
Unit - 1

WEBSITE BASICS, HTML 5, CSS 3, WEB 2.0

1.1 WEB ESSENTIALS

WEBSITES

A website is a set of related web pages typically served from a single web server.

A website is hosted on at least one web server, accessible via a network such as the internet or a private local area network. The pages of a website can usually be accessed from a simple Uniform Resource Locator (URL) otherwise called as web address. The URLs of the pages organize them into a hierarchy.

Terminologies:

Internet: The Internet is a collection of computers around the world connected to each other via high speed series of networks.

World Wide Web (WWW): The World Wide Web – or Web consists of a vast assortment of files and documents that are stored on the computers and written in some form of Hyper Text Markup Language (HTML).

Servers: The computers that store the files are called servers because they can serve requests from many users at the same time.

Browsers: A Web browser is a program that displays Web pages and other documents on the web. Examples: Internet explorer, Firefox, Google Chrome etc.

HTML: HTML, or Hyper Text Markup Language, is the authoring language that describes how a Web page should be displayed by a Web browser. It has two essential features:

- **Hypertext:** When a visitor clicks a link on a Web page, it leads to another web page

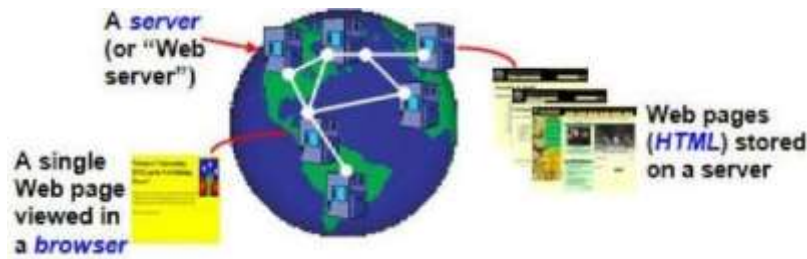


Fig 1.1 Internet and its components

INTERNET

The Internet is a vast, electronic network connecting many millions of computers from every corner of the world. The Internet is a global network of networks.

The Internet is a publicly-accessible network that consists of millions of smaller domestic, academic, business, and government networks. The Internet links are computer networks all over the world so that users can share resources and communicate with each other. People and organizations connect into the Internet so they can access its massive store of shared information.

The internet is a participative medium. Anybody can publish information or create new services. The internet is a **cooperative endeavor** - no organization is in charge of the internet. The following components are essential for an internet connection: Computer, Connection - Phone Line, Cable, DSL, Wireless, Modem, Network Software - TCP/IP, Application Software - Web Browser, Email, etc and Internet Service Provider (ISP).

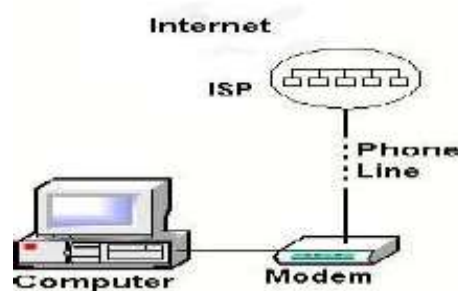


Fig 1.2 Components of internet

Evolution of Internet

The concept of Internet was originated in 1969 and has undergone several technological & infrastructural changes:

- The origin of Internet devised from the concept of Advanced Research Project Agency Network (ARPANET). **ARPANET** was developed by United States Department of Defense.
- Basic purpose of ARPANET was to provide communication among the various bodies of government.
- In 1972, the ARPANET spread over the globe with 23 nodes located at different countries and thus became known as Internet.
- By the time, with invention of new technologies such as TCP/IP protocols, DNS, WWW, browsers, scripting languages etc, Internet provided a medium to publish and access information over the web.

Internet Terminologies

- **Host:** A computer connected to the Internet is commonly referred to as a host.
- **Communication services:** The data is passed back and forth between host computers using packets and protocols, such as electronic mail (e-mail) for messaging, file transfer protocol (FTP) for moving files, telnet for accessing information, hypertext transfer protocol (HTTP) for serving up Web sites, custom protocols, etc. They are called communication services.
- **Internet Service Provider (ISP):** The Internet itself is decentralized-no one is completely responsible or has total control; however, the connection to the Internet is partly controlled by an Internet Service Provider (ISP). Example for ISP: Reliance, Airtel, Idea (IIN) etc.
- **Online:** When the computer is connected to the internet then it is in online.
- **Hyperlinks:** Allow a user to quickly move from one web page to another, even if the pages are on different servers in different parts of the world.
- **Protocols:** They are pre-established means of communication. **Example:** TCP/IP, SMTP.
- **TCP/IP:** TCP is the protocol that establishes a virtual connection between a destination and a source. TCP guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent. Internet Protocol (IP) is responsible for packaging the little packets of information and delivering them.
- **Client/ Server model:** TCP/IP uses the client/server model of communication in which a computer user (a client) requests and is provided a service (such as sending a Web page) by another computer (a server) in the network.

- **IP address:** It is the address of the machine. It is a four byte unique number that identifies a system on the Internet.
- **Domain Name Services (DNS):** They link text to our numeric IP addresses, allowing users to use the DNS as a proxy for the IP address. The IP addresses are often provided by the ISP. Each site must register the name for a cost through a DNS hosting service. DNS host servers then are used to convert our text DNS address to its digital IP address equivalent.
 - **Universal Resource Locators:** URL's are a way of identifying information on a server. A URL gives the protocol, the domain, the directory, and even the file. A URL consists of the following parts: protocol (such as http:// or ftp://), host name (the Web server's IP address or domain name), directory (i.e. folder) and file name
- **World Wide Web:** The World Wide Web consists of all the Web sites and pages served on the Internet via HTTP. It is a hypermedia-based system for browsing Internet sites. It is named the web because it is made of many sites linked together; users can travel from one site to another by clicking on hyperlinks. Text, graphics, sound, and video can all be accessed. **Tim Berners-Lee** invented the World Wide Web in 1989 while working at CERN, the European Particle Physics Laboratory.

BASIC INTERNET PROTOCOLS

Protocol is a set of mutually accepted and implemented rules at both ends of the communications channel for the proper exchange of information.

TCP / IP

Transmission Control Protocol/Internet Protocol (TCP /IP) is a suite of communication protocols used to interconnect network devices on the internet. TCP/IP can also be used as a communications protocol in a private network. TCP/IP specifies how data is exchanged over the internet by providing end-to-end communications that identify how it should be broken into packets, addressed, transmitted, routed and received at the destination.

TCP/IP requires little central management, and it is designed to make networks reliable, with the ability to recover automatically from the failure of any device on the network. TCP defines how applications can create channels of communication across a network. It also manages how a message is assembled into smaller packets before they are then transmitted over the internet and reassembled in the right order at the destination address.

IP defines how to address and route each packet to make sure it reaches the right destination. Each gateway computer on the network checks this IP address to determine where to forward the message. A key element of IP is the IP address, which is simply a 32-bit number. IP addresses are normally written as a sequence of four decimal numbers separated by periods as in 192.0.34.166. When an application on the source computer wants to send

information to a destination, the application calls IP software on the source machine and provides it with data to be transferred along with an IP address for each of the source and destination computers.

The IP software running on the source creates a packet, which is a sequence of bits representing the data to be transferred along with the source and destination IP addresses and some other header information, such as the length of the data. If the destination computer is on the same local network as the source, then the IP software will send the packet to the destination directly via this network.

If the destination is on another network, the IP software will send the packet to a gateway, which is a device that is connected to the source computer's network as well as to at least one other network. The gateway will select a computer on one of the other networks to which it is attached and send the packet on to that computer. This process will continue, until the packet reaches the destination computer.

IP software on that computer will receive the packet and pass its data up to an application that is waiting for the data. TCP, the Transmission Control Protocol, is a higher-level protocol that extends IP to provide additional functionality, including reliable communication based on the concept of a connection.

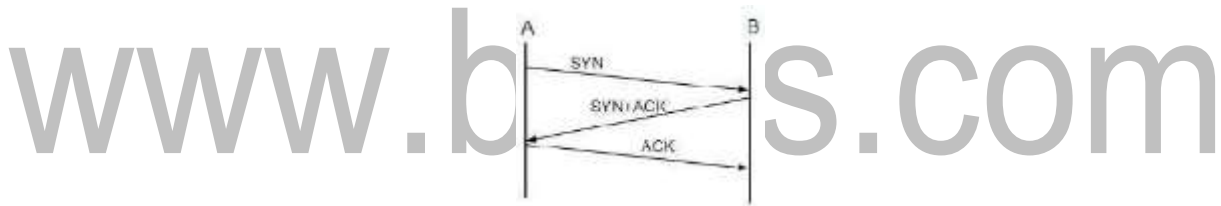


Fig 1.5: Three way handshaking of TCP

A connection is established between TCP software running on two machines by one of the machines sending a connection-request message via IP to the other. If the connection is accepted by B, then B returns a message to A requesting a connection in the other direction. If A responds affirmatively, then the connection is established. Notice that this means that A and B can both send messages. Once a connection has been established, TCP provides reliable data transmission by demanding an acknowledgment for each packet it sends via IP. TCP has contains port to communicate with many different applications on a machine.

