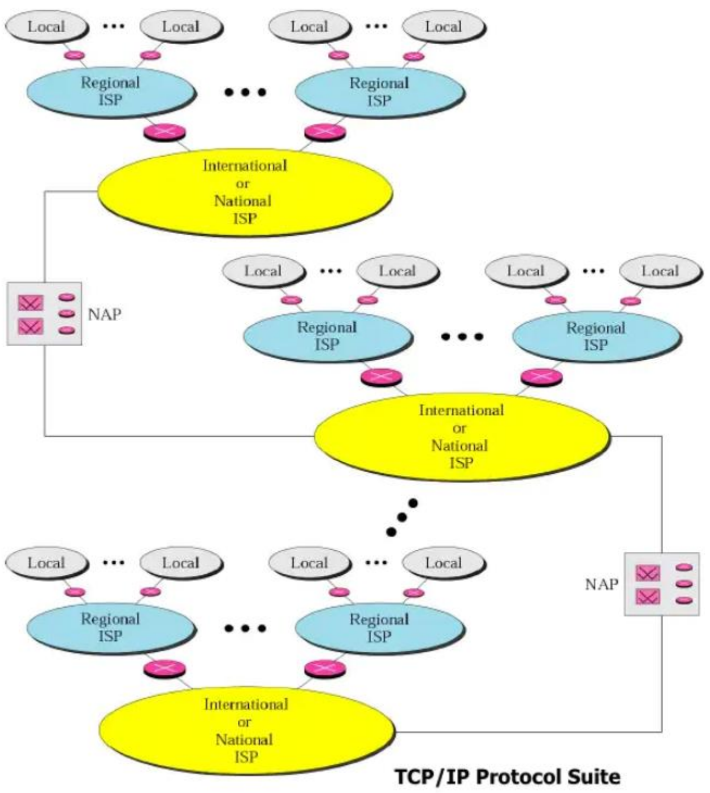
Internet Standards

- A network is a group of connected, communicating devices.
- An internet is two or more networks that can communicate with each other.
- The most notable internet is called the ***Internet***.
- Millions of people are users, internet only came into being in 1969
- ***ARPANET*** (Advanced Research Projects Agency Network), the network that ultimately evolved into what we now know as the Internet.
- January 1, 1983, a new communications protocol called Transfer Control Protocol/Internetwork Protocol (***TCP/IP***) was established.

