HTML

Tutorial

!

Simply Easy Learning

www.tutorialspoint.com

SIMPLE EASY LEARNING

!

! ! !

!

!

!

! !

HTML stands for Hyper Text Markup Language, which is the most widely used language on Web to develop web pages.

HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

!

This tutorial is designed for the aspiring Web Designers and Developers with a need to understand the HTML in enough detail along with its simple overview, and practical examples. This tutorial will give you enough ingredients to start with HTML from where you can take yourself at higher level of expertise.

!

Before proceeding with this tutorial you should have a basic working knowledge with Windows or Linux operating system, additionally you must be familiar with:

- Experience with any text editor like notepad, notepad++, or Editplus etc.
- How to create directories and files on your computer.
- How to navigate through different directories.
- How to type content in a file and save them on a computer.
- Understanding about images in different formats like JPEG, PNG format.

!' ! ! !

©All the content and graphics on this tutorial are the property of tutorialspoint.com. Any content from tutorialspoint.com or this tutorial may not be redistributed or reproduced in any way, shape, or form without the written permission of tutorialspoint.com. Failure to do so is a violation of copyright laws.

This tutorial may contain inaccuracies or errors and tutorialspoint provides no guarantee regarding the accuracy of the site or its contents including this tutorial. If you discover that the tutorialspoint.com site or this tutorial content contains some errors, please contact us at webmaster@tutorialspoint.com

!

! ! !

HTML - Overview 17
Basic HTML Document 17
HTML Tags 18
HTML Document Structure 19
The <!DOCTYPE> Declaration 19
HTML Basic Tags 20
Heading Tags 20
Example 20
Paragraph Tag 21
Example 21
Line Break Tag 21
Example 22
Centering Content 22
Example 22
Horizontal Lines 23
Example 23
Preserve Formatting 23
Example 24
Nonbreaking Spaces 24
Example 24
HTML Elements 25
HTML Tag vs. Element 25
Nested HTML Elements 26
Example 26

HTML Attributes 27
Example 27
Core Attributes 28
The id Attribute 28
Example 28
The title Attribute 28
Example 28
The class Attribute 29
The style Attribute 29
Internationalization Attributes 30
The dir Attribute 30
Example 30
The lang Attribute 31
Example 31
The xml:lang Attribute 31
Generic Attributes 31
HTML Formatting 33
Bold Text 33
Example 33
Italic Text 33
Example 33
Underlined Text 34
Example 34
Strike Text 34
Example 34
Monospaced Font 35
Example 35
Superscript Text 35
Example 35
Subscript Text 36
Example 36
Inserted Text 36
Example 36

Deleted Text 36
Example 37
Larger Text 37
Example 37
Smaller Text 37
Example 37
Grouping Content 38
Example 38
Example 39
HTML Phrase Tags 40
Emphasized Text 40
Example 40
Marked Text 40
Example 40
Strong Text 41
Example 41
Text Abbreviation 41
Example 41
Acronym Element 42
Example 42
Text Direction 42
Example 42
Special Terms 43
Example 43
Quoting Text 43
Example 43
Short Quotations 44
Example 44
Text Citations 44
Example 44
Computer Code 45
Example 45
Keyboard Text 45

tutorialspoint
SIMPLYEASYLEARN

Example 45
Programming Variables 46
Example 46
Program Output 46
Example 46
Address Text 47
Example 47
HTML Meta Tags 48
Adding Meta Tags to Your Documents 48
Specifying Keywords 49
Example 49
Document Description 49
Example 49
Document Revision Date 50
Example 50
Document Refreshing 50
Example 50
Page Redirection 50
Example 51
Setting Cookies 51
Example 51
Setting Author Name 52
Example 52
Specify Character Set 52
Example 52
HTML Comments 54
Example 54
Valid vs Invalid Comments 54
Example 54
Multiline Comments 55
Example 55
Conditional Comments 56
Example 56

Using Comment Tag 56
Example 56
Commenting Script Code 57
Example 57
Commenting Style Sheets 57
Example 57
HTML Images 59
Insert Image 59

Example 69
Nested Tables 70
Example 70
HTML Lists 71
HTML Unordered Lists 71
Example 71
The type Attribute 72
Example 72
Example 72
Example 73
HTML Ordered Lists 73
Example 74
The type Attribute 74
Example 74
Example 75
Example 75
Example 76
Example 76
The start Attribute 77
Example 77
HTML Definition Lists 78
Example 78
HTML Text Links 79
Linking Documents 79
Example 79
The target Attribute 80
Example 80
Use of Base Path 81
Example 81
Linking to a Page Section 81
Setting Link Colors 82
Example 82
Download Links 82

File Download Dialog Box....83
HTML Image Links....84
Example....84
Mouse-Sensitive Images....84
Server-Side Image Maps....85
Client-Side Image Maps....86
Coordinate System....87
HTML Email Links....88
HTML Email Tag....88
Default Settings....89
HTML Frames....90
Disadvantages of Frames....90
Creating Frames....90
Example....90
Example....91
The <frameset> Tag Attributes....92
The <frame> Tag Attributes....93
Browser Support for Frames....94
Frame's name and target attributes....95
HTML IFrames....97
Example....97
The <Iframe> Tag Attributes....98
HTML Blocks....100
Block Elements....100
Inline Elements....100
Grouping HTML Elements....100
The tag....100
Example....100
The tag....101
Example....102
HTML Backgrounds....103
Html Background with Colors....103
Example....103

Html Background with Images 104
Example 104
Patterned & Transparent Backgrounds 105
Example 105
HTML Colors 107
HTML Color Coding Methods 107
HTML Colors - Color Names 107
W3C Standard 16 Colors 107
Example 108
HTML Colors - Hex Codes 108
Example 109
HTML Colors - RGB Values 109
Example 110
Browser Safe Colors 111
HTML Fonts 114
Set Font Size 114
Example 114
Relative Font Size 115
Example 115
Setting Font Face 116
Example 116
Specify alternate font faces 116
Setting Font Color 117
Example 117
The <basefont> Element: 117
Example 117
HTML Forms 119
Form Attributes 119
HTML Form Controls 120
Text Input Controls 120
Single-line text input controls 121
Example 121
Attributes 121

Password input controls..... 122
Example 122
Attributes 122
Multiple-Line Text Input Controls 123

tutorialspoint

HTML Marquees 136
Syntax 136
The <marquee> Tag Attributes 136
Example – 1 137
Example – 2 138
Example – 3 138
Example – 4 138
HTML Header 140
The HTML <title> Tag 140
The HTML <meta> Tag 141
The HTML <base> Tag 141
The HTML <link> Tag 142
The HTML <style> Tag 143
The HTML <script> Tag 143
HTML Stylesheet 145
Example 145
External Style Sheet 146
Example 146
Internal Style Sheet 147
Example 147
Inline Style Sheet 148
Example 148
HTML Javascript 149
External Javascript 149
Example 149
Internal Script 150
Example 150
Event Handlers 151
Hide Scripts from Older Browsers 152
The <noscript> Element 152
Default Scripting Language 153
HTML Layouts 154
HTML Layout - Using Tables 154

Example 154
Multiple Columns Layout - Using Tables 155
Example 155
HTML Layouts - Using DIV, SPAN 156

HTML URL Encoding 227
Example 227
ASCII control characters encoding 228
Non-ASCII control characters encoding 230
Reserved characters encoding 237
Unsafe characters encoding 238
Language ISO Codes 240
Language Codes: ISO 639, Microsoft 240
Language Codes: ISO 639, Macintosh 249
HTML Character Encodings 260
HTML Deprecated Tags 263
HTML Deprecated Attributes 264

!

!

!

!.!

CHAPTER 1

HTML stands for _yper_ext _arkup _anguage, and it is the most widely used language to write Web Pages.

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

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

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

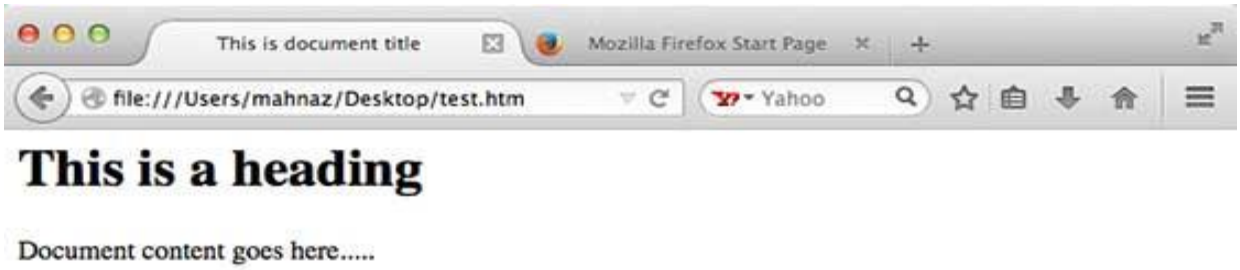
! ! !

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

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

Let's save it in an HTML file / using your favorite text editor. Finally open it using a web browser like Internet Explorer or Google Chrome, or Firefox etc. It must show the following output:

tutorialspoint



! !

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

Above example of HTML document uses following tags:

!	!
<!DOCTYPE...>	This tag defines the document type and HTML version.
<html>	This tag encloses the complete HTML document and mainly comprises of document header which is represented by /// and document body which is represented by /// tags.
<head>	This tag represents the document's header which can keep other HTML tags like <title>, <link> etc.
<title>	The tag is used inside the <head> tag to mention the document title.
<body>	This tag represents the document's body which keeps other HTML tags like <h1>, , <p> etc.

tutorialspoint

<h1>	This tag represents the heading.
<p>	This tag represents a paragraph.

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

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

! ! !

A typical HTML document will have following structure:

```
Document declaration tag
<html>
   <head>
      Document header related tags
   </head>

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

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

! " ! !

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

```
<!DOCTYPE html>
```

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

CHAPTER 2

!

! ! !

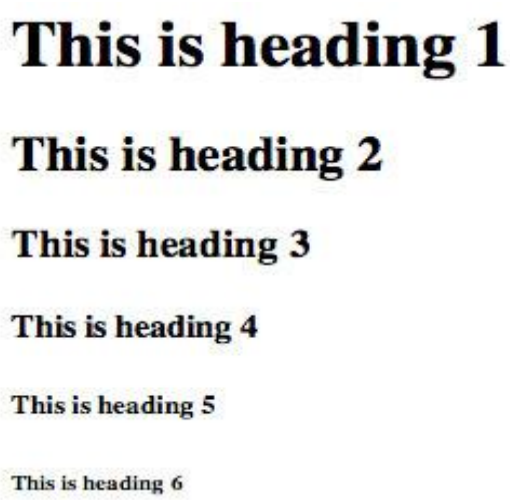
! !

Any document starts with a heading. You can use different sizes for your headings. HTML also have six levels of headings, which use the elements **2** -! -! -! -! -! ! . While displaying any heading, browser adds one line before and one line after that heading.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Heading Example</title>
</head>
<body>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>
</body>
</html>
```

This will produce following result:



!

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Paragraph Example</title>
</head>
<body>
<p>Here is a first paragraph of text.</p>
<p>Here is a second paragraph of text.</p>
<p>Here is a third paragraph of text.</p>
</body>
</html>
```

This will produce following result:

Here is a first paragraph of text.

Here is a second paragraph of text.

Here is a third paragraph of text.

! ! !

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

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Line Break Example</title>
</head>
<body>
<p>Hello<br />
You delivered your assignment on time.<br />
Thanks<br />
Mahnaz</p>
</body>
</html>
```

This will produce following result:

Hello
You delivered your assignment ontime.
Thanks
Mahnaz

! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Centring Content Example</title>
</head>
<body>
<p>This text is not in the center.</p>
<center>
<p>This text is in the center.</p>
</center>
</body>
</html>
```

This will produce following result:

This text is not in the center.

This text is in the center.

! !

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

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Horizontal Line Example</title>
</head>
<body>
<p>This is paragraph one and should be on top</p>
<hr />
<p>This is paragraph two and should be at bottom</p>
</body>
</html>
```

This will produce following result:

This is paragraph one and should be on top

This is paragraph two and should be at bottom

Again ! tag is an example of the element, where you do not need opening and closing tags, as there is nothing to go in between them.

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

! !

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 .

Any text between the opening tag and the closing tag will preserve the formatting of the source document.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Preserve Formatting Example</title>
</head>
<body>
<pre>
function testFunction( strText ){
   alert (strText)
}
</pre>
</body>
</html>
```

This will produce following result:

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

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

! !

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

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

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Nonbreaking Spaces Example</title>
</head>
<body>
<p>An example of this technique appears in the movie "12 Angry Men."</p>
</body>
</html>
```

CHAPTER 3

!

! !

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

! !	!	! !
	This is paragraph content.	
<h1>	This is heading content.	</h1>
	This is division content.	

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

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

! ! / !

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

For example is starting tag of a paragraph and is closing tag of the same paragraph but ! ! is a paragraph element.

tutorialspoint

! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Nested Elements Example</title>
</head>
<body>
<h1>This is <i>italic</i> heading</h1>
<p>This is <u>underlined</u> paragraph</p>
</body>
</html>
```

This will display following result:

This is *italic* heading

This is underlined paragraph

! !

tutorialspoint

CHAPTER 4

!

! !

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

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

- The name is the property you want to set. For example, the paragraph <p> element in the example carries an attribute whose name is , which you can use to indicate the alignment of paragraph on the page.
- The value is what you want the value of the property to be set and always put within quotations. The below example shows three possible values of align attribute: -! and .

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Align Attribute Example</title>
</head>
<body>
<p align="left">This is left aligned</p>
<p align="center">This is center aligned</p>
<p align="right">This is right aligned</p>
</body>
</html>
```

This will display following result:

This is left aligned

This is center aligned

This is right aligned

! !

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

- id
- title
- class
- style

! ! !

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

- If an element carries an id attribute as a unique identifier it is possible to identify just that element and its content.
- If you have two elements of the same name within a Web page (or style sheet), you can use the id attribute to distinguish between elements that have the same name.

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

!

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

!

! ! !

The attribute gives a suggested title for the element. The syntax for the !attribute is similar as explained for attribute:

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>The title Attribute Example</title>
```

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

This will produce following result:

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

! ! !

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

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

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

! ! !

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

```
<!DOCTYPE html>
<html>
<head>
<title>The style Attribute</title>
</head>
<body>
<p style="font-family:arial; color:#FF0000;">Some text...</p>
</body>
</html>
```

This will produce following result:

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

! !

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

- dir
- lang
- xml:lang

!

! ! !

The 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:

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

!

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

This will produce following result:

This is how IE 5 renders right-to-left directed text.

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.

id	User Defined	Names an element for use with Cascading Style Sheets.
class	User Defined	Classifies an element for use with Cascading Style Sheets.
width	Numeric Value	Specifies the width of tables, images, or table cells.
height	Numeric Value	Specifies the height of tables, images, or table cells.
title	User Defined	"Pop-up" title of the elements.

We will see related examples as we will proceed to study other HTML tags.

!

CHAPTER 5

! ! !

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

! !

Anything that appears within /// element is displayed in bold as shown below:

!

```
<! DOCTYPE html>
<html>
<head>
<title>Bold Text Example</title>
</head>
<body>
<p>The following word uses a <b>bold</b> typeface.</p>
</body>
</html>
```

This will produce following result:

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

! !

Anything that appears within /// element is displayed in italicized as shown below:

!

```
<!DOCTYPE html>
<html>
<head>
<title>Italic Text Example</title>
</head>
<body>
<p>The following word uses a <i>italicized</i> typeface.</p>
</body>
```

```
</html>
```

This will produce following result:

The following word uses a *italicized* typeface.

! !

Anything that appears within /// element is displayed with underline as shown below:

!

```
<!DOCTYPE html>
<html>
<head>
<title>Underlined Text Example</title>
</head>
<body>
<p>The following word uses a <u>underlined</u> typeface.</p>
</body>
</html>
```

This will produce following result:

The following word uses a underlined typeface.

! !

Anything that appears within /// element is displayed with strikethrough, which is a thin line through the text as shown below:

!

```
<!DOCTYPE html>
<html>
<head>
<title>Strike Text Example</title>
</head>
<body>
<p>The following word uses a <strike>strikethrough</strike> typeface.</p>
</body>
</html>
```

This will produce following result:

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

! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Monospaced Font Example</title>
</head>
<body>
<p>The following word uses a <tt>monospaced</tt> typeface.</p>
</body>
</html>
```

This will produce following result:

The following word uses a monospaced typeface.

! !

The content of a 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.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Superscript Text Example</title>
</head>
<body>
<p>The following word uses a <sup>superscript</sup> typeface.</p>
</body>
</html>
```

This will produce following result:

The following word uses a superscript typeface.

! !

The content of a 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.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Subscript Text Example</title>
</head>
<body>
<p>The following word uses a <sub>subscript</sub> typeface.</p>
</body>
</html>
```

This will produce following result:

The following word uses a subscript typeface.

! !

Anything that appears with-in /// element is displayed as inserted text.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Inserted Text Example</title>
</head>
<body>
<p>I want to drink <del>cola</del> <ins>wine</ins></p>
</body>
</html>
```

This will produce following result:

I want to drink ~~cola~~ wine

! !

Anything that appears within /// element is displayed as deleted text.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Deleted Text Example</title>
</head>
<body>
<p>I want to drink <del>cola</del> <ins>wine</ins></p>
</body>
</html>
```

This will produce following result:

I want to drink ~~cola~~ wine

! !

The content of the element is displayed one font size larger than the rest of the text surrounding it as shown below:

!

```
<!DOCTYPE html>
<html>
<head>
<title>Larger Text Example</title>
</head>
<body>
<p>The following word uses a <big>big</big> typeface.</p>
</body>
</html>
```

This will produce following result:

The following word uses a big typeface.

! !

The content of the element is displayed one font size smaller than the rest of the text surrounding it as shown below:

!

```
<!DOCTYPE html>
<html>
<head>
<title>Smaller Text Example</title>
```

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

This will produce following result:

The following word uses a small typeface.

! !

The and 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 element to indicate that all of the elements within that element relate to the footnotes. You might then attach a style to this element so that they appear using a special set of style rules.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Div Tag Example</title>
</head>
<body>
<div id="menu" align="middle" >
<a href="/index.htm">HOME</a> |
<a href="/about/contact_us.htm">CONTACT</a> |
<a href="/about/index.htm">ABOUT</a>
</div>

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

This will produce following result:

HOME | CONTACT | ABOUT

CONTENT ARTICLES

Actual content goes here.....

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Span Tag Example</title>
</head>
<body>
<p>This is the example of <span style="color:green">span tag</span> and the <span style="co
lor:red">div tag</span> along with CSS</p>
</body>
</html>
```

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.

tutorialspoint

CHAPTER 6

!

! ! !

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

! !

Anything that appears within /// element is displayed as emphasized text.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Emphasized Text Example</title>
</head>
<body>
<p>The following word uses a <em>emphasized</em> typeface.</p>
</body>
</html>
```

This will produce following result:

The following word uses a *emphasized* typeface.

! !

Anything that appears within /// element, is marked with yellow ink.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Marked Text Example</title>
</head>
```

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

This will produce following result:

The following word has been marked with yellow.

! !

Anything that appears within /// element is displayed as important text.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Strong Text Example</title>
</head>
<body>
<p>The following word uses a <strong>strong</strong> typeface.</p>
</body>
</html>
```

This will produce following result:

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

! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Text Abbreviation</title>
</head>
<body>
<p>My best friend's name is <abbr title="Abhishek">Abhy</abbr>.</p>
</body>
</html>
```

This will produce following result:

My best friend's name is Abhy.

! !

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

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Acronym Example</title>
</head>
<body>
<p>This chapter covers marking up text in <acronym>XHTML</acronym>.</p>
</body>
</html>
```

This will produce following result:

This chapter covers marking up text in XHTML.

