

## What is HTML?

Rs = 20/-

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

- **Hypertext** refers to the way in which Web pages (HTML documents) are linked together. When you click a link in a Web page, you are using hypertext.
- **Markup Language** describes how HTML works. With a markup language, you simply "mark up" a text document with tags that tell a Web browser how to structure it to display.

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

## Why to learn HTML?

It is possible to create webpages without knowing anything about the HTML source behind the page.

There are excellent editors on the market that will take care of the HTML parts. All you need to do is layout the page.

However, if you want to make it above average in webdesign, it is strongly recommended that you understand these tags.

The most important benefits are:

You can use tags the editor does not support.

You can read the code of other people's pages, and "borrow" the cool effects.

You can do the work yourself, when the editor simply refuses to create the effects you want.

## Words to Know

- **Tag** - tag is a command the web browser interprets. Tags look like this: <tag>
- **Element** - A complete tag, having an opening <tag> and a closing </tag>.
- **Attribute** - Used to modify the value of the HTML element. Elements will often have multiple attributes.

## HTML Elements

An HTML element is everything from the start tag to the end tag:

Start tag *	Element content	End tag *
<p>	This is a paragraph	</p>

<a href="default.htm" > This is a link </a>

<br />

\* The start tag is often called the **opening tag**. The end tag is often called the **closing tag**.

## **HTML Element Syntax**

- An HTML element starts with a **start tag / opening tag**
- An HTML element ends with an **end tag / closing tag**
- The **element content** is everything between the start and the end tag
- Some HTML elements have **empty content**
- Empty elements are **closed in the start tag**
- Most HTML elements can have **attributes**

## **Nested HTML Elements**

Most HTML elements can be nested (can contain other HTML elements).

HTML documents consist of nested HTML elements.

## **Don't Forget the End Tag**

Some HTML elements might display correctly even if you forget the end tag:

<p>This is a paragraph

<p>This is a paragraph

The example above works in most browsers, because the closing tag is considered optional.

Never rely on this. Many HTML elements will produce unexpected results and/or errors if you forget the end tag .

## **Empty HTML Elements**

HTML elements with no content are called empty elements.

<br> is an empty element without a closing tag (the <br> tag defines a line break).

**Tip:** In XHTML, all elements must be closed. Adding a slash inside the start tag, like <br />, is the proper way of closing empty elements in XHTML (and XML).

## **Use Lowercase Tags**

HTML tags are not case sensitive: <P> means the same as <p>. Many web sites use uppercase HTML tags.

World Wide Web Consortium (W3C) **recommends** lowercase in HTML 4, and **demands** lowercase tags in XHTML.

## Attributes

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

The value of the attribute should be put in double quotation marks, and is separated from the name by the equals sign.

Many HTML tags have a unique set of their own attributes. Right now we want to focus on a set of generic attributes.

## Core Attributes:

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

- id
- title
- class
- style

## The id Attribute:

The *id* attribute can be used to uniquely identify any element within a page ( or style sheet ). There are two primary reasons that you might want to use an id attribute on an element:

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

For now, the id attribute could be used to distinguish between two paragraph elements, like so:

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

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

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

## The title Attribute:

The *title* attribute gives a suggested title for the element. The behavior of this attribute will depend upon the element that carries it, although it is often displayed as a tooltip or while the element is loading.

**The class Attribute:**

The *class* attribute is used to associate an element with a style sheet, and specifies the class of element. You learn more about the use of the class attribute when you will learn Cascading Style Sheet (CSS). The value of the attribute may also be a space-separated list of class names. For example:

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

**The style Attribute:**

The style attribute allows you to specify CSS rules within the element. For example:

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

**Generic Attributes:**

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

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

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

## Use Lowercase Attributes

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.

Newer versions of (X)HTML will demand lowercase attributes.

## What do I need to create HTML?

- **Computer**
- **Text or HTML editor.**
  - Adobe Dreamweaver.
  - SeaMonkey, Coffee Cup (Windows) and TextPad (Windows).
  - Notepad (for Windows), Pico (for Linux), or Simpletext/Text Edit/Text Wrangler (Mac).
- **Web Browser.**
  - Internet Explorer, Firefox, Chrome, opera ...etc.

## Do I need to be online?

No, you do not need to be online to create web pages. You can create web pages on your local machine. You only need to go online when you want to publish your web page to the web

## HTML Documents = Web Pages

- HTML documents describe web pages
- HTML documents contain HTML tags and plain text
- HTML documents are also called web pages

## .HTM or .HTML File Extension?

When you save an HTML file, you can use either the .htm or the .html file extension. There is no difference, it is entirely up to you.

## Basic HTML Document Structure

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

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

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

## Doctypes

A doctype declaration refers to the rules for the markup language, so that the browsers render the content correctly.

### Example

An HTML document with a doctype of HTML 4.01 Transitional:

```
1. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
2. "http://www.w3.org/TR/html4/loose.dtd">  
3. <html>  
4.   <head>  
5.     <title>Title of the document</title>  
6.   </head>  
7.  
8.   <body>  
9.     The content of the document.....  
10.  </body>  
11.  
12. </html>
```

### HTML Different Doctypes

The doctype declaration is not an HTML tag; it is an instruction to the web browser about what version of the markup language the page is written in.

The doctype declaration refers to a Document Type Definition (DTD). The DTD specifies the rules for the markup language, so that the browsers render the content correctly.

The doctype declaration should be the very first thing in an HTML document, before the <html> tag.

**Tip:** Always add a doctype to your pages. This helps the browsers to render the page correctly!

#### HTML 4.01 Strict

This DTD contains all HTML elements and attributes, but does NOT INCLUDE presentational or deprecated elements (like font and center). Framesets are not allowed:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">
```

#### HTML 4.01 Transitional

This DTD contains all HTML elements and attributes, INCLUDING presentational and deprecated elements (like font). Framesets are not allowed:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

### HTML 4.01 Frameset

This DTD is equal to HTML 4.01 Transitional, but allows the use of frameset content:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

### The <html> Element:

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

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

- <head>
- <body>

### The <head> Element:

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

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

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

### HTML Base tag.

1. <html>
2. <head>
3. <title>Practice HTML base tag</title>
- 4.
5. <base href="http://www.sekharit.com" target="\_blank" />
- 6.
7. </head>
8. <body>
- 9.
10. <p>This tag is using base path to display the image.</p>

```
11. 
12.
13. <p>This link will open in new window because we have used _blank in base tag.</p>
14. <p>Try to click this <a href="/php/index.htm">PHP Tutorial</a>
15.
16. </body>
17. </html>
```

## HTML Link tag

```
<link rel="stylesheet" type="text/css" href="/style.css" />
```

## HTML Style tag.

Style sheets 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) is a style sheet mechanism that has been specifically developed to meet the needs of Web designers and users.

With 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 (;).