Internet today

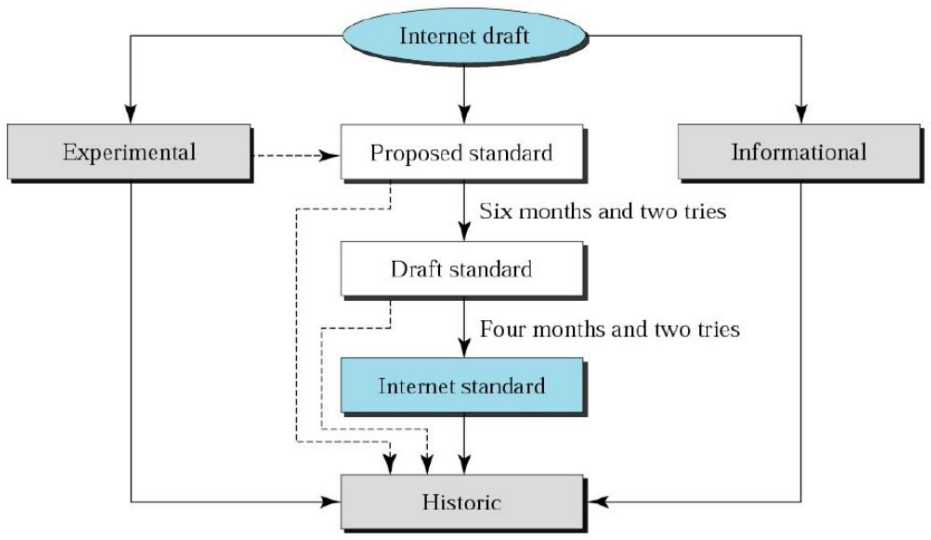


TCP/IP Protocol Suite

- Internet Standards are developed through the cooperation of standards creation committees, forums, and government regulatory agencies.
- An Internet standard is a thoroughly tested specification.
- There is a strict procedure by which a specification attains Internet standard status.
- A specification begins as an Internet draft, working document with no official status and a six-month lifetime. A draft may be published as a Request for Comment (RFC)
- There are four streams of RFCs:
  - (1) IETF – Internet Engineering Task Force
  - (2) IRTF – Internet Research Task Force
  - (3) IAB – Internet Architecture Board under ISOC
  - (4)Independent submission

Maturity Levels

Maturity levels of an RFC



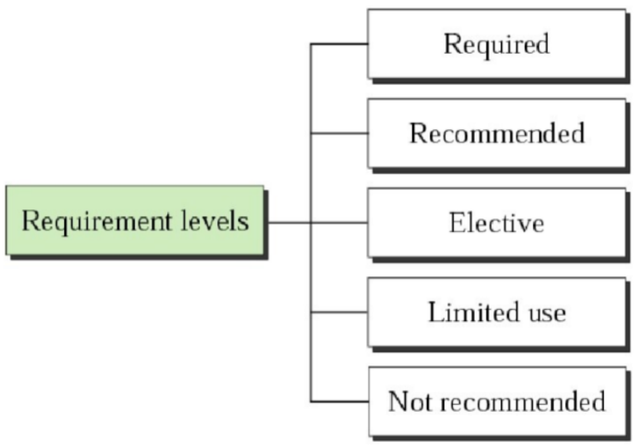
An RFC, during its lifetime, falls into one of six maturity levels

- **Proposed Standard**. A proposed standard is a specification that is stable, well understood, and of sufficient interest to the internet community.
- **Draft Standard.** A proposed standard is elevated to draft standard status after atleast two successful independent and interoperable implementations.
- **Internet Standard.** A draft standard reaches Internet standard after demonstrations of successful implementation.
- **Historic**. They either have been superseded by later specifications or have never passed the necessary maturity levels to become an internet standard.
- **Experimental.** Experimental situation must not be implemented in any functional Internet service.
- **Informational.** An RFC classified as informational contains general, historical, or tutorial information related to the Internet.

Requirements of an RFC

- **Required**. Must be implemented by all Internet systems. Ex: IP and ICMP are required protocols.
- **Recommended**. Recommended because of its usefulness. Ex: FTP and TELNET
- **Elective**. Not required and not recommended. However, a system can use it for its own benefit.
- **Limited Use**. Should be used only in limited situations.
- **Not recommended.** Is inappropriate for general use.

Requirement levels of an RFC

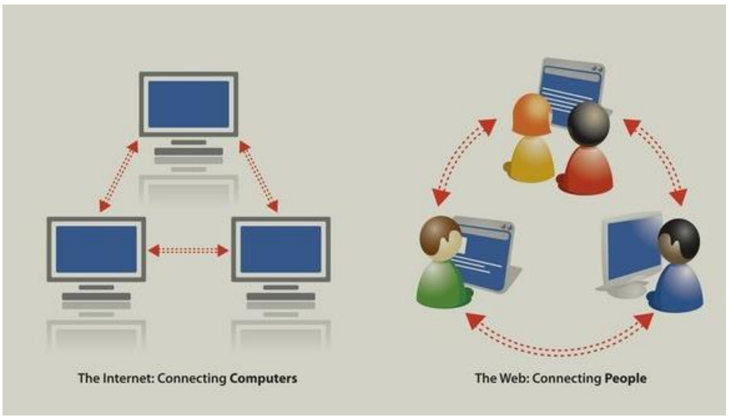


Introduction to WWW

- World Wide Web, which is also known as a Web, is a collection of websites or web pages stored in web servers and connected to local computers through the internet.
- These websites contain text pages, digital images, audios, videos, etc.
- Users can access the content of these sites from any part of the world over the internet using their devices such as computers, laptops, cell phones, etc.
- The building blocks of the Web are web pages which are formatted in HTML and connected by links called "hypertext" or hyperlinks and accessed by HTTP.
- A web page is given an online address called a Uniform Resource Locator (URL).
- A particular collection of web pages that belong to a specific URL is called a website, e.g., www.facebook.com, www.google.com, etc.