UDP, DNS and Domain Names

UDP is connectionless and unreliable protocol. It doesn't require making a connection with the host to exchange data. Since UDP is unreliable protocol, there is no mechanism for ensuring that data sent is received. UDP transmits the data in form of a datagram.

UDP provides protocol port used i.e. UDP message contains both source and destination port number, that makes it possible for UDP software at the destination to deliver the message to correct application program.

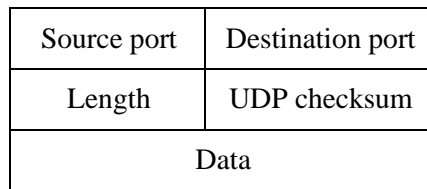


Fig 1.3: UDP Datagram

Differences between TCP and UDP

TCP	UDP
TCP is a connection-oriented protocol.	UDP is a connectionless protocol.
As a message makes its way across the internet from one computer to another. This is connection based.	UDP is also a protocol used in message transport or transfer. This is not connection based which means that one program can send a load of packets to another and that would be the end of the relationship.
TCP is suited for applications that require high reliability, and transmission time is relatively less critical	UDP is suitable for applications that need fast, efficient transmission, such as games. UDP's stateless nature is also useful for servers that answer small queries from huge numbers of clients.
TCP is used by HTTP, HTTPS, FTP, SMTP, Telnet	UDP is used by DNS, DHCP, TFTP, SNMP, RIP, VOIP.
TCP rearranges data packets in the order specified.	UDP has no inherent order as all packets are independent of each other. If ordering is required, it has to be managed by the application layer.
The speed for TCP is slower than UDP.	UDP is faster because there is no error-checking for packets.
There is absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent.	There is no guarantee that the messages or packets sent would reach at all.
TCP header size is 20 bytes	UDP Header size is 8 bytes.
Data is read as a byte stream, no distinguishing indications are transmitted to signal message (segment) boundaries	Packets are sent individually and are checked for integrity only if they arrive. Packets have definite boundaries which are honored upon receipt, meaning a read operation at the receiver socket will yield an entire message as it was originally sent

TCP is heavy-weight. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP is lightweight. There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP.
TCP does Flow Control. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP does not have an option for flow control
TCP does error checking	UDP does error checking, but no recovery options.
Acknowledge segments	No Acknowledgment
Handshaking is done	No handshake (connectionless protocol)

File Transfer Protocol (FTP)

FTP is used to copy files from one host to another. FTP offers the mechanism for the same in following manner:

- FTP creates two processes such as **Control Process and Data Transfer Process** at both ends i.e. at client as well as at server.
- FTP establishes two different connections: one is for data transfer and other is for control information.
- Control connection is made between control processes while Data Connection is made between data transfer process.
- FTP uses port 21 for the control connection and Port 20 for the data connection.

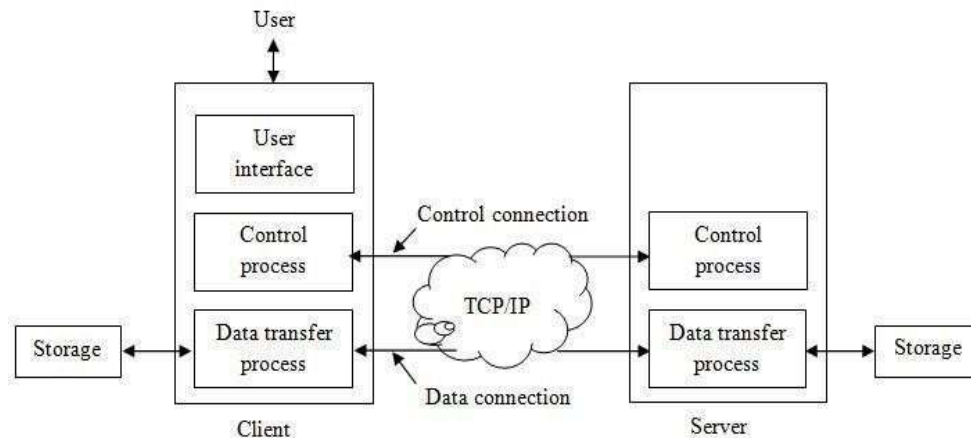


Fig 1.4 FTP

Trivial File Transfer Protocol (TFTP)

Trivial File Transfer Protocol is also used to transfer the files but it transfers the files **without authentication**. Unlike FTP, TFTP does not separate control and data information. Since there is no authentication exists, TFTP lacks in security features therefore it is not recommended to use TFTP.

TFTP makes use of UDP for data transport. Each TFTP message is carried in separate UDP datagram. The first two bytes of a TFTP message specify the type of message. The TFTP session is initiated when a TFTP client sends a request to upload or download a file. The request is sent from an ephemeral UDP port to the UDP port 69 of a TFTP server.

Differences between FTP and TFTP

FTP	TFTP
Authentication is done before transferring files.	No authentication is done before transferring files
The underlying protocol employed is TCP.	The underlying protocol employed is UDP.
Port 20 is used as control port and 21 for data transfers.	Port numbers: 3214, 69, 4012 are used.
Reliable data transfer is provided.	Unreliable data transfer

Telnet

Telnet is a protocol used to log in to remote computer on the internet. There are a number of Telnet clients having user friendly user interface. The following diagram shows a person is logged in to computer A, and from there, he remotely logged into another computer B.

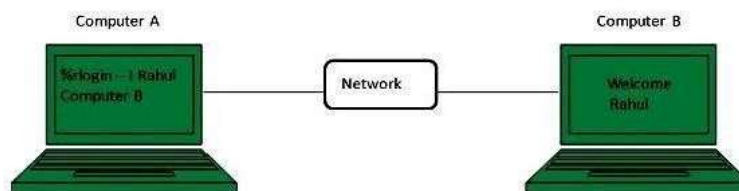


Fig 1.5 Telnet

Hyper Text Transfer Protocol (HTTP)

HTTP is a communication protocol. It defines mechanism for communication between browser and the web server. It is also called request and response protocol because the communication between browser and server takes place in request and response pairs. It is **astateless** protocol (i.e.) the history of the communication between server and client is not stored in any form.

1.5 CASCADING STYLE SHEETS (CSS)

CSS helps to define the presentation of HTML elements as a separate file known as CSS file having .css extension. CSS helps to change formatting of any HTML element by just making changes at one place. All changes made would be reflected automatically to all of the web pages of the website in which that element appeared.

CSS Rules

CSS Rules are the styles that we have to create in order to create style sheets. These rules define appearance of associated HTML element.

Selector {property: value;}

- ❖ Selector is HTML element to which CSS rule is applied.
- ❖ Property specifies the attribute that the user wants to change corresponding to the selector.
- ❖ Property can take specified value. Property and Value are separated by a colon (:). Each declaration is separated by semi colon (;).

Examples: i) P { color : red;} ii) h1 (color : green; font-style : italic } iii) body { color : cyan; font-family : Arial; font- style : 16pt}

Embedding CSS into HTML

Following are the four ways to add CSS to HTML documents:

➤ **Inline Style Sheets**

Inline Style Sheets are included with HTML element i.e. they are placed inline with the element. To add inline CSS, we have to declare style attribute which can contain any CSS property.

Syntax: <Tagname STYLE = “ Declaration1 ; Declaration2 “> </Tagname>	Example: <p style="color: blue; text-align: left; font-size: 15pt">
---	---

➤ **Embedded Style Sheets**

Embedded Style Sheets are used to apply same appearance to all occurrence of a specific element. These are defined in element by using the <style> element. The

<style> element must include type attribute. The value of type attribute specifies what type of syntax it includes when rendered by the browser.

| Syntax: | Example: |
|--|---|
| <pre><head><title> </title> <style type =”text/css”>CSS Rules/Styles....</head></pre> | <pre><style type="text/css"> p { color:green; text-align: left; font-size: 10pt} h1 { color: red; font-weight: bold} </style></pre> |

➤ External Style Sheets

External Style Sheets are the separate .css files that contain the CSS rules. These files can be linked to any HTML documents using <link> tag with rel attribute.

Syntax:

```
<head><link rel= “stylesheet” type=”text/css” href= “url of css file”> </head>
```

In order to create external css and link it to HTML document, follow the following steps:

- ❖ Create a CSS file and define all CSS rules for several HTML elements. Let’s name this file as external.css.

```
p { Color: orange; text-align: left; font-size: 10pt;}
h1 { Color: orange; font-weight: bold;}
```

- ❖ Now create HTML document and name it as externaldemo.html.

```
<html><head> <title> External Style Sheets Demo </title>
<link rel="stylesheet" type="text/css" href="external.css"> </head>
<body> <h1> External Style Sheets</h1>
```

```
<p>External Style Sheets are the separate .css files that contain the CSS
rules.</p> </body> </html>
```

➤ Imported Style Sheets

Imported Style Sheets allow us to import style rules from other style sheets. To import CSS rules we have to use @import before all the rules in a style sheet.