```
<p style="color:red;font-size:24px;">Using Style Sheet Rules</p>
```

There are three ways of using a style sheet in an HTML document:

## External Style Sheet:

If you have to give same look and feel to many pages then it is a good idea to keep all the style sheet rules in a single style sheet file and include this file in all the HTML pages. You can include a style sheet file into HTML document using <link> element. Below is an example:

```
<head>
  <link rel="stylesheet" type="text/css" href="yourstyle.css">
</head>
```

## Internal Style Sheet:

If you want to apply Style Sheet rules to a single document only then you can include those rules into that document only. Below is an example:

```
<head>
  <style type="text/css">
```



```
body{background-color: pink;}
p{color:blue; 20px;font-size:24px;}
</style>
</head>
```

## Inline Style Sheet:

You can apply style sheet rules directly to any HTML element. This should be done only when you are interested to make a particular change in any HTML element only. To use inline styles you use the style attribute in the relevant tag. Below is an example:

```
<p style="color:red;font-size:24px;">Using Style Sheet Rules</p>
```

## HTML Script tag.

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

You can write your Event Handlers using any of the scripting language and then you can trigger those functions using HTML attributes.

## External Script:

If you have to use a single script functionality among many HTML pages then it is a good idea to keep that function in a single script file and then include this file in all the HTML pages. You can include a style sheet file into HTML document using <script> element. Below is an example:

```
<head>
  <script src="yourfile.js" type="text/javascript" />
</head>
```

## Internal Script:

You can write your script code directly into your HTML document. Usually we keep script code in header of the document using <script> tag, otherwise there is no restriction and you can put your source code anywhere in the document. You can specify whether to make a script run automatically (as soon as the page loads), or after the user has done something (like click on a link). Below is an example this would write a *Hello Javascript!* message as soon as the page loads.:

```
<head>
  <title>Internal Script</title>
</head>
<body>
```

```
<script type="text/javascript">
    alert("Hello Javascript!")
</script>
</body>
```

This will produce following result:

Hello Javascript!

## Writing Event Handler:

It is very easy to write an event handler. Following example explains how to write an event handler. Let's write one simple function *myAlert* in the header of the document. We will call this function when any user will bring mouse over a paragraph written in the example.

```
<head>
    <title>Event Handler Example t</title>
    <script type="text/javascript">
        function myAlert()
        {
            alert("I am an event handler....");
            return;
        }
    </script>
</head>
<body>
    <span onmouseover="myAlert();" >
        Bring your mouse here to see an alert
    </span>
</body>
```

Now this will produce following result. Bring your mouse over this line and see the result:

Bring your mouse here to see an alert

## The <noscript> Element:

You can also provide alternative info for users whose browsers don't support scripts and for users who have disabled scripts. You do this using the <noscript> tag.

```
<noscript >
    Please enable javascript
</noscript>
```

## Default Scripting Language

You can specify a default scripting language for all your *script* tags to use. This saves you from having to specify the language everytime you use a script tag within the page. Below is the example:

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

## HTML Meta tag.

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

Metadata provided by using meta tag is a very important part of the web. It can assist search engines in finding the best match when a user performs a search. Search engines will often look at any metadata attached to a page - especially keywords - and rank it higher than another page with less relevant metadata, or with no metadata at all.

## Adding Meta Tags to Your Documents:

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

Attribute	Description
Name	Name for the property. Can be anything. Examples include, keywords, description, author, revised, generator etc.
content	Specifies the property's value.
scheme	Specifies a scheme to use to interpret the property's value (as declared in the content attribute).
http-equiv	Used for http response message headers. For example http-equiv can be used to refresh the page or to set a cookie. Values include content-type, expires, refresh and set-cookie.

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

## Meta Tag Examples:

Let's see few important usage of Meta Tags.

**Specifying Keywords:**

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

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

**Document Description:**

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

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

**Document Revision date:**

This information tells about last time the document was updated.

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

**Document Refreshing:**

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

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

**Page Redirection:**

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

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

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

## Setting Cookies:

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

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

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

## Setting Author Name:

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

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

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

## HTML Comment

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

Comments do not nest, and the double-dash sequence `--` may not appear inside a comment except as part of the closing `-->` tag. You must also make sure that there are no spaces in the start-of-comment string.

For example: Given line is a valid comment in HTML

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

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

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

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

## Multiline Comments:

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

For example:

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

## Using Comment tag

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

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

## Paragraph Tag <p>

Publishing any kind of written works requires the use of a paragraph. The paragraph tag is very basic and a great introductory tag for beginner's because of its simplicity.

The `<p>` tag defines a paragraph. Using this tag places a blank line above and below the text of the paragraph. These automated blank lines are examples of how a tag "marks" a paragraph and the web browser automatically understands how to display the paragraph text because of the paragraph tag.

## HTML Code:

```
<p>Avoid losing floppy disks with important school...</p>  
<p>For instance, let's say you had a HUGE school...</p>
```

## Two HTML Paragraphs:

Avoid losing floppy disks with important school/work projects on them. Use the web to keep your content so you can access it from anywhere in the world. It's also a quick way to write reminders or notes to yourself. With simple html skills, (the ones you have gained so far) it is easy.

For instance, let's say you had a HUGE school or work project to complete. Off hand, the easiest way to transfer the documents from your house could be a 3.5" floppy disk. However, there is an alternative. Place your documents, photos, essays, or videos onto your web server and access them from anywhere in the world.

## HTML - Paragraph Justification

Paragraphs can be formatted in HTML much the same as you would expect to find in a word processing program. Here the align attribute is used to "justify" our paragraph.

### HTML Code:

```
<p align="justify">For instance, let's say you had a HUGE school or work...</p>
```

### Justified Text Alignment:

For instance, let's say you had a HUGE school or work project to complete. Off hand, the easiest way to transfer the documents from your house could be a 3.5" floppy disk. However, there is an alternative. Place your documents, photos, essays, or videos onto your web server and access them from anywhere in the world.

## HTML - Paragraph Centering

### HTML Code:

```
<p align="center">For instance, let's say you had a HUGE school or work...</p>
```

### Centered Text Alignment:

For instance, let's say you had a HUGE school or work project to complete. Off hand, the easiest way to transfer the documents from your house could be a 3.5" floppy disk. However, there is an alternative. Place your documents, photos, essays, or videos onto your web server and access them from anywhere in the world.

Each line of the paragraph has now been centered inside the display window.

## HTML - Paragraph Align Right

### HTML Code:

```
<p align="right">For instance, let's say you had a HUGE school or work...</p>
```

### Right Text Alignment:

For instance, let's say you had a HUGE school or work project to complete. Off hand, the easiest way to transfer the documents from your house could be a 3.5" floppy disk. However, there is an alternative. Place your documents, photos, essays, or videos onto your web server and access them from anywhere in the world.

Every line of the paragraph above is now aligned to the right hand side of the display box.

## Create Headings - The <h> Elements:

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

Example:

```
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>
```

This will display following result:

**This is heading 1**

**This is heading 2**

**This is heading 3**

**This is heading 4**

This is heading 5

*This is heading 6*

## Line Breaks

Line breaks are different then most of the tags we have seen so far. A line break ends the line you are currently on and resumes on the next line. Placing `<br />` within the code is the same as pressing the return key in a word processor. Use the `<br />` tag within the `<p>` (paragraph) tag.

### HTML Code:

```
<p>
    Will Mateson<br />
    Box 61<br />
    Cleveland, Ohio<br />
</p>
```

### Address:

Will Mateson  
Box 61  
Cleveland, Ohio

We have created a possible address for a letter head. The line break tag will also come in handy toward the end of our letter.



**HTML Code:**

```
<p>Sincerely,<br />
<br />
<br />
Kevin Sanders<br />
Vice President</p>
```

**Closing Letter:**

Sincerely,

Kevin Sanders  
Vice President

**HTML Horizontal Rule**

Use the <hr /> tag to display lines across the screen. Note: the horizontal rule tag has no ending tag like the line break tag.

**HTML Code::**

```
<hr />
Use
<hr /><hr />
Them
<hr />
Sparingly
<hr />
```

**Display::**

---

Use

---

Them

---

Sparingly

Aside from our exaggerated example, the horizontal rule tag can come in handy when publishing work. A table of contents or perhaps a bibliography.

**HTML Code:**

```
<hr />
<p>1. "The Hobbit", JRR Tolkein.<br />
2. "The Fellowship of the Ring" JRR Tolkein.</p>
```

1. "The Hobbit", JRR Tolkein.
2. "The Fellowship of the Ring" JRR Tolkein.

**Lists:**

As you can see, all this tag does is draw a line across your content, and used properly, its results can be outstanding.

You can list out your items, subjects or menu in the form of a list. HTML gives you three different types of lists.

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

**HTML Unordered Lists:**

An unordered list is a collection of related items that have no special order or sequence. The most common unordered list you will find on the Web is a collection of hyperlinks to other documents.

This list is created by using `<ul>` tag. Each item in the list is marked with a butllet. The bullet itself comes in three flavors: squares, discs, and circles. The default bullet displayed by most web browsers is the traditional full disc.

One Movie list is given below:

```
<center>
  <h2>Movie List</h2>
</center>

<ul>
  <li>Ram Teri Ganga Meli</li>
  <li>Mera Naam Jocker</li>
  <li>Titanic</li>
  <li>Ghost in the ship</li>
</ul>
```

This will produce following result:

## Movie List

- Ram Teri Ganga Meli
- Mera Naam Jocker
- Titanic
- Ghost in the ship

You can use *type* attribute to specify the type of bullet you like. By default its is a disc. Following are the possible way:

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

<ul type="square">	<ul type="disc">	<ul type="circle">
<ul style="list-style-type: none"> <li>▪ Hindi</li> <li>▪ English</li> <li>▪ Maths</li> <li>▪ Physics</li> </ul>	<ul style="list-style-type: none"> <li>• Hindi</li> <li>• English</li> <li>• Maths</li> <li>• Physics</li> </ul>	<ul style="list-style-type: none"> <li>○ Hindi</li> <li>○ English</li> <li>○ Maths</li> <li>○ Physics</li> </ul>

## HTML Ordered Lists:

The typical browser formats the contents of an ordered list just like an unordered list, except that the items are numbered instead of bulleted. The numbering starts at one and is incremented by one for each successive ordered list element tagged with <li>

This list is created by using <ol> tag. Each item in the list is marked with a number.

One Movie list is given below:

```
<center>
  <h2>Movie List</h2>
</center>

<ol>
  <li>Ram Teri Ganga Meli</li>
  <li>Mera Naam Jocker</li>
  <li>Titanic</li>
  <li>Ghost in the ship</li>
</ol>
```

This will produce following result:

## Movie List

1. Ram Teri Ganga Meli
2. Mera Naam Jocker
3. Titanic
4. Ghost in the ship

You can use *type* attribute to specify the type of numbers you like. By default its is a generic numbers. Following are the other possible way:

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

<code>&lt;ol type="I"&gt;</code>	<code>&lt;ol type="i"&gt;</code>	<code>&lt;ol type="a"&gt;</code>	<code>&lt;ol type="A"&gt;</code>
I. Hindi	i. Hindi	a. Hindi	A. Hindi
II. English	ii. English	b. English	B. English
III. Maths	iii. Maths	c. Maths	C. Maths
IV. Physics	iv. Physics	d. Physics	D. Physics

You can use *start* attribute to specify the beginning of any index. By default its is a first number or character. In the following example index starts from 5:

```

<center>
  <h2>Movie List</h2>
</center>
<ol start="5">
  <li>Ram Teri Ganga Meli</li>
  <li>Mera Naam Jocker</li>
  <li>Titanic</li>
  <li>Ghost in the ship</li>
</ol>

```

This will produce following result:

### Movie List

5. Ram Teri Ganga Meli
6. Mera Naam Jocker
7. Titanic
8. Ghost in the ship

### HTML Definition Lists:

HTML and XHTML also support a list style entirely different from the ordered and unordered lists we have discussed so far - definition lists . Like the entries you find in a dictionary or encyclopedia, complete with text, pictures, and other multimedia elements, the Definition List is the ideal way to present a glossary, list of terms, or other name/value list.

Definition List makes use of following three tags.

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

Example:

```
<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>
```

This will produce following result:

#### HTML

This stands for Hyper Text Markup Language

#### HTTP

This stands for Hyper Text Transfer Protocol

## HTML - Formatting Elements w/ Tags

As you begin to place more and more elements onto your web site, it will become necessary to make minor changes to the formatting of those elements. In our [HTML Attributes](#) lesson we discussed ways to add some flavor with attributes and align elements within other elements. Several tags exist to further amplify text elements. These formatting tags can make text bold, italic, sub/superscripted, and more.

### Bold, Italic and More

#### HTML Code:

```
<p>An example of <b>Bold Text</b></p>
<p>An example of <em>Emphasized Text</em></p>
<p>An example of <strong>Strong Text</strong></p>
<p>An example of <i>Italic Text</i></p>
<p>An example of <sup>superscripted Text</sup></p>
<p>An example of <sub>subscripted Text</sub></p>
<p>An example of <del>struckthrough Text</del></p>
<p>An example of <code>Computer Code Text</code></p>
```

#### HTML Formatting:

An example of **Bold Text**

An example of *Emphasized Text*

An example of **Strong Text**

An example of *Italic Text*

An example of <sup>superscripted Text</sup>

An example of <sub>subscripted Text</sub>

An example of ~~struckthrough Text~~

An example of `Computer Code Text`

All of these tags add a pinch of flavor to paragraph elements. They can be used with any text type element.

### Preserve Formatting - The <pre> Element:

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

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

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

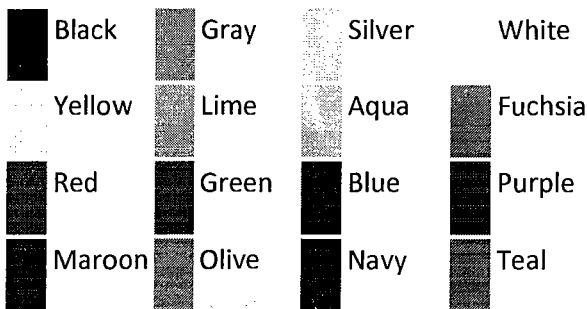
This will produce following result:

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

## HTML Color Coding System - Color Names

There are 3 different methods to set color. The simplest being the **Generic** terms of colors. Examples: black, white, red, green, and blue. Generic colors are preset HTML coded colors where the value is simply the name of each color. Here is a sample of the most widely supported colors and their respective name values.

### The 16 Basic Colors:



## HTML Coloring System - RGB Values

We do not recommend that you use RGB for safe web design because non-IE browsers do not support HTML RGB. However, if you plan on learning CSS then you should glance over this topic.

RGB stands for Red, Green, Blue. Each can have a value from 0 (none of that color) to 255 (fully that color). The format for RGB is - rgb(RED, GREEN, BLUE), just like the name implies. Below is an example of RGB in use, but if you are not using a browser that supports it, do not worry, that is just one of the problems with HTML RGB.

### Red, Green, and Blue Values:

bgcolor="rgb(255,255,255)" White

bgcolor="rgb(255,0,0)" Red

bgcolor="rgb(0,255,0)" Green

bgcolor="rgb(0,0,255)" Blue

## HTML Coloring System - Hexadecimal

The hexadecimal system is complex and difficult to understand at first. Rest assured that the system becomes much, MUCH easier with practice and as a blossoming web developer, it is critical to understand hexadecimal

to be capable of using them in your own web publications. They are far more reliable and widely compatible among web browsers and are the standard for colors on the internet.

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).

Here's a hexadecimal you might see in an HTML document.

### My First Hexadecimal:

```
bgcolor="#RRGGBB"
```

## HTML Color Code - Breaking the Code

The following table shows how letters are incorporated into the hexadecimal essentially extending the numbers system to 16 values. Hang in there it all makes sense shortly.

### Hexadecimal Color Values:

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

*So use letters as numbers?* We will answer this question as we dive into the converting hexadecimal to regular numbers. Let's have a look at real Hexadecimal.

### A Real Hexadecimal:

```
bgcolor="#FFFFFF"
```

The letter "F" is the maximum amount we can send each color and as you may deduce, this color (#FFFFFF) represents the color white. A formula exists to calculate the numeric equivalent of a hexadecimal.

## HTML - Font and Basefont

The <font> tag is used to add style, size, and color to the text on your site. Use the *size*, *color*, and *face* attributes to customize your fonts. Use a <basefont> tag to set all of your text to the same size, face, and color.

**The font and basefont tags are deprecated and should not be used. Instead, use css styles to manipulate your font.**

## Font Size

Set the size of your font with *size*. The range of accepted values is from 1(smallest) to 7(largest). The default size of a font is 3.



**HTML Code:**

```
<p>  
  <font size="5">Here is a size 5 font</font>  
</p>
```

**Font Size:**

Here is a size 5 font.

**Font Color**

Set the color of your font with *color*.

**HTML Code:**

```
<font color="#990000">This text is hexcolor #990000</font>  
<br />
```

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

**Font Color:**

This text is hexcolor #990000

This text is red

**Font Face**

Choose a different font face using any font you have installed. Be aware that if the user viewing the page doesn't have the font installed, they will not be able to see it. Instead they will default to Times New Roman. An option is to choose a few that are similar in appearance.

**HTML Code:**