Difference between World Wide Web and Internet

- Internet is entirely different from WWW.
- It is a worldwide network of devices like computers, laptops, tablets, etc.
- When we send an email or when we chat, we are using the internet.


The Internet: Connecting Computers
The Web: Connecting People

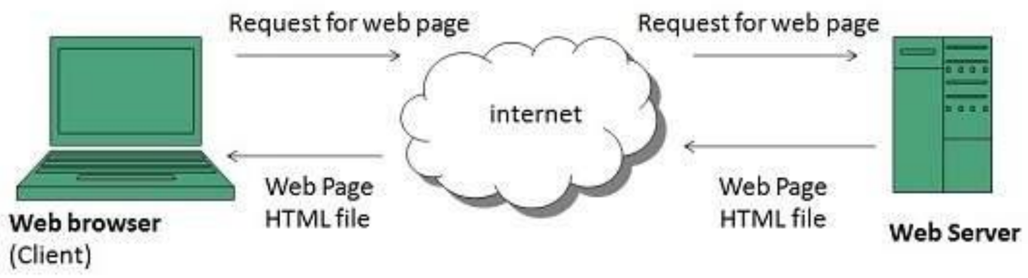
History of the World Wide Web

- The World Wide Web was invented by a British scientist, ***Tim Berners-Lee*** in 1989.
- He was working at CERN at that time.
- The World Wide Web was invented by a British scientist, ***Tim Berners-Lee*** (in the picture)in 1989.
- He was working at CERN at that time.
- Originally, it was developed by him to fulfill the need of automated information sharing between scientists across the world.
- In March 1989, Tim Berners-Lee wrote the first proposal for the World Wide Web. Later.
- He wrote another proposal in May 1990.
- After a few months, in November 1990, along with Robert Cailliau, it was formalized as a management proposal.
- The project was named "hypertext project" called World Wide Web in which a web of hypertext documents could be viewed by browsers.
- His proposal included the three main technologies (**HTML, URL, and HTTP**).
- In 1990, Tim Berners-Lee was able to run the first Web server and browser at CERN to demonstrate his ideas.
- In 1993, the National Center for Supercomputing Applications (NCSA) introduced the first version of its Mosaic browser.

- April 1993, CERN made the source code of WWW available on a royalty-free basis and thus made it free software.
- In 1994, Berners-Lee left CERN and joined MIT and established the International World Wide Web Consortium (W3C)

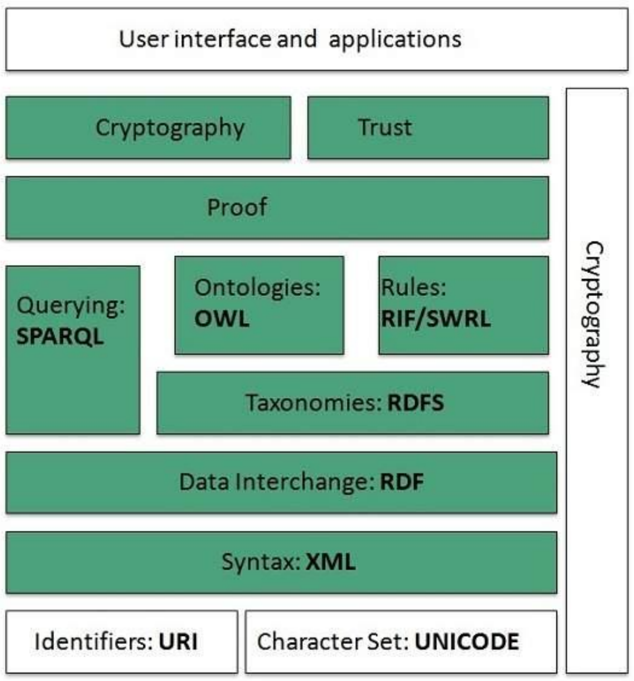
How the World Wide Web Works?