Syntax:	Example:
<pre><head><title> Title Information </title> <style type=”text/css”></pre>	<pre><html><head> <title> External Style Sheets Demo</pre>

<pre>@import URL (cssfilepath) ... CSS rules... </style> </head></style></pre>	<pre></title><style> @import url(external.css); </style> </head> <body> <h1> External Style Sheets</h1> <p>External Style Sheets.</p> </body> </html></pre>
--	---

CSS imported

```
<!DOCTYPE html><html><head><style>
p.ex1 { font: 15px arial, sans-serif;}
p.ex2 {font:italic bold 12px/30px Georgia, serif;}</style></head>
<body>
<p class="ex1">This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.</p>
<p class="ex2">This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.</p></body></html>
```

Output:

This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.

This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.

CSS - Pseudo Classes

CSS pseudo-classes are used to add special effects to some selectors.

Syntax:selector:pseudo-class {property: value }

CSS classes can also be used with pseudo-classes:

Syntax:selector.class:pseudo-class {property: value }

There are following most commonly used pseudo-classes:

Value	Description
:link	Use this class to add special style to an unvisited link.
:visited	Use this class to add special style to a visited link.
:hover	Use this class to add special style to an element when you mouse over it.
:active	Use this class to add special style to an active element.
focus	Use this class to add special style to an element while the element has focus.
:first-child	Use this class to add special style to an element that is the first child of some other element.
:lang	Use this class to specify a language to use in a specified element.

While defining pseudo-classes in a <style>...</style> block, following points should be noted:

- a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.
- a:active MUST come after a:hover in the CSS definition in order to be effective.
- Pseudo-class names are not case-sensitive.
- Pseudo-class are different from CSS classes but they can be combined.

The :link pseudo-class

```
<style type="text/css">
a:link {color:#000000}
</style>
<a href="/html/index.htm">Black Link</a>
```

The :visited pseudo-class

```
<style type="text/css">
a:visited {color: #006600}
</style>
<a href="/html/index.htm">Click this link</a>
```

When the link is clicked, it will change its color to green.

The :hover pseudo-class

```
<style type="text/css">
a:hover {color: #FFCC00}
</style>
<a href="/html/index.htm">Bring Mouse Here</a>
```

When the mouse is moved over this link and it changes its color to yellow.

The :active pseudo-class

```
<style type="text/css">
a:active {color: #FF00CC}
</style>
<a href="/html/index.htm">Click This Link</a>
```

When this link is clicked the color will be changed to pink.

Inheritance in CSS

In CSS, inheritance controls what happens when no value is specified for a property on an element. The inherit keyword allows authors to explicitly specify inheritance. It works on both inherited and non-inherited properties.

Inherited property: When no value for an inherited property has been specified on an element, the element gets the computed value of that property on its parent element. Only the root element of the document gets the initial value given in the property's summary.

Example: color

Non inherited property: When no value for a non-inherited property has been specified on an element, the element gets the initial value of that property

Example: border

The following are the property values:

- **Inherit:** Sets the property value applied to a selected element to be the same as that of its parent element.
- **Initial:** Sets the property value applied to a selected element to be the same as the value set for that element in the browser's default style sheet. If no value is set by the browser's default style sheet and the property is naturally inherited, then the property value is set to inherit instead.

- **Unset:** Resets the property to its natural value, which means that if the property is naturally inherited it acts like inherit, otherwise it acts like initial.
- **Revert:** Reverts the property to the value it would have had if the current origin had not applied any styles to it. In other words, the property's value is set to the user stylesheet's value for the property (if one is set), otherwise, the property's value is taken from the user agent's default stylesheet.

<pre>Default link color <li class="my-class-1">Inherit the link color <li class="my-class-2">Reset the link color <li class="my-class-3">Unset the link color body { color: green;} .my-class-1 a { color: inherit;} .my-class-2 a { color: initial;} .my-class-3 a { color: unset; }</pre>	<p>Result:</p> <ul style="list-style-type: none">• Default link color• Inherit the link color• Reset the link color• Unset the link color
---	--

- Set the color of the <body> to green.
- As the color property is naturally inherited, all child elements of body will have the same green color. It's worth noting that browsers set the color of links to blue by default instead of allowing the natural inheritance of the color property, so the first link in our list is blue.
- The second rule sets links within an element with the class my-class-1 to inherit its color from its parent. In this case, it means that the link inherits its color from its parent, which, by default inherits its color from its own parent, which ultimately inherits its color from the <body> element, which had its color set to green by the first rule.
- The third rule selects any links within an element with the class my-class-2 and sets their color to initial. Usually, the initial value set by browsers for the text color is black, so this link is set to black.
- The last rule selects all links within an element with the class my-class-3 and sets their color to unset — we unset the value. Because the color property is a naturally inherited property it acts exactly like setting the value to inherit. As a consequence, this link is set to the same color as the body — green.


CSS Background:

Defines how the background image will behave when scrolling the page. The background image will scroll with the page. It will also position and resize itself according to

the element it's applied to. The background property is specified as one or more background layers, separated by commas. Each layer may include zero or one occurrences of any of the following values: <attachment>, <bg-image>, <position>, <bg-size> and <repeat-style>


The <bg-size> value may only be included immediately after <position>, separated with the '/' character, like this: "center/80%".

The <box> value may be included zero, one, or two times. If included once, it sets both background-origin and background-clip. If it is included twice, the first occurrence sets background-origin, and the second sets background-clip. The <background-color> value may only be included in the last layer specified.

<pre><p class="topbanner"> Starry sky
 Twinkle twinkle
 Starry sky</p> <p class="warning">Here is a paragraph<p> .warning { background: pink; } .topbanner { background: url("https://mdn.mozillademos.org/files/11983/starsolid.gif ") #99f repeat-y fixed; }</pre>	
---	---

Border:

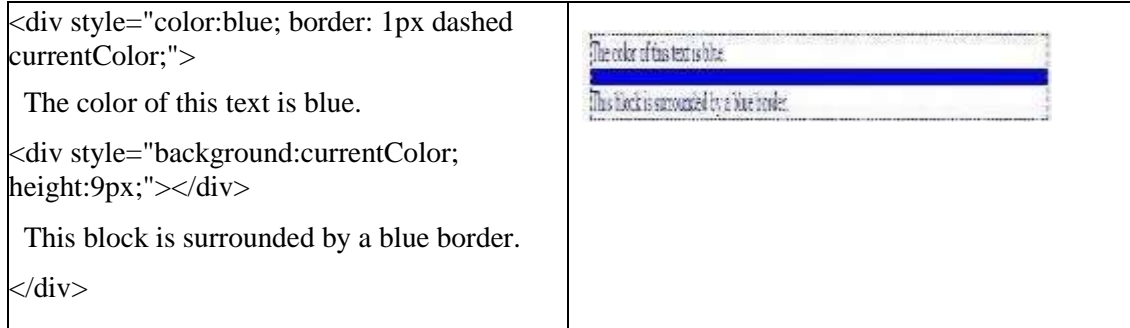
The border shorthand CSS property sets an element's border. It sets the values of border-width, border-style, and border-color.

<pre><div>I have a border, an outline, AND a box shadow! Amazing, isn't it?</div> div { border: 0.5rem outset pink; outline: 0.5rem solid khaki; box-shadow: 0 0 0 2rem skyblue; border-radius: 12px; font: bold 1rem sans-serif; margin: 2rem; padding: 1rem; outline-offset: 0.5rem; }</pre>	
---	---

CSS Colors:

CSS Color is a CSS module that deals with colors, color types, color blending, opacity, and how you can apply these colors and effects to HTML content. Not all CSS properties that take a <color> as a value are part of this module, but they do depend upon it. The <color> CSS data type represents a color in the RGB color space. A <color> may also include an alpha-channel transparency value, indicating how the color should composite with

its background. A `<color>` can be defined in any of the following ways: Using a keyword, Using the RGB cubic-coordinate system and Using the HSL cylindrical-coordinate system



CSS Shadows:

CSS can add shadow to text and to elements.

