HTML Tutorial

Tutorialspoint.com

HTML or Hypertext Markup Language, is the most widely used language on Web.

Technically, HTML is not a programming language, but rather a markup language.

This tutorial gives a complete understanding on HTML.

Send your feedback using Contact Us Form

HTML Introduction

Before your begin:

Before you begin, it's important that you know Windows or Unix. A working knowledge of Windows or Unix makes it much easier to learn HTML.

You should be familiar with:

- Basic word processing using any text editor.
- How to create directories and files.
- How to navigate through different directories.
- Basic understaning on internet browsing using a browser like Internet Explorer or Firefox etc.

Introducing HTML:

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

- **Hypertext** refers to the way in which Web pages (HTML documents) are linked together. When you click a link in a Web page, you are using hypertext.
- **Markup Language** describes how HTML works. With a markup language, you 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.

All you need to do to use HTML is to learn what type of markup to use to get the results you want.

Creating HTML Document:

Creating an HTML document is easy. To begin coding HTML you need only two things: a simple-text editor and a web browser. Notepad is the most basic of simple-text editors and you will probably code a fair amount of HTML with it.

You can use our HTML Online Editor to learn HTML. Here are the simple steps to create a baisc HTML document:

- Open Notepad or another text editor.
- At the top of the page type <html>.
- On the next line, indent five spaces and now add the opening header tag: <head>.
- On the next line, indent ten spaces and type <title> </title>.
- Go to the next line, indent five spaces from the margin and insert the closing header tag: </head>.
- Five spaces in from the margin on the next line, type<body>.
- Now drop down another line and type the closing tag right below its mate: </body>.
- Finally, go to the next line and type </html>.
- In the File menu, choose Save As.
- In the Save as Type option box, choose All Files.
- Name the file template.htm.
- Click Save.

You have basic HTML document now, to see some result put the following code in title and body tags.

```
<html>
<head>
<title>This is document title</title>
</head>
<body>
<h1>This is a heading</h1>
<p>Document description goes here.....</p>
</body>
</html>
```

Now you have created one **HTML page** and you can use a Web Browser to open this HTML file to see the result. Hope you understood that Web Pages are nothing but they are simple HTML files with some content which can be rendered using Web Browsers.

Here <html>, <head>,...<p>, <h1> etc. are called HTML tags. HTML tags are building blocks of an HTML document nd we will learn all the HTML tags in subsequent chapters.

NOTE: One HTML file can have extension as **.htm** or **.html**. So you can use either of them based on your comfort.

HTML Document Structure:

An HTML document starts and ends with <html> and >/html> tags. These tags tell the browser that the entire document is composed in HTML. Inside these two tags, the document is split into two sections:

- The <head>...</head> elements, which contain information about the document such as title of the document, author of the document etc. Information inside this tag does not display outside.
- The <body>...</body> elements, which contain the real content of the document that you see on your screen.

HTML Tags and Elements:

HTML language is a markup language and we use many tags to markup text. In the above example you have seen <html>, <body> etc. are called HTML tags or HTML elements.

Every tag consists of a tag name, sometimes followed by an optional list of tag attributes , all placed between opening and closing brackets (< and >). The simplest tag is nothing more than a name appropriately enclosed in brackets, such as <head> and <i>. More complicated tags contain one or more attributes , which specify or modify the behaviour of the tag.

According to the HTML standard, tag and attribute names are not case-sensitive. There's no difference in effect between <head>, <Head>, <HEAD>, or even <HeaD>; they are all equivalent. But with XHTML, case is important: all current standard tag and attribute names are in lowercase.

HTML is Forgiving?

A very good quality associated with all the browsers is that they would not give any error if you have not put any HTML tag or attribute properly. They will just ignore that tag or attribute and will apply only correct tags and attributes before displaying the result.

We can not say, HTML is forgiving because this is just a markup language and required to format documents.

HTML Basic Tags

The basic structure for all HTML documents is simple and should include the following minimum elements or tags:

- **<html>** - The main container for HTML pages
- **<head>** - The container for page header information
- **<title>** - The title of the page
- **<body>** - The main body of the page

Remember that before an opening <html> tag, an XHTML document can contain the optional XML declaration, and it should always contain a DOCTYPE declaration indicating which version of XHTML it uses.

Now we will explain each of these tags one by one. In this tutorial you will find the terms element and tag are used interchangeably.

The <html> Element:

The <html> element is the containing element for the whole HTML document. Each HTML document should have one <html> and each document should end with a closing </html> tag.

Following two elements appear as direct children of an <html> element:

- <head>
- <body>

As such, start and end HTML tags enclose all the other HTML tags you use to describe the Web page.

The <head> Element:

The <head> element is just a container for all other header elements. It should be the first thing to appear after the opening <html> tag.

Each <head> element should contain a <title> element indicating the title of the document, although it may also contain any combination of the following elements, in any order:

- The <base> tag is used to areate a "base" url for all links on the page. Check HTML Base tag.
- The <object> tag is designed to include images, JavaScript objects, Flash animations, MP3 files, QuickTime movies and other components of a page. Check HTML Object tag.
- The <link> tag is used to link to an external file, such as a style sheet or JavaScript file. Check HTML Link tag.
- The <style> tag is used to include CSS rules inside the document. Check HTML Style tag.
- The <script> tag is used to include JAVAScript or VBScript inside the document. Check HTML Script tag.
- The <meta> tag includes information about the document such as keywords and a description, which are particularly helpful for search applications. Check HTML Meta tag.

Example:

Following is the example of head tag.

```
<head>
<title>HTML Basic tags</title>
<meta name="Keywords" content="HTML, Web Pages" />
<meta name="description" content="HTML Basic Tags" />
<base href="http://www.tutorialspoint.com" />
<link rel="stylesheet" type="text/css" href="tp.css" />
<script type="text/javascript">
_uacct = "UA-232293";
urchinTracker();
</script>
</head>
```

The <title> Element:

You should specify a title for every page that you write inside the <title> element. This element is a child of the <head> element). It is used in several ways:

- It displays at the very top of a browser window.
- It is used as the default name for a bookmark in browsers such as IE and Netscape.
- Its is used by search engines that use its content to help index pages.

Therefore it is important to use a title that really describes the content of your site. The <title> element should contain only the text for the title and it may not contain any other elements.

Example:

Here is the example of using title tag.

```
<head>
<title>HTML Basic tags</title>
</head>
```

The <body> Element:

The <body> element appears after the <head> element and contains the part of the Web page that you actually see in the main browser window, which is sometimes referred to as body content.

A <body> element may contain anything from a couple of paragraphs under a heading to more complicated layouts containing forms and tables.

Most of what you will be learning in this and the following five chapters will be written between the opening <body> tag and closing </body> tag.

Example:

Here is the example of using body tag.

```
<body>
   <p>This is a paragraph tag.</p>
</body>
```

Putting all together:

Now if we will put all these tags together, it will constitute a complete HTML document as follows:

```
<html>

<head>
<title>HTML Basic tags</title>
<meta name="Keywords" content="HTML, Web Pages" />
<meta name="description" content="HTML Basic Tags" />
<base href="http://www.tutorialspoint.com" />
<link rel="stylesheet" type="text/css" href="tp.css" />
<script type="text/javascript">
_uacct = "UA-232293";
urchinTracker();
</script>
</head>

<body>
   <p>This is a paragraph tag.</p>
</body>

</html>
```

To Become more comfortable - Do Online Practice

HTML Meta Tags

HTML lets you specify metadata - information about a document rather than document content - in a variety of ways. The META element can be used to include name/value pairs describing properties of the HTML document, such as author, Expiry Date, a list of key words, author etc.