- The Web works as per the internet's basic client-server format as shown in the image.
- The servers store and transfer web pages or information to user's computers on the network when requested by the users.
- A web server is a software program which serves the web pages requested by web users using a browser.
- The computer of a user who requests documents from a server is known as a client. Browser, which is installed on the user' computer, allows users to view the retrieved documents.



WWW Architecture

WWW architecture is divided into several layers as shown in the following diagram:



Identifiers and Character Set

- **Uniform Resource Identifier (URI)** is used to uniquely identify resources on the web

- **UNICODE** makes it possible to built web pages that can be read and write in human languages.

Syntax

- **XML (Extensible Markup Language)** helps to define common syntax in semantic web.

Data Interchange

- **Resource Description Framework (RDF)** framework helps in defining core representation of data for web. RDF represents data about resource in graph form.

Taxonomies

- **RDF Schema (RDFS)** allows more standardized description of **taxonomies** and other **ontological** constructs.

Ontologies

Web Ontology Language (OWL) offers more constructs over RDFS. It comes in following three versions:

  - OWL Lite for taxonomies and simple constraints.
  - OWL DL for full description logic support.
  - OWL for more syntactic freedom of RDF

- **RIF** and **SWRL** offers rules beyond the constructs that are available from **RDFs** and **OWL.**
- Simple Protocol and **RDF Query Language (SPARQL)** is SQL like language used for querying RDF data and OWL Ontologies.

Proof

- This layer used to prove that the information comes from a trusted source.
- All semantic and rules that are executed at layers below Proof and their result will be used to prove deductions.

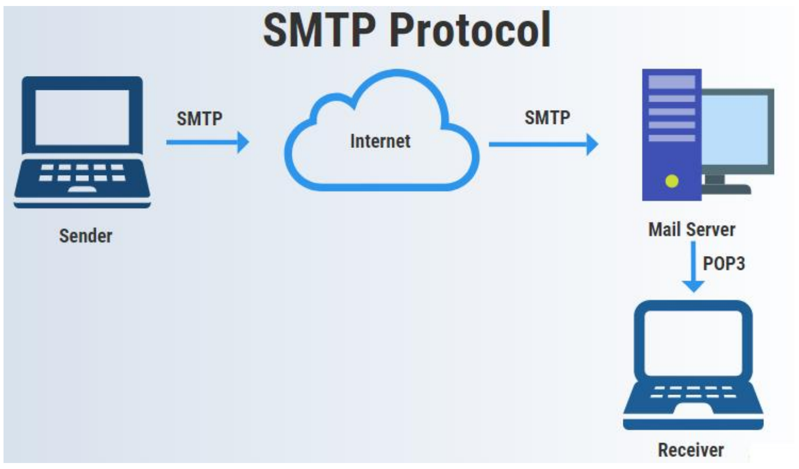
Cryptography

- **Cryptography** means such as digital signature for verification of the origin of sources is used.

User Interface and Applications

- On the top of layer **User interface and Applications** layer is built for user interaction.

SMTP



- The SMTP protocol is a Simple Mail Transfer Protocol that defines

1. **MTA** (Mail Transfer Agent) - client to send mail and
2. **MTA** (Mail Transfer Agent) - server to receive the mail.

