a paragraph.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Event Handlers Example</title>
      <base href = "https://www.tutorialspoint.com/" />

      <script type = "text/JavaScript">
         function EventHandler() {
            alert("I'm event handler!!");
         }
      </script>
   </head>

   <body>
      <p onmouseover = "EventHandler();">Bring your mouse here to
see an alert</p>
   </body>

</html>
```

Now This will produce the following result. Bring your mouse over this line and see the result −

Hide Scripts from Older Browsers

Although most (if not all) browsers these days support JavaScript, but still some older browsers don't. If a browser doesn't support JavaScript, instead of running your script, it would display the code to the user. To prevent this, you can simply place HTML comments around the script as shown below.

```
JavaScript Example:
<script type = "text/JavaScript">
   <!--
      document.write("Hello JavaScript!");
   //-->
</script>

VBScript Example:
<script type = "text/vbscript">
   <!--
      document.write("Hello VBScript!")
   '-->
```

```
</script>
```

The <noscript> Element

You can also provide alternative info to the users whose browsers don't support scripts and for those users who have disabled script option their browsers. You can do this using the **<noscript>** tag.

```
JavaScript Example:
<script type = "text/JavaScript">
   <!--
      document.write("Hello JavaScript!");
   //-->
</script>

<noscript>Your browser does not support JavaScript!</noscript>

VBScript Example:
<script type = "text/vbscript">
   <!--
      document.write("Hello VBScript!")
   '-->
</script>

<noscript>Your browser does not support VBScript!</noscript>
```

Default Scripting Language

There may be a situation when you will include multiple script files and ultimately using multiple <script> tags. You can specify a default scripting language for all your *script* tags. This saves you from specifying the language every time you use a script tag within the page. Below is the example −

```
<meta http-equiv = "Content-Script-Type" content =
"text/JavaScript" />
```

Note that you can still override the default by specifying a language within the script tag.

HTML - Layouts

A webpage layout is very important to give better look to your website. It takes considerable time to design a website's layout with great look and feel.

Now-a-days, all modern websites are using CSS and JavaScript based framework to come up with responsive and dynamic websites but you can create a good layout using simple HTML tables or division tags in combination with other formatting tags. This chapter will give you few examples on how to create a simple but working layout for your webpage using pure HTML and its attributes.

HTML Layout - Using Tables

The simplest and most popular way of creating layouts is using HTML <table> tag. These tables are arranged in columns and rows, so you can utilize these rows and columns in whatever way you like.

Example

For example, the following HTML layout example is achieved using a table with 3 rows and 2 columns but the header and footer column spans both columns using the colspan attribute −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Layout using Tables</title>
   </head>

   <body>
      <table width = "100%" border = "0">

         <tr>
            <td colspan = "2" bgcolor = "#b5dcb3">
               <h1>This is Web Page Main title</h1>
            </td>
         </tr>
         <tr valign = "top">
            <td bgcolor = "#aaa" width = "50">
               <b>Main Menu</b><br />
               HTML<br />
               PHP<br />
               PERL...
            </td>

            <td bgcolor = "#eee" width = "100" height = "200">
               Technical and Managerial Tutorials
            </td>
         </tr>
         <tr>
            <td colspan = "2" bgcolor = "#b5dcb3">
               <center>
                  Copyright © 2007 Tutorialspoint.com
               </center>
            </td>
         </tr>

      </table>
   </body>

</html>
```

This will produce the following result −

Multiple Columns Layout - Using Tables

You can design your webpage to put your web content in multiple pages. You can keep your content in middle column and you can use left column to use menu and right column can be used to put advertisement or some other stuff. This layout will be very similar to what we have at our website tutorialspoint.com.

Example

Here is an example to create three column layout −

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>Three Column HTML Layout</title>
   </head>

   <body>
      <table width = "100%" border = "0">

         <tr valign = "top">
            <td bgcolor = "#aaa" width = "20%">
               <b>Main Menu</b><br />
               HTML<br />
               PHP<br />
               PERL...
            </td>

            <td bgcolor = "#b5dcb3" height = "200" width = "60%">
               Technical and Managerial Tutorials
            </td>

            <td bgcolor = "#aaa" width = "20%">
               <b>Right Menu</b><br />
               HTML<br />
               PHP<br />
               PERL...
            </td>
         </tr>

      <table>
   </body>

</html>
```

This will produce the following result −

HTML Layouts - Using DIV, SPAN

The <div> element is a block level element used for grouping HTML elements. While the <div> tag is a block-level element, the HTML <span> element is used for grouping elements at an inline level.

Although we can achieve pretty nice layouts with HTML tables, but tables weren't really designed as a layout tool. Tables are more suited to presenting tabular data.

Note − This example makes use of Cascading Style Sheet (CSS), so before understanding this example you need to have a better understanding on how CSS works.

Example

Here we will try to achieve same result using <div> tag along with CSS, whatever you have achieved using <table> tag in previous example.

Live Demo

```
<!DOCTYPE html>
<html>

   <head>
      <title>HTML Layouts using DIV, SPAN</title>
   </head>

   <body>
      <div style = "width:100%">

         <div style = "background-color:#b5dcb3; width:100%">
            <h1>This is Web Page Main title</h1>
         </div>

         <div style = "background-color:#aaa; height:200px;
width:100px; float:left;">
            <div><b>Main Menu</b></div>
            HTML<br />
            PHP<br />
            PERL...
         </div>

         <div style = "background-color:#eee; height:200px;
width:350px; float:left;" >
            <p>Technical and Managerial Tutorials</p>
         </div>

         <div style = "background-color:#aaa; height:200px;
width:100px; float:right;">
            <div><b>Right Menu</b></div>
            HTML<br />
            PHP<br />
            PERL...
         </div>

         <div style = "background-color:#b5dcb3; clear:both">
```

```
            <center>
               Copyright © 2007 Tutorialspoint.com
            </center>
         </div>

      </div>
   </body>

</html>
```

This will produce the following result −

You can create better layout using DIV, SPAN along with CSS. For more information on CSS, please refer to CSS Tutorial.