! !

The element stands for Bi-Directional Override and it is used to override the current text direction.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Text Direction Example</title>
</head>
<body>
<p>This text will go left to right.</p>
<p><bdo dir="rtl">This text will go right to left.</bdo></p>
</body>
</html>
```

This will produce following result:

This text will go left to right.
.tfel ot thgir og lliw txet sihT

! !

The / element (or HTML Definition Element) allows you to specify that you are introducing a special term. Its usage is similar to italic words in the midst of a paragraph.

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Special Terms Example</title>
</head>
<body>
<p>The following word is a <dfn>special</dfn> term.</p>
</body>
</html>
```

This will produce following result:

The following word is a *special* term.

! !

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

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Blockquote Example</title>
</head>
<body>
<p>The following description of XHTML is taken from the W3C Web site:</p>
<blockquote>XHTML 1.0 is the W3C's first Recommendation for XHTML, following on from earlie
r work on HTML 4.01, HTML 4.0, HTML 3.2 and HTML 2.0.</blockquote>
</body>
```

```
</html>
```

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.

! !

The element is used when you want to add a double quote within a sentence.

!

!

```
<!DOCTYPE html>
<html>
<head>
<title>Double Quote Example</title>
</head>
<body>
<p>Amit is in Spain, <q>I think I am wrong</q>.</p>
</body>
</html>
```

This will produce following result:

Amit is in Spain, "I think I am wrong".

! !

If you are quoting a text, you can indicate the source placing it between an opening tag and closing tag.

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

!

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

```
<head>
<title>Citations Example</title>
</head>
<body>
<p>This HTML tutorial is derived from <cite>W3 Standard for HTML</cite>.</p>
</body>
</html>
```

This will produce following result:

This HTML tutorial is derived from *W3 Standard for HTML*.

! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Computer Code Example</title>
</head>
<body>
<p>Regular text. <code>This is code.</code> Regular text.</p>
</body>
</html>
```

This will produce following result:

Regular text. `This is code.` Regular text.

! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Keyboard Text Example</title>
</head>
<body>
<p>Regular text. <kbd>This is inside kbd element</kbd> Regular text.</p>
</body>
```

```
</html>
```

This will produce following result:

Regular text. This is inside kbd element Regular text.

! !

This element is usually used in conjunction with the and elements to indicate that the content of that element is a variable.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Variable Text Example</title>
</head>
<body>
<p><code>document.write("<var>user-name</var>")</code></p>
</body>
</html>
```

This will produce following result:

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

! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Program Output Example</title>
</head>
<body>
<p>Result produced by the program is <samp>Hello World!</samp></p>
</body>
</html>
```

This will produce following result:

```
Result produced by the program is Hello World!
```

! !

The element is used to contain any address.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Address Example</title>
</head>
<body>
<address>388A, Road No 22, Jubilee Hills -  Hyderabad</address>
</body>
</html>
```

This will produce following result:

388A, Road No 22, Jubilee Hills - Hyderabad

CHAPTER 7

!

! ! !

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

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

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

! ! ! ! ! !

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

!	!
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 interpret the property's value (as declared in the content attribute).

tutorialspoint

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.

! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

This will produce following result:

```
Hello HTML5!
```

! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="revised" content="Tutorialspoint, 3/7/2014" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

!

! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="revised" content="Tutorialspoint, 3/7/2014" />
<meta http-equiv="refresh" content="5" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

!

! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="revised" content="Tutorialspoint, 3/7/2014" />
<meta http-equiv="refresh" content="5" url=http://www.tutorialspoint.com" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

!

! !

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

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="revised" content="Tutorialspoint, 3/7/2014" />
<meta http-equiv="cookie" content="userid=xyz; expires=Wednesday, 08-Aug-15 23:59:59 GMT;"
/>
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

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

!You can check **PHP and Cookies** tutorial for a complete detail on Cookies.

! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta name="author" content="Mahnaz Mohtashim" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

!

! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Meta Tags Example</title>
<meta name="keywords" content="HTML, Meta Tags, Metadata" />
<meta name="description" content="Learning about Meta Tags." />
<meta ame="author" content="Mahnaz Mohtashim" />
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

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

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

```
<meta http-equiv="Content-Type" content="text/html; charset=Big5" />
</head>
<body>
<p>Hello HTML5!</p>
</body>
</html>
```

CHAPTER 8

!

! !

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

HTML comments are placed in between "..! ///.. tag. So any content placed within <!-- ... --> tag will be treated as comment and will be completely ignored by the browser.

!

```
<!DOCTYPE html>
<html>
<head>  <!-- Document Header Starts -->
<title>This is document title</title>
</head> <!-- Document Header Ends -->
<body>
<p>Document content goes here.....</p>
</body>
</html>
```

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

Document content goes here.....

! ! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Valid Comment Example</title>
</head>
<body>
<!--   This is valid comment -->
<p>Document content goes here.....</p>
</body>
</html>
```

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

```
<!DOCTYPE html>
<html>
<head>
<title>Invalid Comment Example</title>
</head>
<body>
< !--   This is not a valid comment -->
<p>Document content goes here.....</p>
</body>
</html>
```

This will produce following result:

< !-- This is not a valid comment -->

Document content goes here.....

! !

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

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

!

```
<!DOCTYPE html><html>
<head>
<title>Multiline Comments</title>
</head>
<body>
<!--
This is a multiline comment and it can
span through as many as lines you like.
-->
<p>Document content goes here.....</p>
```

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

This will produce following result:

Document content goes here.....

! !

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

!

```
<!DOCTYPE html><html>
<head>
<title>Conditional Comments</title>

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

</head>
<body>
<p>Document content goes here.....</p>
</body>
</html>
```

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

! ! !

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

!

```
<!DOCTYPE html><html>
<head>
<title>Using Comment Tag</title>
</head>
<body>
<p>This is <comment>not</comment> Internet Explorer.</p>
</body>
</html>
```

If you are using IE then it will produce following result:

```
This is Internet Explorer.
```

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

```
This is not Internet Explorer.
```

! ! !

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

!

```
<!DOCTYPE html><html>
<head>
<title>Commenting Script Code</title>
<script>
<!--
   document.write("Hello World!")
//-->
</script>
</head>
<body>
<p>Hello , World!</p>
</body>
</html>
```

This will produce following result:

```
Hello World!
Hello , World!
```

! ! !

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

!

```
<!DOCTYPE html><html>
<head>
<title>Commenting Style Sheets</title>
```

```
<style>
<!--
.example {
  border:1px solid #4a7d49;
}
//-->
</style>
</head>
<body>
<div class="example">Hello , World!</div>
</body>
</html>
```

This will produce following result:

Hello , World!

CHAPTER 9

!

! !

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

! !

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

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

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Using Image in Webpage</title>
</head>
<body>
<p>Simple Image Insert</p>
<img src="test.png" alt="Test Image" />
</body>
</html>
```

This will produce following result:

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

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

! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Using Image in Webpage</title>
</head>
<body>
<p>Simple Image Insert</p>
<img src="images/test.png" alt="Test Image" />
</body>
</html>
```

This will produce following result:

Simple Image Insert

!

! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Set Image Width and Height</title>
```

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

This will produce following result:

Setting image width and height

! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Set Image Border</title>
</head>
<body>
<p>Setting image Border</p>
<img src="test.png" alt="Test Image" border="3"/>
</body>
</html>
```

This will produce following result:

Setting image Border

! ! !

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

! !

```
<!DOCTYPE html>
<html>
<head>
<title>Set Image Alignment</title>
</head>
<body>
<p>Setting image Alignment</p>
<img src="test.png" alt="Test Image" border="3" align="right"/>
</body>
</html>
```

This will produce following result:

Setting image Alignment

! ! !

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

!

! !

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

The HTML tables are created using the tag in which the tag is used to create table rows and the tag is used to create data cells.

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Tables</title>
</head>
<body>
```

your top row as table heading as shown below, otherwise you can use <th> element in any row.

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table Header</title>
</head>
<body>
<table border="1">
<tr>
<th>Name</th>
<th>Salary</th>
</tr>
<tr>
<td>Ramesh Raman</td>
<td>5000</td>
</tr>
<tr>
<td>Shabbir Hussein</td>
<td>7000</td>
</tr>
</table>
</body>
</html>
```

This will produce following result:

Name	Salary
Ramesh Raman	5000
Shabbir Hussein	7000

!

! ! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table Cellpadding</title>
</head>
<body>
```

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

This will produce following result:

Name	Salary
Ramesh Raman	5000
Shabbir Hussein	7000

! ! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table Colspan/Rowspan</title>
</head>
<body>
<table border="1">
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
</tr>
<tr><td rowspan="2">Row 1 Cell 1</td><td>Row 1 Cell 2</td><td>Row 1 Cell 3</td></tr>
<tr><td>Row 2 Cell 2</td><td>Row 2 Cell 3</td></tr>
<tr><td colspan="3">Row 3 Cell 1</td></tr>
</table>
</body>
</html>
```

This will produce 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		

!

! !

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

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

You can also set border color also using attribute.

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table Background</title>
</head>
<body>
<table border="1" bordercolor="green" bgcolor="yellow">
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
</tr>
<tr><td rowspan="2">Row 1 Cell 1</td><td>Row 1 Cell 2</td><td>Row 1 Cell 3</td></tr>
<tr><td>Row 2 Cell 2</td><td>Row 2 Cell 3</td></tr>
<tr><td colspan="3">Row 3 Cell 1</td></tr>
</table>
</body>
</html>
```

This will produce 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		

tutorialspoint

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table Background</title>
</head>
<body>
<table border="1" bordercolor="green" background="/images/test.png">
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
</tr>
<tr><td rowspan="2">Row 1 Cell 1</td><td>Row 1 Cell 2</td><td>Row 1 Cell 3</td></tr>
<tr><td>Row 2 Cell 2</td><td>Row 2 Cell 3</td></tr>
<tr><td colspan="3">Row 3 Cell 1</td></tr>
</table>
</body>
</html>
```

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

!

! ! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table Width/Height</title>
</head>
<body>
<table border="1" width="400" height="150">
<tr>
<td>Row 1, Column 1</td>
<td>Row 1, Column 2</td>
```

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

This will produce following result:

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

! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table Caption</title>
</head>
<body>
<table border="1" width="100%">
<caption>This is the caption</caption>
<tr>
<td>row 1, column 1</td><td>row 1, columnn 2</td>
</tr>
<tr>
<td>row 2, column 1</td><td>row 2, columnn 2</td>
</tr>
</table>
</body>
</html>
```

This will produce following result:

This is the caption

row 1, column 1	row 1, columnn 2
row 2, column 1	row 2, columnn 2

! -! -! ! !

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

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

- !. to create a separate table header.
- !. to indicate the main body of the table.
- !. 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>

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table</title>
</head>
<body>
<table border="1" width="100%">
<thead>
<tr>
<td colspan="4">This is the head of the table</td>
</tr>
</thead>
<tfoot>
<tr>
<td colspan="4">This is the foot of the table</td>
</tr>
</tfoot>
<tbody>
<tr>
<td>Cell 1</td>
<td>Cell 2</td>
<td>Cell 3</td>
<td>Cell 4</td>
</tr>
</tbody>
</table>
</body>
</html>
```

This will produce following result:

This is the head of the table			
Cell 1	Cell 2	Cell 3	Cell 4
This is the foot of the table			

! !

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.

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

This will produce following result:

Name	Salary
Ramesh Raman	5000
Shabbir Hussein	7000

CHAPTER

11

! ! !

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

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

! ! !

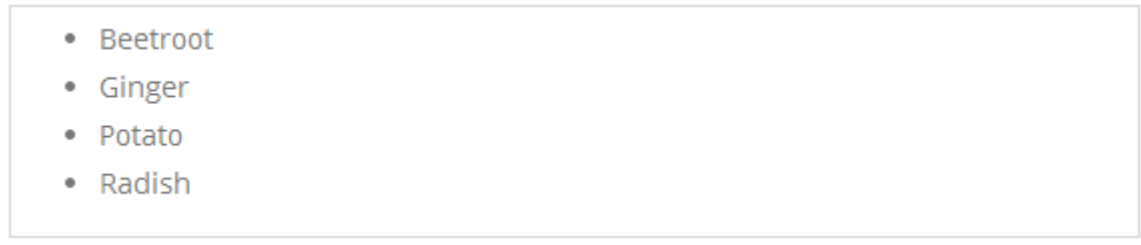
An unordered list is a collection of related items that have no special order or sequence. This list is created by using HTML tag. Each item in the list is marked with a bullet.

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Unordered List</title>
</head>
<body>
<ul>
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ul>
</body>
</html>
```

This will produce following result:

tutorialspoint



! ! !

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

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

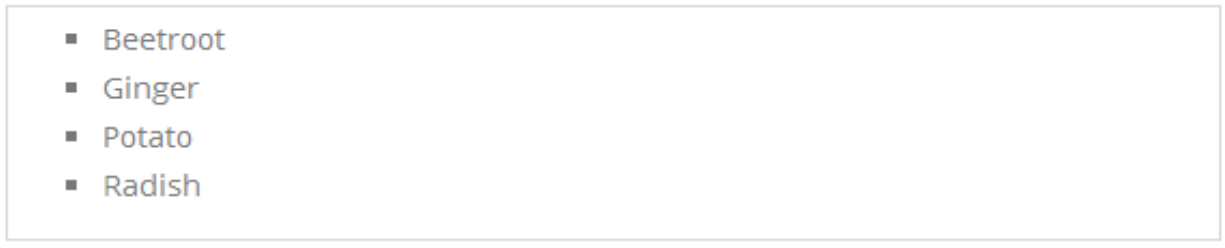
!

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Unordered List</title>
</head>
<body>
   <ul type="square">
   <li>Beetroot</li>
   <li>Ginger</li>
   <li>Potato</li>
   <li>Radish</li>
   </ul>
</body>
</html>
```

This will produce following result:



!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Unordered List</title>
</head>
```

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

This will produce following result:

- Beetroot
- Ginger
- Potato
- Radish

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Unordered List</title>
</head>
<body>
   <ul type="circle">
   <li>Beetroot</li>
   <li>Ginger</li>
   <li>Potato</li>
   <li>Radish</li>
   </ul>
</body>
</html>
```

This will produce following result:

- Beetroot
- Ginger
- Potato
- Radish

! ! !

If you are required to put your items in a numbered list instead of bulleted list then HTML ordered list will be used. This list is created by using tag.

The numbering starts at one and is incremented by one for each successive ordered list element tagged with <li>.

!

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Ordered List</title>
</head>
<body>
<ol>
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ol>
</body>
</html>
```

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

! ! !

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

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

!

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Ordered List</title>
</head>
<body>
   <ol type="1">
   <li>Beetroot</li>
```

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

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Ordered List</title>
</head>
<body>
   <ol type="I">
   <li>Beetroot</li>
   <li>Ginger</li>
   <li>Potato</li>
   <li>Radish</li>
   </ol>
</body>
</html>
```

This will produce following result:

I. Beetroot
II. Ginger
III. Potato
IV. Radish

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Ordered List</title>
</head>
```

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

This will produce following result:

i. Beetroot
ii. Ginger
iii. Potato
iv. Radish

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Ordered List</title>
</head>
<body>
   <ol type="A">
   <li>Beetroot</li>
   <li>Ginger</li>
   <li>Potato</li>
   <li>Radish</li>
   </ol>
</body>
</html>
```

This will produce following result:

A. Beetroot
B. Ginger
C. Potato
D. Radish

!

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

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

```
<head>
<title>HTML Ordered List</title>
</head>
<body>
   <ol type="a">
   <li>Beetroot</li>
   <li>Ginger</li>
   <li>Potato</li>
   <li>Radish</li>
   </ol>
</body>
</html>
```

This will produce following result:

a. Beetroot
b. Ginger
c. Potato
d. Radish

! ! !

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

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

!

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Ordered List</title>
</head>
<body>
   <ol type="i" start="4">
   <li>Beetroot</li>
   <li>Ginger</li>
   <li>Potato</li>
   <li>Radish</li>
   </ol>
</body>
</html>
```

This will produce following result

iv. Beetroot
v. Ginger
vi. Potato
vii. Radish

! ! !

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

Definition List makes use of following three tags.

- !. Defines the start of the list
- !. A term
- !. Term definition
- !. Defines the end of the list

!

!

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

This will produce following result:

HTML
This stands for Hyper Text Markup Language
HTTP
This stands for Hyper Text Transfer Protocol

!

CHAPTER

12

!

! ! !

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

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

I recommend to go through a short tutorial on **Understanding URL**

! !

A link is specified using HTML tag . This tag is called ! and 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 <a> tag.

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Hyperlink Example</title>
</head>
<body>
<p>Click following link</p>
<a href="http://www.tutorialspoint.com" target="_self">Tutorials Point</a>
</body>
</html>
```

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

Click following link
Tutorials Point

! ! !

We have used attribute in our previous example. This attribute is used

possible options:

!	!
_blank	Opens theFlinked document in a new window or tab.
_self	Opens theFlinked document in theFsame frame.
_parent	Opens theFlinked document in theFparent frame.
_top	Opens theFlinked document in theFfull body of theFwindow.
targetframe	Opens theFlinked document in a named*targetframe*.

! ! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Hyperlink Example</title>
<base href="http://www.tutorialspoint.com/">
</head>
<body>
<p>Click following link</p>
<a href="/html/index.htm" target="_blank">HTML Tutorial</a>
</body>
</html>
```

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

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

Click following link
HTML Tutorial

! ! ! ! !

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

First create a link to the place where you want to reach within a webpage and name it using <a...> tag as follows:

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

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

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

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

Go to the Top

!

! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Hyperlink Example</title>
<base href="http://www.tutorialspoint.com/">
</head>
<body alink="#54A250" link="#040404" vlink="#F40633">
<p>Click following link</p>
<a href="/html/index.htm" target="_blank" >HTML Tutorial</a>
</body>
</html>
```

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

Click following link
HTML Tutorial

! !

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:

```
<!DOCTYPE html>
<html>
<head>
<title>Hyperlink Example</title>
</head>
<a href="http://www.tutorialspoint.com/page.pdf">Download PDF File</a>
</body>
</html>
```

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

Download PDF File

! ! ! !

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

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

```
#!/usr/bin/perl
# Addtional HTTP Header
print "Content-Type:application/octet-stream; name=\"FileName\"\r\n";
print "Content-Disposition: attachment; filename=\"FileName\"\r\n\n";
# Open the target file and list down its content as follows
open( FILE, "<FileName" );
while(read(FILE, $buffer, 100)){
   print("$buffer");
}
```

!

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

!

CHAPTER 13

!

! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Image Hyperlink Example</title>
</head>
<body>
<p>Click following link</p>
<a href="http://www.tutorialspoint.com" target="_self">
   <img src="/images/logo.png" alt="Tutorials Point" border="0"/>
</a>
</body>
</html>
```

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

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

. ! !

The HTML and XHTML standards provide a feature that lets you embed many different links inside a single image. You can create different links on the

single image based on different coordinates available on the image. Once different links are attached to different coordinates, we can click different parts of the image to open target documents. Such mouse-sensitive images are known as image maps.

There are two ways to create image maps:

- . ! ! !. This is enabled by the attribute of the <img> tag and requires access to a server and related image-map processing applications.
- . ! ! !. This is created with the attribute of the <img> tag, along with corresponding <map> and <area> tags.