CSS Text Shadow: The CSS text-shadow property applies shadow to text.

```
h1 { text-shadow: 2px 2px;}
```

To add more than one shadow to the text, add a comma-separated list of shadows.

```
h1 { text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;}
```

Box Shadow: The CSS box-shadow property applies shadow to elements.

```
div { box-shadow: 10px 10px grey;}
```

CSS Text

Text Color: The color property is used to set the color of the text. The color is specified by: a color name - like "red", a HEX value - like "#ff0000" or an RGB value - like "rgb(255,0,0)"

- direction-Specifies the text direction/writing direction
- letter-spacing- Increases or decreases the space between characters in a text
- line-height-Sets the line height
- text-align-Specifies the horizontal alignment of text
- text-decoration-Specifies the decoration added to text
- text-indent-Specifies the indentation of the first line in a text-block
- text-shadow- Specifies the shadow effect added to text

- text-transform -Controls the capitalization of text
- text-overflow- Specifies how overflowed content that is not displayed should be signaled to the user
- vertical-align- Sets the vertical alignment of an element
- white-space- Specifies how white-space inside an element is handled
- word-spacing- Increases or decreases the space between words in a text

```
html { font-size: 10px;}
h1 { font-size: 2.6rem; text-transform: capitalize;}
h1 + p { font-weight: bold;}
p { font-size: 1.4rem; color: red; font-family: Helvetica, Arial, sans-serif;}
```

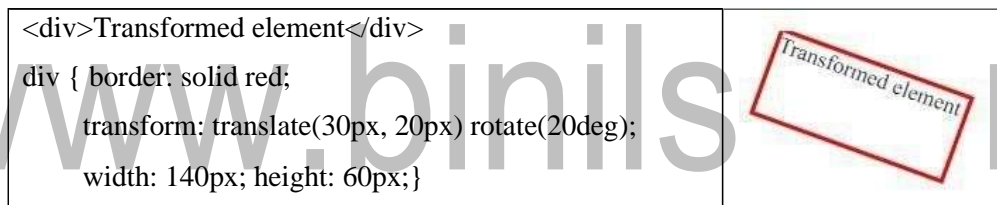
CSS Transformation

The transform CSS property allows to rotate, scale, skew, or translate an element. It modifies the coordinate space of the CSS visual formatting model. Only transformable elements can be transformed. That is, all elements whose layout is governed by the CSS box model except for: non-replaced inline boxes, table-column boxes, and table-column-group boxes.

Properties:

- none -Defines that there should be no transformation
- matrix(n,n,n,n,n,n)-Defines a 2D transformation, using a matrix of six values
- matrix3d(n,n,n,n,n,n,n,n,n,n,n,n,n,n,n,n)-Defines a 3D transformation, using a 4x4 matrix of 16 values
- translate(x,y)-Defines a 2D translation
- translate3d(x,y,z)-Defines a 3D translation
- translateX(x)- Defines a translation, using only the value for the X-axis
- translateY(y)- Defines a translation, using only the value for the Y-axis
- translateZ(z)- Defines a 3D translation, using only the value for the Z-axis
- scale(x,y)- Defines a 2D scale transformation
- scale3d(x,y,z)-Defines a 3D scale transformation
- scaleX(x)-Defines a scale transformation by giving a value for the X-axis

- scaleY(y)-Defines a scale transformation by giving a value for the Y-axis
- scaleZ(z)-Defines a 3D scale transformation by giving a value for the Z-axis
- rotate(angle)-Defines a 2D rotation, the angle is specified in the parameter
- rotate3d(x,y,z,angle)-Defines a 3D rotation
- rotateX(angle)-Defines a 3D rotation along the X-axis
- rotateY(angle)-Defines a 3D rotation along the Y-axis
- rotateZ(angle)-Defines a 3D rotation along the Z-axis
- skew(x-angle,y-angle)-Defines a 2D skew transformation along the X- and the Y-axis
- skewX(angle) Defines a 2D skew transformation along the X-axis
- skewY(angle) Defines a 2D skew transformation along the Y-axis
- perspective(n) Defines a perspective view for a 3D transformed element
- initial Sets this property to its default value.
- inherit Inherits this property from its parent element.



CSS Transitions:

CSS transitions provide a way to control animation speed when changing CSS properties. Instead of having property changes take effect immediately, the transition cause the changes in a property to take place over a period of time. To create a transition effect, you must specify two things:

1. the CSS property you want to add an effect to
2. the duration of the effect. If the duration part is not specified, the transition will have no effect, because the default value is 0.

transition-property: Specifies the name or names of the CSS properties to which transitions should be applied. Only properties listed here are animated during transitions; changes to all other properties occur instantaneously as usual.

transition-duration: Specifies the duration over which transitions should occur. You can specify a single duration that applies to all properties during the transition, or multiple values to allow each property to transition over a different period of time.

Example:transition-duration: 1s

transition-timing-function: Specifies a function to define how intermediate values for properties are computed. Timing functions determine how intermediate values of the transition are calculated. Most timing functions can be specified by providing the graph of the corresponding function, as defined by four points defining a cubic bezier. Some of the functions are:

- ease - specifies a transition effect with a slow start, then fast, then end slowly (this is default)
- linear - specifies a transition effect with the same speed from start to end
- ease-in - specifies a transition effect with a slow start
- ease-out - specifies a transition effect with a slow end
- ease-in-out - specifies a transition effect with a slow start and end
- cubic-bezier(n,n,n,n) - lets to define our own values in a cubic-bezier function

transition-delay: Defines how long to wait between the time a property is changed and the transition actually begins.

CSS Animations

CSS animations make it possible to animate transitions from one CSS style configuration to another. Animations consist of two components, a style describing the CSS animation and a set of keyframes that indicate the start and end states of the animation's style, as well as possible intermediate waypoints.

Advantages of animations over transitions:

- They're easy to use for simple animations; you can create them without even having to know JavaScript.
- The animations run well, even under moderate system load.
- Letting the browser control the animation sequence lets the browser optimize performance and efficiency.

Keyframes are used to create animations. The @keyframes CSS at-rule controls the intermediate steps in a CSS animation sequence by defining styles for keyframes along the animation sequence. This gives more control over the intermediate steps of the animation sequence than transitions. The sub-properties of the animation property are:

- animation-name- Specifies the name of the @keyframes at-rule describing the animation's keyframes.

- animation-duration- Configures the length of time that an animation should take to complete one cycle.
- animation-timing-function- Configures the timing of the animation; that is, how the animation transitions through keyframes, by establishing acceleration curves.
 - animation-delay- Configures the delay between the time the element is loaded and the beginning of the animation sequence.
- animation-iteration-count- Configures the number of times the animation should repeat.
- animation-direction- Configures whether or not the animation should alternate direction on each run through the sequence or reset to the start point and repeat itself.
- animation-fill-mode- Configures what values are applied by the animation before and after it is executing.
- animation-play-state- Lets to pause and resume the animation sequence.
- ```
p { animation-duration: 3s; animation-name: slidein; } @keyframes slidein {
 from { margin-left: 100%; width: 300%; } to {
 margin-left: 0%; width: 100%; }
```

- In this example the style for the <p> element specifies that the animation should take 3 seconds to execute from start to finish, using the animation-duration property, and that the name of the @keyframes at-rule defining the keyframes for the animation sequence is named “slidein”.

## 1.4 HTML

HTML stands for Hyper Text Markup Language. It allows us to organize text, graphics, audio, and video on a web page.

*HTML is a formatting language used to define the appearance and contents of a web page.*

- The word **Hypertext** refers to the text which acts as a link. The word **markup** refers to the symbols that are used to define structure of the text. The markup symbols tells the browser how to display the text and are often called tags. The word **Language** refers to the syntax that is similar to any other language.
- HTML was created by Tim Berners-Lee at CERN.

### 1.2.1 HTML Tags

*Tag is a command that tells the web browser how to display the text, audio, graphics or video on a web page.*

Tags are indicated with pair of angle brackets. They start with a less than (<) character and end with a greater than (>) character. The tag name is specified between the angle brackets. Most of the tags usually occur in pair: the start tag and the closing tag. The start tag is simply the tag name is enclosed in angle bracket whereas the closing tag is specified including

a forward slash (/).Some tags are the empty i.e. they don't have the closing tag.Tags are not case sensitive.

The starting and closing tag name must be the same. For example `<b> hello </i>` is invalid as both are different.If the angle brackets are not specified (`<>`) for a tag, the browser will treat the tag name as a simple text.The tag can also have attributes to provide additional information about the tag to the browser.