The <meta> tag is an empty element and so does not have a closing tag, rather, <meta> tags carry information within attributes, so you need a forward slash character at the end of the element.

Metadata provided by using meta tag is a very important part of the web. It can assist search engines in finding the best match when a user performs a search. Search engines will often look

at any metadata attached to a page - especially keywords - and rank it higher than another page with less relevant metadata, or with no metadata at all.

Adding Meta Tags to Your Documents:

You can add metadata to your web pages by placing <meta> tags between the <head> and </head> tags. The can include the following attributes:

Attribute	Description
Name	Name for the property. Can be anything. Examples include, keywords, description, author, revised, generator etc.
content	Specifies the property's value.
scheme	Specifies a scheme to use to interpret the property's value (as declared in the content attribute).
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.

NOTE: Core attributes for all the elements are discussed in next chapter.

Meta Tag Examples:

Let's see few important usage of Meta Tags.

Specifying Keywords:

We specify keywords which will be used by the search engine to search a web page. So using following tag you can specify important keywords related to your page.

```
<head>
<meta name="keywords" content="HTML, meta tags, metadata" />
</head>
```

Document Description:

This is again important information and many search engine use this information as well while searching a web page. So you should give an appropriate description of the page.

```
<head>
<meta name="description" content="Learn about Meta Tags." />
</head>
```

Document Revision date:

This information tells about last time the document was updated.

```
<head>
<meta name="revised" content="Tutorialspoint, 6/12/2006" />
</head>
```

Document Refreshing:

You can specify a duration after which your web page will keep refreshing. If you want your page keep refreshing after every 10 seconds then use the following syntax.

```
<head>
<meta http-equiv="refresh" content="10" />
</head>
```

Page Redirection:

You can specify a page redirection using Meta Tag. Following is an example of redirecting current page to another page. You can specify a duration after which page will be redirected.

```
<head>
<meta http-equiv="refresh"
        content="10; url=http://www.tutorialspoint.com" />
</head>
```

If you don't provide a duration then page will be redirected immediately.

Setting Cookies:

You can use Meta Tag to store cookies on client side later information can be used by then Web Server to track a site visitor.

```
<head>
<meta http-equiv="cookie" content="userid=xyz;
        expires=Wednesday, 08-Aug-00 23:59:59 GMT; />
</head>
```

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.

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:

```
<head>
<meta name="author" content="Mahnaz Mohtashim" />
</head>
```

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.

HTML Attributes

Attributes are another important part of HTML markup. An attribute is used to define the characteristics of an 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 <font> element in the example carries an attribute whose name is *face*, which you can use to indicate which typeface you want the text to appear in.
- The *value* is what you want the value of the property to be. The first example was supposed to use the Arial typeface, so the value of the *face* attribute is Arial.

The value of the attribute should be put in double quotation marks, and is separated from the name by the equals sign. You can see that a color for the text has been specified as well as the typeface in this <font> element:

```
<font face="arial" color="#CC0000">
```

Many HTML tags have a unique set of their own attributes. These will be discussed as each tag is introduced throughout the tutorial. Right now we want to focus on a set of generic attributes that can be used with just about every HTML Tag in existence.

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 can be used to uniquely identify any element within a page (or style sheet). 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, the id attribute could be used to distinguish between two paragraph elements, like so:

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

Note that there are some special rules for the value of the id attribute, it must:

- Begin with a letter (A.Z or a.z) and can then be followed by any number of letters, digits (0.9), hyphens, underscores, colons, and periods.
- Remain unique within that document; no two attributes may have the same value within that HTML document.

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 or while the element is loading.

For example:

```
<h4 title="Hello HTML!">Titled Heading Tag Example</h4>
```

Above code will generate following result:

Titled Heading Tag Example

Now try to bring your cursor over "Titled Heading Tag Example" and see the result.

The class Attribute:

The *class* attribute is used to associate an element with a style sheet, and specifies the class of element. You learn more about the use of the class attribute when you will learn Casecading 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 CSS rules within the element. For example:

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

Internationalization Attributes:

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

- dir
- lang
- xml:lang

The dir Attribute:

The *dir* attribute allows you to indicate to the browser 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:

```
<html dir=rtl>
<head>
<title>Display Directions</title>
</head>
<body>
This is how IE 5 renders right-to-left directed text.
</body>
</html>
```

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.

When included within the <html> tag, the *lang* attribute specifies the language you've generally used within the document. When used within other tags, the lang attribute specifies the language you used within that tag's content. Ideally, the browser will use *lang* to better render the text for the user.

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:

```
<html lang=en>
<head>
<title>English Language Page</title>
</head>
<body>
This page is using English Language
</body>
</html>
```

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 HTML's 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 an 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 for your 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 Tags

If you want people to read what you have written, then structuring your text well is even more important on the Web than when writing for print. People have trouble reading wide, long, paragraphs of text on Web sites unless they are broken up well.

This section will teach you basic text formatting elements like heading elements and paragraph elements.

Whitespace and Flow:

Before you start to mark up your text, it is best to understand what HTML does when it comes across spaces and how browsers treat long sentences and paragraphs of text.

You might think that if you put several consecutive spaces between two words, the spaces would appear between those words onscreen, but this is not the case; by default, only one space will be displayed. This is known as white *space collapsing*. So you need to use special HTML tags to create multiple spaces.

Similarly, if you start a new line in your source document, or you have consecutive empty lines, these will be ignored and simply treated as one space. So you need to use special HTML tags to create more number of empty lines.

Create Headings - The <hn> Elements:

Any documents starts with a heading. You use different sizes for your headings. HTML also have 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 after that heading.

Example:

```
<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>
```

This will display following result:

This is heading 1

This is heading 2

This is heading 3

This is heading 4

This is heading 5

This is heading 6

To Become more comfortable - Do Online Practice

Create Paragraph - The <p> Element:

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

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

This will produce following result:

Here is a paragraph of text.

Here is a second paragraph of text.

Here is a third paragraph of text.

You can use *align* attribute to align your paragraphs.

```
<p align="left">This is left aligned.</p>
<p align="center">This is center aligned.</p>
```

```
<p align="right">This is right aligned.</p>
<p align="justify">This is jutified. This works when you have multiple lines in your paragraph
and you want to justfy all the lines so that they can look more nice.</p>
```

This will produce following result:

This is left aligned.

This is center aligned.

This is right aligned.

This is jutified. This works when you have multiple lines in your paragraph and you want to justfy all the lines so that they can look more nice.

To Become more comfortable - Do Online Practice

Create Line Breaks - The
 Element:

Whenever you use the <br /> element, anything following it starts on 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.

Note: The <br /> element 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 XHTML

Example:

```
Hello<br />
You come most carefully upon your hour.<br />
Thanks<br />
Mahnaz
```

This will produce following result:

```
Hello

You come most carefully upon your hour.

Thanks

Mahnaz
```

To Become more comfortable - Do Online Practice

Centring Content - The <center> Element:

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

Example:

```
<p>This is not in the center.</p>
<center>
<p>This is in the center.</p>
</center>
```

This will produce following result:

This is not in the center.

This is in the center.

Nonbreaking Spaces:

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

A good 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 ( ) instead of a normal space. For example, when coding the "12 Angry Men" paragraph, you would use something similar to the following code:

```
<p>A good example of this technique appears in the movie "12 Angry Men."</p>
```

Soft Hyphens:

Occasionally, you will want to allow a browser to hyphenate long words to better justify a paragraph. For example, consider the following code and its resulting output.

```
<p style="text-align: justify;"> The morbid fear of the number 13, or triskaidekaphobia, has plagued some important historic figures like Mahamiya and Nanao.</p>
```

In cases where you want a client browser to be able to hyphenate a word if necessary, use the soft hyphen entity (­) to specify where a word should be hyphenated. So above example should be written as follows:

```
<p style="text-align: justify;"> Example for soft hyphen - The morbid fear of the number 13, or tri&shy;skai&shy;deka&shy;phobia, has plagued some important historic figures like Mahamiya and Nanao.</p>
```

This will produce following result:

Example for soft hyphen - The morbid fear of the number 13, or triskaidekaphobia, has plagued some important historic figures like Mahamiya and Nanao.

