Syllogistek
Systems Private Limited

502 SreeNilayam, Lanco Hills Road,
Manikonda, Hyderabad – 500089
08413485772 (landline) / 9701377070
Email : hr@syllogistek.com

Contents

HTML

Short for Hyper Text Markup Language, the authoring language used to create documents on the World Wide Web. **HTML** defines the structure and layout of a Web document by using a variety of tags and attributes.

Definition and Usage

The <!DOCTYPE> declaration must be the very first thing in your HTML document, before the <html> tag.

The <!DOCTYPE> declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in.

In HTML 4.01, the <!DOCTYPE> declaration refers to a DTD, because HTML 4.01 was based on SGML. The DTD specifies the rules for the markup language, so that the browsers render the content correctly.

HTML5 is not based on SGML, and therefore does not require a reference to a DTD.

Tip: Always add the <!DOCTYPE> declaration to your HTML documents, so that the browser knows what type of document to expect.

Browser Support

- Internet Explorer
- Google Chrome
- Mozilla Firefox
- Apple Safari
- Opera

Notes

Tip: The <!DOCTYPE> declaration is NOT case sensitive.

Tip: To check if the HTML of your Web documents is valid, go to http://validator.w3.org/

Common DOCTYPE Declarations

HTML 5

```
<!DOCTYPE html>
```

HTML 4.01 Strict

This DTD contains all HTML elements and attributes, but does NOT INCLUDE presentational or deprecated elements (like font). 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">
```

XHTML 1.0 Strict

This DTD contains all HTML elements and attributes, but does NOT INCLUDE presentational or deprecated elements (like font). Framesets are not allowed. The markup must also be written as well-formed XML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

XHTML 1.0 Transitional

This DTD contains all HTML elements and attributes, INCLUDING presentational and deprecated elements (like font). Framesets are not allowed. The markup must also be written as well-formed XML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

XHTML 1.0 Frameset

This DTD is equal to XHTML 1.0 Transitional, but allows the use of frameset content.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

XHTML 1.1

This DTD is equal to XHTML 1.0 Strict, but allows you to add modules (for example to provide Ruby support for East-Asian languages).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Example