➤ **Basic Tags**

| Tag                                                | Description                                                    |
|----------------------------------------------------|----------------------------------------------------------------|
| <code>&lt;html&gt;&lt;/html&gt;</code>             | Specifies the document as a web page                           |
| <code>&lt;head&gt;&lt;/head&gt;</code>             | Specifies the descriptive information about the web documents. |
| <code>&lt;title&gt;&lt;/title&gt;</code>           | Specifies the title of the web page.                           |
| <code>&lt;body&gt;&lt;/body&gt;</code>             | Specifies the body of a web document.                          |
| <code>&lt;!DOCTYPE&gt;</code>                      | Defines the document type                                      |
| <code>&lt;h1&gt;</code> to <code>&lt;h6&gt;</code> | Defines HTML headings                                          |
| <code>&lt;p&gt;</code>                             | Defines a paragraph                                            |
| <code>&lt;br&gt;</code>                            | Inserts a single line break                                    |
| <code>&lt;hr&gt;</code>                            | Defines a thematic change in the content                       |
| <code>&lt;!--...--&gt;</code>                      | Defines a comment                                              |

**Basic HTML tags**

|                                                                                                                                                                     |                                                           |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|
| <pre>&lt;!DOCTYPE html&gt; &lt;html&gt; &lt;body&gt; &lt;h1&gt;My First Heading&lt;/h1&gt; &lt;p&gt;My first paragraph.&lt;/p&gt; &lt;/body&gt; &lt;/html&gt;</pre> | Output:<br><b>My First Heading</b><br>My first paragraph. |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|

➤ **Formatting Tag**

| Tag               | Description                                                                                         |
|-------------------|-----------------------------------------------------------------------------------------------------|
| <b></b>           | Specifies the text as bold.                                                                         |
| <em></em>         | It is a phrase text. It specifies the emphasized text                                               |
| <strong></strong> | It is a phrase tag. It specifies an important text                                                  |
| <i></i>           | The content of italic tag is displayed in italic.                                                   |
| <sub></sub>       | Specifies the subscripted text                                                                      |
| <sup></sup>       | Defines the superscripted text.                                                                     |
| <ins></ins>       | Specifies the inserted text                                                                         |
| <del></del>       | Specifies the deleted text.                                                                         |
| <mark></mark>     | Specifies the marked text.                                                                          |
| <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                            |
| <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                                                |
| <center>          | Not supported in HTML5. Use CSS instead. Defines centered text                                      |
| <cite>            | Defines the title of a work                                                                         |
| <code>            | Defines a piece of computer code                                                                    |
| <dfn>             | Represents the defining instance of a term                                                          |
| <font>            | Not supported in HTML5. Use CSS instead. Defines font, color, and size for text                     |
| <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> instead. Defines strikethrough text          |
| <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                                                  |

### Formatting tags

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

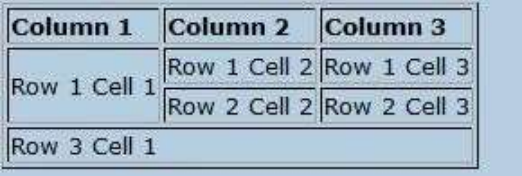


|                                                            |      |
|------------------------------------------------------------|------|
| <code>Text&lt;/code&gt;&lt;/p&gt;<br/>&lt;/html&gt;</code> | Text |
|------------------------------------------------------------|------|

➤ **Table Tags**

| Tag                                            | Description                                                                                |
|------------------------------------------------|--------------------------------------------------------------------------------------------|
| <code>&lt;table&gt;&lt;/table&gt;</code>       | Specifies a table.                                                                         |
| <code>&lt;tr&gt;&lt;/tr&gt;</code>             | Specifies a row in the table.                                                              |
| <code>&lt;th&gt;&lt;/th&gt;</code>             | Specifies header cell in the table.                                                        |
| <code>&lt;td&gt;&lt;/td&gt;</code>             | Specifies the data in an cell of the table.                                                |
| <code>&lt;caption&gt;&lt;/caption&gt;</code>   | Specifies the table caption.                                                               |
| <code>&lt;colgroup&gt;&lt;/colgroup&gt;</code> | Specifies a group of columns in a table for formatting.                                    |
| <code>&lt;thead&gt;</code>                     | Groups the header content in a table                                                       |
| <code>&lt;tbody&gt;</code>                     | Groups the body content in a table                                                         |
| <code>&lt;tfoot&gt;</code>                     | Groups the footer content in a table                                                       |
| <code>&lt;col&gt;</code>                       | Specifies column properties for each column within a <code>&lt;colgroup&gt;</code> element |

**Table tag**

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                                                                      |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| <pre>&lt;html&gt;&lt;table border="1"&gt;&lt;tr&gt; &lt;td&gt;&lt;b&gt;Column 1&lt;/b&gt;&lt;/td&gt; &lt;td&gt;&lt;b&gt;Column 2&lt;/b&gt;&lt;/td&gt; &lt;td&gt;&lt;b&gt;Column 3&lt;/b&gt;&lt;/td&gt;&lt;/tr&gt; &lt;tr&gt;&lt;td rowspan="2"&gt;Row 1 Cell 1&lt;/td&gt; &lt;td&gt;Row 1 Cell 2&lt;/td&gt; &lt;td&gt;Row 1 Cell 3&lt;/td&gt;&lt;/tr&gt; &lt;tr&gt;&lt;td&gt;Row 2 Cell 2&lt;/td&gt; &lt;td&gt;Row 2 Cell 3&lt;/td&gt;&lt;/tr&gt; &lt;tr&gt;&lt;td colspan="3"&gt;Row 3 Cell 1&lt;/td&gt;&lt;/tr&gt; &lt;/table&gt;&lt;/html&gt;</pre> |  |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|

➤ **List tags**

| Tag        | Description                                                             |
|------------|-------------------------------------------------------------------------|
| <ul></ul>  | Specifies an unordered list.                                            |
| <ol></ol>  | Specifies an ordered list.                                              |
| <li></li>  | Specifies a list item.                                                  |
| <dl></dl>  | Specifies a description list.                                           |
| <dt></dt>  | Specifies the term in a description list.                               |
| <dd></dd>  | Specifies description of term in a description list.                    |
| <dir>      | Not supported in HTML5. Use <ul>instead. Defines a directory list.      |
| <menu>     | Defines a list/menu of commands                                         |
| <menuitem> | Defines a command/menu item that the user can invoke from a popup menu. |

**List tags**

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                                                                                                                                                                                                      |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> &lt;html&gt; &lt;ol type="1" value="1"&gt; &lt;li&gt;Arabic Number&lt;/li&gt; &lt;li&gt;Arabic Number&lt;/li&gt;&lt;/ol&gt; &lt;ol type="a" value="1"&gt; &lt;li&gt;Lower Alphabet&lt;/li&gt; &lt;li&gt;Lower Alphabet&lt;/li&gt;&lt;/ol&gt; &lt;ol type="A" value="1"&gt; &lt;li&gt;Upper Alphabet&lt;/li&gt; &lt;li&gt;Upper Alphabet&lt;/li&gt;&lt;/ol&gt; &lt;ol type="i" value="1"&gt; &lt;li&gt;Lower Roman numeral&lt;/li&gt; &lt;li&gt;Lower Roman numeral&lt;/li&gt;&lt;/ol&gt; &lt;ol type="I" value="1"&gt; &lt;li&gt;Upper Romannumeral&lt;/li&gt; &lt;li&gt;Upper Romannumeral&lt;/li&gt; &lt;/ol&gt;&lt;/html&gt; </pre> | <pre> 1. Arabic Number 2. Arabic Number a. Lower Alphabet b. Lower Alphabet A. Upper Alphabet B. Upper Alphabet i. Lower Roman numeral ii. Lower Roman numeral I. Upper Roman numeral II. Upper Roman numeral </pre> |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## ➤ Frames

Frames help us to divide the browser's window into multiple rectangular regions. Each region contains separate html web page and each of them work independently. A set of frames in the entire browser is known as frameset. It tells the browser how to divide browser window into frames and the web pages that each has to load.

### Advantages of Frames

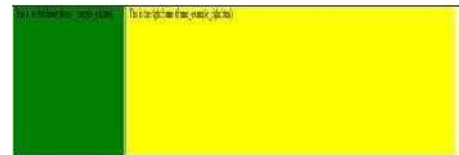
- ❖ Frame provides technical sophisticated appearance to the web site.
- ❖ It facilitates to reduce downloading time and improves the usability of the website.
- ❖ Frames generally include navigation link, header or footers, which help user to find and navigate to required information.
- ❖ It separates content of website from navigation elements, which is useful for website maintenance and content modification.

### Disadvantages of Frames

- ❖ The web developer must be track of more HTML documents linked with main frame.
- ❖ It is difficult to print the entire page, which is developed using frame.

### Frame tags

```
<html><head>
<title>Frameset page</title></head>
<frameset cols = "25%, *"><noframes>
<body>Browser doesn't support frames. Therefore,
this is the noframe version of the
site.</body></noframes>
<frame src ="/html/frame_example_left.html" />
```



Tag	Description
<code>&lt;frameset&gt;&lt;/frameset&gt;</code>	It is replacement of the <code>&lt;body&gt;</code> tag. It doesn't contain the tags that are normally used in <code>&lt;body&gt;</code> element; instead it contains the <code>&lt;frame&gt;</code> element used to add each frame.
<code>&lt;frame&gt;&lt;/frame&gt;</code>	Specifies the content of different frames in a web page.

<pre>&lt;frame src="/html/frame_example_right.html" /&gt; &lt;/frameset&gt;&lt;/html&gt;Leftframe.html&lt;html&gt; &lt;body style="background-color:green"&gt; &lt;p&gt;This is the left frame (frame_example_left.html).&lt;/p&gt;&lt;/body&gt;&lt;/html&gt; Rightframe.html &lt;html&gt;&lt;body style="background-color:yellow"&gt; &lt;p&gt;This is the right frame (frame_example_right.html).&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</pre>	
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

➤ **Forms**

Forms are used to input the values. These values are sent to the server for processing. Forms uses input elements such as text fields, check boxes, radio buttons, lists, submit buttons etc. to enter the data into it.