- SMTP simply defines how data or commands transfer from client to server or server to client
- SMTP uses three phases,
  1. **Connection establishment,**
  2. **Mail transfer and**
  3. **Connection termination**
- It can also perform the following tasks:
  1. It can transmit a message to more than one recipient.
  2. It can attach text, video voice or graphics in the message.
  3. It can transmit messages on networks external to the internet, as well

Components

- SMTP consists of four main components:
  1. Mail user agent (MUA)
  2. Mail submission agent (MSA)
  3. Mail transfer agent (MTA)
  4. Mail delivery agent (MDA)

Types of SMTP Protocol

- The SMTP model can be of the following two types:
  - **End-to-end model**
  - **Store-and-forward model**

How does SMTP Protocol Work?

1. Composition of Mail

- The sender composes an e-mail message with the help of a Mail User Agent (MUA) and then transmits the e-mail.
- MUA helps in sending and receiving mail.
- The email message consists of two parts,
  1. **The body** - comprises the main portion of the message,
  2. **The header** - header comprises the subject, a one-line description of what the body holds and the sender and recipient address information.

2. Submission of Mail

- After the document's structure is complete,
- The customer utilizes the Simple Mail Transfer Protocol and sends the completed document to the server on TCP channel 25.

3. Delivery of Mail

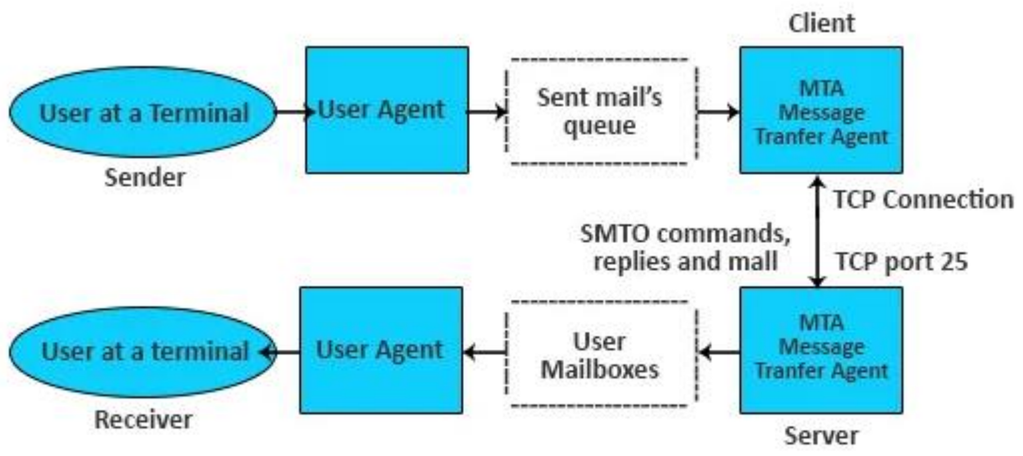
- E-mail addresses contain the recipient's username and the domain name.
- Ex: abc@gmail.com, "abc" is the username of the receiver and "gmail.com" is the domain name.
- If the recipient"s domain name is different from the sender's domain name,
  1. then MSA will send the letter to Mail Transfer Agent (MTA).
  2. MTA looks in the Domain Name System (DNS) to find the target domain.
  3. MTA connects to the server to transmit the message.

4. Receipt and Processing of Mail

- After the server receives the message, it sends it to the Mail Delivery Agent (MDA), which keeps the e-mail until the user retrieves it.

5. Access and Retrieval of Mail

- The email stored in the MDA is retrieved by logging into the MUA with the help of a username and password of the MUA.