. ! ! !

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

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

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

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

It has been generated by the following code snippet:

```
<!DOCTYPE html>
<html>
<head>
<title>ISMAP Hyperlink Example</title>
</head>
<body>
<p>Click following link</p>
<a href="/cgi-bin/ismap.cgi" target="_self">
   <img ismap src="/images/logo.png" alt="Tutorials Point" border="0"/>
```

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

Then the browser sends the following search parameters to the web serversear>

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

!

! !

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

rect = x_1 , y_1 , x_2 , y_2

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

circle = x_c , y_c , radius

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

poly = x_1 , y_1 , x_2 , y_2 , x_3 , y_3 , … x_n , y_n

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

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

CHAPTER 14

! ! ! !

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

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

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

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

! ! ! !

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

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

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

```
Send Email
```

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

! !

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

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

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

```
Send Feedback
```


tutorialspoint

!

! !

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

! ! !

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

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

!

! !

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

!

tutorialspoint

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

This will produce following result:

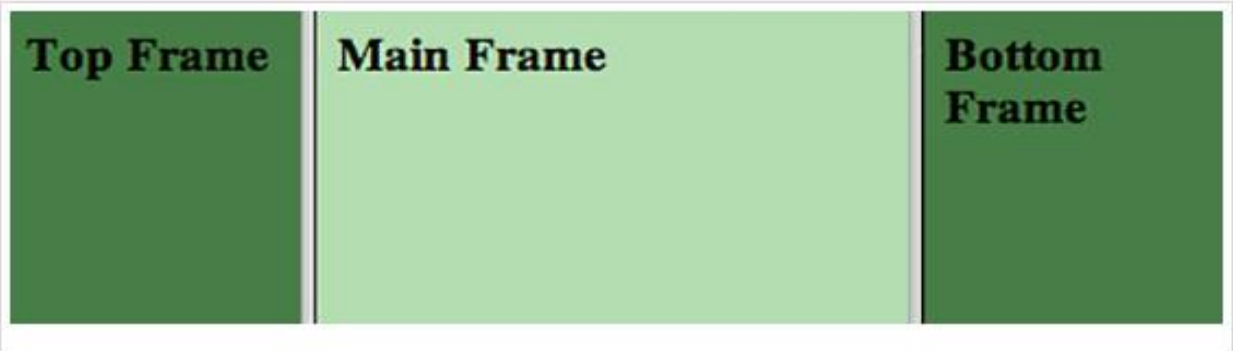
!

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Frames</title>
</head>
<frameset cols="25%,50%,25%">
   <frame name="left" src="/html/top_frame.htm" />
   <frame name="center" src="/html/main_frame.htm" />
   <frame name="right" src="/html/bottom_frame.htm" />
   <noframes>
   <body>
      Your browser does not support frames.
   </body>
   </noframes>
</frameset>
</html>
```

This will produce following result:



!

! ! ! !

Following are important attributes of the <frameset> tag:

!	!
cols	specifies how many columns are contained in the frameset and the size of each column. You can specify the width of each column in one of four ways: • Absolute values in pixels. For example to create three vertical frames, use *cols="100, 500,100"*. • A percentage of the browser window. For example to create three vertical frames, use *cols="10%, 80%,10%"*. • Using a wildcard symbol. For example to create three vertical frames, use *cols="10%, *,10%"*. In this case wildcard takes remainder of the window. • As relative widths of the browser window. For example to create three vertical frames, use *cols="3*,2*,1*"*. This is an alternative to percentages. You can use relative widths of the browser window. Here the window is divided into sixths: the first column takes up half of the window, the second takes one third, and the third takes one sixth.
rows	This attribute works just like the cols attribute and takes the same values, but it is used to specify the rows in the frameset.

	For example to create two horizontal frames, use *rows="10%, 90%"*. You can specify the height of each row in the same way as explained above for columns.
border	This attribute specifies the width of the border of each frame in pixels. For example border="5". A value of zero means no border.
frameborder	This attribute specifies whether a three-dimensional border should be displayed between frames. This attribute takes value either 1 (yes) or 0 (no). For example frameborder="0" specifies no border.
framespacing	This attribute specifies the amount of space between frames in a frameset. This can take any integer value. For example framespacing="10" means there should be 10 pixels spacing between each frames.

!

! ! ! !

Following are important attributes of <frame> tag:

!	**!**
src	This attribute is used to give the file name that should be loaded in the frame. Its value can be any URL. For example, src="/html/top_frame.htm" will load an HTML file available in html directory.
name	This attribute allows you to give a name to a frame. It is used to indicate which frame a document should be loaded into. This is especially important when you want to create links in one frame that load pages into another frame, in which case the second frame needs a name to identify itself as the target of the link.
frameborder	This attribute specifies whether or not the borders of that frame are shown; it overrides the value given in the

	frameborder attribute on the <frameset> tag if one is given, and this can take values either 1 (yes) or 0 (no).
marginwidth	This attribute allows you to specify the width of the space between the left and right of the frame's borders and the frame's content. The value is given in pixels. For example marginwidth="10".
marginheight	This attribute allows you to specify the height of the space between the top and bottom of the frame's borders and its contents. The value is given in pixels. For example marginheight="10".
noresize	By default you can resize any frame by clicking and dragging on the borders of a frame. The noresize attribute prevents a user from being able to resize the frame. For example noresize="noresize".
scrolling	This attribute controls the appearance of the scrollbars that appear on the frame. This takes values either "yes", "no" or "auto". For example scrolling="no" means it should not have scroll bars.
longdesc	This attribute allows you to provide a link to another page containing a long description of the contents of the frame. For example longdesc="framedescription.htm"

!

! ! ! !

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

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

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

(! ! ! ! !

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

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Target Frames</title>
</head>
<frameset cols="200, *">
   <frame src="/html/menu.htm" name="menu_page" />
   <frame src="/html/main.htm" name="main_page" />
   <noframes>
   <body>
      Your browser does not support frames.
   </body>
   </noframes>
</frameset>
</html>
```

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

Following is the content of menu.htm file

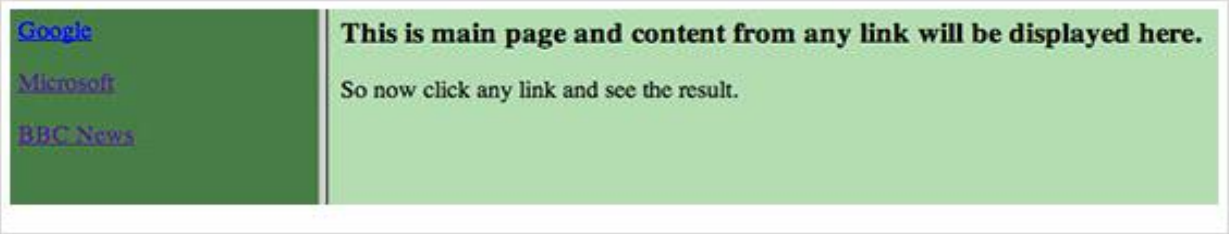
```
<!DOCTYPE html>
<html>
<body bgcolor="#4a7d49">
<a href="http://www.google.com" target="main_page">Google</a>
<br /><br />
<a href="http://www.microsoft.com" target="main_page">Microsoft</a>
<br /><br />
<a href="http://news.bbc.co.uk" target="main_page">BBC News</a>
</body>
</html>
```

Following is the content of main.htm file:

```
<!DOCTYPE html>
<html>
<body bgcolor="#b5dcb3">
```

```
<h3>This is main page and content from any link will be displayed here.</h3>
<p>So now click any link and see the result.</p>
</body>
</html>
```

When we load / file, it produces following result:



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

!	!
_self	Loads the page into the current frame.
_blank	Loads a page into a new browser window.
_parent	Loads the page into the parent window, which in the case of a single frameset, is the main browser window.
_top	Loads the page into the browser window, replacing any current frames.
targetframe	Loads the page into a named targetframe.

!

CHAPTER 16

!

! !

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

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Iframes</title>
</head>
<body>
<p>Document content goes here...</p>
<iframe src="/html/menu.htm" width="555" height="200">
   Sorry your browser does not support inline frames.
</iframe>
<p>Document content also go here...</p>
</body>
</html>
```

This will produce following result:

! ! ! !

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

!	!
src	This attribute is used to give the file name that should be loaded in the frame. Its value can be any URL. For example, src="/html/top_frame.htm" will load an HTML file available in html directory.
name	This attribute allows you to give a name to a frame. It is used to indicate which frame a document should be loaded into. This is especially important when you want to create links in one frame that load pages into an another frame, in which case the second frame needs a name to identify itself as the target of the link.
frameborder	This attribute specifies whether or not the borders of that frame are shown; it overrides the value given in the frameborder attribute on the <frameset> tag if one is given, and this can take values either 1 (yes) or 0 (no).

marginwidth	This attribute allows you to specify the width of the space between the left and right of the frame's borders and the frame's content. The value is given in pixels. For example marginwidth="10".
marginheight	This attribute allows you to specify the height of the space between the top and bottom of the frame's borders and its contents. The value is given in pixels. For example marginheight="10".
noresize	By default you can resize any frame by clicking and dragging on the borders of a frame. The noresize attribute prevents a user from being able to resize the frame. For example noresize="noresize".
scrolling	This attribute controls the appearance of the scrollbars that appear on the frame. This takes values either "yes", "no" or "auto". For example scrolling="no" means it should not have scroll bars.
longdesc	This attribute allows you to provide a link to another page containing a long description of the contents of the frame. For example longdesc="framedescription.htm"

CHAPTER 17

!

! !

All the HTML elements can be categorized into two categories) Block Level Elements) Inline Elements

! !

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

! !

Inline elements, 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>, , , <sup>, <sub>, <big>, <small>, <li>, <ins>, <del>, <code>, <cite>, <dfn>, <kbd>, and <var> elements are all inline elements.

! ! !

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

! ! !

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

!

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

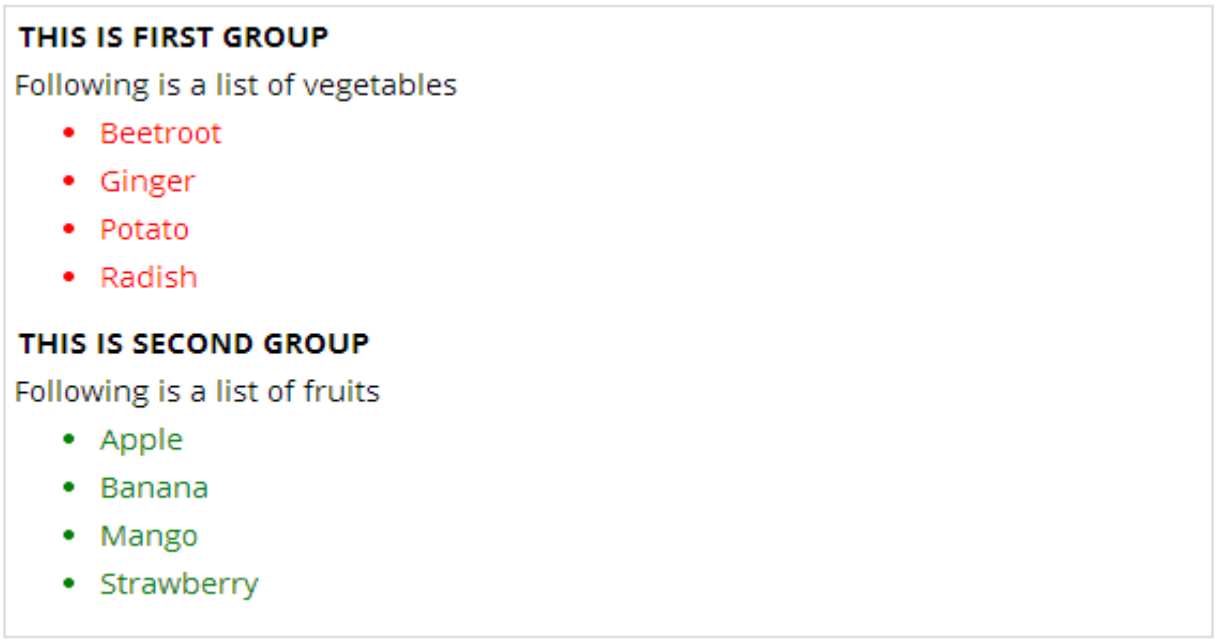
```
<!DOCTYPE html>
<html>
<head>
<title>HTML div Tag</title>
```

```
</head>
<body>

<!-- First group of tags -->
<div style="color:red">
   <h4>This is first group</h4>
   <p>Following is a list of vegetables</p>
   <ul>
   <li>Beetroot</li>
   <li>Ginger</li>
   <li>Potato</li>
   <li>Radish</li>
   </ul>
</div>
<!-- Second group of tags -->
<div style="color:green">
   <h4>This is second group</h4>
   <p>Following is a list of fruits</p>
   <ul>
   <li>Apple</li>
   <li>Banana</li>
   <li>Mango</li>
   <li>Strawberry</li>
   </ul>
</div>

</body>
</html>
```

This will produce following result:



! ! !

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

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML span Tag</title>
</head>
<body>
<p>This is <span style="color:red">red</span> and this is <span style="color:green">green</
span></p>
</body>
</html>
```

This will produce following result:

This is red, and this is green

!

tutorialspoint

CHAPTER 18

! ! !

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

- Html Background with Colors
- Html Background with Images

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

! ! ! !

The attribute is used to control the background of an HTML element, specifically page body and table backgrounds. Following is the syntax to use bgcolor attribute with any HTML tag.

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

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

```
<!-- Format 1 - Use color name -->
<table bgcolor="lime" >

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

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

! ! !

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Background Colors</title>
</head>
<body>
```


tutorialspoint

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

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

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

</body>
</html>
```

This will produce following result:

! ! ! !

The attribute can also be used to control the background of an HTML element, specifically page body and table backgrounds. You can specify an image to set background of your HTML page or table. Following is the syntax to use background attribute with any HTML tag.

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

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

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

!

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

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

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

</body>
</html>
```

This will produce following result:

!

!' ! ! !

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

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Background Images</title>
</head>
<body>

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

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

```
</td></tr>
</table>

</body>
</html>
```

This will produce following result:

CHAPTER 19

!

! !

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

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

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

! ! ! !

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

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

Now we will see these coloring schemes one by one.

! !.! ! !

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

Check a complete list of **HTML Color Name**.

! !2 ! !

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

tutorialspoint

	Black		Gray		Silver		White
	Yellow		Lime		Aqua		Fuchsia
	Red		Green		Blue		Purple
	Maroon		Olive		Navy		Teal

!

! !

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Colors by Name</title>
</head>
<body text="blue" bgcolor="green">
<p>Use different color names for body and table and see the result.</p>
<table bgcolor="black">
<tr>
<td>
<font color="white">This text will appear white on black background.</font>
</td>
</tr>
</table>
</body>
</html>
```

!

! !.! ! !

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

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

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

!	! !
	#000000

tutorialspoint

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

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

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

!	! !
	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)

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Colors by RGB code</title>
</head>
<body text="rgb(0,0,255)" bgcolor="rgb(0,255,0)">
<p>Use different color code for body and table and see the result.</p>
<table bgcolor="rgb(0,0,0)">
<tr>
<td>
<font color="rgb(255,255,255)">This text will appear white on black background.</font>
</td>
</tr>
</table>
</body>
</html>
```

!

! ! !

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

000000	000033	000066	000099	0000CC	0000FF
003300	003333	003366	003399	0033CC	0033FF
006600	006633	006666	006699	0066CC	0066FF
009900	009933	009966			

669900	669933	669966	669999	6699CC	6699FF
66CC00	66CC33	66CC66	66CC99	66CCCC	66CCFF
66FF00	66FF33	66FF66	66FF99	66FFCC	66FFFF
990000	990033	990066	990099	9900CC	9900FF
993300	993333	993366	993399	9933CC	9933FF
996600	996633	996666	996699	9966CC	9966FF
999900	999933	999966	999999	9999CC	9999FF
99CC00	99CC33	99CC66	99CC99	99CCCC	99CCFF
99FF00	99FF33	99FF66	99FF99	99FFCC	99FFFF
CC0000	CC0033	CC0066	CC0099	CC00CC	CC00FF
CC3300	CC3333	CC3366	CC3399	CC33CC	CC33FF
CC6600	CC6633	CC6666	CC6699	CC66CC	CC66FF
CC9900	CC9933	CC9966	CC9999	CC99CC	CC99FF
CCCC00	CCCC33	CCCC66	CCCC99	CCCCCC	CCCCFF
CCFF00	CCFF33	CCFF66	CCFF99	CCFFCC	CCFFFF
FF0000	FF0033	FF0066	FF0099	FF00CC	FF00FF
FF3300	FF3333	FF3366	FF3399	FF33CC	FF33FF
FF6600	FF6633	FF6666	FF6699	FF66CC	FF66FF

FF9900	FF9933	FF9966	FF9999	FF99CC	FF99FF
FFCC00	FFCC33	FFCC66	FFCC99	FFCCCC	FFCCFF
FFFF00	FFFF33	FFFF66	FFFF99	FFFFCC	FFFFFF

!

CHAPTER 20

! ! !

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

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

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

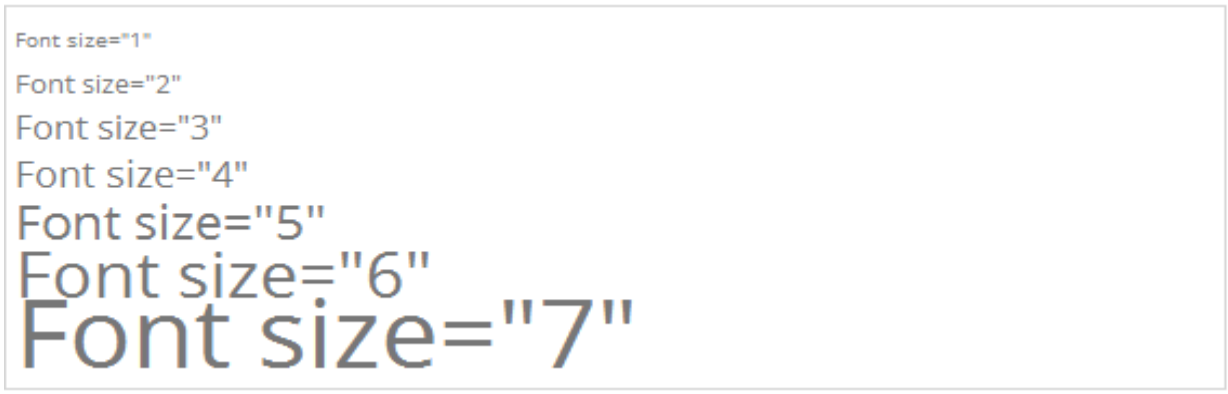
! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Setting Font Size</title>
</head>
<body>
<font size="1">Font size="1"</font><br />
<font size="2">Font size="2"</font><br />
<font size="3">Font size="3"</font><br />
<font size="4">Font size="4"</font><br />
<font size="5">Font size="5"</font><br />
<font size="6">Font size="6"</font><br />
<font size="7">Font size="7"</font>
</body>
</html>
```

This will produce following result:



! ! !

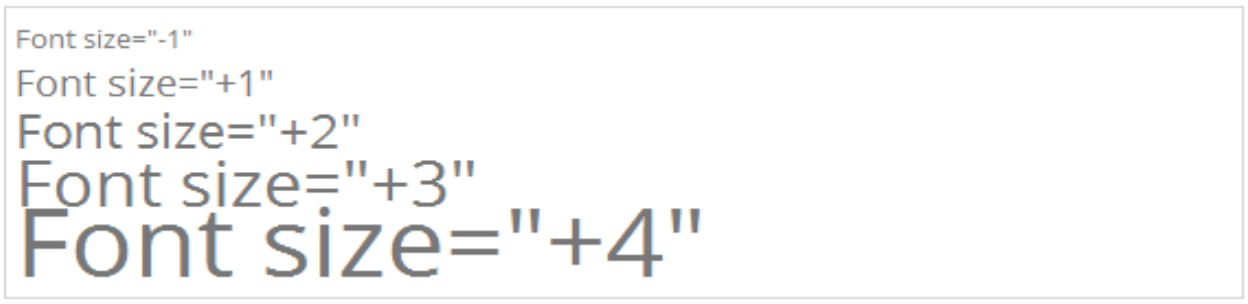
You can specify how many sizes larger or how many sizes smaller than the preset font size should be. You can specify it like ! , ! ! !

.

!