Tag	Description
<form></form>	It is used to create HTML form.
<input></input>	Specifies the input field.
<textarea></textarea>	Specifies a text area control that allows to enter multi-line text.
<label></label>	Specifies the label for an input element.
<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
<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.

<code>&lt;keygen&gt;</code>	Defines a key-pair generator field (for forms)
<code>&lt;output&gt;</code>	Defines the result of a calculation

### Form tags

```
<form action="/html/tags/html_form_tag_action.cfm"
method="get"><fieldset>

<legend>Your Details</legend>

<div>

<label for="first_name">First Name:</label>

<input type="text" name="first_name" value=""
maxlength="100" />
</div>

<div>

<label for="lunch">Lunch:</label>

<input type="radio" name="lunch" value="Meals" />
meals
<input type="radio" name="lunch" value="Tiffin"
/>tiffin</div>

<div>

<label for="drinks">Drinks:</label>

<input type="checkbox" name="drinks" value="fresh
juice" /> fresh juice

<input type="checkbox" name="drinks" value="tea"
/> tea</div>

<div>

<label for="city">Preferred City:</label>

<select name="city">

<option value="Coimbatore">coimbatore</option>

<option value="Chennai">chennai</option>

</select></div>

<div>
```

Your Details

First Name:

Lunch:  
 meals  tiffin

Drinks:  
 fresh juice  tea

Preferred City:  
coimbatore

Comments:

Submit

```
<label for="comments">Comments:</label>

<textarea rows="3" cols="20"
name="comments"></textarea></div>

<div>
<input type="submit" value="Submit" /></div>

</fieldset></form>
```

➤ **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

➤ **Audio /Video**

Tag	Description
<audio>	Defines sound content
<source>	Defines multiple media resources for media elements (<video> and <audio>)
<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

➤ **Styles and Semantics**

Tag	Description
<style>	Defines style information for a document
<div>	Defines a section in a document
<span>	Defines a section in a document
<header>	Defines a header for a document or section
<hgroup>	Defines a group of headings
<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

➤ **Meta information**

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 <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

## Drag and Drop

Drag and Drop is a very interactive and user-friendly concept which makes it easier to move an object to a different location by grabbing it. This allows the user to click and hold the mouse button over an element, drag it to another location, and release the mouse button to drop the element there. In HTML 5 Drag and Drop are much easier to code and any element in it is draggable.

Tag	Description
ondragstart	Calls a function, drag(event), that specifies what data to be dragged
ondragenter	To determine whether or not the drop target is to accept the drop. If the drop is to be accepted, then this event has to be canceled
ondragleave	Occurs when the mouse leaves an element before a valid drop target while the drag is occurring
ondragover	Specifies where the dragged data can be dropped
Ondrop	Specifies where the drop was occurred at the end of the drag operation
ondragend	Occurs when the user has finished dragging an element

## XHTML

XHTML stands for **Extensible Hyper Text Markup Language**. It is the next step in the evolution of the internet. The XHTML 1.0 is the first document type in the XHTML family.

XHTML is almost identical to HTML 4.01 with only few differences. XHTML was developed by W3C to help web developers make the transition from HTML to XML. By migrating to XHTML today, web developers can enter the XML world with all of its benefits.



XHTML has stricter syntax rules in comparison to HTML. XHTML gives you a more consistent, well-structured format so that the webpages can be easily parsed and processed by present and future web browsers.

### Advantages of XHTML

- ❖ XHTML documents are XML conforming as they are readily viewed, edited, and validated with standard XML tools.
- ❖ XHTML documents can be written to operate better than they did before in existing browsers as well as in new browsers.
- ❖ XHTML documents can utilize applications such as scripts and applets that rely upon either the HTML Document Object Model or the XML Document Object Model.

### Migration from HTML to XHTML

#### ➤ DOCTYPE Declaration

All XHTML documents must have a DOCTYPE declaration at the start.

#### **Example:**

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

#### ➤ Case sensitivity

XHTML is case sensitive markup language. All the XHTML tags and attributes need to be written in lower case only.

- `<!-- This is invalid in XHTML -->`
- `<A Href="/xhtml/xhtml_abc.html">XHTML </A>`
- `<!-- Correct XHTML way of writing this is as follows -->`
- `<ahref="/xhtml/xhtml_abc.html">XHTML</a>`: In the example, Href and anchor tag A are having characters which are not in lower case, so it is incorrect.

#### ➤ Closing tags

Each and every XHTML tag should have an equivalent closing tag, even empty elements should also have closing tags.

- `<!-- This is valid in XHTML -->`
- `<p>This paragraph is not written according to XHTML syntax.</p>`

- <!-- This is also valid now -->
- <imgsrc="/images/xhtml.gif" />

➤ **Attribute quotes**

All the values of XHTML attributes must be quoted. Otherwise, the XHTML document is assumed as an invalid document.

<!-- Correct XHTML way of writing this is as follows -->

<imgsrc="/images/xhtml.gif" width="250" height="50" />

➤ **Attribute minimization**

XHTML does not allow attribute minimization. It means attribute and its values must be explicitly stated.

<!-- Correct XHTML way of writing this is as follows -->

<option selected="selected">

HTML style	XHTML style	HTML style	XHTML style
compact	compact="compact"	ismap	ismap="ismap"
Checked	checked="checked"	nohref	nohref="nohref"
Declare	declare="declare"	noshade	noshade="noshade"
readonly	readonly="readonly"	nowrap	nowrap="nowrap"
disabled	disabled="disabled"	multiple	multiple="multiple"
Selected	selected="selected"	noresize	noresize="noresize"
Defer	defer="defer"		

➤ **The id attribute**

The id attribute replaces the name attribute. Instead of using name="name", XHTML prefers to use id="id".

<!-- Correct XHTML way of writing this is as follows -->

<imgsrc="/images/xhtml.gif" id="xhtml\_logo" />

➤ **The language attribute**

The language attribute of the script tag is deprecated.

---

<!-- Correct XHTML way of writing this is as follows -->

```
<script type="text/JavaScript">
```

```
document.write("Hello XHTML!");</script>
```

➤ **Nested tags**

All the XHTML tags must be nested properly otherwise the document will be assumed as an incorrect XHTML document.

<!-- Correct XHTML way of writing this is as follows -->

```
<i>This text is bold and italic</i>
```

➤ **Element prohibitions**

The following elements are not allowed to have any other element inside them. This prohibition applies to all depths of nesting, i.e. it includes all the descendant elements.

Element	Prohibition
<a>	Must not contain other <a> elements.
<pre>	Must not contain the <img>, <object>, <big>, <small>, <sub>, or <sup> elements.
<button>	Must not contain the <input>, <select>, <textarea>, <label>, <button>, <form>, <fieldset>, <iframe> or <isindex> elements.
<label>	Must not contain other <label> elements.
<form>	Must not contain other <form> elements.

**Differences between HTML and XHTML**

HTML	XHTML
HTML or Hyper Text Markup Language is the main markup language for creating web pages and other information that can be displayed in a web browser.	HTML (Extensible HyperText Markup Language) is a family of XML markup languages that mirror or extend versions of the widely used Hypertext Markup Language (HTML), the language in which web pages are written.
It has document file format.	It is a mark- up language.

## 1.2 HTTP Request and HTTP Response

### HTTP Request

HTTP request comprises of lines which contains: Request line, Header Fields and Message body. The first line i.e. the Request line specifies the request method i.e. Get or Post. The second line specifies the header which indicates the domain name of the server from where index.htm is retrieved.

### HTTP Response

Like HTTP request, HTTP response also has certain structure. HTTP response contains: Status line, Headers and Message body.

### Domain Name System (DNS)

- When DNS was not into existence, one had to download a host file containing host names and their corresponding IP address.
- But with increase in number of hosts of internet, the size of host file also increased. This resulted in increased traffic on downloading this file. To solve this problem the DNS system was introduced

*Domain Name System helps to resolve the host name to an address. It uses a hierarchical naming scheme and distributed database of IP addresses and associated names.*

### Domain Name System Architecture

The Domain name system comprises of Domain Names, Domain Name Space, Name Server that have been described below:

#### ➤ Domain Names

Domain Name is a symbolic string associated with an IP address. There are several domain names available. Some of them are generic such as com, edu, gov, net etc, while some country level domain names such as au, in, za, us etc.

#### ➤ Domain Name Space

The domain name space refers a hierarchy in the internet naming structure. This hierarchy has multiple levels (from 0 to 127), with a root at the top. Each domain can be partitioned into sub domains and these can be further partitioned and so on.

#### ➤ Name Server

Name server contains the **DNS database**. This database comprises of various names and their corresponding IP addresses. Since it is not possible for a single server to

maintain entire DNS database, therefore, the information is distributed among many DNS servers.

- Hierarchy of server is same as hierarchy of names.
- The entire name space is divided into the zones

### Zones

Zone is collection of nodes (sub domains) under the main domain. The server maintains a database called zone file for every zone. If the domain is not further divided into sub domains then domain and zone refers to the same thing. The information about the nodes in the sub domain is stored in the servers at the lower levels however; the original server keeps reference to these lower levels of servers.