NOTE: This may notwork with some web browsers.

Preserve Formatting - The <pre> Element:

Sometimes you want your text to follow the exact format of how it is written in the HTML document. In those 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.

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

This will produce following result:

```
function testFunction( strText ){
    alert (strText)
}
```

To Become more comfortable - Do Online Practice

Horizontal Rules - The <hr /> Element

Horizontal rules 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 follows:

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

This will produce following result:

This is paragraph one and should be on top

This is paragraph two and should be at bottom

Again <hr /> 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.

Note: The <hr /> element 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 <hr> it is not valid XHTML

To Become more comfortable - Do Online Practice

Presentational Tags:

If you use a word processor, you are 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 - The Element:

Anything that appears in a ... element is displayed in bold, like the word bold here:

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

This will produce following result:

The following word uses a **bold** typeface.

To Become more comfortable - Do Online Practice

Italic Text - The <i> Element:

Anything that appears in a <i>...</i> element is displayed in italicized, like the word italicized here:

```
<p>The following word uses a <i>italicized</i> typeface.</p>
```

This will produce following result:

The following word uses a *italicized* typeface.

To Become more comfortable - Do Online Practice

Underlined Text - The <u> Element:

Anything that appears in a <u>...</u> element is displayed with underline, like the word underlined here:

```
<p>The following word uses a <u>underlined</u> typeface.</p>
```

This will produce following result:

The following word uses a underlined typeface.

To Become more comfortable - Do Online Practice

Strike Text - The <strike> Element:

Anything that appears in a <strike>...</strike> element is displayed with strikethrough, which is a thin line through the text:

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

This will produce following result:

The following word uses a ~~strikethrough~~ typeface.

To Become more comfortable - <u>Do Online Practice</u>

Monospaced font - The <tt> Element:

The content of a <tt> element is written in monospaced font. Most 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 is the same width.

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

This will produce following result:

The following word uses a `monospaced` typeface.

To Become more comfortable - <u>Do Online Practice</u>

Superscript Text - The <sup> Element:

The content of a <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.

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

This will produce following result:

The following word uses a ^{superscript} typeface.

To Become more comfortable - <u>Do Online Practice</u>

Subscript Text - The <sub> Element:

The content of a <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.

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

This will produce following result:

The following word uses a subscript typeface.

To Become more comfortable - Do Online Practice

Larger Text - The <big> Element:

The content of the <big> element is displayed one font size larger than the rest of the text surrounding it.

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

This will produce following result:

The following word uses a big typeface.

To Become more comfortable - Do Online Practice

Smaller Text - The <small> Element:

The content of the <small> element is displayed one font size smaller than the rest of the text surrounding it.

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

This will produce following result:

The following word uses a small typeface.

To Become more comfortable - Do Online Practice

Grouping - The <div> and <span> Elements :

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.

The <div> element is used to group block-level elements together:

```
<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" bgcolor="white">
<h5>Content Articles</h5>
<p>Actual content goes here.....</p>
</div>
```

This will produce following result:

HOME | CONTACT | ABOUT

Content Articles

Actual content goes here.....

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

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

This will produce following result:

This is the example of span tag and the div tag alongwith CSS

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

To Become more comfortable - Do Online Practice

For a complete list of HTML Tags and related attributes please check reference to HTML Tags Reference.

HTML Phrase Tags

While some of these phrase elements are displayed in a similar manner to the <b>, <i>, <pre>, and <tt> elements you have already seen, they are designed for specific purposes. For example, the <em> and <strong> elements give text emphasis and strong emphasis respectively and there are several elements for marking up quotes.

We will see all phrase tags in this section with examples.

Emphasized Text - The <em> Element:

The content of an <em> element is intended to be a point of emphasis in your document, and it is usually displayed in italicized text. The kind of emphasis intended is on words such as "must" in the following sentence:

```
<p>You <em>must</em> remember to close elements in XHTML.</p>
```

This will produce following result:

You *must* remember to close elements in XHTML.

To Become more comfortable - <u>Do Online Practice</u>

Strong Text - The Element:

The element is intended to show strong emphasis for its content; stronger emphasis than the <em> element. As with the <em> element, the element should be used only when you want to add strong emphasis to part of a document.

<p>You must remember to close elements in XHTML.</p>

This will produce following result:

You **must** remember to close elements in XHTML.

To Become more comfortable - <u>Do Online Practice</u>

Text Abbreviation - The <abbr> Element :

You can indicate when you are using an abbreviated form by placing the abbreviation between opening <abbr> and closing </abbr> tags.

<p>I have a friend called <abbr title="Abhishek">Abhy</abbr>.</p>

This will produce following result:

I have a friend called Abhy.

To Become more comfortable - <u>Do Online Practice</u>

Using Acronym - The <acronym> Element :

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

When possible use a title attribute whose value is the full version of the acronyms on the <acronym> element, and if the acronym is in a different language, include an xml:lang attribute in XHTML documents.

<p>This chapter covers marking up text in <acronym title="Extensible Hypertext Markup Language">XHTML</acronym>.</p>

This will produce following result:

This chapter covers marking up text in XHTML.

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

To Become more comfortable - Do Online Practice

Special Terms - The <dfn> Element :

The <dfn> element allows you to specify that you are introducing a special term. Its use is similar to the words that are in italics in the midst of paragraphs in this book when new key concepts are introduced.

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

```
<p>This tutorial teaches you how mark up your documents for the web using
<dfn>XHTML</dfn>.</p>
```

This will produce following result:

This tutorial teaches you how mark up your documents for the web using *XHTML*.

To Become more comfortable - Do Online Practice

Quoting Text - The <blockquote> Element :

When you want to quote a passage from another source, you should use the <blockquote> element.

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

```
<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>
```

This will produce following result:

The following description of XHTML is taken from the W3C Web site:

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.

You can use the *cite* attribute on the <blockquote> element to indicate the source of the quote.

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

<blockquote cite="http://www.w3.org/markup/"> 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>
```

To Become more comfortable - <u>Do Online Practice</u>

Short Quotations - The <q> Element :

The <q> element is intended to be used when you want to add a quote within a sentence rather than as an indented block on its own.

```
<p>Amit is in Spain, <q>He is their at my home. I think I am wrong</q>.</p>
```

This will produce following result:

Amit is in Spain, He is their at my home. I think I am wrong.

The <q> element can also carry the cite attribute. The value should be a URL pointing to the source of the quote.

To Become more comfortable - <u>Do Online Practice</u>

Citations - The <cite> Element :

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.

```
<p>This HTML Tutorial is derived from <cite>World Wide Web Standard for HTML</cite>.</p>
```

This will produce following result:

This HTML Tutorial is derived from *World Wide Web Standard for HTML*.

To Become more comfortable - <u>Do Online Practice</u>

Computer Code - The <code> Element :

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

```
<h1> <code>This is inside code element</code></h1>
```

This will produce following result:

This is inside code element

To Become more comfortable - <u>Do Online Practice</u>

Keyboard Text - The <kbd> Element :

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

The content of a <kbd> element is usually represented in a monospaced font rather like the content of the <code> element.