```
<!DOCTYPE html>
<html>
<head>
<title>Relative Font Size</title>
</head>
<body>
<font size="-1">Font size="-1"</font><br />
<font size="+1">Font size="+1"</font><br />
<font size="+2">Font size="+2"</font><br />
<font size="+3">Font size="+3"</font><br />
<font size="+4">Font size="+4"</font>
</body>
</html>
```

This will produce following result:



! ! !

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

!

```
<!DOCTYPE html>
<html>
<head>
<title>Font Face</title>
</head>
<body>
<font face="Times New Roman" size="5">Times New Roman</font><br />
<font face="Verdana" size="5">Verdana</font><br />
<font face="Comic sans MS" size="5">Comic Sans MS</font><br />
<font face="WildWest" size="5">WildWest</font><br />
<font face="Bedrock" size="5">Bedrock</font><br />
</body>
</html>
```

This will produce following result:

Times New Roman
Verdana
Comic Sans MS
WildWest
Bedrock

! ! ! !

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

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

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

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

! ! !

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.

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

Example !

```
<!DOCTYPE html>
<html>
<head>
<title>Setting Font Color</title>
</head>
<body>
<font color="#FF00FF">This text is in pink</font><br />
<font color="red">This text is red</font>
</body>
</html>
```

This will produce following result:

! ! !

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

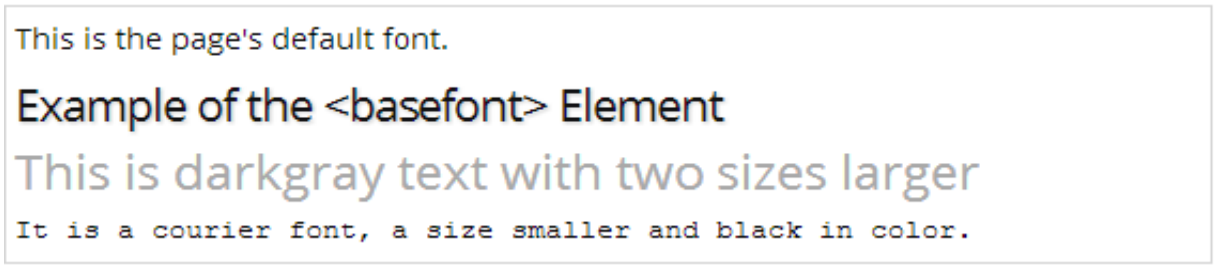
The <basefont> tag also takes color, Fo><br

tutorialspoint

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

</body>
</html>
```

This will produce following result:



CHAPTER 21

! !

! !

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

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

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

The HTML tag is used to create an HTML form and it has following syntax:

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

! !

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

!	!
action	Backend script ready to process your passed data.
method	Method to be used to upload data. The most frequently used are GET and POST methods.

target	Specify the target window or frame 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: • . . . - This is the standard method most forms use in simple scenarios. • . - This is used when you want to upload binary data in the form of files like image, word file etc.

!

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

! ! !

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

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

! ! !

There are three types of text input used on forms:

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

. ! ! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Text Input Control</title>
</head>
<body>
<form >
First name:  <input type="text" name="first_name" />
<br>
Last name:  <input type="text" name="last_name" />
</form>
</body>
</html>
```

This will produce following result:

First name:

Last name:

!

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

!	!
type	Indicates the type of input control and for text input control it will be set to .
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
value	This can be used to provide an initial value inside the control.
size	Allows to specify the width of the text-input control in terms of characters.

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

!

! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Password Input Control</title>
</head>
<body>
<form >
User ID :  <input type="text" name="user_id" />
<br>
Password:  <input type="password" name="password" />
</form>
</body>
</html>
```

This will produce following result:

User ID :

Password:

!

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

!	!
type	Indicates the type of input control and for password input control it will be set to .

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

!

. ! ! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Multiple-Line Input Control</title>
</head>
<body>
<form>
Description : <br />
<textarea rows="5" cols="50" name="description">
Enter description here...
</textarea>
</form>
</body>
</html>
```

This will produce following result:

!

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

!	!
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
rows	Indicates the number of rows of text area box.
cols	Indicates the number of columns of text area box

!

! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Checkbox Control</title>
</head>
<body>
<form>
<input type="checkbox" name="maths" value="on"> Maths
<input type="checkbox" name="physics" value="on"> Physics
</form>
</body>
</html>
```

This will produce following result:

!

Following is the list of attributes for checkbox.

!	!
type	Indicates the type of input control and for checkbox input control it will be set to .
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
value	The value that will be used if the checkbox is selected.
checked	Set to *checked* if you want to select it by default.

! ! !

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Radio Box Control</title>
</head>
<body>
<form>
<input type="radio" name="subject" value="maths"> Maths
<input type="radio" name="subject" value="physics"> Physics
</form>
</body>
</html>
```

This will produce following result:

!

Following is the list of attributes for radio button.

!	!
type	Indicates the type of input control and for checkbox input control it will be set to .
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
value	The value that will be used if the radio box is selected.
checked	Set to *checked* if you want to select it by default.

!

! ! !

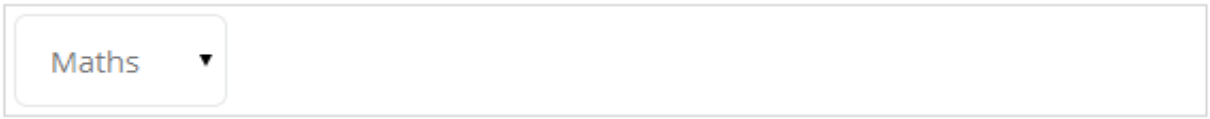
A select box, also called drop down box which provides option to list down various options in the form of drop down list, from where a user can select one or more options.

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>Select Box Control</title>
</head>
<body>
<form>
<select name="dropdown">
<option value="Maths" selected>Maths</option>
<option value="Physics">Physics</option>
</select>
</form>
</body>
</html>
```

This will produce following result:



!

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

!	!
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
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> tag:

!	!
value	The value that will be used if an option in the select box box is selected.
selected	Specifies that this option should be the initially selected value when the page loads.
label	An alternative way of labeling options

tutorialspoint

!

! ! !

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

!	!
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.
image	This creates a clickable button but we can use an image as background of the button.

!

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>File Upload Box</title>
</head>
<body>
<form>
<input type="submit" name="submit" value="Submit" />
<input type="reset" name="reset"  value="Reset" />
<input type="button" name="ok" value="OK"  />
<input type="image" name="imagebutton" src="/html/images/logo.png" />
</form>
</body>
</html>
```

This will produce the following result

!

! ! !

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

CHAPTER

22

!

! ! !

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

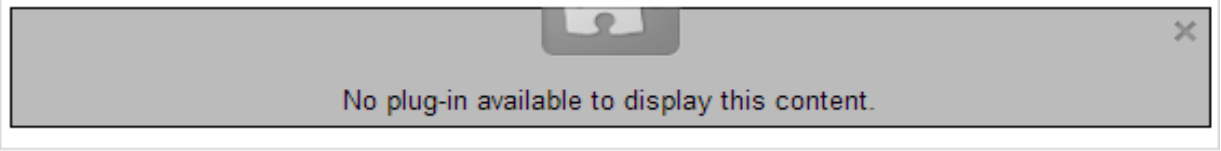
You can also include a tag for the browsers which don't recognize the <embed> tag. You could, for example, use <embed> to display a movie of your choice, and <noembed> to display a single JPG image if browser does not support <embed> tag.

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML embed Tag</title>
</head>
<body>
<embed src="/html/yourfile.mid" width="100%" height="60" >
   <noembed><img src="yourimage.gif" alt="Alternative Media" ></noembed>
</embed>
</body>
</html>
```

This will produce following result:



You should have appropriate plug-in installed to view the correct output. If you will run the same code in Internet Explorer, you will get the following result.

You can put any media file in src attribute. You can try it yourself by giving various types of files.

! ! ! !

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

!	!
align	Determines how to align the object. It can be set to either *center, left or right*.
autostart	This boolean attribute indicates if the media should start automatically. You can set it either true or false.
loop	Specifies if the sound should be played continuously (set loop to true), a certain number of times (a positive value) or not at all (false)
playcount	Specifies the number of times to play the sound. This is alternate option for *loop* if you are using IE.
hidden	Specifies if the multimedia object should be shown on the page. A false value means no and true values means yes.
width	Width of the object in pixels
height	Height of the object in pixels
name	A name used to reference the object.

src	URL of the object to be embedded.
volume	Controls volume of the sound. Can be from 0 (off) to 100 (full volume).

!

! ! !

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

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML embed Tag</title>
</head>
<body>
<embed src="/html/yourfile.swf" width="200" height="200" >
   <noembed><img src="yourimage.gif" alt="Alternative Media" ></noembed>
</embed>
</body>
</html>
```

This will produce following result:

! !

You can use HTML tag to play a soundtrack in the background of your webpage. This tag is supported by Internet Explorer only and most of the other browsers ignore this tag. It downloads and plays an audio file when the

host document is first downloaded by the user and displayed. The background sound file also will replay whenever the user refreshes the browser.

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

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML embed Tag</title>
</head>
<body>
<bgsound src="/html/yourfile.mid">
   <noembed><img src="yourimage.gif" ></noembed>
</bgsound>
</body>
</html>
```

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

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

! ! !

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

Here are few examples:

!.! 2!

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

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

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

!.! !

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

```
<object data="data/test.pdf" type="application/pdf" width="300" height="200">
```

```
  alt : <a href="data/test.pdf">test.htm</a>
</object>
```

!

!.! !

You can specify some parameters related to the document with the tag. Here is an example to embed a wav file:

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

!

!.! !

You can add a flash document as follows:

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" id="penguin"
   codebase="someplace/swflash.cab" width="200" height="300">
   <param name="movie" value="flash/penguin.swf" />
   <param name="quality" value="high" />
   <img src="penguin.jpg" width="200" height="300" alt="Penguin" />
</object>
```

!

!.! !

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

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

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

!

CHAPTER 23

!

! !

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

The HTML <marquee> tag may not be supported by various browsers so it's not recommended to rely on this tag, instead you can use Javascript and CSS to create such effects.

!

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

```
<marquee attribute_name="attribute_value"....more attributes>

One or more lines or text message or image

</marquee>
```

!

! ! ! !

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

!	!
width	This specifies the width of the marquee. This can be a value like 10 or 20% etc.
height	This specifies the height of the marquee. This can be a value like 10 or 20% etc.
direction	This specifies the direction in which marquee should scroll. This can be a value like *up*, *down*, *left* or *right*.

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

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

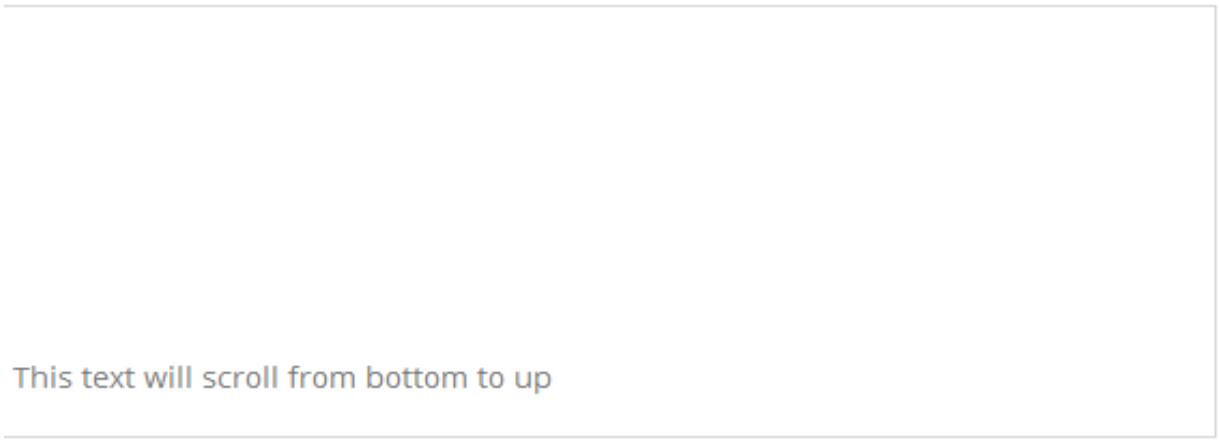
```
<!DOCTYPE html>
<html>
<head>
<title>HTML marquee Tag</title>
</head>
<body>
<marquee>This is basic example of marquee</marquee>
</body>
</html>
```

This will produce following result:

This is basic example of marquee

!

This will produce following result:



!

CHAPTER 24

!

! !

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

```
Document declaration tag
<html>
   <head>
      Document header related tags
   </head>

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

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

! ! ! !

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Title Tag Example</title>
</head>
<body>
<p>Hello, World!</p>
</body>
</html>
```

This will produce following result:

Hello, World!

! ! ! !

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

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Meta Tag Example</title>

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

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

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

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

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

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

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

</head>
<body>
<p>Hello, World!</p>
</body>
</html>
```

This will produce following result:

```
Hello, World!
```

! ! ! !

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

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

```
<!DOCTYPE html>
```

```
<html>
<head>
<title>HTML Base Tag Example</title>
<base href="http://www.tutorialspoint.com/" />
</head>
<body>

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

</body>
</html>
```

This will produce following result:

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

!

! ! ! !

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML link Tag Example</title>
<base href="http://www.tutorialspoint.com/" />
<link rel="stylesheet" type="text/css" href="/css/style.css">
</head>
<body>
<p>Hello, World!</p>
</body>
</html>
```

This will produce following result:

Hello, World!

! ! ! !

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML style Tag Example</title>
<base href="http://www.tutorialspoint.com/" />
<style type="text/css">
.myclass{
   background-color: #aaa;
   padding: 10px;
}
</style>
</head>
<body>
<p
```

```
</html>
```

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

To learn about how Javascript works, kindly check a separate tutorial available at http://www.tutorialspoint.com/javascript

CHAPTER 25

! ! !

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

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML CSS</title>
</head>
<body>
<p><font color="green" size="5">Hello, World!</font></p>
</body>
</html>
```

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML CSS</title>
</head>
<body>
<p style="color:green;font-size:24px;">Hello, World!</p>
</body>
</html>
```

This will produce following result:

Hello, World!

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

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

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

! ! !

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

!

Consider we define a style sheet file / which has following rules:

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

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

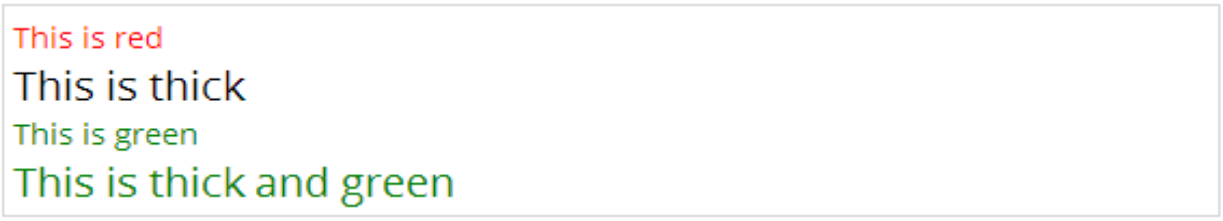
```
<!DOCTYPE html>
<html>
<head>
<title>HTML External CSS</title>
<link rel="stylesheet" type="text/css" href="/html/style.css">
</head>
<body>
<p class="red">This is red</p>

<p class="thick">This is thick</p>
```

```
<p class="green">This is green</p>

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

This will produce following result:



! ! !

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

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Internal CSS</title>
<style type="text/css">
.red{
   color: red;
}
.thick{
   font-size:20px;
}
.green{
   color:green;
}
</style>
</head>
<body>
<p class="red">This is red</p>

<p class="thick">This is thick</p>

<p class="green">This is green</p>

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

This will produce following result:

This is red

This is thick

This is green

This is thick and green

! ! !

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

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

!

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

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Inline CSS</title>
</head>
<body>
<p style="color:red;">This is red</p>

<p style="font-size:20px;">This is thick</p>

<p style="color:green;">This is green</p>

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

This will produce following result:

This is red

This is thick

This is green

This is thick and green

! !

CHAPTER

26

!

! !

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

You can write various small functions, called event handlers, using any of the scripting language and then you can trigger those functions using HTML attributes.

Now a days only and associated frameworks are being used by most of the web developers, VBScript is not even supported by various major browsers.

You can keep Javascript code in a separate file and then include it whereever it's needed, or you can define functionality inside HTML document itself. Let's see both the cases one by one with suitable examples.

! !

If you are going to define a functionality which will be used in various HTML documents then it's better to keep that functionality in a separate Javascript file and then include that file in your HTML documents. A Javascript file will have extension as **/** and it will be included in HTML files using <script> tag.

!

Consider we define a small function using Javascript in **/** which has following code:

```
function Hello()
{
    alert("Hello, World");
}
```

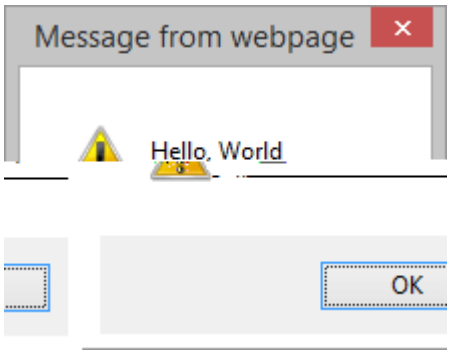
Now let's make use of the above external Javascript file in our following HTML document:

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

```
<head>
<title>Javascript External Script</title>
<script src="/html/script.js" type="text/javascript"/></script>
</head>
<body>
<input type="button" onclick="Hello();" name="ok" value="Click Me"  />
</body>
</html>
```

This will produce following result.

When you will click the button, you will get the following result.



! !

You can write your script code directly into your HTML document. Usually we keep script code in header of the document using <script> tag, otherwise there is no restriction and you can put your source code anywhere in the document but inside <script> tag.

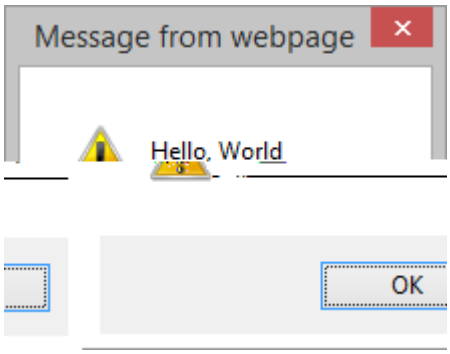
!