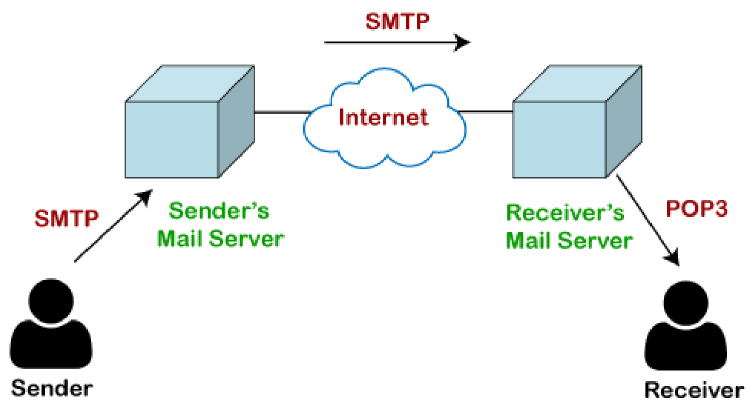
Advantages

1. Easiest way to communicate among several computers.
2. Easy and quick to send emails.
3. SMTP presents reliability for outgoing email messages.
4. Committed server to manage outgoing email messages in organizations.
5. It is easy and simple to connect to SMTP, also offers an easy installation.
6. No limitations and can be used to connect to any system.
7. It doesn't include any development from our end.

POP3 (Post Office Protocol)

- The POP protocol stands for Post Office Protocol.
- As we know that SMTP is used as a message transfer agent.
- When the message is sent, then SMTP is used to deliver the message from the client to the server and then to the recipient server.
- The message is sent from the recipient server to the actual server with the help of the Message Access Agent.
- The Message Access Agent contains two types of protocols,
    1. POP3 and
    2. IMAP.

How is mail transmitted?



- First mail is transmitted to the sender's mail server.
- Then, it is transmitted to the receiver's mail server over the internet.
- The mail is then sent to the user.
- From the sender to the sender's mail server and then to the receiver's mail server is done by SMTP protocol.
- At the receiver's mail server, the POP or IMAP protocol takes the data and transmits to the actual user.
- Since SMTP is a push protocol so it pushes the message from the client to the server.
- The third stage of email communication requires a pull protocol, and POP is a pull protocol.

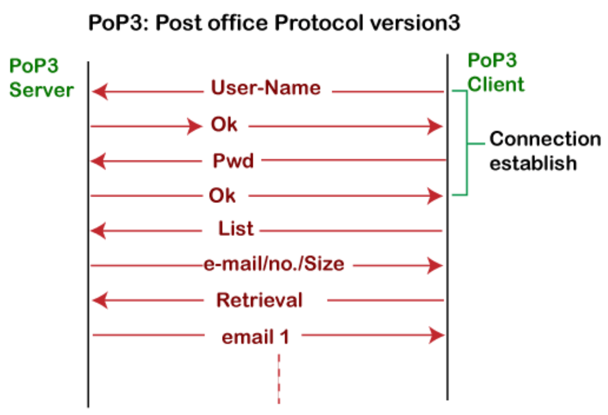
What is POP3?

- The POP3 is a simple protocol and having very limited functionalities.
- In the case of the POP3 protocol, the POP3 client is installed on the recipient system while the POP3 server is installed on the recipient's mail server.

History of POP3 protocol

- 1984 - The first version of post office protocol was first introduced in RFC 918 by the internet engineering task force.
- In 1985, the post office protocol version 2 was introduced in RFC 937,
- In 1988, version 2 war replaced with version 3 in RFC 1081.
- Then, POP3 was revised for the next 10 years before it was published.
- In 1996, finally got published.

Let's understand the working of the POP3 protocol.



To establish the connection between the POP3 server and the POP3 client

1. The POP3 server asks for the user name to the POP3 client.
2. If the username is found in the POP3 server, then it sends the ok message.
3. It then asks for the password from the POP3 client; then the POP3 client sends the password to the POP3 server.
4. If the password is matched, then the POP3 server sends the OK message, and the connection gets established.
5. After the establishment of a connection, the client can see the list of mails on the POP3 mail server.
6. In the list of mails, the user will get the email numbers and sizes from the server.
7. Out of this list, the user can start the retrieval of mail.

Advantages of POP3 protocol

- It allows the users to read the email offline. It requires an internet connection only at the time of downloading emails from the server.
- The POP3 protocol does not require permanent internet connectivity.
- It provides easy and fast access to the emails.
- There is no limit on the size of the email which we receive or send.
- It requires less server storage space.
- It is a simple protocol so it is one of the most popular protocols used today.
- It is easy to configure and use.