```
<h1> <kbd>This is inside kbd element</kbd></h1>
```

This will produce following result:

This is inside kbd element

To Become more comfortable - <u>Do Online Practice</u>

Programming Variables - The <var> Element :

This element is usually used in conjunction with the <pre> and <code> elements to indicate that the content of that element is a variable that can be supplied by a user.

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

This will produce following result:

document.write("*user-name*")

To Become more comfortable - <u>Do Online Practice</u>

Program Output - The <samp> Element :

The <samp> element indicates sample output from a program, script, or the like. Again, it is mainly used when documenting programming concepts. For example:

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

This will produce following result:

Result produced by the program is `Hello World`

To Become more comfortable - <u>Do Online Practice</u>

Addresses - The <address> Element :

The <address> element is used to contain any address. For example:

```
<address>304, Menna Colony, Hyderabad - INDIA, 500032</address>
```

This will produce following result:

304, Menna Colony, Hyderabad - INDIA, 500032

To Become more comfortable - **Do Online Practice**

Block and Inline Elements:

We can categories all the elements into two sections:

- **Block-level elements** - Block-level elements appear on the screen as if they have a carriage return or 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.

The elements which we have not discussed till now, will be discussed in subsequent chapters.

HTML Comments

Comments are piece of code which is ignored by any web browser. It is good practice to comment your 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.

HTML Comment lines are indicated by the special beginning tag <!-- and ending tag --> placed at the beginning and end of EVERY line to be treated as a comment.

Comments do not nest, and 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.

For example: Given line is a valid comment in HTML

```
<!--   This is commented out -->
```

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

```
< !--   This is commented out -->
```

Be careful if you use comments to "comment out" HTML that would otherwise be shown to the user, since some older browsers will still pay attention to angle brackets inside the comment and close the comment prematurely -- so that some of the text that was supposed to be inside the comment mistakenly appears as part of the document.

Multiline Comments:

You have seen how to comment a single line in HTML. You can comment multiple lines by the special beginning tag <!-- and ending tag --> placed before the first line and end of the lastline to be treated as a comment.

For example:

```
<!--
This is a multiline comment <br />
and can span through as many as lines you like.
-->
```

Conditional Comments :

Conditional comments only work in Explorer on Windows, and are thus excellently suited to give special instructions meant only for Explorer on Windows. They are supported from Explorer 5 onwards, and it is even possible to distinguish between 5.0, 5.5 and 6.0.

Conditional comments work as follows:

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

- Their basic structure is the same as an HTML comment (<!-- -->). Therefore all other browsers will see them as normal comments and will ignore them entirely.
- Explorer Windows, though, has been programmed to recognize the special <!--[if IE]> syntax, resolves the if and parses the content of the conditional comment as if it were normal page content.
- Since conditional comments use the HTML comment structure, they can only be included in HTML files, and not in CSS files.

Using Comment tag

There are few browsers who supports <comment> tag to comment a part of code.

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

Commenting Scripts and Style Sheets:

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 to make old browser works properly.

For example:

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

Similarly if you are using Casecading Style Sheet in your HTML code then it is recommended to put that style sheet code inside proper HTML Comments to make old browser works properly.

For example:

```
<style>
<!--
img{
  border:0px;
}
//-->
</style>
```

NOTE: To become familiar with JAVA Script and Cascading Style Sheet you need to refer different tutorial.

To Become more comfortable - <u>Do Online Practice</u>

HTML Fonts

Font face and color depends entirely on the computer and browser that is being used to view your page. But the tag is used to add style, size, and color to the text on your site. 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 page, simply use the tag. The text that follows will remain changed until you close with the tag. You can change any or all of the font attributes at the one time, by including all the required changes within the 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 it should not be used. Its is suggested to use css styles to manipulate your font.

Font Size:

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

Example:

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

This will produce following result:

Font size="1"
Font size="2"

Font size="3"

Font size="4"

Font size="5"

Font size="6"

Font size="7"

SPECIFY THE RELATIVE FONT SIZE. <font size="+n"> or <font size="-n">:
You can specify how many sizes larger or how many sizes smaller than the preset font size should be.

Example:

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

This will produce following result:

Font size="-1"

Font size="+1"

Font size="+2"

Font size="+3"

Font size="+4"

Font Face:

You can set any font you like 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 they will default to Times New Roman of your font with size attribute. See below few examples on using different font face

Example:

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

```
<font face="Bedrock" size="5">Bedrock</font>
```

This will produce following result:

Times New Roman

Verdana

Comic Sans MS

WildWest

Bedrock

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.

Example:

```
<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 that it has available. If none of your selections are installed....then it will display the default font face *Times New Roman*.

Check a complete list of HTML Standard Fonts.

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. Check a complete list of HTML Color Name with Codes.

Example:

```
<font color="#FF00FF">This text is hexcolor #FF00FF</font>
<font color="red">This text is red</font>
```

This will produce following result:

This text is hexcolor #FF00FF
This text is red

To Become more comfortable - Do Online Practice

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> element. You can then use the <font> elements to override the <basefont> settings.

The attributes that the <basefont> element takes are exactly the same as for the <font> element. You can also set the size of fonts relative to the size of the <basefont> by giving them a value of +1 for a size larger or -2 for two sizes smaller

NOTE: This element is deprecated in HTML 4 and will be removed from HTML, the preferred option is to use CSS styles. Your browser may not have support for this tag.

Example:

```
<basefont face="arial, verdana, sans-serif" size="2" color="#ff0000">
<p>This is the page's default font.</p>
<h2>Example of the &lt;basefont&gt; Element</h2>
<p><font size="+2" color="darkgray">Here is some darkgray text
two sizes larger</font></p>
<p><font face="courier" size="-1" color="#000000">Here is a courier
font, a size smaller, in black</font></p>
```

This will produce following result:

This is the page's default font.

Example of the <basefont> Element

Here   is   some   darkgray   text
two sizes larger

Here   is   a   courier
font, a size smaller, in black

As you can see, the default font now takes on the properties specified in the <basefont> element. It is red, size 2, and uses the Arial typeface.

The paragraph after the <h2> element uses a font size two sizes larger than the default size and is gray text, whereas the following paragraph uses a font one size smaller than the default font. You can also see that the color of this font is black (overriding the default).

HTML Images

Images are very important to beautify as well as to depicts many concepts on your web page. Its is true that one single image is worth than thuasands of words. So as a Web Developer you should have clear understanding on how to use images in your web pages.

Insert Image - The <img> Element:

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

```
<img src="image URL" attr_name="attr_value"...more attributes />
```

Image Attributes:

Following are most frequently used attributes for <img> tag.

- **width:** sets width of the image. This will have a value like 10 or 20%etc.
- **height:** sets height of the image. This will have a value like 10 or 20% etc.
- **border:** sets a border around the image. This will have a value like 1 or 2 etc.
- **src:** specifies URL of the image file.
- **alt:** this is an alternate text which will be displayed if image is missing.
- **align:** this sets horizontal alignment of the image and takes value either *left*, *right* or *center*.
- **valign:** this sets vertical alignment of the image and takes value either *top*, *bottom* or *center*.
- **hspace:** horizontal space around the image. This will have a value like 10 or 20%etc.
- **vspace:** vertical space around the image. This will have a value like 10 or 20%etc.
- **name:** name of the image with in the document.
- **id:** id of the image with in the document.
- **style:** this will be used if you are using CSS.
- **title:** specifies a text title. The browser, perhaps flashing the title when the mouse passes over the link.
- **ismap and usemap:** These attributes for the <img> tag tell the browser that the image is a special mouse-selectable visual map of one or more hyperlinks, commonly known as an **image map**. We will see how to use these attributes in Image Links chapter.