```
<!DOCTYPE html>
<html>
<head>
<title>Javascript Internal Script</title>
<base href="http://www.tutorialspoint.com/" />
<script type="text/javascript">
function Hello(){
   alert("Hello, World");
}
</script>
</head>
<body>
<input type="button" onclick="Hello();" name="ok" value="Click Me"  />
</body>
</html>
```

This will produce following result.

Click Me

When you will click the button, you will get the following result.



! !

Event handlers are nothing but simply defined functions which can be called against any mouse or keyboard event. You can define your business logic inside your event handler which can vary from a single to 1000s of line code.

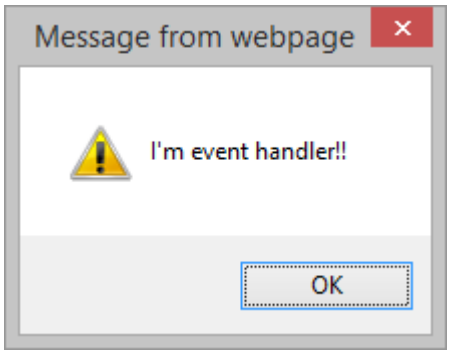
Following example explains how to write an event handler. Let's write one simple function *EventHandler()* in the header of the document. We will call this function when any user brings mouse over a paragraph.

```
<!DOCTYPE html>
<html>
<head>
<title>Event Handlers Example</title>
<base href="http://www.tutorialspoint.com/" />
<script type="text/javascript">
function EventHandler(){
   alert("I'm event handler!!");
}
</script>
</head>
<body>
<p onmouseover="EventHandler();">Bring your mouse here to see an alert</p>
</body>
</html>
```

Now this will produce following result.

Bring your mouse here to see an alert

When you will bring your mouse over this line, you will see the following result.



! ! ! ! !

Although most (if not all) browsers these days support Javascript, but still some older browsers don't. If a browser doesn't support JavaScript, instead of running your script, it would display the code to the user. To prevent this, you can simply place HTML comments around the script as shown below.

```
JavaScript Example:
<script type="text/javascript">
<!--
document.write("Hello Javascript!");
//-->
</script>

VBScript Example:
<script type="text/vbscript">
<!--
document.write("Hello VBScript!")
'-->
</script>
```

!

! ! !

You can also provide alternative info to the users whose browsers don't support scripts and for those users who have disabled script option their browsers. You can do this using the tag.

```
JavaScript Example:
<script type="text/javascript">
<!--
document.write("Hello Javascript!");
//-->
</script>
<noscript>Your browser does not support Javascript!</noscript>
VBScript Example:
<script type="text/vbscript">
<!--
document.write("Hello VBScript!")
```

```
'-->
</script>
<noscript>Your browser does not support VBScript!</noscript>
```

!

! ! !

There may be a situation when you will include multiple script files and ultimately using multiple <script> tags. You can specify a default scripting language for all your *script* tags. This saves you from specifying the language every time you use a script tag within the page. Below is the example:

```
<meta http-equiv="Content-Script-Type" content="text/JavaScript" />
```

Note that you can still override the default by specifying a language within the script tag.

!

tutorialspoint

CHAPTER

27

!

! !

A webpage layout is very important to give better look to your website. It takes considerable time to design a website's layout with great look and feel.

Now a days, all modern websites are using CSS and Javascript based framework to come up with responsive and dynamic websites but you can create a good layout using simple HTML tables or division tags in combination with other formatting tags. This chapter will give you few examples on how to create a simple but working layout for your webpage using pure HTML and its attributes.

! !.! ! !

The simplest and most popular way of creating layouts is using HTML <table> tag. These tables are arranged in columns and rows, so you can utilize these rows and columns in whatever way you like.

!

For example, the following HTML layout example is achieved using a table with 3 rows and 2 columns but the header and footer column spans both columns using the colspan attribute:

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Layout using Tables</title>
</head>
<body>
<table width="100%" border="0">
  <tr>
    <td colspan="2" bgcolor="#b5dcb3">
      <h1>This is Web Page Main title</h1>
    </td>
  </tr>
  <tr valign="top">
    <td bgcolor="#aaa" width="50">
      <b>Main Menu</b><br />
      HTML<br />
      PHP<br />
      PERL...
    </td>
    <td bgcolor="#eee" width="100" height="200">
```

```
        Technical and Managerial Tutorials
    </td>
  </tr>
  <tr>
    <td colspan="2" bgcolor="#b5dcb3">
      <center>
      Copyright © 2007 Tutorialspoint.com
      </center>
    </td>
  </tr>
</table>
</body>
</html>
```

This will produce following result:

!

! ! !.! ! !

You can design your webpage to put your web content in multiple pages. You can keep your content in middle column and you can use left column to use menu and right column can be used to put advertisement or some other stuff. This layout will be very similar to what we have at our website tutorialspoint.com.

!

Here is an example to create three column layout:

```
<!DOCTYPE html>
<html>
<head>
<title>Three Column HTML Layout</title>
</head>
```

```
<body>
<table width="100%" border="0">
  <tr valign="top">
    <td bgcolor="#aaa" width="20%">
      <b>Main Menu</b><br />
      HTML<br />
      PHP<br />
      PERL...
    </td>
    <td bgcolor="#b5dcb3" height="200" width="60%">
        Technical and Managerial Tutorials
    </td>
    <td bgcolor="#aaa" width="20%">
      <b>Right Menu</b><br />
      HTML<br />
      PHP<br />
      PERL...
    </td>
   </tr>
<table>
</body>
</html>
```

This will produce following result:

! !.! ! -! !

The element is a block level element used for grouping HTML elements. While the tag is a block-level element, the HTML span element is used for grouping elements at an inline level.

Although we can achieve pretty nice layouts with HTML tables, but tables weren't really designed as a layout tool. Tables are more suited to presenting tabular data.

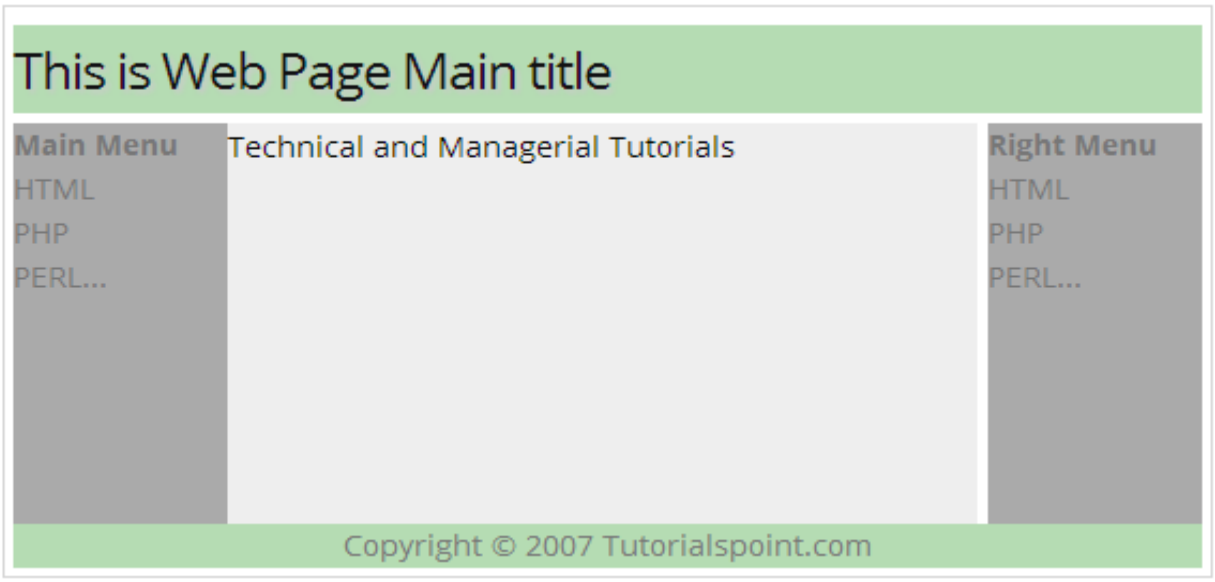
This example makes use of Cascading Style Sheet (CSS), so before understanding this example you need to have a better understanding on how CSS works.

!

Here we will try to achieve same result using tag along with CSS, whatever you have achieved using <table> tag in previous example.

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Layouts using DIV, SPAN</title>
</head>
<body>
<div style="width:100%">
  <div style="background-color:#b5dcb3; width:100%">
      <h1>This is Web Page Main title</h1>
  </div>
  <div style="background-color:#aaa; height:200px;width:100px;float:left;">
      <div><b>Main Menu</b></div>
      HTML<br />
      PHP<br />
      PERL...
  </div>
  <div style="background-color:#eee; height:200px;width:350px;float:left;">
    <p>Technical and Managerial Tutorials</p>
  </div>
  <div style="background-color:#aaa; height:200px;width:100px;float:right;">
      <div><b>Right Menu</b></div>
      HTML<br />
      PHP<br />
      PERL...
  </div>
  <div style="background-color:#b5dcb3;clear:both">
  <center>
      Copyright © 2007 Tutorialspoint.com
  </center>
  </div>
</div>
</body>
</html>
```

This will produce following result:



You can create better layout using DIV, SPAN along with CSS. For more information on CSS, please refer to CSS Tutorial.

!

!

CHAPTER

28

! ! !

Following tags have been introduced in older versions of HTML but all the tags marked with are part of HTML-5.

!	!	!
<!--...-->	Specifies a comment	
<!DOCTYPE>	Specifies the document type	
<a>	Specifies an anchor	
<abbr>	Specifies an abbreviation	
<acronym>	Specifies an acronym	
<address>	Specifies an address element	
<applet>	/ Specifies an applet	
<area>	Specifies an area inside an image map	
<article>	Specifies an article	

tutorialspoint

<aside>	Specifies some content loosely related to the page content. If it is removed, the remaining content still makes sense	
<audio>	Specifies a sound content	
<b>	Specifies bold text	
<base>	Specifies a base URL for all the links in a page	
<basefont>	/ Specifies a base font	
<bdo>	Specifies the direction of text display	
<bdi>	Represents text that must be isolated from its surrounding for bidirectional text formatting. It allows embedding a span of text with a different, or unknown, directionality	
<bgsound>	Specifies background music	
<big>	Specifies big text	
<blink>	Specifies a text which blinks	
<blockquote>	Specifies a long quotation	

<body>	Specifies the body element	
 	Inserts a single line break	
<button>	Specifies a push button	
<canvas>	For making graphics with a script	
<caption>	Specifies a table caption	
	/ Specifies centered text	
<cite>	Specifies a citation	
<code>	Specifies computer code text	
<col>	Specifies attributes for table columns	
<colgroup>	Specifies groups of table columns	
<comment>	Puts a comment in the document	
<datalist>	A list of options for input values	
<dd>	Specifies a definition description	

<del>	Specifies deleted text	
<dfn>	Specifies a definition term	
<dialog>	Specifies a dialog box or window	
<dir>	/ Specifies a directory list	
	Specifies a section in a document	
<dl>	Specifies a definition list	
<dt>	Specifies a definition term	
	Specifies emphasized text	
<embed>	Specifies a container for an external (non-HTML) application	
<fieldset>	Specifies a fieldset	
<figcaption>	Specifies a caption for a <figure> element	
<figure>	Specifies self-contained content	

<font>	/ Specifies text font, size, and color	
<footer>	Specifies a footer for a document or section	
<form>	Specifies a form	
<frame>	Specifies a sub window (a frame)	
<frameset>	Specifies a set of frames	
<h1> to <h6>	Specifies header 1 to header 6	
<head>	Specifies information about the document	
<header>	Specifies a header for a document or section	
<hr>	Specifies a horizontal rule	
<html>	Specifies an html document	
<i>	Specifies italic text	
<iframe>	Specifies an inline sub window (frame)	
<ilayer>	Specifies an inline layer	

<img>	Specifies an image	
<input>	Specifies an input field	
<ins>	Specifies inserted text	
<isindex>	/ Specifies a single-line input field	
<kbd>	Specifies keyboard text	
<keygen>	Generate key information in a form	
<label>	Specifies a label for a form control	
<layer>	Specifies a layer	
<legend>	Specifies a title in a fieldset	
	Specifies a list item	
<link>	Specifies a resource reference	
<main>	Specifies the main or important content in the document. There is only one element in the document	
<map>	Specifies an image map	

<mark>	Specifies a text highlighted for reference purposes, that is for its relevance in another context	
<marquee>	Creates a scrolling-text marquee	
<menu>	/ Specifies a menu list	
<menuitem>	Specifies a command/menu item that the user can invoke from a popup menu	
<meter>	Specifies a scalar measurement within a known range (a gauge)	
<multicol>	Specifies a multicolumn text flow	
<nav>	Specifies a section that contains only navigation links	
<nobr>	No breaks allowed in the enclosed text	
<noembed>	Specifies content to be presented by browsers that do not support the <embed> tag	
<noframes>	Specifies a noframe section	

<noscript>	Specifies a noscript section	
<object>	Specifies an embedded object	
	Specifies an ordered list	
<optgroup>	Specifies an option group	
<option>	Specifies an option in a drop-down list	
<output>	Specifies the result of a calculation	
<p>	Specifies a paragraph	
<param>	Specifies a parameter for an object	
<plaintext>	/ Render the remainder of the document as preformatted plain text	
<pre>	Specifies preformatted text	
<progress>	Specifies a completion progress of a task	
<q>	Specifies a short quotation	

<rp>	Specifies to show browsers that do not support the ruby element	
<rt>	Specifies a text ruby annotation	
<ruby>	Specifies a ruby annotation	
<s>	/ Specifies strikethrough text	
<samp>	Specifies sample computer code	
<script>	Specifies a script	
<section>	Specifies a section in a document	
<select>	Specifies a selectable list	
<spacer>	Specifies a white space	
<small>	Specifies small text	
<source>	Specifies a media resource for media elements, defined inside video or audio elements	
	Specifies a section in a document	

<strike>	/ Specifies strikethrough text	
	Specifies strong text	
<style>	Specifies a style definition	
<sub>	Specifies subscripted text	
<summary>	Specifies a summary, caption, or legend for a given <details>	
<sup>	Specifies superscripted text	
<table>	Specifies a table	
<tbody>	Specifies a table body	
<td>	Specifies a table cell	
<textarea>	Specifies a text area	
<tfoot>	Specifies a table footer	
<th>	Specifies a table heading	
<thead>	Specifies a table header	
<time>	Specifies a date and time <details>	

<title>	Specifies the document title	
<tr>	Specifies a table row	
<track>	Specifies text tracks used in media players	
<tt>	Specifies teletype text	
<u>	/ Specifies underlined text	
	Specifies an unordered list	
<var>	Specifies a variable	
<video>	Specifies text tracks used in media players	
<wbr>	Indicates a potential word break point within a <nobr> section	
<xmp>	/ Specifies preformatted text	

CHAPTER 29

! ! !

There are few HTML attributes which are standard and associated to all the HTML tags. These attributes are listed here with a brief description.

! !

Not valid in base, head, html, meta, param, script, style, and title elements.

!	. !	!
accesskey		Specifies a shortcut key for an element to be used in place of keyboard.
class		The class of the element
contenteditable	Yes	Boolean attribute to specify whether the element is editable or not.
contextmenu	Yes	Specifies a context menu for an element.
data-*	Yes	Used to store custom data associated with the element.
draggable	Yes	Boolean attribute to specify whether the element can be dragged or not.
dropzone	Yes	Specifies whether the dragged data is copied, moved, or linked, when dropped.

hidden	Yes	Specifies whether element should be visible or not.
id		A unique id for the element
spellcheck	Yes	Specifies if the element must have its spelling or grammar checked.
style		An inline style definition
tabindex		Specifies the tab order of an element.
title		A text to display in a tool tip
translate	Yes	Boolean attribute specifies whether the content of an element should be translated or not

!

! !

The attribute indicates the language being used for the enclosed content. The language is identified using the ISO standard language abbreviations, such as for , for , and so on. RFC 1766 (http://www.ietf.org/rfc/rfc1766.txt) describes these codes and their formats.

Not valid in base, br, frame, frameset, hr, iframe, param, and script elements.

!	!	!
dir	ltr | rtl	Sets the text direction
lang	language_code	Sets the language code

CHAPTER 30

! ! ! !

When users visit your website, they do things like click various links, bring mouse over text and images etc. These are examples of what we call events in Javascript and VBScript terminologies.

We can write our event handlers using Javascript or VBScript and can specify some actions to be taken against these events. Though these are the events but they will be specified as attributes for the HTML tags.

The HTML 4.01 specification had defined only 19 events but later HTML-5 has added many other events which we have listed down here:

! ! !

Following events have been introduced in older versions of HTML but all the tags marked with are part of HTML-5.

!	. !	!
onafterprint		Triggers after a document is printed
onbeforeprint		Triggers before a document is printed
onbeforeonload		Triggers before a document loads
onerror		Triggers when an error occurs
onhaschange		Triggers when a document has changed

tutorialspoint

onload		Triggers when a document loads
onmessage	HTML5	Triggers when a message is triggered
onoffline	HTML5	Triggers when a document goes offline
ononline	HTML5	Triggers when a document comes online
onpagehide	HTML5	Triggers when a window is hidden
onpageshow	HTML5	Triggers when a window becomes visible
onpopstate	HTML5	Triggers when a window's history changes
onredo	HTML5	Triggers when a document performs a redo
onresize	HTML5	Triggers when a window is resized
onstorage	HTML5	Triggers when a document loads
onundo	HTML5	Triggers when a document performs an undo
onunload		Triggers when a user leaves the document

!

! ! !

Following events have been introduced in older versions of HTML but all the tags marked with HTML5 are part of HTML-5.

!	. !	!

tutorialspoint

onblur		Triggers when a window loses focus
onchange		Triggers when an element changes
oncontextmenu		Triggers when a context menu is triggered
onfocus		Triggers when a window gets focus
onformchange		Triggers when a form changes
onforminput		Triggers when a form gets user input
oninput		Triggers when an element gets user input
oninvalid		Triggers when an element is invalid
onreset		Triggers when a form is reset
onselect		Triggers when an element is selected
onsubmit		Triggers when a form is submitted

! ! !

Following events have been introduced in older versions of HTML.

!

!	. !	!
onkeydown		Triggers when a key is pressed

tutorialspoint

onkeypress		Triggers when a key is pressed and released
onkeyup		Triggers when a key is released

!

! ! !

Following events have been introduced in older versions of HTML but all the tags marked with are part of HTML-5.

!	. !	!
onclick		Triggers on a mouse click
ondblclick		Triggers on a mouse double-click
ondrag		Triggers when an element is dragged
ondragend		Triggers at the end of a drag operation
ondragenter		Triggers when an element has been dragged to a valid drop target
ondragleave		Triggers when an element leaves a valid drop target
ondragover		Triggers when an element is being dragged over a valid drop target
ondragstart		Triggers at the start of a drag operation
ondrop		Triggers when a dragged element is being dropped

onmousedown		Triggers when a mouse button is pressed
onmousemove		Triggers when the mouse pointer moves
onmouseout		Triggers when the mouse pointer moves out of an element
onmouseover		Triggers when the mouse pointer moves over an element
onmouseup		Triggers when a mouse button is released
onmousewheel		Triggers when the mouse wheel is being rotated
onscroll		Triggers when an element's scrollbar is being scrolled

!

! ! !

Following events have been introduced in older versions of HTML but all the tags marked with are part of HTML-5.

!	. !	!
onabort		Triggers on an abort event
oncanplay		Triggers when a media can start play, but might has to stop for buffering
oncanplaythrough		Triggers when a media can be played to the end, without stopping for buffering

tutorialspoint

ondurationchange		Triggers when the length of a media is changed
onemptied		Triggers when a media resource element suddenly becomes empty.
onended		Triggers when a media has reached the end
onerror		Triggers when an error occurs
onloadeddata		Triggers when media data is loaded

onreadystatechange		Triggers when the ready-state changes
onseeked		Triggers when the seeking attribute of a media element is no longer true, and the seeking has ended
onseeking		Triggers when the seeking attribute of a media element is true, and the seeking has begun
onstalled		Triggers when there is an error in fetching media data
onsuspend		Triggers when the browser has been fetching media data, but stopped before the entire media file was fetched
ontimeupdate		Triggers when a media changes its playing position
onvolumechange		Triggers when a media changes the volume, also when volume is set to "mute"
onwaiting		Triggers when a media has stopped playing, but is expected to resume

CHAPTER 31

! ! !

Fonts are specific to platform. You will have different look and feel of a web page on different machines running different operating systems like Windows, Linux or Mac iOS. Here we are giving a list of fonts which are available in various operating systems.

HTML <font> tag is deprecated in version 4.0 onwards and now all fonts are set by using CSS. Here is the simple syntax of setting font of a body of web page.