```
<p>  
  <font face="Bookman Old Style, Book Antiqua, Garamond">This paragraph  
  has had its font...</font>  
</p>
```

**Font Face:**

This paragraph has had its font formatted by the font tag!

**HTML Entities**

Some characters are reserved in HTML.

It is not possible to use the less than (<) or greater than (>) signs in your text, because the browser will mix them with tags.

To actually display reserved characters, we must use character entities in the HTML source code.


A character entity looks like this:

`&entity_name;`

OR

`&#entity_number;`

To display a less than sign we must write: **&lt;** or **&#60;**;

 **Tip:** The advantage of using an entity name, instead of a number, is that the name is easier to remember. However, the disadvantage is that browsers may not support all entity names (the support for entity numbers is very good).

## Non-breaking Space

A common character entity used in HTML is the non-breaking space (`&nbsp;`).

Browsers will always truncate spaces in HTML pages. If you write 10 spaces in your text, the browser will remove 9 of them, before displaying the page. To add spaces to your text, you can use the `&nbsp;` character entity.

## HTML Useful Character Entities

**Note:** Entity names are case sensitive!

Result	Description	Entity Name	Entity Number
	non-breaking space	<code>&amp;nbsp;</code>	<code>&amp;#160;</code>
<	less than	<code>&amp;lt;</code>	<code>&amp;#60;</code>
>	greater than	<code>&amp;gt;</code>	<code>&amp;#62;</code>
&	ampersand	<code>&amp;amp;</code>	<code>&amp;#38;</code>
¢	cent	<code>&amp;cent;</code>	<code>&amp;#162;</code>
£	pound	<code>&amp;pound;</code>	<code>&amp;#163;</code>
¥	yen	<code>&amp;yen;</code>	<code>&amp;#165;</code>
€	euro	<code>&amp;euro;</code>	<code>&amp;#8364;</code>
§	section	<code>&amp;sect;</code>	<code>&amp;#167;</code>

©	copyright	&copy;	&#169;
®	registered trademark	&reg;	&#174;
™	trademark	&trade;	&#8482;

## HTML Color - bgcolor

The bgcolor attribute is used to control the background of an HTML element, specifically page and table backgrounds. Bgcolor can be placed within several of the HTML tags. However, we suggest you only use it for your page's main background (<body>) and in tables.

### Syntax

<TAGNAME bgcolor="value">

Quick and dirty, here is how to change the background of your web page. Just use the bgcolor attribute in the <body> tag and you are golden.

### HTML Code:

```
<body bgcolor="Silver">
<p>We set the background...</p>
</body>
```

## HTML - Background

Images can be placed within elements of HTML. Tables, paragraphs, and bodys may all have a background image. To accomplish this, we use the background attribute as follows.

### HTML Code:

```
<table height="100" width="150"
  background="http://www.tizag.com/pics/htmlT/background.jpg" >
<tr><td>This table has a background image</td></tr>
</table>
```

## HTML - Background Repeat

In the first example we happen to be lucky because our image and our table had exactly the same size pixel dimensions. Everything looks great. When your HTML element is larger than the dimensions of your picture, the image simply begins to repeat itself.

### HTML Code:

```
<table height="200" width="300"
  background="http://www.tizag.com/pics/htmlT/background.jpg" >
```

```
<tr><td>This table has a background image</td></tr>
</table>
```

It is obvious this is often not the desired outcome, however, it can also be quite useful as you will see in the following example.

## HTML - Patterned Backgrounds

Repeating a generic image as a background doesn't have much practical use. We either need to find an image to fit exactly as our background or have an image editing program to adjust the dimensions of our image.

From a different angle, we can use this default attribute to our benefit say if we wanted to have some sort of pattern as our background. In an image editing program such as Adobe Photosop, or Paint Shop Pro, we could create a very small (perhaps 4X4 pixels) and create a couple of basic patterns.

### 4x4 Image:

-

Now here is the same image set as the background to our same table.

### HTML Code:

```
<table height="100" width="150"
  background="http://www.tizag.com/pics/htmlT/pattern.jpg" >
<tr><td>This table has a background patterned image</td></tr>
</table>
```

## HTML - Transparent Background

Another great technique, along the same lines as the patterned images, is that of transparent, colored backgrounds. Most image editors have some sort of transparency device to create images that appear see through. We're not going to cover how to do this with every single program, however, most of the time all you need to do is fill your canvas with the color you would like and then set the opacity to something below 100%. Then make sure you save your file as a gif not a jpeg, and all systems should be go.

Now that you have had the crash course on creating transparent files, you place them onto your websites the exact same way as you would a repeating background.

### HTML Code:

```
<table background="http://www.tizag.com/pics/htmlT/transparent.gif" >
<tr><td>This table has a red transparent background image</td></tr>
</table>
```

## HTML - Div Element(s)

The <div> tag is nothing more than a container for other tags. Much like the body tag, Div elements are block elements and work behind the scenes grouping other tags together. Use only the following attributes with your div element,

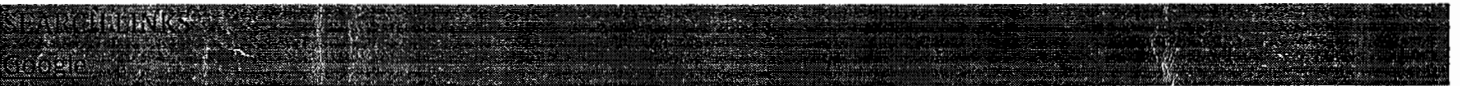
- id
- width
- height
- title
- style

For the purpose of this example, we have included the *style* attribute in order to color our div tag in order to bring a stronger visualization for our viewers.

### HTML Code:

```
<body>
<div style="background: green">
<h5 >SEARCH LINKS</h5>
<a target="_blank" href="http://www.google.com">Google</a>
</div>
</body>
```

### HTML Div Element:



Above is a great visual about how a div plays the role of a container for other HTML elements, applying a background color/image is the only real way to visualize your div tags.

## HTML - Links and Anchors

The web got its spidery name from the plentiful connections between web sites. These connections are made using anchor tags to create links. Text, Images, and Forms may be used to create these links.

### HTML - Hypertext Reference (href)

The *href* attribute defines reference that the link refers to. Basically this is where the user will be taken if they wish to click this link.

Hypertext references can be Internal, Local, or Global.

- Internal - Links to anchors on the current page
- Local - Links to other pages within your domain

- Global - Links to other domains outside of your site

### HTML Code:

Internal - href="#anchorname"

Local - href="../pics/picturefile.jpg"

Global - href=http://www.sekharit.com/

## HTML - Text Links

Use the <a></a> tags to define the start and ending of an anchor. Decide what type of href attribute you need and place this attribute into the opening tag. The text you place between the opening and closing tags will be shown as the link on a page. Use the demonstration below as a reference.

### HTML Code:

<a href="http://www.sekharit.com/" target="\_blank" >Tizag Home</a>

<a href="http://www.espn.com/" target="\_blank" >ESPN Home</a>

<a href="http://www.yahoo.com/" target="\_blank" >Yahoo Home</a>

### Global Link:

[Tizag Home](#) [ESPN Home](#) [Yahoo Home](#)

## HTML - Link Targets

The target attribute defines whether to open the page in a separate window, or to open the link in the current browser window.

### HTML Code:

target="\_blank" Opens new page in a new browser window

\_self" Loads the new page in current window

\_parent" Loads new page into a frame that is superior to where the link lies

\_top" Loads new page into the current browser window, cancelling all frames

The example below shows how you would link to ESPN.COM, a popular sports web site. The target attribute is added to allow the browser to open ESPN in a new window, so that the viewer can remain at our web site. Here's the example.

### HTML Code:

<a href="http://www.ESPN.com" target="\_blank">ESPN.COM</a>

**\_blank Target:**ESPN.COM**HTML - Anchors**

To link to sections of your existing page a name must be given to the anchor. In the example below, we've created a mini Table of Contents for this page. By placing blank anchors just after each heading, and naming them, we can then create reference links to those sections on this page as shown below.

First, the headings of this page contain blank, named anchors. They look like this.

**Tizag's Own Code:**

```
<h2>HTML Links and Anchors<a name="top"></a></h2>
```

```
<h2>HTML Text Links<a name="text"></a></h2>
```

```
<h2>HTML Email<a name="email"></a></h2>
```

Now create the reference links, placing the pound symbol followed by the name of the anchor in the href of the new link.

**Anchor Code:**

```
<a href="#top">Go to the Top</a>
```

```
<a href="#text">Learn about Text Links</a>
```

```
<a href="#email">Learn about Email Links</a>
```

**Local Links:**[Go to the Top](#)[Learn about Text Links](#)[Learn about Email Links](#)**HTML - Email Links**

Creating an email link is simple. If you want somebody to mail you about your site a good way to do it is place an email link with a subject already in place for them.

**HTML Code:**

```
<a href="mailto:email@sekharit.com?subject=Feedback" >Email@sekharit.com</a>
```

**Email Links:**[Email@sekharit.com](mailto:Email@sekharit.com)

In some circumstances it may be necessary to fill in the body of the Email for the user as well.

**HTML Code:**

```
<a href="mailto:email@sekharit.com?subject=Feedback&body=Sweet site!">  
Email@sekharit.com</a>
```

**Complete Email:**

[Email@sekharit.com](mailto:Email@sekharit.com)

**HTML - Download Links**

Placing files available for download is done in exactly the same fashion as placing text links. Things become complicated if we want to place image links available for download. The best solution for images is to use a thumbnail link that we discuss in the next lesson.

**HTML Code:**

```
<a href="http://www.sekharit.com/pics/htmlT/blanktext.zip">Text Document</a>
```

**Download a Text Document:**

[Text Document](#)

**HTML - Default Links; Base**

Use the <base> tag in the *head* element to set a default URL for all links on a page to go to. It's always a good idea to set a base tag just incase your links become bugged somewhere down the line. Usually set your base to your home page.

**HTML Code:**

```
<head>  
  
<base href="http://www.sekharit.com/">  
  
</head>
```

**HTML - Images**

Images are a staple of any web designer, so it is very important that you understand how to use them properly. Use the <img /> tag to place an image on your web page.

**HTML Code:**

```

```

**Image:**



## HTML - Image src

Above we have defined the *src* attribute. Src stands for *source*, the source of the image or more appropriately, where the picture file is located. As with links described in a previous lesson, you may use any standard URL to properly point the src attribute to a local or external source.

There are two ways to define the source of an image. First you may use a standard URL.

(src=http://www.Tizag.com/pics/htmlT/sunset.gif) As your second choice, you may copy or upload the file onto your web server and access it locally using standard directory tree methods. (src="../sunset.gif") The location of this picture file is in relation to your location of your .html file.

### URL Types:

Local Src	Location Description
src="sunset.gif"	picture file resides in same directory as .html file
src="../sunset.gif"	picture file resides in previous directory as .html file
src="../pics/sunset.gif"	picture file resides in the <i>pic</i> directory in a previous directory as .html file

A URL cannot contain drive letters, since a src URL is a relational source interpretation based on the location of your .html file and the location of the picture file. Therefore something like src="C:\\www\\web\\pics\\" will not work. Pictures must be uploaded along with your .html file to your web server.

Each method has its pros and cons, for instance using the URL of pictures on other sites poses a problem if the web master(s) of the other site happen to change the physical location of the picture file. Copying the file directly to your web server solves this problem, however, as you continue to upload picture files to your system, you may eventually run short on hard drive space. Use your best judgement to meet your needs.

## HTML - Alternative Attribute

The *alt* attribute specifies alternate text to be displayed if for some reason the browser cannot find the image, or if a user has image files disabled. Text only browsers also depend on the alt attribute since they cannot display pictures.

### HTML Code:

```

```

### Alternative Text:

## HTML - Image Height and Width

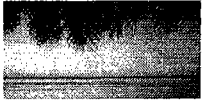
To define the height and width of the image, rather than letting the browser compute the size, use the *height* and *width* attributes.

### HTML Code:

```

```

### Height and Width:



Above we have defined the *src*, *height* and *width* attributes. By informing the browser of the image dimensions it knows to set aside a place for that image. Without defining an image's dimensions your site may load poorly; text and other images will be moved around when the browser finally figures out how big the picture is supposed to be and then makes room for the picture.

## Vertically and Horizontally Align Images

Use the *align* and *valign* attributes to place images within your body, tables, or sections.

1. *align* (Horizontal)
  - right
  - left
  - center
2. *valign* (Vertical)
  - top
  - bottom
  - center

Below is an example of how to align an image to the right of a paragraph.

### HTML Code:

```
<p>This is paragraph 1, yes it is...</p>
```

```
<p>
```

```

```

```
The image will appear along the...isn't it?
```

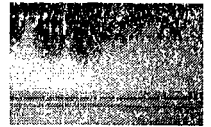
```
</p>
```