A Simple Example:

```
<img src="http://www.tutorialspoint.com/images/html.gif" alt="HTML Tutorial" />
```

This will produce following result:

Image Attributes - width, height, title, border and align:

Now let us try to set some more attributes:

```
<img src="http://www.tutorialspoint.com/images/html.gif"
alt="HTML Tutorial" width="100" height="100"
```

```
border="2" align="right" title="HTML Tutorial" />
```

This will produce following result:

Remember that all the images will have a border by default. In our examples its not showing because our global style sheet has set **img {border:0px;}** which means that no border will be displayed till it is mentioned explicitly.

You can remove an image border by setting **border="0"** or through CSS by setting **img {border:0px;}**.

To Become more comfortable with other image attributes - Do Online Practice

Wrapping text around images:

Example 1:

```
<p>This is the first paragraph that appears above the paragraph with the image!</p>

<p><img src="http://www.tutorialspoint.com/images/html.gif" width="75" height="75" alt="HTML Tutorial" align="right">

The image will appear along the right hand side of the paragraph. As you can see this is very nice for adding a little eye candy that relates to the specified paragraph.</p>

<p>The left and right image-alignment values tell the browser to place an image against the left or right margin, respectively, of the current text flow. The browser then renders subsequent document content in the remaining portion of the flow adjacent to the image. The net result is that the document content following the image gets wrapped around the image. </p>
```

This will produce following result:

This is the first paragraph that appears above the paragraph with the image!

The image will appear along the right hand side of the paragraph. As you can see this is very nice for adding a little eye candy that relates to the specified paragraph.

The left and right image-alignment values tell the browser to place an image against the left or right margin, respectively, of the current text flow. The browser then renders subsequent document content in the remaining portion of the flow adjacent to the image. The net result is that the document content following the image gets wrapped around the image.

Example 2:

You can use vspace or hspace attributes if you want to keep some distance between text and image. Let us revise above example:

```
<p>This is the first paragraph that appears above the paragraph with the image!</p>

<p><img src="http://www.tutorialspoint.com/images/html.gif" vspace="10" hspace="15"
width="75" height="75" alt="HTML Tutorial" align="right">

The image will appear along the right hand side of the paragraph. As you can see this is very
nice for adding a little eye candy that relates to the specified paragraph.</p>

<p>The left and right image-alignment values tell the browser to place an image against the
left or right margin, respectively, of the current text flow. The browser then renders subsequent
document content in the remaining portion of the flow adjacent to the image. The net result is
that the document content following the image gets wrapped around the image. </p>
```

This will produce following result:

This is the first paragraph that appears above the paragraph with the image!

The image will appear along the right hand side of the paragraph. As you can see this is very nice for adding a little eye candy that relates to the specified paragraph.

The left and right image-alignment values tell the browser to place an image against the left or right margin, respectively, of the current text flow. The browser then renders subsequent document content in the remaining portion of the flow adjacent to the image. The net result is that the document content following the image gets wrapped around the image.

For a complete list of image attributes please check reference to HTML Image Tag.

HTML Text Links

Web pages can contain 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 your any web page.

In this tutorial you will learn how to create text links between the different pages of your site, links within pages of your sites, and how to link to other sites (or external sites). If you want to know more about URL then check Understanding URL Tutorial.

Linking Documents - The <a> Element:

A link is specified using the <a> element. This element is called **anchor tag** as well. Anything between the opening <a> tag and the closing 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 this tag.

```
<a href="Document URL" attr_name="attr_value"...more attributes />
```

Anchor Attributes:

Following are most frequently used attributes for <a> tag.

- **href:** specifies the URL of the target of a hyperlink. Its value is any valid document URL, absolute or relative, including a fragment identifier or a JavaScript code fragment.
- **target:** specify where to display the contents of a selected hyperlink. If set to "_blank" then a new window will be opened to display the loaded page, if set to "_top" or "_parent" then same window will be used to display the loaded document, if set to "_self" then loads the new page in current window. By default its "_self".
- **name & id:** attributes places a label within a document. When that label is used in a link to that document, it is the equivalent of telling the browser to goto that label.
- **event:** attributes like *onClick*, *onMouseOver* etc. are used to trigger any Javascript ot VBscript code.
- **title:** attribute lets you specify a title for the document to which you are linking. The value of the attribute is any string, enclosed in quotation marks. The browser might use it when displaying the link, perhaps flashing the title when the mouse passes over the link.
- **accesskey:** attribute attribute provides a keyboard shortcut that can be used to activate a link. For example, you could make the T key an access key so that when the user presses either the Alt or Ctrl key on his keyboard (depending on his operating system) along with the T key, the link gets activated.

A Simple Example:

```
<a href="http://www.tutorialspoint.com/" target="_blank" >TP Home</a> |
<a href="http://www.amrood.com/" target="_self" >AMROOD Home</a> |
<a href="http://www.change-images.com/" target="_top" >Change Images Home</a>
```

This will produce following result, Click and come back to proceed with rest of the tutorial:

Tutorials Point | AMROOD | Change Images

Base Path for Links:

It is not required to give a complete URL for every link. You can get rid of it if you will use <base> tag in your 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.

For example we have used following base tag in all the pages at tutorialspoint.com:

```
<head>

<base href="http://www.tutorialspoint.com/">

</head>
```

So now if you will use **<a href="/html/index.htm"** then it will be considered as **<a href="http://www.tutorialspoint.com/html/index.htm"**.

Linking to a Page Section:

You can create a link to a particular section of a page by using *name* attribute. Here we will create three links with-in this page itself.

First create a link to reach to the top of this page. Here is the code we have used for the title heading *HTML Text Links*

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

Now you have a place where you can reach. To reach to this place use the following code with-in this document anywhere:

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

This will produce following link and you try using this link to reach to the top of this page:

Go to the Top

NOTE: Here we are using relative path. You can give complete URL and then # and then link name eg. http://www.tutorialspoint.com/html/html_text_links.htm#top

You can use this type of URL in any other page to reach directly to a particular section.

Setting Link Colors:

You can set colors of your links, active links and visited links using *link*, *alink* and *vlink* attributes of <body> tag. But it is recommended to use CSS to set colors of links, visited links and active links.

Following is the example we have used for our web side tutorialspoint.com

```
a:link     {color:#900B09; background-color:transparent}
a:visited  {color:#900B09; background-color:transparent}
a:active   {color:#FF0000; background-color:transparent}
a:hover    {color:#FF0000; background-color:transparent}
```

You can refer to Style Sheet Tutorial for a complete understanding on CSS.

Otherwise you can use <body> tag to set link colors. Here is the syntax.

```
<body alink="#FF0000" link="#900B09" vlink="#900B09">
.......
</body>
```

Create 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:

```
<a href="http://www.example.com/file.pdf">Download File</a>
```

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

Download File

You can not make an image download able until you follow the following procedure.

How To Raise a "File Download" Dialog Box ?

Sometime it is desired that you want to give option where a use will click a link and it will pop up a "File Download" box to the user in stead of displaying actual content. This is very easy and will be achived through HTTP header.

This HTTP header will be different from the header mentioned in previous section.

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

```perl
#!/usr/bin/perl

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

# Actual File Content will go hear.
open( FILE, "<FileName" );
while(read(FILE, $buffer, 100) )
{
   print("$buffer");
}
```

For more detail on CGI go through our another tutorial PERL and CGI.

HTML Image Links

Previous chapters has tought you how to create hyper text link using text and how to use images in your web page. Now we will learn how to use images to create hyper links. See example below:

```html
<a href="http://www.tutorialspoint.com/index.htm" target="_self" >
<img src="/images/home.gif" alt="Tutorials Point Home" border="0"/>
</a>
```

This will create following hyperlink at tutorialspoint.com home.