```
body { font-family: "new century schoolbook"; }

or

<body style="font-family:new century schoolbook;">
```

!

!

!

```
<!DOCTYPE html>
<html>
<head>
<title>Font Setting Using CSS</title>
</head>
<body>

<p>Change any of the style and try it.</p>
<div style="font-family:verdana;">This is demo for font family</div>
<br />
<div style="font-size:120%;">This is demo for font size</div>
<br />
<div style="font-size:14pt;">This is demo for font size</div>

</body>
</html>
```

This will produce following result:

Change any of the style and try it.
This is demo for font family

This is demo for font size

This is demo for font size

!

! ! ! !

!

!	!	!
Andale Mono	Arial	Arial Bold
Arial Italic	Arial Bold Italic	Arial Black
Comic Sans MS	Comic Sans MS Bold	Courier New
Courier New Bold	Courier New Italic	Courier New Bold Italic
Georgia	Georgia Bold	Georgia Italic
Georgia Bold Italic	Impact	Lucida Console
Lucida Sans Unicode	Marlett	Minion Web
Symbol	Times New Roman	Times New Roman Bold
Times New Roman Italic	Times New Roman Bold Italic	Tahoma
Trebuchet MS	Trebuchet MS Bold	Trebuchet MS Italic
Trebuchet MS Bold Italic	Verdana	Verdana Bold

tutorialspoint

Verdana Italic	Verdana Bold Italic	Webdings

You can check example fonts here: **Microsoft Fonts Examples**. You can also have more information on Microsoft Fonts at http://www.microsoft.com/typography/fonts.

!

! ! ! !

!	!	!
American Typewriter	Andale Mono	
Arial	Arial Bla	
Baskerville	B	
Copperplate		
Luci		Optima
		Webdings
		Times
	Papyrus	Times New Roman
	Zapf Dingbats	Zapfino!

Hoefler Text Ornaments	Capitals	Charcoal
Gadget	Sand	

Following is the list of fonts supported by Macintosh System 7 and higher versions

You can check example fonts here: **Mac Fonts Examples**

! ! ! !

Following is the list of fonts supported by most Unix System variants

!	!	!
Charter	Clean	Courier
Fixed	Helvetica	Lucida
Lucida bright	Lucida Typewriter	New Century Schoolbook
Symbol	Terminal	Times
Utopia		

You can check example fonts here: **Unix Fonts Examples**

tutorialspoint

CHAPTER 32

! ! !

There are 2^7 = 128 printable characters which can be represented by different 7-BIT ASCII codes. Another set of characters are not for HTML representation but they are devised to control hardware.

Following tables list down all the 7-BIT ASCII codes and their equivalent HTML Entity Codes.

If you want to see equivalent HEX, OCT and extended set of ASCII codes then check next chapter.

. ! ! ! !
!

! !	!	! ! !
	space	
!	exclamation mark	!
"	quotation mark	"
#	number sign	#
$	dollar sign	$
%	percent sign	%
&	ampersand	&

tutorialspoint

'	apostrophe	'
(	left parenthesis	(
)	right parenthesis	)
*	asterisk	*
+	plus sign	+
,	comma	,
-	hyphen	-
.	period	.
/	slash	/
0	digit 0	0
1	digit 1	1
2	digit 2	2
3	digit 3	3
4	digit 4	4
5	digit 5	5
6	digit 6	6
7	digit 7	7
8	digit 8	8

9	digit 9	9
:	colon	:
;	semicolon	;
<	less-than	<
=	equals-to	=
>	greater-than	>
?	question mark	?
@	at sign	@
A	uppercase A	A
B	uppercase B	B
C	uppercase C	C
D	uppercase D	D
E	uppercase E	E
F	uppercase F	F
G	uppercase G	G
H	uppercase H	H
I	uppercase I	I
J	uppercase J	J

K	uppercase K	K
L	uppercase L	L
M	uppercase M	M
N	uppercase N	N
O	uppercase O	O
P	uppercase P	P
Q	uppercase Q	Q
R	uppercase R	R
S	uppercase S	S
T	uppercase T	T
U	uppercase U	U
V	uppercase V	V
W	uppercase W	W
X	uppercase X	X
Y	uppercase Y	Y
Z	uppercase Z	Z
[	left square bracket	[
\	backslash	\

]	right square bracket	]
^	caret	^
_	underscore	_
`	grave accent	`
a	lowercase a	a
b	lowercase b	b
c	lowercase c	c
d	lowercase d	d
e	lowercase e	e
f	lowercase f	f
g	lowercase g	g
h	lowercase h	h
i	lowercase i	i
j	lowercase j	j
k	lowercase k	k
l	lowercase l	l
m	lowercase m	m
n	lowercase n	n

o	lowercase o	o
p	lowercase p	p
q	lowercase q	q
r	lowercase r	r
s	lowercase s	s
t	lowercase t	t
u	lowercase u	u
v	lowercase v	v
w	lowercase w	w
x	lowercase x	x
y	lowercase y	y
z	lowercase z	z
{	left curly brace	{
|	vertical bar		
}	right curly brace	}
~	tilde	~

. ! ! ! ! !
!

! !	!	! ! !
NUL	null character	&#00;
SOH	start of header	&#01;
STX	start of text	&#02;
ETX	end of text	&#03;
EOT	end of transmission	&#04;
ENQ	enquiry	&#05;
ACK	acknowledge	&#06;
BEL	bell (ring)	&#07;
BS	backspace	&#08;
HT	horizontal tab	
LF	line feed
VT	vertical tab	&#11;
FF	form feed	
CR	carriage return
SO	shift out	&#14;
SI	shift in	&#15;
DLE	data link escape	&#16;

DC1	device control 1	&#17;
DC2	device control 2	&#18;
DC3	device control 3	&#19;
DC4	device control 4	&#20;
NAK	negative acknowledge	&#21;
SYN	synchronize	&#22;
ETB	end transmission block	&#23;
CAN	cancel	&#24;
EM	end of medium	&#25;
SUB	substitute	&#26;
ESC	escape	&#27;
FS	file separator	&#28;
GS	group separator	&#29;
RS	record separator	&#30;
US	unit separator	&#31;
DEL	delete (rubout)	&#127;

CHAPTER 33

! ! !

ASCII stands for American Standard Code for Information Interchange. There are 128 standard ASCII codes, each of which can be represented by a 7 digit binary number: 0000000 through 1111111.

Extended ASCII adds an additional 128 characters that vary between computers, programs and fonts.

! ! ! !

!	!	!	!	!	!	!
0	000	00	00000000	NUL	&#000;	Null char
1	001	01	00000001	SOH	&#001;	Start of Heading
2	002	02	00000010	STX	&#002;	Start of Text
3	003	03	00000011	ETX	&#003;	End of Text
4	004	04	00000100	EOT	&#004;	End of Transmission
5	005	05	00000101	ENQ	&#005;	Enquiry
6	006	06	00000110	ACK	&#006;	Acknowledgment
7	007	07	00000111	BEL	&#007;	Bell
8	010	08	00001000	BS	&#008;	Back Space

tutorialspoint

9	011	09	00001001	HT		Horizontal Tab
10	012	0A	00001010	LF
Line Feed
11	013	0B	00001011	VT	&#011;	Vertical Tab
12	014	0C	00001100	FF		Form Feed
13	015	0D	00001101	CR
Carriage Return
14	016	0E	00001110	SO	&#014;	Shift Out / X-On
15	017	0F	00001111	SI	&#015;	Shift In / X-Off
16	020	10	00010000	DLE	&#016;	Data Line Escape
17	021	11	00010001	DC1	&#017;	Device Control 1 (oft. XON)
18	022	12	00010010	DC2	&#018;	Device Control 2
19	023	13	00010011	DC3	&#019;	Device Control 3 (oft. XOFF)
20	024	14	00010100	DC4	&#020;	Device Control 4
21	025	15	00010101	NAK	&#021;	Negative Acknowledgement
22	026	16	00010110	SYN	&#022;	Synchronous Idle
23	027	17	00010111	ETB	&#023;	End of Transmit Block
24	030	18	00011000	CAN	&#024;	Cancel
25	031	19	00011001	EM	&#025;	End of Medium

26	032	1A	00011010	SUB	&#026;	Substitute
27	033	1B	00011011	ESC	&#027;	Escape
28	034	1C	00011100	FS	&#028;	File Separator
29	035	1D	00011101	GS	&#029;	Group Separator
30	036	1E	00011110	RS	&#030;	Record Separator
31	037	1F	00011111	US	&#031;	Unit Separator
32	040	20	00100000			Space
33	041	21	00100001	!	!	Exclamation mark
34	042	22	00100010	"	"	Double quotes
35	043	23	00100011	#	#	Number
36	044	24	00100100	$	$	Dollar
37	045	25	00100101	%	%	Procenttecken
38	046	26	00100110	&	&	Ampersand
39	047	27	00100111	'	'	Single quote
40	050	28	00101000	(	(	Open parenthesis
41	051	29	00101001	)	)	Close parenthesis
42	052	2A	00101010	*	*	Asterisk
43	053	2B	00101011	+	+	Plus

44	054	2C	00101100	,	,	Comma
45	055	2D	00101101	-	-	Hyphen
46	056	2E	00101110	.	.	Period, dot or full stop
47	057	2F	00101111	/	/	Slash or divide
48	060	30	00110000	0	0	Zero
49	061	31	00110001	1	1	One
50	062	32	00110010	2	2	Two
51	063	33	00110011	3	3	Three
52	064	34	00110100	4	4	Four
53	065	35	00110101	5	5	Five
54	066	36	00110110	6	6	Six
55	067	37	00110111	7	7	Seven
56	070	38	00111000	8	8	Eight
57	071	39	00111001	9	9	Nine
58	072	3A	00111010	:	:	Colon
59	073	3B	00111011	;	;	Semicolon
60	074	3C	00111100	<	<	Less than
61	075	3D	00111101	=	=	Equals

62	076	3E	00111110	>	>	Greater than
63	077	3F	00111111	?	?	Question mark
64	100	40	01000000	@	@	At symbol
65	101	41	01000001	A	A	Uppercase A
66	102	42	01000010	B	B	Uppercase B
67	103	43	01000011	C	C	Uppercase C
68	104	44	01000100	D	D	Uppercase D
69	105	45	01000101	E	E	Uppercase E
70	106	46	01000110	F	F	Uppercase F
71	107	47	01000111	G	G	Uppercase G
72	110	48	01001000	H	H	Uppercase H
73	111	49	01001001	I	I	Uppercase I
74	112	4A	01001010	J	J	Uppercase J
75	113	4B	01001011	K	K	Uppercase K
76	114	4C	01001100	L	L	Uppercase L
77	115	4D	01001101	M	M	Uppercase M
78	116	4E	01001110	N	N	Uppercase N
79	117	4F	01001111	O	O	Uppercase O

80	120	50	01010000	P	P	Uppercase P
81	121	51	01010001	Q	Q	Uppercase Q
82	122	52	01010010	R	R	Uppercase R
83	123	53	01010011	S	S	Uppercase S
84	124	54	01010100	T	T	Uppercase T

98	142	62	01100010	b	b	Lowercase b
99	143	63	01100011	c	c	Lowercase c
100	144	64	01100100	d	d	Lowercase d
101	145	65	01100101	e	e	Lowercase e
102	146	66	01100110	f	f	Lowercase f
103	147	67	01100111	g	g	Lowercase g
104	150	68	01101000	h	h	Lowercase h
105	151	69	01101001	i	i	Lowercase i
106	152	6A	01101010	j	j	Lowercase j
107	153	6B	01101011	k	k	Lowercase k
108	154	6C	01101100	l	l	Lowercase l
109	155	6D	01101101	m	m	Lowercase m
110	156	6E	01101110	n	n	Lowercase n
111	157	6F	01101111	o	o	Lowercase o
112	160	70	01110000	p	p	Lowercase p
113	161	71	01110001	q	q	Lowercase q
114	162	72	01110010	r	r	Lowercase r
115	163	73	01110011	s	s	Lowercase s

116	164	74	01110100	t	t	Lowercase t
117	165	75	01110101	u	u	Lowercase u
118	166	76	01110110	v	v	Lowercase v
119	167	77	01110111	w	w	Lowercase w
120	170	78	01111000	x	x	Lowercase x
121	171	79	01111001	y	y	Lowercase y
122	172	7A	01111010	z	z	Lowercase z
123	173	7B	01111011	{	{	Opening brace
124	174	7C	01111100	|			Vertical bar
125	175	7D	01111101	}	}	Closing brace
126	176	7E	01111110	~	~	Equivalency sign (tilde)
127	177	7F	01111111		&#127;	Delete

!

! ! !

Below is set of additional 128 Extended ASCII Codes according to ISO 8859-1, also called ISO Latin-1.

!	!	!	!	!	!	!	!
128	200	80	10000000	€	€	Euro sign
129	201	81	10000001			

130	202	82	10000010	‚	‚	Single low-9 quotation mark
131	203	83	10000011	ƒ	ƒ	Latin small letter f with hook
132	204	84	10000100	„	„	Double low-9 quotation mark
133	205	85	10000101	…	…	Horizontal ellipsis
134	206	86	10000110	†	†	Dagger
135	207	87	10000111	‡	‡	Double dagger
136	210	88	10001000	ˆ	ˆ	Modifier letter circumflex accent
137	211	89	10001001	‰	‰	Per mille sign
138	212	8A	10001010	Š	Š	Latin capital letter S with caron
139	213	8B	10001011	‹	‹	Single left-pointing angle quotation
140	214	8C	10001100	Œ	Œ	Latin capital ligature OE
141	215	8D	10001101			
142	216	8E	10001110	Ž	Ž	Latin capital letter Z with caron
143	217	8F	10001111			

144	220	90	10010000			
145	221	91	10010001	‘	‘	Left single quotation mark
146	222	92	10010010	’	’	Right single quotation mark
147	223	93	10010011	“	“	Left double quotation mark
148	224	94	10010100	”	”	Right double quotation mark
149	225	95	10010101	•	•	Bullet
150	226	96	10010110	–	–	En dash
151	227	97	10010111	—	—	Em dash
152	230	98	10011000	˜	˜	Small tilde
153	231	99	10011001	™	™	Trade mark sign
154	232	9A	10011010	š	š	Latin small letter S with caron
155	233	9B	10011011	›	›	Single right-pointing angle quotation mark
156	234	9C	10011100	œ	œ	Latin small ligature oe
157	235	9D	10011101			
158	236	9E	10011110	ž	ž	Latin small letter z with caron

159	237	9F	10011111	Ÿ	Ÿ	Latin capital letter Y with diaeresis
160	240	A0	10100000			Non-breaking space
161	241	A1	10100001	¡	¡	Inverted exclamation mark
162	242	A2	10100010	¢	¢	Cent sign
163	243	A3	10100011	£	£	Pound sign
164	244	A4	10100100	¤	¤	Currency sign
165	245	A5	10100101	¥	¥	Yen sign
166	246	A6	10100110	¦	¦	Pipe, Broken vertical bar
167	247	A7	10100111	§	§	Section sign
168	250	A8	10101000	¨	¨	Spacing diaeresis - umlaut
169	251	A9	10101001	©	©	Copyright sign
170	252	AA	10101010	ª	ª	Feminine ordinal indicator
171	253	AB	10101011	«	«	Left double angle quotes
172	254	AC	10101100	¬	¬	Not sign
173	255	AD	10101101			Soft hyphen

174	256	AE	10101110	®	®	Registered trade mark sign
175	257	AF	10101111	¯	¯	Spacing macron - overline
176	260	B0	10110000	°	°	Degree sign
177	261	B1	10110001	±	±	Plus-or-minus sign
178	262	B2	10110010	²	²	Superscript two - squared
179	263	B3	10110011	³	³	Superscript three - cubed
180	264	B4	10110100	´	´	Acute accent - spacing acute
181	265	B5	10110101	µ	µ	Micro sign
182	266	B6	10110110	¶	¶	Pilcrow sign - paragraph sign
183	267	B7	10110111	·	·	Middle dot - Georgian comma
184	270	B8	10111000	¸	¸	Spacing cedilla
185	271	B9	10111001	¹	¹	Superscript one
186	272	BA	10111010	º	º	Masculine ordinal indicator
187	273	BB	10111011	»	»	Right double angle quotes

188	274	BC	10111100	¼	¼	Fraction one quarter
189	275	BD	10111101	½	½	Fraction one half
190	276	BE	10111110	¾	¾	Fraction three quarters
191	277	BF	10111111	¿	¿	Inverted question mark
192	300	C0	11000000	À	À	Latin capital letter A with grave
193	301	C1	11000001	Á	Á	Latin capital letter A with acute
194	302	C2	11000010	Â	Â	Latin capital letter A with circumflex
195	303	C3	11000011	Ã	Ã	Latin capital letter A with tilde
196	304	C4	11000100	Ä	Ä	Latin capital letter A with diaeresis
197	305	C5	11000101	Å	Å	Latin capital letter A with ring above
198	306	C6	11000110	Æ	Æ	Latin capital letter AE
199	307	C7	11000111	Ç	Ç	Latin capital letter C with cedilla
200	310	C8	11001000	È	È	Latin capital letter E with grave

213	325	D5	11010101	Õ	Õ	Latin capital letter O with tilde
214	326	D6	11010110	Ö	Ö	Latin capital letter O with diaeresis
215	327	D7	11010111	×	×	Multiplication sign
216	330	D8	11011000	Ø	Ø	Latin capital letter O with slash
217	331	D9	11011001	Ù	Ù	Latin capital letter U with grave
218	332	DA	11011010	Ú	Ú	Latin capital letter U with acute
219	333	DB	11011011	Û	Û	Latin capital letter U with circumflex
220	334	DC	11011100	Ü	Ü	Latin capital letter U with diaeresis
221	335	DD	11011101	Ý	Ý	Latin capital letter Y with acute
222	336	DE	11011110	Þ	Þ	Latin capital letter THORN
223	337	DF	11011111	ß	ß	Latin small letter sharp s - ess-zed
224	340	E0	11100000	à	à	Latin small letter a with grave

225	341	E1	11100001	á	á	Latin small letter a with acute
226	342	E2	11100010	â	â	Latin small letter a with circumflex
227	343	E3	11100011	ã	ã	Latin small letter a with tilde
228	344	E4	11100100	ä	ä	Latin small letter a with diaeresis
229	345	E5	11100101	å	å	Latin small letter a with ring above
230	346	E6	11100110	æ	æ	Latin small letter ae
231	347	E7	11100111	ç	ç	Latin small letter c with cedilla
232	350	E8	11101000	è	è	Latin small letter e with grave
233	351	E9	11101001	é	é	Latin small letter e with acute
234	352	EA	11101010	ê	ê	Latin small letter e with circumflex
235	353	EB	11101011	ë	ë	Latin small letter e with diaeresis
236	354	EC	11101100	ì	ì	Latin small letter i with grave

237	355	ED	11101101	í	í	Latin small letter i with acute
238	356	EE	11101110	î	î	Latin small letter i with circumflex
239	357	EF	11101111	ï	ï	Latin small letter i with diaeresis
240	360	F0	11110000	ð	ð	Latin small letter eth
241	361	F1	11110001	ñ	ñ	Latin small letter n with tilde
242	362	F2	11110010	ò	ò	Latin small letter o with grave
243	363	F3	11110011	ó	ó	Latin small letter o with acute
244	364	F4	11110100	ô	ô	Latin small letter o with circumflex
245	365	F5	11110101	õ	õ	Latin small letter o with tilde
246	366	F6	11110110	ö	ö	Latin small letter o with diaeresis
247	367	F7	11110111	÷	÷	Division sign
248	370	F8	11111000	ø	ø	Latin small letter o with slash
249	371	F9	11111001	ù	ù	Latin small letter u with grave

250	372	FA	11111010	ú	ú	Latin small letter u with acute
251	373	FB	11111011	û	û	Latin small letter u with circumflex
252	374	FC	11111100	ü	ü	Latin small letter u with diaeresis
253	375	FD	11111101	ý	ý	Latin small letter y with acute
254	376	FE	11111110	þ	þ	Latin small letter thorn
255	377	FF	11111111	ÿ	ÿ	Latin small letter y with diaeresis

CHAPTER 34

! ! !

The following table shows the 16 color names that were introduced in HTML 3.2:

! !	! !	!	!
aqua	#00ffff		Demo
black	#000000		Demo
blue	#0000ff		Demo
fuchsia	#ff00ff		Demo
green	#008000		Demo
gray	#808080		Demo
lime	#00ff00		Demo
maroon	#800000		Demo
navy	#000080		Demo
olive	#808000		Demo
purple	#800080		Demo
red	#ff0000		Demo

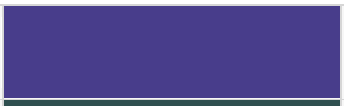
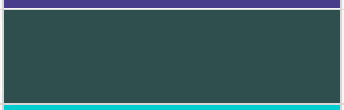
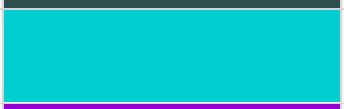
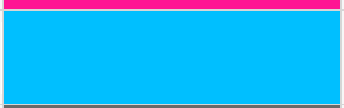
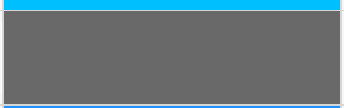
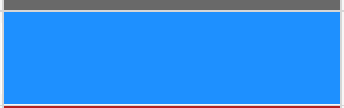
silver	#c0c0c0		Demo
teal	#008080		Demo
white	#ffffff		Demo
yellow	#ffff00		Demo

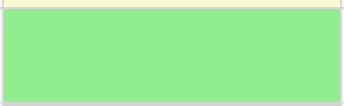
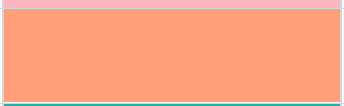
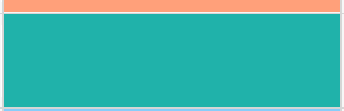
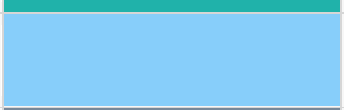
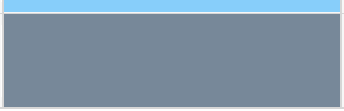
There are other colors which are not part of HTML or XHTML but they are supported by most of the versions of major browsers.

! !	! !	!	!
aliceblue	#f0f8ff		Demo
antiquewhite	#faebd7		Demo
aquamarine	#7fffd4		Demo
azure	#f0ffff		Demo
beige	#f5f5dc		Demo
bisque	#ffe4c4		Demo
blanchedalmond	#ffebcd		Demo
blueviolet	#8a2be2		Demo
brown	#a52a2a		Demo
burlywood	#deb887		Demo
cadetblue	#5f9ea0		Demo
chartreuse	#7fff00		Demo

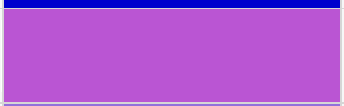
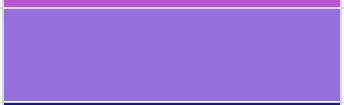
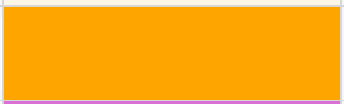
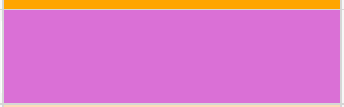
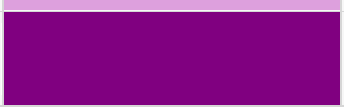
chocolate	#d2691e		Demo
coral	#ff7f50		Demo
	#6495ed		Demo
cornsilk	#fff8dc		Demo
crimson	#dc143c		Demo
cyan	#00ffff		Demo

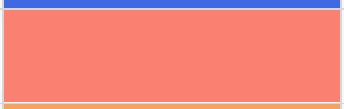
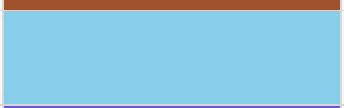
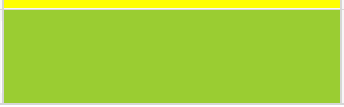
tutorialspoint

darkslateblue	#483d8b		Demo
darkslategray	#2f4f4f		Demo
darkturquoise	#00ced1		Demo
darkviolet	#9400d3		Demo
deeppink	#ff1493		Demo
deepskyblue	#00bfff		Demo
dimgray	#696969		Demo
dodgerblue	#1e90ff		Demo
firebrick	#b22222		Demo
floralwhite	#fffaf0		Demo
forestgreen	#228b22		Demo
gainsboro	#dcdcdc		Demo
ghostwhite	#f8f8ff		Demo
gold	#ffd700		Demo
goldenrod	#daa520		Demo
gray	#808080		Demo
greenyellow	#adff2f		Demo
honeydew	#f0fff0		Demo
hotpink	#ff69b4		Demo

indianred	#cd5c5c		Demo
indigo	#4b0082		Demo
ivory	#fffff0		Demo
khaki	#f0e68c		Demo
lavender	#e6e6fa		Demo
lavenderblush	#fff0f5		Demo
lawngreen	#7cfc00		Demo
lemonchiffon	#fffacd		Demo
lightblue	#add8e6		Demo
lightcoral	#f08080		Demo
lightcyan	#e0ffff		Demo
lightgoldenrodyellow	#fafad2		Demo
lightgreen	#90ee90		Demo
lightgrey	#d3d3d3		Demo
lightpink	#ffb6c1		Demo
lightsalmon	#ffa07a		Demo
lightseagreen	#20b2aa		Demo
lightskyblue	#87cefa		Demo
lightslategray	#778899		Demo

tutorialspoint

lightsteelblue	#b0c4de		Demo
lightyellow	#ffffe0		Demo
limegreen	#32cd32		Demo
linen	#faf0e6		Demo
magenta	#ff00ff		Demo
mediumblue	#0000cd		Demo
mediumorchid	#ba55d3		Demo
mediumpurple	#9370db		Demo
midnightblue	#191970		Demo
mistyrose	#ffe4e1		Demo
moccasin	#ffe4b5		Demo
oldlace	#fdf5e6		Demo
orange	#ffa500		Demo
orchid	#da70d6		Demo
peachpuff	#ffdab9		Demo
peru	#cd853f		Demo
pink	#ffc0cb		Demo
plum	#dda0dd		Demo
purple	#800080		Demo

rosybrown	#bc8f8f		Demo
royalblue	#4169e1		Demo
salmon	#fa8072		Demo
sandybrown	#f4a460		Demo
seagreen	#2e8b57		Demo
sienna	#a0522d		Demo
skyblue	#87ceeb		Demo
slateblue	#6a5acd		Demo
steelblue	#4682b4		Demo
tan	#d2b48c		Demo
thistle	#d8bfd8		Demo
tomato	#ff6347		Demo
violet	#ee82ee		Demo
wheat	#f5deb3		Demo
whitesmoke	#f5f5f5		Demo
yellow	#ffff00		Demo
yellowgreen	#9acd32		Demo

!

tutorialspoint

! !

Some characters are reserved in HTML and they have special meaning when used in HTML document. For example, you cannot use the greater than and less than signs or angle brackets within your HTML text because the browser will treat them differently and will try to draw a meaning related to HTML tag.

HTML processors must support following five special characters listed in the table that follows.