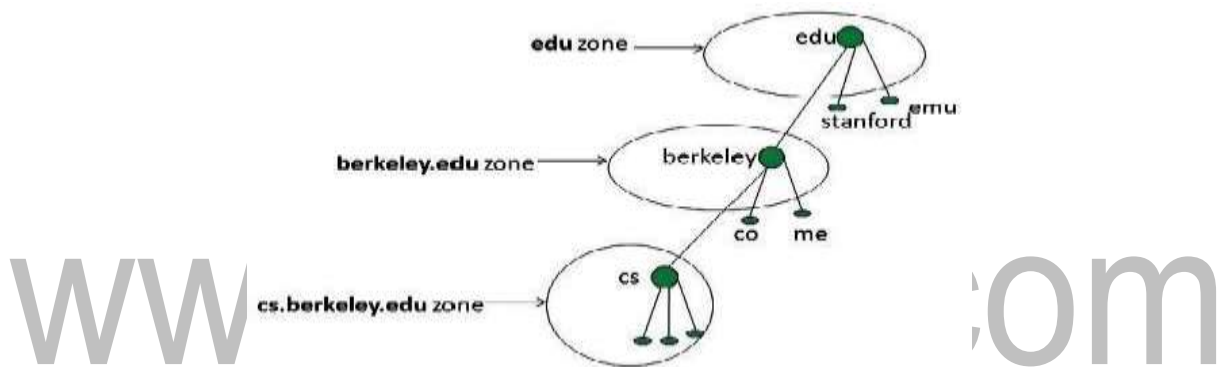


Fig 1.6 Zones in DNS

### ➤ Types of Name Servers

Following are the three categories of Name Servers that manages the entire Domain Name System:

- **Root Server:** Root Server is the top level server which consists of the entire DNS tree. It does not contain the information about domains but delegates the authority to the other server.
- **Primary Server:** Primary Server stores a file about its **zone**. It has authority to create, maintain, and update the zone file.
- **Secondary Server:** Secondary Server transfers complete information about a zone from another server which may be primary or secondary server. The secondary server does not have authority to create or update a zone file.

### ➤ DNS Working

DNS translates the domain name into IP address automatically. Following are the steps in domain resolution process:

1. When we type www.abc.com into the browser, it asks the local DNS Server for its IP address. Here the local DNS is at ISP end.
2. When the local DNS does not find the IP address of requested domain name, it forwards the request to the root DNS server and again enquires about IP address of it.
3. The root DNS server replies with delegation that 'I do not know the IP address of www.abc.com but know the IP address of DNS Server'.
4. The local DNS server then asks the com DNS Server the same question.
5. The com DNS Server replies the same that it does not know the IP address of www.abc.com but knows the address of server that contains www.abc.com.
6. Then the local DNS asks the abc.com DNS server the same question.
7. Then abc.com DNS server replies with IP address of www.abc.com.
8. Now, the local DNS sends the IP address of www.abc.com to the computer that sends the request.
9. When a DNS server receives a DNS reply it cache the information in the reply in its local memory. This is called **DNScache**.

### **HTTP Request Message**

Whenever a URL is entered in the address box of the browser, the browser translates the URL into a request message according to the HTTP protocol; and sends the request message to the server. When the request message reaches the server, the server can take either one of these actions:

- The server interprets the request received, maps the request into a file under the server's document directory, and returns the file requested to the client.
- The server interprets the request received, maps the request into a program kept in the server, executes the program, and returns the output of the program to the client.
- The request cannot be satisfied, the server returns an error message.

The following are the parts of the HTTP request message:

**Request-Line or Start line:** This begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF (Carriage Return and Line Feed). The elements are separated by space.

---

**Request Method:** This indicates the method to be performed on the resource identified by the given Request-URI. The method is case-sensitive and should always be mentioned in uppercase. The following are the methods:

- **GET** - return the resource specified by the Request-URI as the body of a response message.
- **POST**- pass the body of this request message on as data to be processed by the resource specified by the Request-URI.
- **HEAD**- return the same HTTP header fields that would be returned if a GET method were used, but not return the message body that would be returned to a GET (this provides information about a resource without the communication overhead of transmitting the body of the response, which may be quite large).
- **OPTIONS**- return (in Allow header field) a list of HTTP methods that may be used to access the resource specified by the Request-URI.
- **PUT**- store the body of this message on the server and assign the specified Request-URI to the data stored so that future GET request messages containing this Request-URI will receive this data in their response messages.
- **DELETE**- respond to future HTTP request messages that contain the specified Request-URI with a response indicating that there is no resource associated with this Request-URI.
- **TRACE**- return a copy of the complete HTTP request message, including start line, header fields, and body, received by the server. Used primarily for test purposes.

### Header Fields

The request-header fields allow the client to pass additional information about the request, and about the client itself, to the server. These fields act as request modifiers. Here is a list of some important Request-header fields that can be used based on the requirement: Accept-Charset, Accept-Encoding, Accept-Language, Authorization, Expect, From, Host, If-Match, If-Modified-Since, If-None-Match, If-Range, If-Unmodified-Since, Max-Forwards, Proxy-Authorization, Range, Referer, TE, User-Agent.

### Multimedia Internet Mail Extension (MIME) type

MIME are standards that are used to pass variety of types of information through internet message protocol. It has two-parts specified as the content type of the message.

**Examples:** text/html and image/jpeg. The following are the content types supported by the HTTP:

- **Application**- Data that does not fit within another content type and that is intended to be processed by application software, or that is itself an executable binary.
- **Audio**- Audio data. Subtype defines audio format.

- Image- Image data, typically static. Subtype defines image format. Requires appropriate software and hardware in order to be displayed.
- Message- Another document that represents a MIME-style message.
- Video- Animated images, possibly with synchronized sound.
- Model- Structured data, generally numeric, representing physical or behavioral models.
- Multipart- Multiple entities, each with its own header and body.
- Text- Displayable as text. That is, a human can read this document without the need for special software, although it may be easier to read with the assistance of other software.

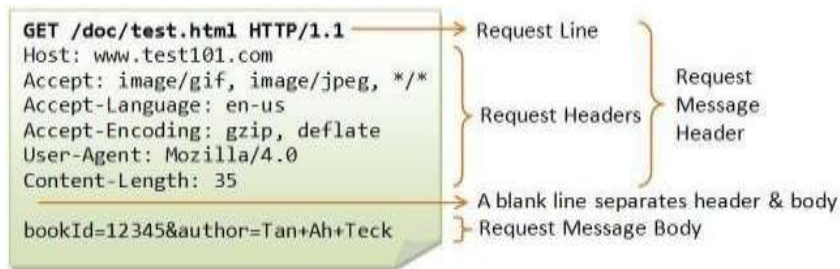


Fig 1.7 HTTP Request Message

### HTTP Response Message

The following are contents of HTTP Response message:

- **Status line:** It has three fields namely HTTP version, numeric status code and text string which informs about the information represented by numeric status code.

Status code	Recommended reason phrase	Usual meaning
200	OK	Request processed normally.
301	Moved Permanently	URI for the requested resource has been changed. All future requests should be made to URI contained in the location header field of the response. Most browsers will automatically send a second request to the new URI and display the second response.
307	Temporary redirect	URI for the request has been changed at least temporarily. This request should be fulfilled by making a second request to the URI contained in the location header field of the response.



401	Unauthorized	The resource is password protected, and the user has not yet supplied a valid password.
403	Forbidden	The resource is present on the server but is read protected.
404	Not found	No resource corresponding to the given request-URI was found at this server.
500	Internal server error	Server software detected an internal failure.

- Header field(s) (one or more): It contains Connection, ContentType, and Content-Length, are also valid in response messages. The Content-Type of a response can be any one of the MIME type values specified by the Accept header field of the corresponding request. Some of the common response header fields are:
  - Host- Specify authority portion of URL
  - User-Agent -A string identifying the browser or other software that is sending the request.
  - Accept MIME- types of documents that are acceptable as the body of the response, possibly with indication of preference ranking.
  - Accept Language- Specifies preferred language(s) for the response body. A server may have several translations of a document, and among these should return the one that has the highest preference rating in this header field.
  - Accept-Encoding- Specifies preferred encoding(s) for the response body.
  - Accept-Charset- Allows the client to express preferences to a server that can return a document using various character sets.
  - Connection- Indicates whether or not the client would like the TCP connection kept open after the response is sent.
  - Keep-Alive- Number of seconds TCP connection should be kept open.
  - Content-Type -The MIME type of the document contained in the message body, if one is present.
  - Content-Length- Number of bytes of data in the message body, if one is present.
  - Referer -The URI of the resource from which the browser obtained the Request-URI value for this HTTP request.
- Blank line
- Message body (optional)

**Cache control:** Web browsers automatically cache on the client machine many of the resources that they request from servers via HTTP. But information in a cache can become invalid. One approach to guaranteeing that a cached copy of a resource is valid is for the client to ask the server whether or not the client's copy is valid. This can be done with relatively little communication by sending an HTTP request for the resource using the HEAD method, which returns only the status line and header portion of the response.