This was the simpletest 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 provide a feature that lets you embed many different links inside the same image. Clicking different areas of the image causes the browser to link to different target documents. Such mouse-sensitive images known as *image maps*.

There are two ways to create image maps:

- **A server-side image maps:** is enabled by the *ismap* attribute for the <img> tag and requires access to a server and related image-map processing applications.
- **A client-side image maps:** is created with the *usemap* attribute for the <img> tag, along with corresponding <map> and <area> tags.

Server-Side Image Maps:

You add an image to an anchor simply by placing an <img> tag within the body of the <a> tag. Make that embedded image into a mouse-sensitive one by adding the *ismap* attribute to the <img> tag. This special <img> attribute tells the browser that the image is a special map containing more than one link.

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 document 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 amap file or cgi 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 50 pixels over and 30 pixels down from the upper-left corner of the image displayed from the following link:

```
<a href="/cgi-bin/logo.map" target="_self" >
<img ismap src="/images/html.gif"
       alt="HTML" border="0"/>
</a>
```

Then the browser sends the following search parameters to the HTTP server which can be processed by cgi script or map file and you can link whatever you like to these coordinates:

```
/cgi-bin/logo.map?50,30
```

To Become more comfortable - Do Online Practice

NOTE: Converting the coordinates into a specific document is handled by the server side application, either cgi programme or special map files provided by seb server. For more detail Check Using HTML ismap.

Client-Side Image Maps:

Client side image maps are enabled by the *usemap* attribute for 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 /> element as normal, except it carries an extra attribute called usemap. The value of the usemap attribute is the value of the name attribute on the <map> element, which you are about to meet, preceded by a pound or hash sign.

The <map> element actually creates the map for the image and usually follows directly after the <img /> element. It acts as a container for the <area /> elements that actually define the clickable hotspots. The <map> element carries only one attribute, the name attribute, which is the name that identifies the map. This is how the <img /> element knows which <map> element to use.

The <area> element specifies the shape and the coordinates that define the boundaries of each clickable hotspot. Here's an example from the image map:

```
<img src=/images/html.gif alt="HTML Map" border="0" usemap="#html"/>

<!-- Create  Mappings -->
<map name="html">
   <area shape="circle"
         coords="154,150,59" href="link1.htm" alt="link 1"
                    target="_self" />
   <area shape="poly"
         coords="272,79,351,79,351,15,486,15,486,218,272,218,
                292,166,292,136,270,76" alt="link 2"
                    href="link2.htm" target="_self" />
   <area shape="rect"
         coords="325,224,488,286" alt="link 3"
                    href="link3.htm" target="_self" />
</map>
```

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. Therefore, a rectangle which goes from 10,5 to 20,25 would have the attribute *coords="10,5,20,25"*. A rectangle which defines the upper-left quarter of an image might use *coords="0,0,50%,50%"*.

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"*; one centered at the image's center and having a diameter of half the image would be defined by *coords="50%,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". A "line" is always drawn from the coordinates of the last point to the coordinates of the first point in order to close the polygon.

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.

To Become more comfortable - Do Online Practice

HTML Tables

Tables are very useful to arrange in HTML and they are used very frequently by almost all web developers. Tables are just like spreadsheets and they are made up of rows and columns.

You will create a table in HTML/XHTML by using <table> tag. Inside <table> element the table is written out row by row. A row is contained inside a <tr> tag . which stands for table row. And each cell is then written inside the row element using a <td> tag . which stands for table data.

Example:

```
<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>
```

This will produce following result:

Row 1, Column 1	Row 1, Column 2
Row 2, Column 1	Row 2, Column 2

NOTE: In the above example *border* is an attribute of <table> and it will put border across all the cells. If you do not need a border then you cal use *border="0"*. The border attribute and other attributes also mentione din this session are deprecated and they have been replaced by CSS. So it is recommended to use CSS instead of using any attribute directly.

To Become more comfortable - Do Online Practice

Table Heading - The <th> Element:

Table heading can be defined using <th> element. This tag will be put to replace <td> tag which is used to represent actual data. Normally you will put your top row as table heading as shown below, otherwise you can use <th> element at any place:

```
<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>
```

This will produce following result. You can see its making heading as a bold one:

Name	Salary
Ramesh Raman	5000
Shabbir Hussein	7000

To Become more comfortable - <u>Do Online Practice</u>

NOTE: Each cell must, however, have either a <td> or a <th> element in order for the table to display correctly even if that element is empty.

Table Cellpadding and Cellspacing:

There are two attribiutes called *cellpadding* and *cellspacing* which you will use to adjust the white space in your table cell. Cellspacing defines the width of the border, while cellpadding represents the distance between cell borders and the content within. Following is the example:

```
<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>
```

This will produce following result:

Name	Salary
Ramesh Raman	5000
Shabbir Hussein	7000

To Become more comfortable - Do Online Practice

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. Following is the example:

```
<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>
```

This will produce following result:

Column 1	Column 2	Column 3
Row 1 Cell 1	Row 1 Cell 2	Row 1 Cell 3
	Row 2 Cell 2	Row 2 Cell 3
Row 3 Cell 1		

To Become more comfortable - Do Online Practice

Tables Backgrounds

You can set table background using of the following two ways:

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

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

Here is an example of using *bgcolor* attribute:

```
<table border="5" bordercolor="green" bgcolor="gray">
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
</tr>
<tr><td rowspan="2">Row 1 Cell 1</td>
<td bgcolor="red">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>
```

This will produce following result:

Column 1	Column 2	Column 3
Row 1 Cell 1	Row 1 Cell 2	Row 1 Cell 3
	Row 2 Cell 2	Row 2 Cell 3
Row 3 Cell 1		

To Become more comfortable - <u>Do Online Practice</u>

Here is an example of using *background* attribute:

```
<table border="1" background="/images/home.gif">
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
</tr>
<tr><td rowspan="2">Row 1 Cell 1</td>
<td bgcolor="red">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" background="/images/pattern1.gif">
Row 3 Cell 1
</td></tr>
</table>
```

This will produce following result:

Column 1	Column 2	Column 3
Row 1 Cell 1	Row 1 Cell 2	Row 1 Cell 3
	Row 2 Cell 2	Row 2 Cell 3
Row 3 Cell 1		

To Become more comfortable - <u>Do Online Practice</u>

Table height and width:

You can set a table width and height using *width* and *height* attrubutes. You can specify table width or height in terms of integer value or in terms of percentage of available screen area. Following is the example:

```
<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>
```

This will produce following result:

Row 1, Column 1	Row 1, Column 2	
Row 2, Column 1	Row 2, Column 2	

To Become more comfortable - Do Online Practice

Using Table Caption:

The *caption* tags will serve as a title or explanation and show up at the top of the table. This tag is depracated in newer version of HTML/XHTML.

```
<table border="1">
<caption>This is the caption</caption>
<tr>
<td>row 1, column 1</td><td>row 1, columnn 2</td>
</tr>
</table>
```

This will produce following result:

This is the caption	
row 1, column 1	row 1, columnn 2

Using a 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 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>

```
<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>
<tr>
...more rows here containing four cells...
</tr>
</tbody>
<tbody>
<tr>
<td>Cell 1</td>
<td>Cell 2</td>
<td>Cell 3</td>
<td>Cell 4</td>
</tr>
<tr>
...more rows here containing four cells...
</tr>
</tbody>
</table>
```

This will produce following result:

This is the head of the table			
This is the foot of the table			
Cell 1	Cell 2	Cell 3	Cell 4
...more rows here containing four cells...			
Cell 1	Cell 2	Cell 3	Cell 4
...more rows here containing four cells...			