```
<!DOCTYPE html>
<html>
<head>
<title>Title of the document</title>
</head>

<body>
The content of the document.....
```

```
</body>

</html>
```

HTML Tags

Basic HTML

Tag	Description
<!DOCTYPE>	Defines the document type
<html>	Defines an HTML document
<head>	Defines information about the document
<title>	Defines a title for the document
<body>	Defines the document's body
<h1> to <h6>	Defines HTML headings
<p>	Defines a paragraph
 	Inserts a single line break
<hr>	Defines a thematic change in the content
<!--...-->	Defines a comment

Formatting

Tag	Description
<acronym>	Not supported in HTML5. Use <abbr> instead.
	Defines an acronym
<abbr>	Defines an abbreviation or an acronym
<address>	Defines contact information for the author/owner of a document/article
<b>	Defines bold text
<bdi>	Isolates a part of text that might be formatted in a different direction from other text outside it
<bdo>	Overrides the current text direction
<big>	Not supported in HTML5. Use CSS instead.
	Defines big text
<blockquote>	Defines a section that is quoted from another source
	Not supported in HTML5. Use CSS instead.
	Defines centered text
<cite>	Defines the title of a work
<code>	Defines a piece of computer code
<del>	Defines text that has been deleted from a document
<dfn>	Represents the defining instance of a term
	Defines emphasized text
<font>	Not supported in HTML5. Use CSS instead.

	Defines font, color, and size for text
<i>	Defines a part of text in an alternate voice or mood
<ins>	Defines a text that has been inserted into a document
<kbd>	Defines keyboard input
<mark>	Defines marked/highlighted text
<meter>	Defines a scalar measurement within a known range (a gauge)
<pre>	Defines preformatted text
<progress>	Represents the progress of a task
<q>	Defines a short quotation
<rp>	Defines what to show in browsers that do not support ruby annotations
<rt>	Defines an explanation/pronunciation of characters (for East Asian typography)
<ruby>	Defines a ruby annotation (for East Asian typography)
<s>	Defines text that is no longer correct
<samp>	Defines sample output from a computer program
<small>	Defines smaller text
<strike>	Not supported in HTML5. Use <del> or <s> instead.
	Defines strikethrough text
	Defines important text
<sub>	Defines subscripted text
<sup>	Defines superscripted text
<template>	Defines a template
<time>	Defines a date/time
<tt>	Not supported in HTML5. Use CSS instead.
	Defines teletype text
<u>	Defines text that should be stylistically different from normal text
<var>	Defines a variable
<wbr>	Defines a possible line-break

Forms and Input

Tag	Description
<form>	Defines an HTML form for user input
<input>	Defines an input control
<textarea>	Defines a multiline input control (text area)
<button>	Defines a clickable button
<select>	Defines a drop-down list
<optgroup>	Defines a group of related options in a drop-down list

<option>	Defines an option in a drop-down list
<label>	Defines a label for an <input> element
<fieldset>	Groups related elements in a form
<legend>	Defines a caption for a <fieldset> element
<datalist>	Specifies a list of pre-defined options for input controls
<output>	Defines the result of a calculation

Frames

Tag	Description
<frame>	Not supported in HTML5.
	Defines a window (a frame) in a frameset
<frameset>	Not supported in HTML5.
	Defines a set of frames
<noframes>	Not supported in HTML5.
	Defines an alternate content for users that do not support frames
<iframe>	Defines an inline frame

Images

Tag	Description
<img>	Defines an image
<map>	Defines a client-side image-map
<area>	Defines an area inside an image-map
<canvas>	Used to draw graphics, on the fly, via scripting (usually JavaScript)
<figcaption>	Defines a caption for a <figure> element
<figure>	Specifies self-contained content
<picture>	Defines a container for multiple image resources
<svg>	Defines a container for SVG graphics

Audio / Video

Tag	Description
<audio>	Defines sound content
<source>	Defines multiple media resources for media elements (<video>, <audio> and <picture>)
<track>	Defines text tracks for media elements (<video> and <audio>)
<video>	Defines a video or movie

Links

Tag	Description

<a>	Defines a hyperlink
<link>	Defines the relationship between a document and an external resource (most used to link to style sheets)
<nav>	Defines navigation links

Lists

Tag	Description
	Defines an unordered list
	Defines an ordered list
	Defines a list item
<dir>	Not supported in HTML5. Use instead.
	Defines a directory list
<dl>	Defines a description list
<dt>	Defines a term/name in a description list
<dd>	Defines a description of a term/name in a description list

Tables

Tag	Description
<table>	Defines a table
<caption>	Defines a table caption
<th>	Defines a header cell in a table
<tr>	Defines a row in a table
<td>	Defines a cell in a table
<thead>	Groups the header content in a table
<tbody>	Groups the body content in a table
<tfoot>	Groups the footer content in a table
<col>	Specifies column properties for each column within a <colgroup> element
<colgroup>	Specifies a group of one or more columns in a table for formatting

Styles and Semantics

Tag	Description
<style>	Defines style information for a document
	Defines a section in a document
	Defines a section in a document
<header>	Defines a header for a document or section
<footer>	Defines a footer for a document or section
<main>	Specifies the main content of a document
<section>	Defines a section in a document
<article>	Defines an article
<aside>	Defines content aside from the page content

<details>	Defines additional details that the user can view or hide
<dialog>	Defines a dialog box or window
<summary>	Defines a visible heading for a <details> element
<data>	Links the given content with a machine-readable translation

Meta Info

Tag	Description
<head>	Defines information about the document
<meta>	Defines metadata about an HTML document
<base>	Specifies the base URL/target for all relative URLs in a document
<basefont>	Not supported in HTML5. Use CSS instead.
	Specifies a default color, size, and font for all text in a document

Programming

Tag	Description
<script>	Defines a client-side script
<noscript>	Defines an alternate content for users that do not support client-side scripts
<applet>	Not supported in HTML5. Use <embed> or <object> instead.
	Defines an embedded applet
<embed>	Defines a container for an external (non-HTML) application
<object>	Defines an embedded object
<param>	Defines a parameter for an object

Example

```
<!DOCTYPE html>
<html>
<!--Only the head and body elements are supposed to be direct descendants of the html element.
   All others should be descendants of either the head or body-->
<head>
   <!--The head element must be a direct descendant of the html element-->
   <title>Your Webpage Title Goes Here</title>
</head>
<body>
   <!--The body element contains the full visible content of the web page-->
   <div class="header">
      <!--The header typically includes your logo, tagline, and may contain a nav element-->
      <div class=”nav”>
         <!--The nav element isn't used for every single link but for navigational menus-->
      </nav>
   </div>
   <div class="main">
```