```
<p>This is the third paragraph that appears...</p>
```

## Image Wrap Arond:

This is paragraph 1, yes it is. I think this paragraph serves as a nice example to show how this image alignment works.

The image will appear along the right hand side of the paragraph. As you can see this is very nice for adding a little eye candy that relates to the specified paragraph. If we were talking about beautiful tropical sunsets, this picture would be perfect. But we aren't talking about that, so it's rather a waste, isn't it?



This is the third paragraph that appears below the paragraph with the image!

## Images as Links

This will be a quick review of the [links - image lesson](#). Images are very useful for links and can be created with the HTML below.

### HTML Code:

```
<a href="http://www.tizag.com/">  
    
</a>
```

### Image Links:



Now your image will take you to our home page when you click it. Change it to your home page URL.

## HTML forms

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

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

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

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

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

Most frequently used form attributes are:

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

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

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

## HTML Forms - Text Input Controls:

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

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

### Single-line text input controls:

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

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

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

```
Last name: <input type="text" name="last_name" />
<input type="submit" value="submit" />
</form>
```

This will produce following result:

First name:

Last name:

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

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

```
<html>
  <head>
    <title>Practice HTML input text tag</title>
  </head>
  <body>

    <p>Try with different input text</p>

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

  </body>
</html>
```

## Password input controls::

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

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

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

This will produce following result:

Login :

Password :

```
<html>
  <head>
    <title>Practice HTML input password tag</title>
  </head>
  <body>

    <form action="" method="get">
      Login : <input type="text" name="login" />
      <br>
      Password: <input type="text" name="password" />
      <input type="reset" value="reset" />
    </form>

  </body>
</html>
```

## Multiple-Line Text Input Controls:

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

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

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

This will produce following result:

Description :

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

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

```
<html>
  <head>
    <title>Practice HTML input textarea tag</title>
  </head>
  <body>
    <p>Try with different cols and rows</p>
    <form action="" method="get">
      Description : <br />
      <textarea rows="5" cols="50" name="description">
        Enter description here...
      </textarea>
      <input type="submit" value="submit" />
    </form>
  </body>
</html>
```

## HTML Forms - Creating Button:

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

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

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

Here is the example:

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

```
<input type="button" value="Button" />
</form>
```

This will produce following result:

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

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

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

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

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

This will produce following result:

## HTML Forms - Checkboxes Control:

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

Here is example HTML code for a form with two checkboxes

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

The result of this code is the following form

Naresh i Technologies, Opp. Satyam Theatre, Ameerpet, Hyderabad, Ph: 040-23746666, 23734842

An ISO 9001 : 2000 Certified Company



☐ Maths ☐ Physics

---

Following is the list of important checkbox attributes:

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

```
<html>
  <head>
    <title>Practice HTML input checkbox tag</title>
  </head>
  <body>
    <p>Try with different checkbox and different attributes</p>
    <form action="" method="get">
      <input type="checkbox" name="maths" value="on" /> Maths
      <input type="checkbox" name="physics" value="on" /> Physics
      <input type="reset" value="reset" />
    </form>
  </body>
</html>
```

## HTML Forms - Raidobox Control:

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

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

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

The result of this code is the following form

☐ Maths ☐ Physics

---

Following is the list of important radiobox attributes:

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

```
<html>
  <head>
    <title>Practice HTML input radiobox tag</title>
  </head>
  <body>
    <p>Try with different radiobox and different attributes</p>
    <form action="" method="get">
      <input type="radio" name="subject" value="maths" /> Maths
      <input type="radio" name="subject" value="physics" /> Physics
      <input type="reset" value="reset" />
    </form>
  </body>
</html>
```

## HTML Forms - Select box Control:

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

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

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

The result of this code is the following form

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

- **name:** This is the name for the control.
- **size:** This can be used to present a scrolling list box.
- **multiple:** If set to "multiple" then allows a user to select multiple items from the menu.

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

- **value:** The value that is sent to the server if this option is selected.
- **selected:** Specifies that this option should be the initially selected value when the page loads.
- **label:** An alternative way of labeling options.

## HTML Forms - File Select Boxes:

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

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

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

The result of this code is the following form

-

## HTML Forms - Hidden Controls:

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

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

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

This will produce following result:

This is page 10

---

## HTML Forms - Submit and Reset Button:

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

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

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

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

First name:

Last name:

## Table

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

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

### Example:

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

This will produce following result:

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

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

### Table Heading - The `<th>` Element:

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

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

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

Name	Salary
Ramesh Raman	5000
Shabbir Hussein	7000

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

## Table Cellpadding and Cellspacing:

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

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

This will produce following result:

Name	Salary
Ramesh Raman	5000
Shabbir Hussein	7000

## Colspan and Rowspan Attributes:

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

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

This will produce following result:

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

## Tables Backgrounds

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

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

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

Here is an example of using *bgcolor* attribute:

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

This will produce following result:

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

Here is an example of using *background* attribute:

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

This will produce following result:

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

## Table height and width:

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

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

This will produce following result:

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



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

## Using Table Caption:

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

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

This will produce following result:

This is the caption

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

## Using a Header, Body, and Footer:

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

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

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

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

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

```

<td colspan="4">This is the foot of the table</td>
</tr>
</tfoot>
<tbody>
<tr>
<td>Cell 1</td>
<td>Cell 2</td>
<td>Cell 3</td>
<td>Cell 4</td>
</tr>
<tr>
...more rows here containing four cells...
</tr>
</tbody>
<tbody>
<tr>
<td>Cell 1</td>
<td>Cell 2</td>
<td>Cell 3</td>
<td>Cell 4</td>
</tr>
<tr>
...more rows here containing four cells...
</tr>
</tbody>
</table>

```

This will produce following result:

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

## Nested Tables:

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

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