To Become more comfortable - Do Online Practice

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>.

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

```
<table border="1">
<tr>
<td>
        <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>
</td>
<td>
        <ul>
        <li>This is another cell</li>
        <li>Using list inside this cell</li>
        </ul>
</td>
</tr>
<tr>
<td>Row 2, Column 1</td>
<td>Row 2, Column 2</td>
</tr>
</table>
```

This will produce following result:

Name	Salary	• This is another cell • Using list inside this cell	
Ramesh Raman	5000		
Shabbir Hussein	7000		
Row 2, Column 1		Row 2, Column 2	

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.

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.

NOTE: It is recommended to use CSS to set background or text colors.

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 make 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 sepecify 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 Netscape and IE. 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

To Become more comfortable - Do Online Practice

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, Jasc Paintshop Pro or even using Advanced Paint Brush.

Each hexadecimal code will be preceded by a pound or hash sign #. Following are the examples to use Hexadecimal notation.

Color	Color HEX

	#000000
	#FF0000
	#00FF00
	#0000FF
	#FFFF00
	#00FFFF
	#FF00FF
	#C0C0C0
	#FFFFFF

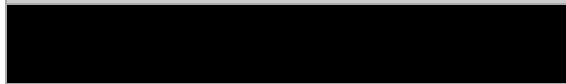To Become more comfortable - Do Online Practice

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 the example 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)

To Become more comfortable - Do Online Practice

Building Color Codes:

You can build millions of color codes using our Color Code Builder. Check our HTML Color Code Builder. To use this tool you would need a Java Enabled Browser.

Browser Safe Colors:

Here is the list of 216 colors which are supposed to be most safe and computer independent colors. These colors very from hexa code 000000 to FFFFFF. These color are safe to use because they ensure that all computers would display the colors correctly when running a 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	

HTML Forms

HTML Forms are required when you want to collect some data from the site visitor. For example registration information: name, email address, credit card, etc.

A form will take input from the site visitor and then will post your back-end application such as CGI, ASP Script or PHP script etc. Then your back-end application will do required processing on that data in whatever way you like.

Form elements are like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc. which are used to take information from the user.

A simple syntax of using <form> is as follows:

```
<form action="back-end script" method="posting method">
    form elements like input, textarea etc.
</form>
```

Most frequently used form attributes are:

- **name:** This is the name of the form.
- **action:** Here you will specify any script URL which will receive uploaded data.
- **method:** Here you will specify method to be used to upload data. It can take various values but most frequently used are GET and POST.
- **target:** It specifies the target page where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.
- **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 like:
    - **application/x-www-form-urlencoded** - This is the standard method most forms use. It converts spaces to the plus sign and non-alphanumeric characters into the hexadecimal code for that character in ASCII text.
    - **mutlipart/form-data** - This allows the data to be sent in parts, with each consecutive part corresponding the a form control, in the order they appear in the form. Each part can have an optional content-type header of its own indicating the type of data for that form control.

Please refer to PERL & CGI for a detail on data uploading using CGI.

There are different types of form controls that you can use to collect data from a visitor to your site.

- Text input controls
- Buttons
- Checkboxes and radio buttons
- Select boxes
- File select boxes
- Hidden controls
- Submit and reset button

HTML Forms - Text Input Controls:

There are actually three types of text input used on forms:

- **Single-line text input controls:** Used for items that require only one line of user input, such as search boxes or names. They are created using the <input> element.
- **Password input controls:** Single-line text input that mask the characters a user enters.
- **Multi-line text input controls:** Used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created with the <textarea> element.

Single-line text input controls:

Single-line text input controls are created using an <input> element whose type attribute has a value of text.

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

```
<form action="/cgi-bin/hello_get.cgi" method="get">
First name:
<input type="text" name="first_name" />
<br>
Last name:
<input type="text" name="last_name" />
<input type="submit" value="submit" />
</form>
```

This will produce following result:

First name:
Last name:
submit

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

- **type:** Indicates the type of input control you want to create. This element is also used to create other form controls such as radio buttons and checkboxes.
- **name:** Used to give the name part of the name/value pair that is sent to the server, representing each form control and the value the user entered.
- **value:** Provides an initial value for the text input control that the user will see when the form loads.
- **size:** Allows you to specify the width of the text-input control in terms of characters.
- **maxlength:** Allows you to specify the maximum number of characters a user can enter into the text box.

To Become more comfortable - Do Online Practice

Password input controls::

This is also a form of single-line text input controls are created using an <input> element whose type attribute has a value of password.

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

```
<form action="/cgi-bin/hello_get.cgi" method="get">
Login :
<input type="text" name="login" />
<br>
Password:
<input type="text" name="password" />
<input type="submit" value="submit" />
</form>
```

This will produce following result:

Login :

Password :

submit

To Become more comfortable - Do Online Practice
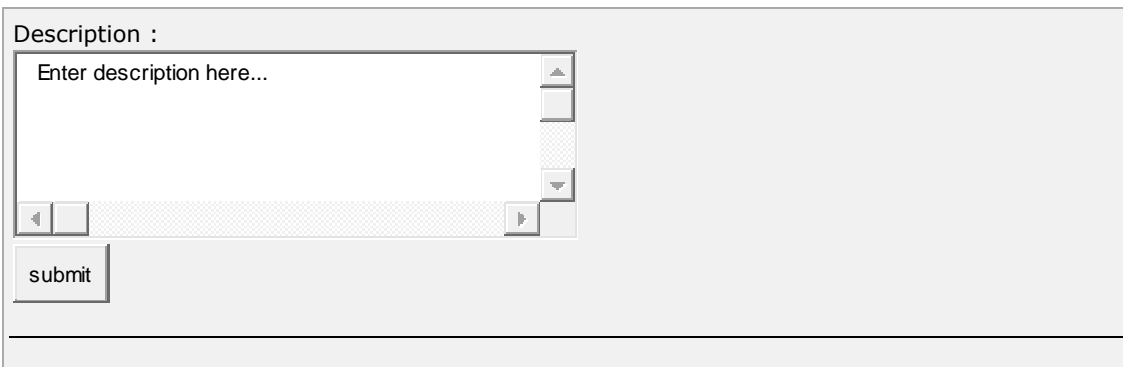
Multiple-Line Text Input Controls:

If you want to allow a visitor to your site to enter more than one line of text, you should create a multiple-line text input control using the <textarea> element.

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

```
<form action="/cgi-bin/hello_get.cgi" method="get">
Description : <br />
<textarea rows="5" cols="50" name="description">
Enter description here...
</textarea>
<input type="submit" value="submit" />
</form>
```

This will produce following result:

Description :

Enter description here...

submit

Following is the detail of above used attributes for <textarea> tag.

- **name:** The name of the control. This is used in the name/value pair that is sent to the server.
- **rows:** Indicates the number of rows of text area box.
- **cols:** Indicates the number of columns of text area box.

To Become more comfortable - <u>Do Online Practice</u>

HTML Forms - Creating Button:

There are various ways in HTML to create clickable buttons. You can create clickable button using <input> tag.

When you use the <input> element to create a button, the type of button you create is specified using the type attribute. The type attribute can take the following values:

- **submit:** This creates a button that automatically submits a form.
- **reset:** This creates a button that automatically resets form controls to their initial values.
- **button:** This creates a button that is used to trigger a client-side script when the user clicks that button.

Here is the example:

```
<form action="http://www.example.com/test.asp" method="get">
<input type="submit" name="Submit" value="Submit" />
<br /><br />
<input type="reset" value="Reset" />
<input type="button" value="Button"  />
</form>
```

This will produce following result:

```
Submit


Reset
```

You can use an image to create a button. Here is the syntax:

```
<form action="http://www.example.com/test.asp" method="get">
<input type="image" name="imagebutton" src="URL" />
</form>
```

Here *src* attribiute specifies a location of the image on your webserver.