```
        <!--The main element cannot be used inside of anything other than the body element.
            It is intended to hold the main content of the page.-->
        <div class="nav">
            <!--You can use a nav element just about anywhere-->
        </div>
        <div class="article">
            <!--If your web page contains a blog post or news article it makes sense to wrap the whole article in article
tags-->
            <div class="aside">
                <!--The aside tag can be used within an article or outside of it.
                    It is used to mark content that is related but not central to the main content of the page-->
            </div>
            <div class="section">
                <!--Sections are used to separate major parts of an element,
                    such as chapters of an HTML eBook, -->
            </div>
            <div class="address">
                <!--An address element inside of an article element is used to provide contact info for the author of the
article-->
            </div>
        </div>
        <div class="aside">
            <!--The aside element would also be used to mark a sidebar if used outside of the main element-->
            <div class="section">
                <!--Within a sidebar you could use section elements to identify the different parts of the sidebar. For example,
you could put adds in one section, related posts in a second section, and a newsletter signup form in a third section
element.-->
            </div>
        </div>
    </div>
    <div class="footer">
        <!--The footer typically contains links to things like About Us, Privacy Policy, Contact Us and so forth. It may also
contain a nav, address, section, or aside element.-->
        <div class="address">
            <!--Put an address element in the footer and you're indicating that the        contact info within the element is for
the owner of the website rather than the author of the article.-->
        </div>
    </div>
</body>
</html>
```

CSS - Cascade style sheet

align-content	Specifies the alignment between the lines inside a flexible container when the items do not use all available space
align-items	Specifies the alignment for items inside a flexible container
align-self	Specifies the alignment for selected items inside a flexible container
All	Resets all properties (except unicode-bidi and direction)
Animation	A shorthand property for all the *animation-** properties
animation-delay	Specifies a delay for the start of an animation
animation-direction	Specifies whether an animation should be played forwards, backwards or in alternate cycles
animation-duration	Specifies how long an animation should take to complete one cycle
animation-fill-mode	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)
animation-iteration-count	Specifies the number of times an animation should be played
animation-name	Specifies a name for the @keyframes animation

animation-play-state	Specifies whether the animation is running or paused
animation-timing-function	Specifies the speed curve of an animation
backface-visibility	Defines whether or not the back face of an element should be visible when facing the user
Background	A shorthand property for all the *background-** properties
background-attachment	Sets whether a background image scrolls with the rest of the page, or is fixed
background-blend-mode	Specifies the blending mode of each background layer (color/image)
background-clip	Defines how far the background (color or image) should extend within an element
background-color	Specifies the background color of an element
background-image	Specifies one or more background images for an element
background-origin	Specifies the origin position of a background image
background-position	Specifies the position of a background image
background-repeat	Sets if/how a background image will be repeated
background-size	Specifies the size of the background images
Border	A shorthand property for *border-width*, *border-style* and *border-color*
border-bottom	A shorthand property for *border-bottom-width*, *border-bottom-style* and *border-bottom-color*
border-bottom-color	Sets the color of the bottom border
border-bottom-left-radius	Defines the radius of the border of the bottom-left corner
border-bottom-right-radius	Defines the radius of the border of the bottom-right corner
border-bottom-style	Sets the style of the bottom border
border-bottom-width	Sets the width of the bottom border
border-collapse	Sets whether table borders should collapse into a single border or be separated
border-color	Sets the color of the four borders
border-image	A shorthand property for all the *border-image-** properties
border-image-outset	Specifies the amount by which the border image area extends beyond the border box
border-image-repeat	Specifies whether the border image should be repeated, rounded or stretched
border-image-slice	Specifies how to slice the border image
border-image-source	Specifies the path to the image to be used as a border
border-image-width	Specifies the width of the border image
border-left	A shorthand property for all the *border-left-** properties
border-left-color	Sets the color of the left border
border-left-style	Sets the style of the left border
border-left-width	Sets the width of the left border
border-radius	A shorthand property for the four *border-*-radius* properties
border-right	A shorthand property for all the *border-right-** properties
border-right-color	Sets the color of the right border
border-right-style	Sets the style of the right border
border-right-width	Sets the width of the right border

border-spacing	Sets the distance between the borders of adjacent cells
border-style	Sets the style of the four borders
border-top	A shorthand property for *border-top-width, border-top-style* and *border-top-color*
border-top-color	Sets the color of the top border
border-top-left-radius	Defines the radius of the border of the top-left corner
border-top-right-radius	Defines the radius of the border of the top-right corner
border-top-style	Sets the style of the top border
border-top-width	Sets the width of the top border
border-width	Sets the width of the four borders
Bottom	Sets the elements position, from the bottom of its parent element
box-decoration-break	Sets the behavior of the background and border of an element at page-break, or, for in-line elements, at line-break.