Disadvantages of POP3 protocol

- If the emails are downloaded from the server, then all the mails are deleted from the server by default.
- Transferring the mail folder from the local machine to another machine can be difficult.
- High risk of a virus attack.
- The mails are stored on the local machine, so anyone who sits on your machine can access the email folder.

File Transfer Protocol

- FTP stands for File transfer protocol.
- FTP is a standard internet protocol provided by TCP/IP used for transmitting the files from one host to another.
- Within the TCP/IP suite, FTP is considered an application layer protocol.
- It is mainly used for transferring the web page files from their creator to the computer that acts as a server for other computers on the internet.
- FTP is still commonly used to transfer files behind the scenes for other applications, such as banking services.
- It is also sometimes used to download new applications via web browsers.

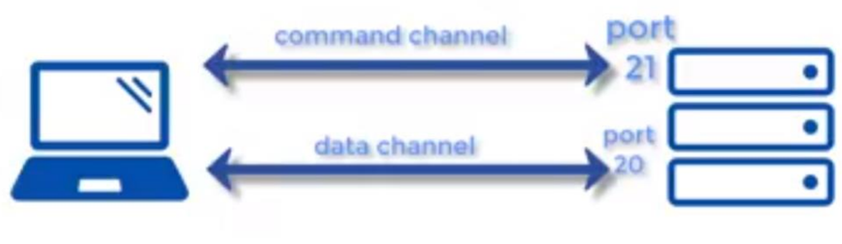
Objectives of FTP

- It provides the sharing of files.

- It is used to encourage the use of remote computers.
- It transfers the data more reliably and efficiently.

How does FTP work?

- FTP is a client-server protocol that relies on two communications channels
    - A **command channel** for controlling the conversation and
    - A **data channel** for transmitting file content.



- Here is how a typical FTP transfer works:
- A user typically needs to log on to the FTP server (Login not required in anonymous FTP).
- The client initiates a conversation with the server when the user requests to download a file.
- Using FTP, a client can upload, download, delete, rename, move and copy files on a server.

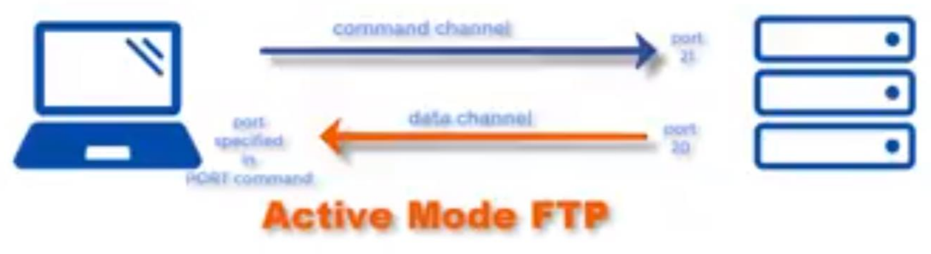
FTP sessions work in active or passive modes:

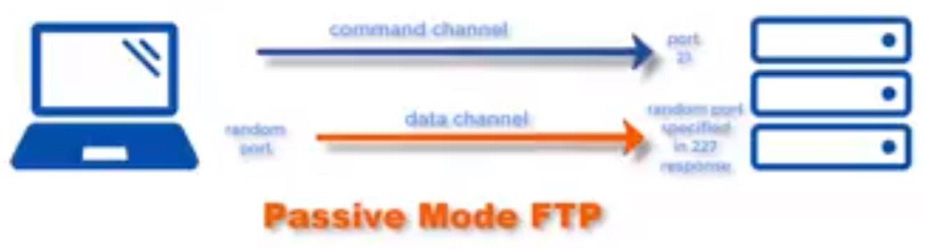
Active mode

- Client initiates a session via a command channel request.
- The server creates a data connection back to the client and begins transferring data.