You can use <button> element to create various buttons. Here is the syntax:

```
<form action="http://www.example.com/test.asp" method="get">
<button type="submit">Submit</button>
<br /><br />
<button type="reset"> Reset </button>
```

```
<button type="button"> Button </button>
</form>
```
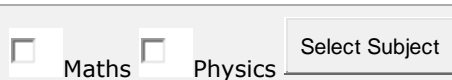
This will produce following result:

Submit

ResetButton

HTML Forms - Checkboxes Control:

Checkboxes are used when more than one option is required to be selected. They are created using <input> tag as shown below.

Here is example HTML code for a form with two checkboxes

```
<form action="/cgi-bin/checkbox.cgi" method="get">
<input type="checkbox" name="maths" value="on"> Maths
<input type="checkbox" name="physics" value="on"> Physics
<input type="submit" value="Select Subject" />
</form>
```

The result of this code is the following form

☐ Maths ☐ Physics Select Subject

Following is the list of important checkbox attributes:

- **type:** Indicates that you want to create a checkbox.
- **name:** Name of the control.
- **value:** The value that will be used if the checkbox is selected. More than one checkbox should share the same name only if you want to allow users to select several items from the same list.
- **checked:** Indicates that when the page loads, the checkbox should be selected.

To Become more comfortable - <u>Do Online Practice</u>

HTML Forms - Raidobox Control:

Radio Buttons are used when only one option is required to be selected. They are created using <input> tag as shown below:

Here is example HTML code for a form with two radio button:

```
<form action="/cgi-bin/radiobutton.cgi" method="post">
<input type="radio" name="subject" value="maths" /> Maths
<input type="radio" name="subject" value="physics" /> Physics
```

```
<input type="submit" value="Select Subject" />
</form>
```

The result of this code is the following form

Maths   Physics   Submit

Following is the list of important radiobox attributes:

- **type:** Indicates that you want to create a radiobox.
- **name:** Name of the control.
- **value:** Used to indicate the value that will be sent to the server if this option is selected.
- **checked:** Indicates that this option should be selected by default when the page loads.

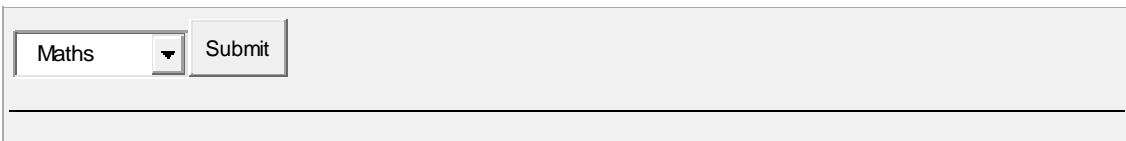To Become more comfortable - Do Online Practice

HTML Forms - Select box Control:

Drop Down Box is used when we have many options available to be selected but only one or two will be selected..

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

```
<form action="/cgi-bin/dropdown.cgi" method="post">
<select name="dropdown">
<option value="Maths" selected>Maths</option>
<option value="Physics">Physics</option>
</select>
<input type="submit" value="Submit" />
</form>
```

The result of this code is the following form

Maths   Submit

Following is the list of important attributes of <select>:

- **name:** This is the name for the control.
- **size:** This can be used to present a scrolling list box.
- **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>:

- **value:** The value that is sent to the server if this option is selected.
- **selected:** Specifies that this option should be the initially selected value when the page loads.

- **label:** An alternative way of labeling options.

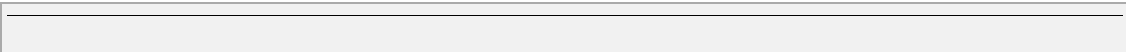To Become more comfortable - <u>Do Online Practice</u>

HTML Forms - File Select Boxes:

If you want to allow a user to upload a file to your web site from his computer, you will need to use a file upload box, also known as a file select box. This is also created using the <input> element.

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

```
<form action="/cgi-bin/hello_get.cgi" method="post"
      name="fileupload" enctype="multipart/form-data">
<input type="file" name="fileupload" accept="image/*" />
</form>
```

The result of this code is the following form

To Become more comfortable - <u>Do Online Practice</u>

HTML Forms - Hidden Controls:

If you will want to pass information between pages without the user seeing it. Hidden form controls remain part of any form, but the user cannot see them in the Web browser. They should not be used for any sensitive information you do not want the user to see because the user could see this data if she looked in the source of the page.

Following hidden form is being used to keep current page number. When a user will click next page then the value of hidden form will be sent to the back-end application and it will decide which page has be displayed next.

```
<form action="/cgi-bin/hello_get.cgi"
        method="get" name="pages">
<p>This is page 10</p>
<input type="hidden" name="pgaenumber" value="10" />
<input type="submit" value="Next Page" />
</form>
```

This will produce following result:

This is page 10

Next Page

To Become more comfortable - <u>Do Online Practice</u>

HTML Forms - Submit and Reset Button:

These are special buttons which can be created using <input> When submit button is clicked then Forms data is submitted to the back-end application. When reset button is clicked then all the forms control are reset to default state.

You already have seen submit button above, we will give one reset example here:

```
<form action="/cgi-bin/hello_get.cgi" method="get">
First name:
<input type="text" name="first_name" />
<br>
Last name:
<input type="text" name="last_name" />
<input type="submit" value="Submit" />
<input type="reset" value="Reset" />
</form>
```

This will produce following result. Type something and click reset button.

First name: [   ]

Last name: [   ]

[submit] [Reset]

For complete Tutorial: http://www.tutorialspoint.com/html

List of Tutorials from TutorialsPoint.com	
▪ **Learn JSP**	▪ **Learn ASP.Net**
▪ **Learn Servlets**	▪ **Learn HTML**
▪ **Learn log4j**	▪ **Learn HTML5**
▪ **Learn iBATIS**	▪ **Learn XHTML**
▪ **Learn Java**	▪ **Learn CSS**
▪ **Learn JDBC**	▪ **Learn HTTP**
▪ **Java Examples**	▪ **Learn JavaScript**
▪ **Learn Best Practices**	▪ **Learn jQuery**
▪ **Learn Python**	▪ **Learn Prototype**
▪ **Learn Ruby**	▪ **Learn script.aculo.us**
▪ **Learn Ruby on Rails**	▪ **Web Developer's Guide**
▪ **Learn SQL**	▪ **Learn RADIUS**
▪ **Learn MySQL**	▪ **Learn RSS**
▪ **Learn AJAX**	▪ **Learn SEO Techniques**
▪ **Learn C Programming**	▪ **Learn SOAP**
▪ **Learn C++ Programming**	▪ **Learn UDDI**
▪ **Learn CGI with PERL**	▪ **Learn Unix Sockets**
▪ **Learn DLL**	▪ **Learn Web Services**

- **Learn ebXML**
- **Learn Euphoria**
- **Learn GDB Debugger**
- **Learn Makefile**
- **Learn Parrot**
- **Learn Perl Script**
- **Learn PHP Script**
- **Learn Six Sigma**
- **Learn SEI CMMI**
- **Learn WiMAX**
- **Learn Telecom Billing**
- **Learn XML-RPC**
- **Learn UML**
- **Learn UNIX**
- **Learn WSDL**
- **Learn i-Mode**
- **Learn GPRS**
- **Learn GSM**
- **Learn WAP**
- **Learn WML**
- **Learn Wi-Fi**

webmaster@TutorialsPoint.com