box-shadow	Attaches one or more shadows to an element
box-sizing	Defines how the width and height of an element are calculated: should they include padding and borders, or not
break-after	Specifies the page-, column-, or region-break behavior after the generated box
break-before	Specifies the page-, column-, or region-break behavior before the generated box
break-inside	Specifies the page-, column-, or region-break behavior inside the generated box
caption-side	Specifies the placement of a table caption
caret-color	Specifies the color of the cursor (caret) in inputs, textareas, or any element that is editable
@charset	Specifies the character encoding used in the style sheet
Clear	Specifies on which sides of an element floating elements are not allowed to float
Clip	Clips an absolutely positioned element
Color	Sets the color of text
column-count	Specifies the number of columns an element should be divided into
column-fill	Specifies how to fill columns, balanced or not
column-gap	Specifies the gap between the columns
column-rule	A shorthand property for all the *column-rule-** properties
column-rule-color	Specifies the color of the rule between columns
column-rule-style	Specifies the style of the rule between columns
column-rule-width	Specifies the width of the rule between columns
column-span	Specifies how many columns an element should span across
column-width	Specifies the column width
Columns	A shorthand property for *column-width* and *column-count*
Content	Used with the :before and :after pseudo-elements, to insert generated content
counter-increment	Increases or decreases the value of one or more CSS counters
counter-reset	Creates or resets one or more CSS counters
Cursor	Specifies the mouse cursor to be displayed when pointing over an element

Direction	Specifies the text direction/writing direction
Display	Specifies how a certain HTML element should be displayed
empty-cells	Specifies whether or not to display borders and background on empty cells in a table
Filter	Defines effects (e.g. blurring or color shifting) on an element before the element is displayed
Flex	A shorthand property for the *flex-grow, flex-shrink*, and the *flex-basis* properties
flex-basis	Specifies the initial length of a flexible item
flex-direction	Specifies the direction of the flexible items
flex-flow	A shorthand property for the *flex-direction* and the *flex-wrap* properties
flex-grow	Specifies how much the item will grow relative to the rest
flex-shrink	Specifies how the item will shrink relative to the rest
flex-wrap	Specifies whether the flexible items should wrap or not
Float	Specifies whether or not a box should float
Font	A shorthand property for the *font-style, font-variant, font-weight, font-size/line-height*, and the *font-family* properties
@font-face	A rule that allows websites to download and use fonts other than the "web-safe" fonts
font-family	Specifies the font family for text
font-feature-settings	Allows control over advanced typographic features in OpenType fonts
@font-feature-values	Allows authors to use a common name in font-variant-alternate for feature activated differently in OpenType
font-kerning	Controls the usage of the kerning information (how letters are spaced)
font-language-override	Controls the usage of language-specific glyphs in a typeface
font-size	Specifies the font size of text
font-size-adjust	Preserves the readability of text when font fallback occurs
font-stretch	Selects a normal, condensed, or expanded face from a font family
font-style	Specifies the font style for text
font-synthesis	Controls which missing typefaces (bold or italic) may be synthesized by the browser
font-variant	Specifies whether or not a text should be displayed in a small-caps font
font-variant-alternates	Controls the usage of alternate glyphs associated to alternative names defined in @font-feature-values
font-variant-caps	Controls the usage of alternate glyphs for capital letters
font-variant-east-asian	Controls the usage of alternate glyphs for East Asian scripts (e.g Japanese and Chinese)
font-variant-ligatures	Controls which ligatures and contextual forms are used in textual content of the elements it applies to
font-variant-numeric	Controls the usage of alternate glyphs for numbers, fractions, and ordinal markers
font-variant-position	Controls the usage of alternate glyphs of smaller size positioned as superscript or subscript regarding the baseline of the font

font-weight	Specifies the weight of a font
Grid	A shorthand property for the *grid-template-rows, grid-template-columns, grid-template-areas, grid-auto-rows, grid-auto-columns*, and the *grid-auto-flow*properties
grid-area	Either specifies a name for the grid item, or this property is a shorthand property for the *grid-row-start*, *grid-column-start*, *grid-row-end*, and *grid-column-end*properties
grid-auto-columns	Specifies a default column size
grid-auto-flow	Specifies how auto-placed items are inserted in the grid
grid-auto-rows	Specifies a default row size
grid-column	A shorthand property for the *grid-column-start* and the *grid-column-end*properties
grid-column-end	Specifies where to end the grid item
grid-column-gap	Specifies the size of the gap between columns
grid-column-start	Specifies where to start the grid item
grid-gap	A shorthand property for the *grid-row-gap* and *grid-column-gap* properties
grid-row	A shorthand property for the *grid-row-start* and the *grid-row-end* properties
grid-row-end	Specifies where to end the grid item
grid-row-gap	Specifies the size of the gap between rows
grid-row-start	Specifies where to start the grid item
grid-template	A shorthand property for the *grid-template-rows*, *grid-template-columns* and *grid-areas* properties
grid-template-areas	Specifies how to display columns and rows, using named grid items
grid-template-columns	Specifies the size of the columns, and how many columns in a grid layout
grid-template-rows	Specifies the size of the rows in a grid layout
hanging-punctuation	Specifies whether a punctuation character may be placed outside the line box
Height	Sets the height of an element
Hyphens	Sets how to split words to improve the layout of paragraphs
image-rendering	Gives a hint to the browser about what aspects of an image are most important to preserve when the image is scaled
@import	Allows you to import a style sheet into another style sheet