```

<table border="1">
<tr>
<td>
    <table border="1">
    <tr>
    <th>Name</th>
    <th>Salary</th>
    </tr>
    <tr>
    <td>Ramesh Raman</td>
    <td>5000</td>
    </tr>
    <tr>
    <td>Shabbir Hussein</td>
    <td>7000</td>
    </tr>
    </table>
</td>
<td>
    <ul>
    <li>This is another cell</li>
    <li>Using list inside this cell</li>
    </ul>
</td>
</tr>
<tr>
<td>Row 2, Column 1</td>
<td>Row 2, Column 2</td>
</tr>
</table>

```

This will produce following result:

Name	Salary	<ul style="list-style-type: none"> <li>This is another cell</li> <li>Using list inside this cell</li> </ul>
Ramesh Raman	5000	
Shabbir Hussein	7000	
Row 2, Column 1	Row 2, Column 2	

## HTML Frames

With frames, you can display more than one HTML document in the same browser window. Each HTML document is called a frame, and each frame is independent of the others.

The disadvantages of using frames are:

- Frames are not expected to be supported in future versions of HTML
- Frames are difficult to use. (Printing the entire page is difficult).

- The web developer must keep track of more HTML documents

## The HTML frameset Element

The frameset element holds one or more frame elements. Each frame element can hold a separate document.

The frameset element states HOW MANY columns or rows there will be in the frameset, and HOW MUCH percentage/pixels of space will occupy each of them.

## The HTML frame Element

The <frame> tag defines one particular window (frame) within a frameset.

In the example below we have a frameset with two columns.

The first column is set to 25% of the width of the browser window. The second column is set to 75% of the width of the browser window. The document "frame\_a.htm" is put into the first column, and the document "frame\_b.htm" is put into the second column:

```
<frameset cols="25%,75%">
  <frame src="frame_a.htm" />
  <frame src="frame_b.htm" />
</frameset>
```

**Note:** The frameset column size can also be set in pixels (cols="200,500"), and one of the columns can be set to use the remaining space, with an asterisk (cols="25%,\*").

## Basic Notes - Useful Tips

**Tip:** If a frame has visible borders, the user can resize it by dragging the border. To prevent a user from doing this, you can add noresize="noresize" to the <frame> tag.

**Note:** Add the <noframes> tag for browsers that do not support frames.

**Important:** You cannot use the <body></body> tags together with the <frameset></frameset> tags! However, if you add a <noframes> tag containing some text for browsers that do not support frames, you will have to enclose the text in <body></body> tags! See how it is done in the first example below.

**How to use the <noframes> tag (for browsers that do not support frames):**

```
<html>

<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm" />
  <frame src="frame_b.htm" />
  <frame src="frame_c.htm" />

<noframes>
```

```
<body>Your browser does not handle frames!</body>
</noframes>

</frameset>

</html>
```

**How to create a frameset with three documents, and how to mix them in rows and columns.**

```
<html>

<frameset rows="50%,50%">

  <frame src="frame_a.htm" />
  <frameset cols="25%,75%">
    <frame src="frame_b.htm" />
    <frame src="frame_c.htm" />
  </frameset>
</frameset>

</html>
```

**Frameset with noresize="noresize"**

```
<html>

<frameset rows="50%,50%">

  <frame noresize="noresize" src="frame_a.htm" />
  <frame noresize="noresize" src="frame_b.htm" />
</frameset>

</html>
```

**Navigation**

```
<html>

<frameset cols="120,*">

  <frame src="tryhtml_contents.htm" />
  <frame src="frame_a.htm" name="showframe" />
</frameset>

</html>
```

## Website Layouts

Most websites have put their content in multiple columns (formatted like a magazine or newspaper).

Multiple columns are created by using <table> or <div> tags. Some CSS are normally also added to position elements, or to create backgrounds or colorful look for the pages.

## HTML Layouts - Using Tables

The simplest way of creating layouts is by using the HTML <table> tag.

The following example uses a table with 3 rows and 2 columns - the first and last row spans both columns using the colspan attribute:

### Example

```
<html>
<body>

<table width="500" border="0">
<tr>
<td colspan="2" style="background-color:#FFA500;">
<h1>Main Title of Web Page</h1>
</td>
</tr>

<tr valign="top">
<td style="background-color:#FFD700;width:100px;text-align:top;">
<b>Menu</b><br />
HTML<br />
CSS<br />
JavaScript
</td>
<td style="background-color:#EEEEEE;height:200px;width:400px;text-align:top;">
Content goes here</td>
</tr>

<tr>
<td colspan="2" style="background-color:#FFA500;text-align:center;">
Copyright © 2011 W3Schools.com</td>
</tr>
</table>

</body>
</html>
```

The HTML code above will produce the following result:

## Main Title of Web Page

Menu Content goes here

HTML

CSS

JavaScript

Copyright © 2011 W3Schools.com

**Note:** Even though it is possible to create nice layouts with HTML tables, tables were designed for presenting tabular data - NOT as a layout tool!

## HTML Layouts - Using Div Elements

The div element is a block level element used for grouping HTML elements.

The following example uses five div elements to create a multiple column layout, creating the same result as in the previous example:

### Example

```
<html>
```

```
<body>
```

```
<div id="container" style="width:500px">
```

```
<div id="header" style="background-color:#FFA500;">
```

```
<h1 style="margin-bottom:0;">Main Title of Web Page</h1></div>
```

```
<div id="menu" style="background-color:#FFD700;height:200px;width:100px;float:left;">
```

```
<b>Menu</b><br />
```

```
HTML<br />
```

```
CSS<br />
```

```
JavaScript</div>
```

```
<div id="content" style="background-color:#EEEEEE;height:200px;width:400px;float:left;">
```

```
Content goes here</div>
```

Naresh i Technologies, Opp. Satyam Theatre, Ameerpet, Hyderabad, Ph: 040-23746666, 23734842

An ISO 9001 : 2000 Certified Company

```
<div id="footer" style="background-color:#FFA500;clear:both;text-align:center;">
Copyright © 2011 W3Schools.com</div>

</div>

</body>
</html>
```

The HTML code above will produce the following result:

## Main Title of Web Page

Menu

HTML

CSS

JavaScript

Content goes here

Copyright © 2011 W3Schools.com

## HTML Layout - Useful Tips

**Tip:** The biggest advantage of using CSS is that, if you place the CSS code in an external style sheet, your site becomes MUCH EASIER to maintain. You can change the layout of all your pages by editing one file.

**Tip:** Because advanced layouts take time to create, a quicker option is to use a template. Search Google for free website templates (these are pre-built website layouts you can use and customize)