```
<body>
&lt;div id=&quot;character&quot;&gt;
</body>
</html>
```

This will produce following result:

```
<div id="character">
```

There is also a long list of special characters in HTML 4.0. In order for these to appear in your document, you can use either the numerical codes or the entity names. For example, to insert a copyright symbol you can use either of the following:

```
&copy; 2007
or
&#169; 2007
```

! .2! ! !
!

!	!	! !	! !
	non-breaking space		
¡	inverted exclamation mark	¡	¡
¤	currency	¤	¤
¢	cent	¢	¢
£	pound	£	£
¥	yen	¥	¥
¦	broken vertical bar	¦	¦
§	section	§	§

¨	spacing diaeresis	¨	¨
©	copyright	©	©
ª	feminine ordinal indicator	ª	ª
«	angle quotation mark (left)	«	«
¬	negation	¬	¬

¹	superscript 1	¹	¹
º	masculine ordinal indicator	º	º
»	angle quotation mark (right)	»	»
¼	fraction 1/4	¼	¼
½	fraction 1/2	½	½
¾	fraction 3/4	¾	¾
¿	inverted question mark	¿	¿
×	multiplication	×	×
÷	division	÷	÷

! . 2! ! !
!

!	!	! !	! !
À	capital a, grave accent	À	À
Á	capital a, acute accent	Á	Á
Â	capital a, circumflex accent	Â	Â
Ã	capital a, tilde	Ã	Ã
Ä	capital a, umlaut mark	Ä	Ä
Å	capital a, ring	Å	Å

Æ	capital ae	Æ	Æ
Ç	capital c, cedilla	Ç	Ç
È	capital e, grave accent	È	È
É			

Ù	capital u, grave accent	Ù	Ù
Ú	capital u, acute accent	Ú	Ú
Û	capital u, circumflex accent	Û	Û
Ü	capital u, umlaut mark	Ü	Ü
Ý	capital y, acute accent	Ý	Ý
Þ	capital THORN, Icelandic	Þ	Þ
ß	small sharp s, German	ß	ß
à	small a, grave accent	à	à
á	small a, acute accent	á	á
â	small a, circumflex accent	â	â
ã	small a, tilde	ã	ã
ä	small a, umlaut mark	ä	ä
å	small a, ring	å	å
æ	small ae	æ	æ
ç	small c, cedilla	ç	ç
è	small e, grave accent	è	è
é	small e, acute accent	é	é
ê	small e, circumflex accent	ê	ê

ë	small e, umlaut mark	ë	ë
ì	small i, grave accent	ì	ì
í	small i, acute accent	í	í
î	small i, circumflex accent	î	î
ï	small i, umlaut mark	ï	ï
ð	small eth, Icelandic	ð	ð
ñ	small n, tilde	ñ	ñ
ò	small o, grave accent	ò	ò
ó	small o, acute accent	ó	ó
ô	small o, circumflex accent	ô	ô
õ	small o, tilde	õ	õ
ö	small o, umlaut mark	ö	ö
ø	small o, slash	ø	ø

þ	small thorn, Icelandic	þ	þ
ÿ	small y, umlaut mark	ÿ	ÿ

! ! ! ! ! !
!

!	!	! !	! !
Œ	capital ligature OE	Œ	Œ
œ	small ligature oe	œ	œ
Š	capital S with caron	Š	Š
š	small S with caron	š	š
Ÿ	capital Y with diaeres	Ÿ	Ÿ
ˆ	modifier letter circumflex accent	ˆ	ˆ
˜	small tilde	˜	˜
	en space		
	em space		
	thin space		
	zero width non-joiner	‌	‌
	zero width joiner	‍	‍
	left-to-right mark	‎	‎

	right-to-left mark	‏	‏
–	en dash	–	–
—	em dash	—	—
‘	left single quotation mark	‘	‘
’	right single quotation mark	’	’
,	single low-9 quotation mark	‚	‚
“	left double quotation mark	“	“
”	right double quotation mark	”	”
„	double low-9 quotation mark	„	„
†	dagger	†	†
‡	double dagger	‡	‡

CHAPTER

36

! ! !

MIME (Multipurpose Internet Mail Extension) media types were originally devised so that e-mails could include information other than plain text. MIME media types indicate the following things:

- How different parts of a message, such as text and attachments, are combined into the message.
- The way in which each part of the message is specified.
- The way different items are encoded for transmission so that even software that was designed to work only with ASCII text can process the message.

Now MIME types are not just for use with e-mail; they have been adopted by Web servers as a way to tell Web browsers what type of material was being sent to them so that they can cope with that kind of messages correctly.

MIME content types consist of two parts:

- Main type
- Sub-type

The main type is separated from the subtype by a forward slash character. For example, text/html for HTML.

This chapter is organized for the main types:

- **text**
- **image**
- **multipart**
- **audio**
- **video**
- **message**
- **model**
- **application**

For example, the text main type contains types of plain text files, such as:

- text/plain for plain text files
- text/html for HTML files
- text/rtf for text files using rich text formatting

MIME types are officially supposed to be assigned and listed by the Internet Assigned Numbers Authority (IANA).

Many of the popular MIME types in this list (all those begin with "x-") are not assigned by the IANA and do not have official status. You can see the list of official MIME types at http://www.iana.org/assignments/media-types/. Those preceded with **/** are vendor-specific.

When specifying the MIME type of a content-type field you can also indicate the character set for the text being used. If you do not specify a character set, the default is US-ASCII. For example:

```
content-type:text/plain; charset=iso-8859-1
```

CHAPTER

37

! ! !

URL encoding is the practice of translating unprintable characters or characters with special meaning within URLs to a representation that is unambiguous and universally accepted by web browsers and servers. These characters include:

- ! ! - Unprintable characters typically used for output control. Character ranges 00-1F hex (0-31 decimal) and 7F (127 decimal). A complete encoding table is given below.
- . ! ! - These characters are beyond the ASCII character set of 128 characters. This range is part of the ISO-Latin character set and includes the entire "top half" of the ISO-Latin set 80-FF hex (128-255 decimal). A complete encoding table is given below.
- ! - These are special characters such as the dollar sign, ampersand, plus, common, forward slash, colon, semi-colon, equals sign, question mark, and "at" symbol. All of these can have different meanings inside a URL so need to be encoded. A complete encoding table is given below.
- ! - These are space, quotation marks, less than symbol, greater than symbol, pound character, percent character, Left Curly Brace, Right Curly Brace , Pipe, Backslash, Caret, Tilde, Left Square Bracket , Right Square Bracket, Grave Accent. These characters present the possibility of being misunderstood within URLs for various reasons. These characters should always be encoded. A complete encoding table is given below.

The encoding notation replaces the desired character with three characters: a percent sign and two hexadecimal digits that correspond to the position of the character in the ASCII character set.

!

One of the most common special characters is a white space. You can't type a space in a URL directly. A space position in the character set is 20

tutorialspoint

hexadecimal. So you can use %20 in place of a space when passing your request to the server.