Isolation	Defines whether an element must create a new stacking content
justify-content	Specifies the alignment between the items inside a flexible container when the items do not use all available space
@keyframes	Specifies the animation code
Left	Specifies the left position of a positioned element
letter-spacing	Increases or decreases the space between characters in a text
line-break	Specifies how/if to break lines
line-height	Sets the line height
list-style	Sets all the properties for a list in one declaration
list-style-image	Specifies an image as the list-item marker

list-style-position	Specifies the position of the list-item markers (bullet points)
list-style-type	Specifies the type of list-item marker
Margin	Sets all the margin properties in one declaration
margin-bottom	Sets the bottom margin of an element
margin-left	Sets the left margin of an element
margin-right	Sets the right margin of an element
margin-top	Sets the top margin of an element
max-height	Sets the maximum height of an element
max-width	Sets the maximum width of an element
@media	Sets the style rules for different media types/devices/sizes
min-height	Sets the minimum height of an element
min-width	Sets the minimum width of an element
mix-blend-mode	Specifies how an element's content should blend with its direct parent background
object-fit	Specifies how the contents of a replaced element should be fitted to the box established by its used height and width
object-position	Specifies the alignment of the replaced element inside its box
Opacity	Sets the opacity level for an element
Order	Sets the order of the flexible item, relative to the rest
Orphans	Sets the minimum number of lines that must be left at the bottom of a page when a page break occurs inside an element
Outline	A shorthand property for the *outline-width, outline-style*, and the *outline-color*properties
outline-color	Sets the color of an outline
outline-offset	Offsets an outline, and draws it beyond the border edge
outline-style	Sets the style of an outline
outline-width	Sets the width of an outline
Overflow	Specifies what happens if content overflows an element's box
overflow-wrap	Specifies whether or not the browser may break lines within words in order to prevent overflow (when a string is too long to fit its containing box)
overflow-x	Specifies whether or not to clip the left/right edges of the content, if it overflows the element's content area
overflow-y	Specifies whether or not to clip the top/bottom edges of the content, if it overflows the element's content area
Padding	A shorthand property for all the *padding-** properties
padding-bottom	Sets the bottom padding of an element
padding-left	Sets the left padding of an element
padding-right	Sets the right padding of an element
padding-top	Sets the top padding of an element
page-break-after	Sets the page-break behavior after an element
page-break-before	Sets the page-break behavior before an element
page-break-inside	Sets the page-break behavior inside an element
Perspective	Gives a 3D-positioned element some perspective
perspective-origin	Defines at which position the user is looking at the 3D-positioned element

pointer-events	Defines whether or not an element reacts to pointer events
Position	Specifies the type of positioning method used for an element (static, relative, absolute or fixed)
Quotes	Sets the type of quotation marks for embedded quotations
Resize	Defines if (and how) an element is resizable by the user
Right	Specifies the right position of a positioned element
scroll-behavior	Specifies whether to smoothly animate the scroll position in a scrollable box, instead of a straight jump
tab-size	Specifies the width of a tab character
table-layout	Defines the algorithm used to lay out table cells, rows, and columns
text-align	Specifies the horizontal alignment of text
text-align-last	Describes how the last line of a block or a line right before a forced line break is aligned when text-align is "justify"
text-combine-upright	Specifies the combination of multiple characters into the space of a single character
text-decoration	Specifies the decoration added to text
text-decoration-color	Specifies the color of the text-decoration
text-decoration-line	Specifies the type of line in a text-decoration
text-decoration-style	Specifies the style of the line in a text decoration
text-indent	Specifies the indentation of the first line in a text-block
text-justify	Specifies the justification method used when text-align is "justify"
text-orientation	Defines the orientation of the text in a line
text-overflow	Specifies what should happen when text overflows the containing element
text-shadow	Adds shadow to text
text-transform	Controls the capitalization of text
text-underline-position	Specifies the position of the underline which is set using the text-decoration property
Top	Specifies the top position of a positioned element
Transform	Applies a 2D or 3D transformation to an element
transform-origin	Allows you to change the position on transformed elements
transform-style	Specifies how nested elements are rendered in 3D space
Transition	A shorthand property for all the *transition-** properties
transition-delay	Specifies when the transition effect will start
transition-duration	Specifies how many seconds or milliseconds a transition effect takes to complete
transition-property	Specifies the name of the CSS property the transition effect is for
transition-timing-function	Specifies the speed curve of the transition effect
unicode-bidi	Used together with the direction property to set or return whether the text should be overridden to support multiple languages in the same document
user-select	Specifies whether the text of an element can be selected
vertical-align	Sets the vertical alignment of an element
Visibility	Specifies whether or not an element is visible
white-space	Specifies how white-space inside an element is handled
Widows	Sets the minimum number of lines that must be left at the top of a page when a page break occurs inside an element

Width	Sets the width of an element
word-break	Specifies how words should break when reaching the end of a line
word-spacing	Increases or decreases the space between words in a text
word-wrap	Allows long, unbreakable words to be broken and wrap to the next line
writing-mode	Specifies whether lines of text are laid out horizontally or vertically
z-index	Sets the stack order of a positioned element

