05

Introduction to HTML

Glister E-Learning Center

Glister Eduversity

05

Contents/Index

Source/Reference:

https://www.phptpoint.com/html-tutorial-pdf/

www.tutorialspoint.com

HTML Introduction

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

Hypertext Markup Language (HTML) is the underlying markup language of the World Wide Web.

It’s the common thread that ties together virtually every Web site, from large scale corporate sites such as Microsoft’s to single-page classroom projects at the local grade school.

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

This tutorial gives a complete understanding on HTML and helps you to create exciting Websites.

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**ypertext **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.

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>
```

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 the entire 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 behavior 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 cannot 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.
- The <object> tag is designed to include images, JavaScript objects, Flash animations, MP3 files, QuickTime movies and other components of a page.
- The <link> tag is used to link to an external file, such as a style sheet or JavaScript file.
- The <style> tag is used to include CSS rules inside the document.
- The <script> tag is used to include JAVAScript or VBScript inside the document
- The <meta> tag includes information about the document such as keywords and a description, which are particularly helpful for search applications.

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" />
```

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>
```

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>
```

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";
```

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

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

Meta Tag Examples:

Let's see few important usages 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" />
```

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." />
```

Document Revision date:

This information tells about last time the document was updated.

```
<head>

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

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" />
```

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" />
```

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; />
```

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.

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" />
```

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 Cascading Style Sheet (CSS). So for now you can avoid it.

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

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

The style Attribute:

The style attribute allows you to specify 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>
```

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>
```

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

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

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>
```

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.
```

Create Line Breaks - The <br /> 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 />
```

This will produce following result:

```
Hello



You come most carefully upon your hour.

Thanks
```

Centering 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.
```

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 (&nbsp;) 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&nbsp;Angry&nbsp;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 (&shy;) 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 not work 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)
```

This will produce following result:

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

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 />
```

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

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 <b> Element:

Anything that appears in a <b>...</b> 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.
```

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.
```

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.
```

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.
```

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.
```

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.
```

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.
```

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.
```

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.
```

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 id="content" align="left" bgcolor="white">

<h5>Content Articles</h5>

<p>Actual content goes here.....</p>
```

This will produce following result:

```
HOME | CONTACT | ABOUT


Content Articles
```

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.

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.
```

Strong Text - The <strong> Element:

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

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

This will produce following result:

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

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.
```

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.

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*.

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
```

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>
```

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.

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*.

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
```

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
```

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")
```

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
```

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
```

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.

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://                                /images/html.gif"
alt="HTML Tutorial" width="100" height="100"
```

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

This will produce following result:

Create Web Pages
With
HTML
FOR THE WORLD WIDE WEB

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;}**.

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>
```

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

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>
```

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

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

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 </a> tag becomes part of the link and a user can click that part to reach to the linked document.

Following is the simple syntax to use this tag.

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

==============