```
http://www.example.com/new%20pricing.htm
```

This URL actually retrieves a document named *"new pricing.htm"* from the www.example.com

! ! ! !

This includes the encoding for character ranges 00-1F hex (0-31 decimal) and 7F (127 decimal)

!	! !	!	! !
0	00		

tutorialspoint

12	0c		%0c
13	0d	carriage return	%0d
14	0e		%0e
15	0f		%0f
16	10		%10
17	11		%11
18	12		%12
19	13		%13
20	14		%14
21	15		%15
22	16		%16
23	17		%17
24	18		%18
25	19		%19
26	1a		%1a
27	1b		%1b
28	1c		%1c
29	1d		%1d

30	1e		%1e
31	1f		%1f
127	7f		%7f

. ! ! ! !

This includes the encoding for the entire "top half" of the ISO-Latin set 80-FF hex (128-255 decimal.)

!	! !	!	! !
128	80	€	%80
129	81	•	%81
130	82	‚	%82
131	83	*ƒ*	%83
132	84	„	%84
133	85	…	%85
134	86	†	%86
135	87	‡	%87
136	88	ˆ	%88
137	89	‰	%89
138	8a	Š	%8a

139	8b	‹	

157	9d	•	%9d
158	9e	ž	%9e
159	9f	Ÿ	%9f
160	a0		%a0
161	a1	¡	%a1
162	a2	¢	%a2
163	a3	£	%a3
164	a4	¤	%a4
165	a5	¥	%a5
166	a6	¦	%a6
167	a7	§	%a7
168	a8	¨	%a8
169	a9	©	%a9
170	aa	ª	%aa
171	ab	«	%ab
172	ac	¬	%ac
173	ad		%ad
174	ae	®	%ae

tutorialspoint

175	af	¯	%af
176	b0	°	%b0
177	b1	±	%b1
178	b2	²	%b2
179	b3	³	%b3
180	b4	´	%b4
181	b5	µ	%b5
182	b6	¶	%b6
183	b7	·	%b7
184	b8	¸	%b8
185	b9	¹	%b9
186	ba	º	%ba
187	bb	»	%bb
188	bc	¼	%bc
189	bd	½	%bd
190	be	¾	%be
191	bf	¿	%bf
192	c0	À	%c0

193	c1	Á	%c1
194	c2	Â	%c2
195	c3	Ã	%c3
196	c4	Ä	%c4
197	c5	Å	%c5
198	c6	Æ	%v6
199	c7	Ç	%c7
200	c8	È	%c8
201	c9	É	%c9
202	ca	Ê	%ca
203	cb	Ë	%cb
204	cc	Ì	%cc
205	cd	Í	%cd
206	ce	Î	%ce
207	cf	Ï	%cf
208	d0	Ð	%d0
209	d1	Ñ	%d1
210	d2	Ò	%d2

211	d3	Ó	%d3
212	d4	Ô	%d4
213	d5	Õ	%d5
214	d6	Ö	%d6
215	d7	×	%d7
216	d8	Ø	%d8
217	d9	Ù	%d9
218	da	Ú	%da
219	db	Û	%db
220	dc	Ü	%dc
221	dd	Ý	%dd
222	de	Þ	%de
223	df	ß	%df
224	e0	à	%e0
225	e1	á	%e1
226	e2	â	%e2
227	e3	ã	%e3
228	e4	ä	%e4

229	e5	å	%e5
230	e6	æ	%e6
231	e7	ç	%e7
232	e8	è	%e8
233	e9	é	%e9
234	ea	ê	%ea
235	eb	ë	%eb
236	ec	ì	%ec
237	ed	í	%ed
238	ee	î	%ee
239	ef	ï	%ef
240	f0	ð	%f0
241	f1	ñ	%f1
242	f2	ò	%f2
243	f3	ó	%f3
244	f4	ô	%f4
245	f5	õ	%f5
246	f6	ö	%f6

247	f7	÷	%f7
248	f8	ø	%f8
249	f9	ù	%f9
250	fa	ú	%fa
251	fb	û	%fb
252	fc	ü	%fc
253	fd	ý	%fd
254	fe	þ	%fe
255	ff	ÿ	%ff

! ! !

Following is the table to be used to encode reserved characters.

!	! !	!	! !
36	24	$	%24
38	26	&	%26
43	2b	+	%2b
44	2c	,	%2c
47	2f	/	%2f
58	3a	:	%3a

126	7e	~	%7e
91	5b	[	%5b
93	5d	]	%5d
96	60	`	%60

CHAPTER 38

! ! !

The following is a draft list of language code correspondences between ISO codes, Microsoft codes, and Macintosh codes. Source of this information is Unicode Consortium.

! ! ! -! !

!

!	! !	! !	! !
Abkhazian	ab		
Afar	aa		
Afrikaans	af	LANG_AFRIKAANS	0x36
Albanian	sq	LANG_ALBANIAN	0x1c
Amharic	am	(no constant defined)	0x5e
Arabic	ar	LANG_ARABIC	0x01
Armenian	hy	LANG_ARMENIAN	0x2b
Assamese	as	LANG_ASSAMESE	0x4d
Aymara	ay		

Azerbaijani	az	LANG_AZERI	0x2c
Bashkir	ba		
Basque	eu	LANG_BASQUE	0x2d
Bengali (Bangla)	bn	LANG_BENGALI	0x45
Bhutani	dz		
Bihari	bh		
Bislama	bi		
Breton	br		
Bulgarian	bg	LANG_BULGARIAN	0x02
Burmese	my	(no constant defined)	0x55
Byelorussian (Belarusian)	be	LANG_BELARUSIAN	0x23
Cambodian	km	(no constant defined)	0x53
Catalan	ca	LANG_CATALAN	0x03
Cherokee		(no constant defined)	0x5c
Chewa			
Chinese (Simplified)	zh	LANG_CHINESE (SUBLANG_CHINESE_SIMPLIFIED)	0x04 (0x0804)

Chinese (Traditional)	zh	LANG_CHINESE (SUBLANG_CHINESE_TRADITIONAL)	0x04 (0x0404)
Corsican	co		
Croatian	hr	LANG_CROATIAN	0x1a
Czech	cs	LANG_CZECH	0x05
Danish	da	LANG_DANISH	0x06
Divehi		LANG_DIVEHI	0x65
Dutch	nl	LANG_DUTCH	0x13
Edo		(no constant defined)	0x66
English	en	LANG_ENGLISH	0x09
Esperanto	eo		
Estonian	et	LANG_ESTONIAN	0x25
Faeroese	fo	LANG_FAEROESE	0x38
Farsi	fa	LANG_FARSI	0x29
Fiji	fj		
Finnish	fi	LANG_FINNISH	0x0b
Flemish		LANG_DUTCH (SUBLANG_DUTCH_BELGIAN)	0x13 (0x0813)
French	fr	LANG_FRENCH	0x0c

Frisian	fy	(no constant defined)	0x62
Fulfulde		(no constant defined)	0x67
Galician	gl	LANG_GALICIAN	0x56
Gaelic (Scottish)	gd	(no constant defined)	0x3c (0x043c)
Gaelic (Manx)	gv		
Georgian	ka	LANG_GEORGIAN	0x37
German	de	LANG_GERMAN	0x07
Greek	el	LANG_GREEK	0x08
Greenlandic	kl		
Guarani	gn	(no constant defined)	0x74
Gujarati	gu	LANG_GUJARATI	0x47
Hausa	ha	(no constant defined)	0x68
Hawaiian		(no constant defined)	0x75
Hebrew	he, iw*	LANG_HEBREW	0x0d
Hindi	hi	LANG_HINDI	0x39
Hungarian	hu	LANG_HUNGARIAN	0x0e
Ibibio		(no constant defined)	0x69

Icelandic	is	LANG_ICELANDIC	0x0f
Igbo		(no constant defined)	0x70
Indonesian	id, in*	LANG_INDONESIAN	0x21
Interlingua	ia		
Interlingue	ie		
Inuktitut	iu	(no constant defined)	0x5d
Inupiak	ik		
Irish	ga	(no constant defined)	0x3c (0x083c)
Italian	it	LANG_ITALIAN	0x10
Japanese	ja	LANG_JAPANESE	0x11
Javanese	jv		
Kannada	kn	LANG_KANNADA	0x4b
Kanuri		(no constant defined)	0x71
Kashmiri	ks	LANG_KASHMIRI	0x60
Kazakh	kk	LANG_KAZAK	0x3f
Kinyarwanda (Ruanda)	rw		
Kirghiz	ky	LANG_KYRGYZ	0x40

Kirundi (Rundi)	rn		
Konkani		LANG_KONKANI	0x57
Korean	ko	LANG_KOREAN	0x12
Kurdish	ku		
Laothian	lo	(no constant defined)	0x54
Latin	la	(no constant defined)	0x76
Latvian (Lettish)	lv	LANG_LATVIAN	0x26
Limburgish (Limburger)	li		
Lingala	ln		
Lithuanian	lt	LANG_LITHUANIAN	0x27
Macedonian	mk	LANG_MACEDONIAN	0x2f
Malagasy	mg		
Malay	ms	LANG_MALAY	0x3e
Malayalam	ml	LANG_MALAYALAM	0x4c
		LANG_MANIPURI	0x58
Maltese	mt	(no constant defined)	0x3a
Maori	mi		

Marathi	mr	LANG_MARATHI	0x4e
Moldavian	mo		
Mongolian	mn	LANG_MONGOLIAN	0x50
Nauru	na		
Nepali	ne	LANG_NEPALI	0x61
Norwegian	no	LANG_NORWEGIAN	0x14
Occitan	oc		
Oriya	or	LANG_ORIYA	0x48
Oromo (Afan, Galla)	om	(no constant defined)	0x72
Papiamentu		(no constant defined)	0x79
Pashto (Pushto)	ps	(no constant defined)	0x63
Polish	pl	LANG_POLISH	0x15
Portuguese	pt	LANG_PORTUGUESE	0x16
Punjabi	pa	LANG_PUNJABI	0x46
Quechua	qu		
Rhaeto-Romance	rm	(no constant defined)	0x17
Romanian	ro	LANG_ROMANIAN	0x18

Russian	ru	LANG_RUSSIAN	0x19
Sami (Lappish)		(no constant defined)	0x3b
Samoan	sm		
Sangro	sg		
Sanskrit	sa	LANG_SANSKRIT	0x4f
Serbian	sr	LANG_SERBIAN (SUBLANG_SERBIAN_LATIN or SUBLANG_SERBIAN_CYRILLIC)	0x1a (0x081a or 0x0c1a)
Serbo-Croatian	sh		
Sesotho	st		
Setswana	tn		
Shona	sn		
Sindhi	sd	LANG_SINDHI	0x59
Sinhalese	si	(no constant defined)	0x5b
Siswati	ss		
Slovak	sk	LANG_SLOVAK	0x1b
Slovenian	sl	LANG_SLOVENIAN	0x24
Somali	so	(no constant defined)	0x77

Spanish	es	LANG_SPANISH	0x0a
Sundanese	su		
Swahili (Kiswahili)	sw	LANG_SWAHILI	0x41
Swedish	sv	LANG_SWEDISH	0x1d
Syriac		LANG_SYRIAC	0x5a
Tagalog	tl	(no constant defined)	0x64
Tajik	tg	(no constant defined)	0x28
Tamazight		(no constant defined)	0x5f
Tamil	ta	LANG_TAMIL	0x49
Tatar	tt	LANG_TATAR	0x44
Telugu	te	LANG_TELUGU	0x4a
Thai	th	LANG_THAI	0x1e
Tibetan	bo	(no constant defined)	0x51
Tigrinya	ti	(no constant defined)	0x73
Tonga	to		
Tsonga	ts	(no constant defined)	0x31
Turkish	tr	LANG_TURKISH	0x1f

Turkmen	tk	(no constant defined)	0x42
Twi	tw		
Uighur	ug		
Ukrainian	uk	LANG_UKRAINIAN	0x22
Urdu	ur	LANG_URDU	0x20
Uzbek	uz	LANG_UZBEK	0x43
Venda		(no constant defined)	0x33
Vietnamese	vi	LANG_VIETNAMESE	0x2a
Volap?k	vo		
Welsh	cy	(no constant defined)	0x52
Wolof	wo		
Xhosa	xh	(no constant defined)	0x34
Yi		(no constant defined)	0x78
Yiddish	yi, ji*	(no constant defined)	0x3d
Yoruba	yo	(no constant defined)	0x6a
Zulu	zu	(no constant defined)	0x35

! ! ! -! !

!	! !	! !	! !
Abkhazian	ab		
Afar	aa		
Afrikaans	af	langAfricaans	141
Albanian	sq	langAlbanian	36
Amharic	am	langAmharic	85
Arabic	ar	langArabic	12
Armenian	hy	langArmenian	51
Assamese	as	langAssamese	68
Aymara	ay	langAymara	134
Azerbaijani	az	langAzerbaijani(Cyrllic), langAzerbaijanAr(Arabic)	49(C), 50(A)
Bashkir	ba		
Basque	eu	langBasque	129
Bengali (Bangla)	bn	langBengali	67
Bhutani	dz	langDzongkha	137
Bihari	bh		
Bislama	bi		

Breton	br	langBreton	142
Bulgarian	bg	langBulgarian	44
Burmese	my	langBurmese	77
Byelorussian (Belarusian)	be	langByelorussian	46
Cambodian	km	langKhmer	78
Catalan	ca	langCatalan	130
Cherokee			
Chewa		langChewa	92
Chinese (Simplified)	zh	langSimpChinese	33
Chinese (Traditional)	zh	langTradChinese	19
Corsican	co		
Croatian	hr	langCroatian	18
Czech	cs	langCzech	38
Danish	da	langDanish	7
Divehi			
Dutch	nl	langDutch	4
Edo			

English	en	langEnglish	0
Esperanto	eo	langEsperanto	94
Estonian	et	langEstonian	27
Faeroese	fo	langFaeroese	30
Farsi	fa	langFarsi, langPersian	31
Fiji	fj		
Finnish	fi	langFinnish	13
Flemish		langFlemish	34
French	fr	langFrench	1
Frisian	fy		
Fulfulde			
Galician	gl	langGalician	140
Gaelic (Scottish)	gd	langScottishGaelic	144
Gaelic (Manx)	gv	langManxGaelic	145
Georgian	ka	langGeorgian	52
German	de	langGerman	2
Greek	el	langGreek (monotonic), langGreekPoly (polytonic)	14(m), 148(p)

Greenlandic	kl		
Guarani	gn	langGuarani	133
Gujarati	gu	langGujarati	69
Hausa	ha		
Hawaiian			
Hebrew	he, iw*	langHebrew	10
Hindi	hi	langHindi	21
Hungarian	hu	langHungarian	26
Ibibio			
Icelandic	is	langIcelandic	15
Igbo			
Indonesian	id, in*	langIndonesian	81
Interlingua	ia		
Interlingue	ie		
Inuktitut	iu	langInuktitut	143
Inupiak	ik		
Irish	ga	langIrishGaelic (normal), langIrishGaelicScr (dots above)	35, 146

Italian	it	langItalian	3
Japanese	ja	langJapanese	11
Javanese	jv	langJavaneseRom	138
Kannada	kn	langKannada	73
Kanuri			
Kashmiri	ks	langKashmiri	61
Kazakh	kk	langKazakh	48
Kinyarwanda (Ruanda)	rw	langKiryarwanda (langRuanda)	90
Kirghiz	ky	langKirghiz	54
Kirundi (Rundi)	rn	langRundi	91
Konkani			
Korean	ko	langKorean	23
Kurdish	ku	langKurdish	60
Laothian	lo	langLao	79
Latin	la	langLatin	131
Latvian (Lettish)	lv	langLatvian	28
Limburgish (Limburger)	li		

Lingala	ln		
Lithuanian	lt	langLithuanian	24
Macedonian	mk	langMacedonian	43
Malagasy	mg	langMalagasy	93
Malay	ms	langMalayRoman(Latin), langMalayArabic(Arabic)	83(L), 84(A)
Malayalam	ml	langMalayalam	72
Maltese	mt	langMaltese	16
Maori	mi		
Marathi	mr	langMarathi	66
Moldavian	mo	langMoldavian	53
Mongolian	mn	langMongolian(Mongolian), langMongolianCyr(Cyrillic)	57(M), 58(C)
Nauru	na		
Nepali	ne	langNepali	64
Norwegian	no	langNorwegian	9
Occitan	oc		
Oriya	or	langOriya	71

Oromo (Afan, Galla)	om	langOromo (langGalla)	87
Papiamentu			
Pashto (Pushto)	ps	langPashto	59
Polish	pl	langPolish	25
Portuguese	pt	langPortuguese	8
Punjabi	pa	langPunjabi	70
Quechua	qu	langQuechua	132
Rhaeto-Romance	rm		
Romanian	ro	langRomanian	37
Russian	ru	langRussian	32
Sami (Lappish)		langSami (langLappish)	29
Samoan	sm		
Sangro	sg		
Sanskrit	sa	langSanskrit	65
Serbian	sr	langSerbian	42
Serbo-Croatian	sh		
Sesotho	st		

Setswana	tn		
Shona	sn		
Sindhi	sd	langSindhi	62
Sinhalese	si	langSinhalese	76
Siswati	ss		
Slovak	sk	langSlovak	39
Slovenian	sl	langSlovenian	40
Somali	so	langSomali	88
Spanish	es	langSpanish	6
Sundanese	su	langSundaneseRom	139
Swahili (Kiswahili)	sw	langSwahili	89
Swedish	sv	langSwedish	5
Syriac			
Tagalog	tl	langTagalog	82
Tajik	tg	langTajiki	55
Tamazight			
Tamil	ta	langTamil	74

Tatar	tt	langTatar	135
Telugu	te	langTelugu	75
Thai	th	langThai	22
Tibetan	bo	langTibetan	63
Tigrinya	ti	langTigrinya	86
Tonga	to	langTongan	147
Tsonga	ts		
Turkish	tr	langTurkish	17
Turkmen	tk	langTurkmen	56
Twi	tw		
Uighur	ug	langUighur	136
Ukrainian	uk	langUkrainian	45
Urdu	ur	langUrdu	20
Uzbek	uz	langUzbek	47
Venda			
Vietnamese	vi	langVietnamese	80
Volap?k	vo		
Welsh	cy	langWelsh	128

Wolof	wo		
Xhosa	xh		
Yi			
Yiddish	yi, ji*	langYiddish	41
Yoruba	yo		
Zulu	zu		

CHAPTER

39

! ! !

Character encoding is a method of converting bytes into characters. To validate or display an HTML document properly, a program must choose a proper character encoding.

The most common character set or character encoding in use on computers is ASCII! (American Standard Code for Information Interchange), and this is probably the most widely used character set for encoding text electronically.

ASCII encoding supports only the upper and lowercase Latin alphabet, the numbers 0-9, and some extra characters which make a total of 128 characters in all. You can have a look at complete set of Printable ASCII Characters

However, many languages use either accented Latin characters or completely different alphabets. ASCII does not address these characters; therefore you need to learn about character encodings if you want to use any non-ASCII characters.

The International Standards Organization created a range of character sets to deal with different national characters. For the documents in English and most other Western European languages, the widely supported encoding ISO-8859-1 is used.

Here is the list of Character Set being used around the world along with their description.

! !	!
ISO-8859-1	Latin alphabet part 1 Covering North America, Western Europe, Latin America, The Caribbean, Canada, Africa
ISO-8859-2	Latin alphabet part 2 Covering Eastern Europe

tutorialspoint

ISO-8859-3	Latin alphabet part 3 Covering SE Europe, Esperanto, miscellaneous others
ISO-8859-4	Latin alphabet part 4 Covering Scandinavia/Baltics (and others not in ISO-8859-1)
ISO-8859-5	Latin/Cyrillic alphabet part 5
ISO-8859-6	Latin/Arabic alphabet part 6
ISO-8859-7	Latin/Greek alphabet part 7
ISO-8859-8	Latin/Hebrew alphabet part 8
ISO-8859-9	Latin 5 alphabet part 9 Same as ISO-8859-1 except Turkish characters replace Icelandic ones
ISO-8859-10	Latin 6 Latin 6 Lappish, Nordic, and Eskimo
ISO-8859-15	The same as ISO-8859-1 but with more characters added
ISO-2022-JP	Latin/Japanese alphabet part 1
ISO-2022-JP-2	Latin/Japanese alphabet part 2
ISO-2022-KR	Latin/Korean alphabet part 1

The Unicode Consortium was then set up to devise a way to show all characters of different languages, rather than have these different incompatible character codes for different languages.

Therefore, if you want to create documents that use characters from multiple character sets, you will be able to do so using the single Unicode character encodings.

Unicode, therefore, specifies encodings that can deal with a string in special ways so as to make enough space for the huge character set it encompasses. These are known as UTF-8, UTF-16, and UTF-32.

! !	!
UTF-8	A Unicode Translation Format that comes in 8-bit units that is, it comes in bytes. A character in UTF8 can be 1 to 4 bytes long, making UTF8 variable width.
UTF-16	A Unicode Translation Format comes in 16-bit units that is, it comes in shorts. It can be 1 or 2 shorts long, making UTF16 variable width.
UTF-32	A Unicode Translation Format that comes in 32-bit units. That is, it comes in longs. It is a fixed-width format and is always 1 "long" in length.

The first 256 characters of Unicode character sets correspond to the 256 characters of ISO-8859-1.

By default, HTML 4 processors should support UTF-8, and XML processors are supposed to support UTF-8 and UTF-16; therefore all XHTML-compliant processors should also support UTF-16.

tutorialspoint

CHAPTER 40

! ! !

A complete list of deprecated HTML tags and attributes are given here. All the tags have been ordered alphabetically along with their equivalent tag or alternate CSS option.

!	!	!
<applet>	/ Specifies an applet	<object>
<basefont>	/ Specifies a base font	
	/ Specifies centered text	text-align
<dir>	/ Specifies a directory list	
<embed>	/ Embeds an application in a document	<object>
<font>	/ Specifies text font, size, and color	font-family, font-size
<isindex>	/ Specifies a single-line input field	
<listing>	/ Specifies listing of items	<pre>
<menu>	/ Specifies a menu list	

tutorialspoint

<plaintext>	/ Specifies plaintext	<pre>
<s>	/ Specifies strikethrough text	text-decoration
<strike>	/ Specifies strikethrough text	text-decoration
<u>	/ Specifies underlined text	text-decoration
<xmp>	/ Specifies preformatted text	<pre>

! ! !

Following is the list of deprecated HTML attributes and alternative CSS options available.

!	!	!
align	Specifies the positioning of an element	text-align, float & vertical-align
alink	Specifies the color of an active link or selected link	Active
background	Specifies background image	background-image
bgcolor	Specifies background color	background-color
border	Specifies a border width of any element	border-width
clear	Indicates how the browser should display the line after the element	clear
height	Specifies the height of body and other elements	height

hspace	Specifies the amount of whitespace or padding that should appear left or right of an element	padding
language	Specifies the scripting language being used	type
link	Specifies the default color of all links in the document	link
nowrap	Prevents the text from wrapping within that table cell	white-space
start	Indicates the number at which a browser should start numbering a list	counter-reset
text	Specifies the color of body text	color
type	Specifies the type of list in tag	list-style-type
vlink	Specifies the color of visited links	visited
vspace	Specifies the amount of whitespace or padding that should appear above or below an element	padding
width	Specifies the width of body and other elements	width