```
div {
  background-color: yellow;
  color: red;
  all: initial;
}
```

JavaScript HTML DOM

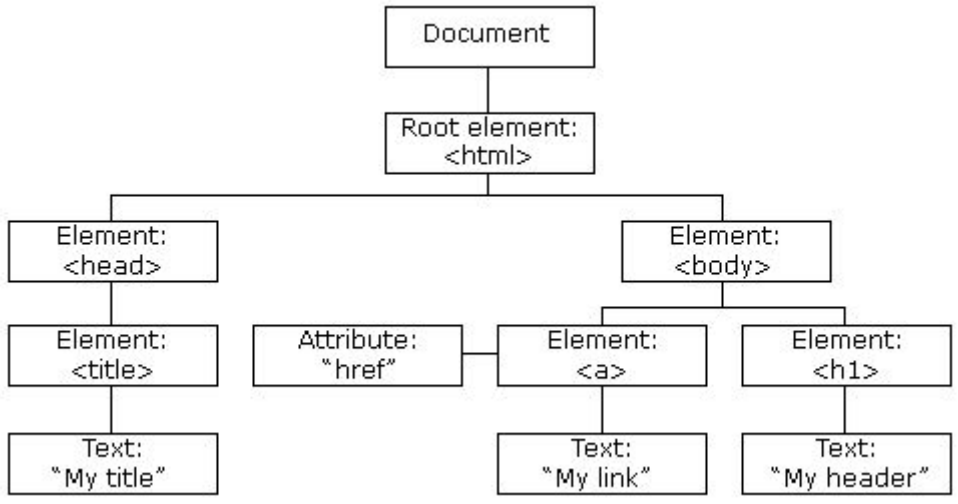
With the HTML DOM, JavaScript can access and change all the elements of an HTML document.

The HTML DOM (Document Object Model)

When a web page is loaded, the browser creates a **D**ocument **O**bject **M**odel of the page.

The **HTML DOM** model is constructed as a tree of **Objects**:

DOM Object Tree



With the object model, JavaScript gets all the power it needs to create dynamic HTML:

- JavaScript can change all the HTML elements in the page
- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles in the page
- JavaScript can remove existing HTML elements and attributes

- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can create new HTML events in the page

JQuery

jQuery is a fast and concise JavaScript Library created by John Resig in 2006 with a nice motto: **Write less, do more**. jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is a JavaScript toolkit designed to simplify various tasks by writing less code. Here is the list of important core features supported by jQuery −

- **DOM manipulation** − The jQuery made it easy to select DOM elements, negotiate them and modifying their content by using cross-browser open source selector engine called **Sizzle**.
- **Event handling** − The jQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.
- **AJAX Support** − The jQuery helps you a lot to develop a responsive and featurerich site using AJAX technology.
- **Animations** − The jQuery comes with plenty of built-in animation effects which you can use in your websites.
- **Lightweight** − The jQuery is very lightweight library - about 19KB in size (Minified and gzipped).
- **Cross Browser Support** − The jQuery has cross-browser support, and works well in IE 6.0+, FF 2.0+, Safari 3.0+, Chrome and Opera 9.0+
- **Latest Technology** − The jQuery supports CSS3 selectors and basic XPath syntax.

There are two ways to use jQuery.

- **Local Installation** − You can download jQuery library on your local machine and include it in your HTML code.
- **CDN Based Version** − You can include jQuery library into your HTML code directly from Content Delivery Network (CDN).

Local Installation

- Go to the https://jquery.com/download/ to download the latest version available.
- Now put downloaded **jquery-2.1.3.min.js** file in a directory of your website, e.g. /jquery.

Example

Now you can include *jquery* library in your HTML file as follows −

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript" src = "/jquery/jquery-2.1.3.min.js">
    </script>

    <script type = "text/javascript">
      $(document).ready(function() {
        document.write("Hello, World!");
      });
```

```
    </script>
  </head>

  <body>
    <h1>Hello</h1>
  </body>
</html>
```