**Character Sets:** A character set defines the mapping between these integers, or code points, and characters. Each US-ASCII character can be represented by a 7-bit integer, which is convenient in part because the messages transmitted by the Internet Protocol are viewed as streams of 8-bit bytes, and therefore each character can be represented by a single byte. The Unicode Standard's Basic Multilingual Plane (BMP), uses characters in every modern language, uses 16-bit character codes, and the full character code space of the Unicode Standard extends to 21-bit integers.

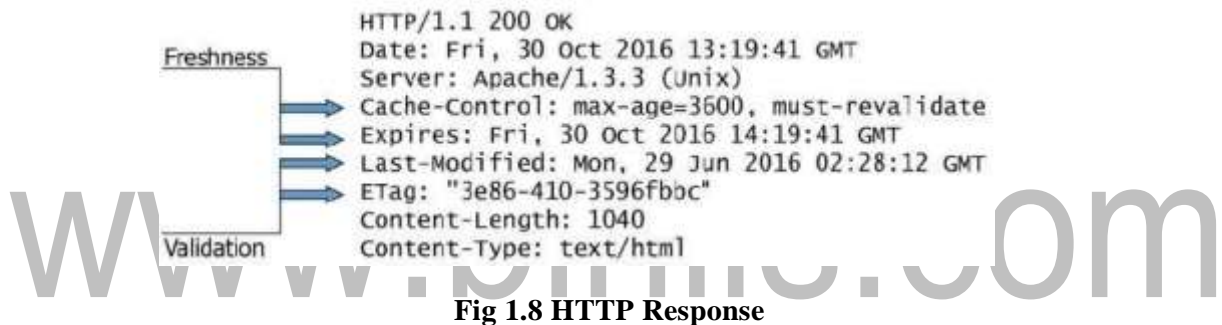


Fig 1.8 HTTP Response

## 1.3 Web Clients and Web Servers

### Web Clients

**A web client is software that accesses a web server by sending an HTTP request message and processing the resulting HTTP response.**

A web client is an application that communicates with a web server, using Hypertext Transfer Protocol (HTTP). Any web client that is designed to directly support user access to web servers is known as a **user agent**. Browser is the most commonly used web client. The most common interface to the World Wide Web is a browser, such as Mosaic, Netscape Navigator, or Internet Explorer.

#### **Functions of browser:**

The primary function of a web browser is to render HTML, the code used to design or mark up webpages. Each time a browser loads a web page, it processes the HTML, which may include text, links, and references to images and other items, such as cascading style sheets and JavaScript functions. The browser processes these items, then renders them in the browser window.

### Parts of browser:

Each webpage has an address. This is indicated in the Address Bar. Most webpages will have titles. The actual webpage itself is displayed in the Main Webpage Window. You can reload a page by hitting the "Refresh" button. This will update it to the latest version. If a webpage takes a long time to load you can stop loading it by hitting the "Stop" button.



Fig 1.9 Browser Window

### Uniform Resource Locators (URL)

The URI (Uniform Resource Identifier) is a string that associates a particular resource on the web. There are two types of URI:

#### ➤ URN (Uniform Resource Name)

This identifies the resource using unique names. They do not signify the location of the resource. It consists of three parts: Scheme name, Namespace identifier and Namespace string

**Syntax:**Urn: name: resource name

**Example:** URN: ISBN: 5427877

#### ➤ URL (Uniform Resource Locator )

*Uniform Resource Locator (URL) refers to a web address which uniquely identifies a document over the internet.*

- 
- This document can be a web page, image, audio, video or anything else present on the web.

## URL Types

There are two forms of URL as listed below:

- **Absolute URL:**

Absolute URL is a complete address of a resource on the web. This completed address comprises of protocol used, server name, path name and file name.

**Example:** `http:// www.abc.com / xyz /index.htm`.

Here http is the protocol, abc.com is the server name and index.htm is the file name.

The protocol part tells the web browser how to handle the file. Other protocols also that can be used to create URL are: FTP, https, Gophe, mailto, news

- **Relative URL**

Relative URL is a partial address of a webpage. Unlike absolute URL, the protocol and server part are omitted from relative URL. Relative URLs are used for internal links i.e. to create links to file that are part of same website as the WebPages on which you are placing the link.

## Differences between Absolute and Relative URL

Absolute URL	Relative URL
Used to link web pages on different websites	Used to link web pages within the same website.
Difficult to manage.	Easy to Manage.
Changes when the server name or directory name changes.	Remains same even if we change the server name or directory name.
Take time to access	Comparatively faster to access.

## Web Servers

*The computer that supplies files or services to the requesting computer over the internet is called as a web server.*

The request to the web pages is sent by the browser to the server. The server will transfer the requested page to the computer over the internet.

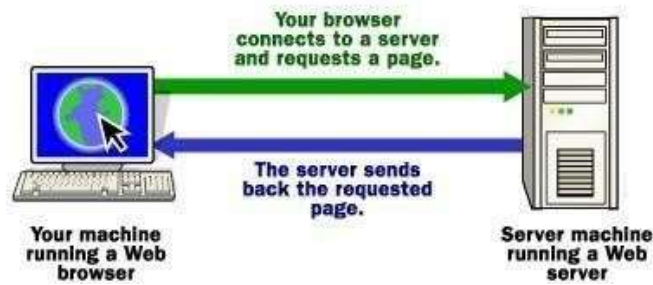


Fig 1.10 Web server

### Working of web servers

The browser broke the URL into three parts: protocol ("http"), server name ("www.abc.com") and file name ("web-server.htm"). The browser communicated with a name server to translate the server name "www.abc.com" into an IP Address, which it uses to connect to the server machine.

Following the HTTP protocol, the browser sends a GET request to the server, asking for the file <http://www.abc.com/web-server.htm>. The server machine transfers the HTML content to the browser. The browser reads the HTML tags and formats the page onto the screen. In general, all of the machines on the Internet can be categorized as two types: **servers and clients**. The machines that provide services (like Web servers or FTP servers) to other machines are **servers**. The machines that are used to connect to those services are **clients**. It is possible and common for a machine to be both a server and a client.

A server machine may provide one or more services on the internet. For example, a server machine might have software running on it that allows it to act as a Web server, an e-mail server and an FTP server. Clients that connect to a server machine do so with a specific intent, so clients direct their requests to specific software running on the overall server machine. The web server contains **log** records that store information about server activity.

**Access log file** contains information about every HTTP requests processes by the server. **Message log file** contains a variety of debugging and other information generated by the web applications and the web server. The standard input, output and error streams are also logged.

### Differences between web sites and web servers

Web Sites	Web Servers
A website is a set of linked documents associated with a particular person, organization or topic that is held on a computer system and can be accessed as part of the world wide web.	The web server is a computer program, which delivers content, such as websites or web pages. It responds to the request for web pages.

All the web sites reside on the web server.	The web server is a computer with high configuration.
Web sites can contain text files, images, videos and audios.	Web servers are hardware or software unit.

### Features of web servers:

1. The server calls on TCP software and waits for connection requests to one or more ports.
2. When a connection request is received, the server dedicates a subtask to handling this connection.
3. The subtask establishes the TCP connection and receives an HTTP request.
4. The subtask examines the Host header field of the request to determine which virtual host should receive this request and invokes software for this host.
5. The virtual host software maps the Request-URI field of the HTTP request start line to a resource on the server.
6. If the resource is a file, the host software determines the MIME type of the file and creates an HTTP response that contains the file in the body of the response message.
7. If the resource is a program, the host software runs the program, providing it with information from the request and returning the output from the program as the body of an HTTP response message.
8. The server normally logs information about the request and response—such as the IP address of the requester and the status code of the response—in a plain-text file.
9. If the TCP connection is kept alive, the server subtask continues to monitor the connection until a certain length of time has elapsed, the client sends another request, or the client initiates a connection close.

### Configuring a server

The following are to be given higher importance while configuring a server:

- IP addresses and TCP ports that may be used to connect to this server.
- Number of subtasks that will be created when the server is initialized.
- Maximum number of threads that will be allowed to exist simultaneously
- Maximum number of TCP connection requests that will be queued if the server is already running its maximum number of threads.
- Length of time the server will wait after serving an HTTP request over a TCP connection before closing the connection if another request is not received.

## Logging

Web log file is log file automatically created and maintained by a web server. Every hit to the Web site, including each view of a HTML document, image or other object, is logged. The raw web log file format is essentially one line of text for each hit to the web site. This contains information about who was visiting the site, where they came from, and exactly what they were doing on the web site. The following are the key fields:

- Directory- Directory where log file will be written
- Pattern Information- to be written to the log
- Prefix- String that will be used to begin log filename
- Resolve Hosts -Whether IP addresses (False value) or host names (True value) should be written to the log file
- Rotatable -Whether or not date should be added to file name and file should be automatically rotated each day
- Suffix- String that will be used to end log

www.binils.com