This will produce following result −

CDN Based Version

You can include jQuery library into your HTML code directly from Content Delivery Network (CDN). Google and Microsoft provides content deliver for the latest version.

We are using Google CDN version of the library throughout this tutorial.

Example

Now let us rewrite above example using jQuery library from Google CDN.

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript"
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
    </script>

    <script type = "text/javascript">
      $(document).ready(function() {
        document.write("Hello, World!");
      });
    </script>
  </head>
  <body>
    <h1>Hello</h1>
  </body>
</html>
```

This will produce following result −

How to Call a jQuery Library Functions?

As almost everything, we do when using jQuery reads or manipulates the document object model (DOM), we need to make sure that we start adding events etc. as soon as the DOM is ready.

If you want an event to work on your page, you should call it inside the $(document).ready() function. Everything inside it will load as soon as the DOM is loaded and before the page contents are loaded.

To do this, we register a ready event for the document as follows −

```
$(document).ready(function() {
  // do stuff when DOM is ready
});
```

To call upon any jQuery library function, use HTML script tags as shown below −

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript"
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
    </script>

    <script type = "text/javascript" language = "javascript">
      $(document).ready(function() {
        $("div").click(function() {alert("Hello, world!");});
      });
```

```
    </script>
  </head>

  <body>
    <div id = "mydiv">
      Click on this to see a dialogue box.
    </div>
  </body>
</html>
```

This will produce following result −

How to Use Custom Scripts?

It is better to write our custom code in the custom JavaScript file : **custom.js**, as follows −

```
/* Filename: custom.js */
$(document).ready(function() {

  $("div").click(function() {
    alert("Hello, world!");
  });
});
```

Now we can include custom.js file in our HTML file as follows −

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript"
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
    </script>

    <script type = "text/javascript" src = "/jquery/custom.js">
    </script>
  </head>

  <body>
    <div id = "mydiv">
      Click on this to see a dialogue box.
    </div>
  </body>
</html>
```

This will produce following result −

Using Multiple Libraries

You can use multiple libraries all together without conflicting each others. For example, you can use jQuery and MooTool javascript libraries together. You can check jQuery noConflict Method for more detail.

JQuery Selector:

The selectors are very useful and would be required at every step while using jQuery. They get the exact element that you want from your HTML document.

1	Name Selects all elements which match with the given element Name.
2	#ID Selects a single element which matches with the given ID.
3	.Class

	Selects all elements which match with the given Class.
4	Universal (*) Selects all elements available in a DOM.
5	Multiple Elements E, F, G Selects the combined results of all the specified selectors E, F or G.

JQuery get() Method

The jQuery get() method sends asynchronous http GET request to the server and retrieves the data.

$.get(url, [data],[callback]);
Parameters Description:

- url: request url from which you want to retrieve the data
- data: data to be sent to the server with the request as a query string
- callback: function to be executed when request succeeds

The following example shows how to retrieve data from a text file.

Example: jQuery get() Method

```
$.get('/data.txt',  // url
    function (data, textStatus, jqXHR) {  // success callback
        alert('status: ' + textStatus + ', data:' + data);
   });
```

In the above example, first parameter is a url from which we want to retrieve the data. Here, we want to retrieve data from a txt file located at mydomain.com/data.txt. Please note that you don't need to give base address.

Internally, jQuery get() method calls ajax() method only.

The second parameter is a callback function that will be executed when this GET request succeeds. This callback function includes three parameters data, textStatus and jQuery wrapper of XMLHttpRequest object. Data contains response data, textStatus contains status of request and jqXHR is a jQuery XMLHttpRequest object which you can use for further process.

The following example shows how to retrieve JSON data using get() method.

Example: Retrieve JSON Data using get()

```
$.get('/jquery/getjsondata', {name:'Steve'}, function (data, textStatus, jqXHR) {
    $('p').append(data.firstName);
});

<p></p>
```

In the above example, first parameter is a url from which we want to retrieve JSON data. This url can be a web service or any other url that returns data in JSON format.

The second parameter is data to be sent to the server as a query string. We have specified name parameter with value 'Steve' in the JSON format. So now, the request url would look like http://mydomain.com/jquery/getjsondata?name=Arghya

The third parameter is a callback function that will be executed when this GET request succeeds.

JQuery post() Method

This is a shorthand Ajax function, which is equivalent to:

```
$.ajax({
  type: "POST",
  url: url,
  data: data,
  success: success,
  dataType: dataType
});
```

The success callback function is passed the returned data, which will be an XML root element or a text string depending on the MIME type of the response. It is also passed the text status of the response.

As of jQuery 1.5, the success callback function is also passed a "jqXHR" object (in **jQuery 1.4**, it was passed the XMLHttpRequest object).

Most implementations will specify a success handler:

```
$.post( "ajax/test.html", function( data ) {
  $( ".result" ).html( data );
});
```

This example fetches the requested HTML snippet and inserts it on the page.

Pages fetched with POST are never cached, so the cache and ifModified options in jQuery.ajaxSetup() have no effect on these requests.

The jqXHR Object

As of jQuery 1.5, all of jQuery's Ajax methods return a superset of the XMLHTTPRequest object. This jQuery XHR object, or "jqXHR," returned by $.get() implements the Promise interface, giving it all the properties, methods, and behavior of a Promise (see Deferred object for more information). The jqXHR.done() (for success), jqXHR.fail() (for error), and jqXHR.always() (for completion, whether success or error; added in jQuery 1.6) methods take a function argument that is called when the request terminates. For information about the arguments this function receives, see the jqXHR Objectsection of the $.ajax() documentation.

The Promise interface also allows jQuery's Ajax methods, including $.get(), to chain multiple .done(), .fail(), and .always() callbacks on a single request, and even to assign these callbacks after the request may have completed. If the request is already complete, the callback is fired immediately.

```
// Assign handlers immediately after making the request,
// and remember the jqxhr object for this request
var jqxhr = $.post( "example.php", function() {
  alert( "success" );
})
  .done(function() {
```

```
    alert( "second success" );
  })
  .fail(function() {
    alert( "error" );
  })
  .always(function() {
    alert( "finished" );
  });

// Perform other work here ...

// Set another completion function for the request above
jqxhr.always(function() {
  alert( "second finished" );
});
```

Deprecation Notice

The jqXHR.success(), jqXHR.error(), and jqXHR.complete() callback methods are **removed as of jQuery 3.0**. You can use jqXHR.done(), jqXHR.fail(), and jqXHR.always() instead.

Additional Notes:

- Due to browser security restrictions, most "Ajax" requests are subject to the same origin policy; the request can not successfully retrieve data from a different domain, subdomain, port, or protocol.
- If a request with jQuery.post() returns an error code, it will fail silently unless the script has also called the global .ajaxError() method. Alternatively, as of jQuery 1.5, the .error() method of the jqXHR object returned by jQuery.post() is also available for error handling.

Examples:

Request the test.php page, but ignore the return results.

```
$.post( "test.php" );
```

Request the test.php page and send some additional data along (while still ignoring the return results).

```
$.post( "test.php", { name: "John", time: "2pm" } );
```

Pass arrays of data to the server (while still ignoring the return results).

```
$.post( "test.php", { 'choices[]': [ "Jon", "Susan" ] } );
```

Send form data using Ajax requests

```
$.post( "test.php", $( "#testform" ).serialize() );
```

Alert the results from requesting test.php (HTML or XML, depending on what was returned).

```
$.post( "test.php", function( data ) {
  alert( "Data Loaded: " + data );
});
```

Alert the results from requesting test.php with an additional payload of data (HTML or XML, depending on what was returned).

```
$.post( "test.php", { name: "John", time: "2pm" })
  .done(function( data ) {
    alert( "Data Loaded: " + data );
  });
```

Post to the test.php page and get content which has been returned in json format (<?php echo json_encode(array("name"=>"John","time"=>"2pm")); ?>).

```
$.post( "test.php", { func: "getNameAndTime" }, function( data ) {
  console.log( data.name ); // John
  console.log( data.time ); // 2pm
}, "json");
```

Post a form using Ajax and put results in a div

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>jQuery.post demo</title>
  <script src="https://code.jquery.com/jquery-1.10.2.js"></script>
</head>
<body>

<form action="/" id="searchForm">
  <input type="text" name="s" placeholder="Search...">
  <input type="submit" value="Search">
</form>
<!-- the result of the search will be rendered inside this div -->
<div id="result"></div>

<script>
// Attach a submit handler to the form
$( "#searchForm" ).submit(function( event ) {

  // Stop form from submitting normally
  event.preventDefault();

  // Get some values from elements on the page:
  var $form = $( this ),
    term = $form.find( "input[name='s']" ).val(),
    url = $form.attr( "action" );

  // Send the data using post
  var posting = $.post( url, { s: term } );

  // Put the results in a div
  posting.done(function( data ) {
    var content = $( data ).find( "#content" );
    $( "#result" ).empty().append( content );
  });
});
</script>

</body>
```

```
</html>
```

JQuery ready() Method

Use ready() to make a function available after the document is loaded:

```
$(document).ready(function(){
  $("button").click(function(){
    $("p").slideToggle();
  });
});
```

The ready event occurs when the DOM (document object model) has been loaded.

Because this event occurs after the document is ready, it is a good place to have all other jQuery events and functions. Like in the example above.

The ready() method specifies what happens when a ready event occurs.

Tip: The ready() method should not be used together with <body onload="">.

Two syntaxes can be used:

$(document).ready(*function*)

The ready() method can only be used on the current document, so no selector is required:

$(*function*)

Parameter	**Description**
function	Required. Specifies the function to run after the document is loaded