Passive mode

- The server uses the command channel to send the client the information it needs to open a data channel.
- Because passive mode has the client initiating all connections, it works well across firewalls and network address translation gateways.



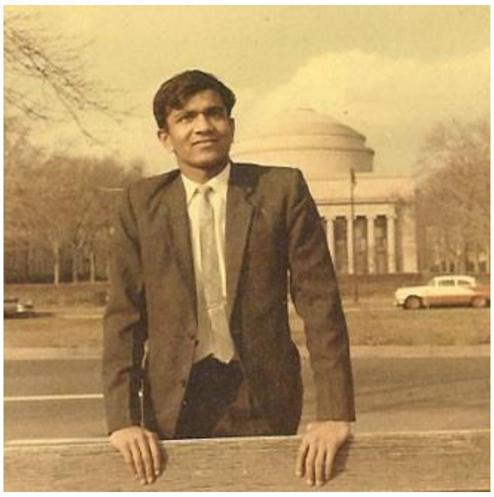


Advantages of FTP:

- **Speed:** FTP is one of the fastest way to transfer the files from one computer to another computer.
- **Efficient:** It is more efficient as we do not need to complete all the operations to get the entire file.
- **Security:** FTP server is more secure.
- **Back & forth movement:** FTP allows us to transfer the files back and forth.

Disadvantages of FTP:

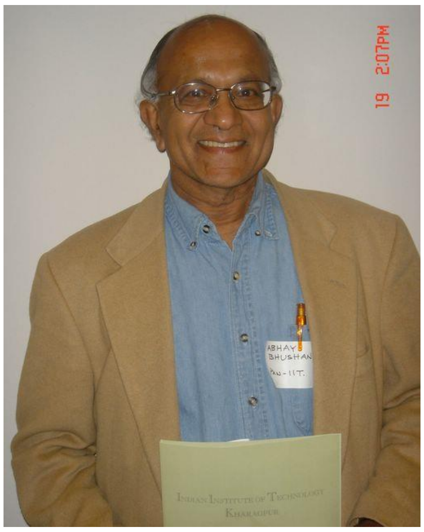
- The standard requirement of the industry is that all the FTP transmissions should be encrypted.
- However, not all the FTP providers are equal and not all the providers offer encryption.
- It also doesn't allow you to run simultaneous transfers to multiple receivers.
- Passwords and file contents are sent in clear text that allows unwanted eavesdropping.
- It is not compatible with every system.



FTP Extras

a major contributor to the development of the Internet **TCP/IP architecture**
Author of **File Transfer Protocol** and early **email** versions an Indian **Abhay Bhusan**

first batch (1960–65) from the Indian Institute of Technology Kanpur



Overview of HTTP

- HTTP stands for **HyperText Transfer Protocol**.
- It is a protocol used to access the data on the World Wide Web (www).
- The HTTP protocol can be used to transfer the data in the form of plain text, hypertext, audio, video, and so on.
- Allows us to use in a hypertext environment where there are rapid jumps from one document to another document.
- HTTP is similar to the FTP as it also transfers the files from one host to another host.
    - HTTP is simpler than FTP as HTTP uses only one connection.
- HTTP is used to carry the data in the form of MIME-like format.
- HTTP is similar to SMTP as the data is transferred between client and server.
- SMTP messages are stored and forwarded while HTTP messages are delivered immediately.

The Basic Characteristics of HTTP

- It is the protocol that allows web servers and browsers to exchange data over the web.
- It is a request response protocol.
- It uses the reliable TCP connections by default on TCP port 80.
- It is stateless means each request is considered as the new request. In other words, server doesn't recognize the user by default.

Features of HTTP

- **Connectionless protocol**
    - HTTP is a connectionless protocol.
    - HTTP client initiates a request and waits for a response from the server.
    - The server processes the request and sends back the response to the HTTP client after which the client disconnects the connection.
    - The connection between client and server exist only during the current request and response time only.
- **Media independent**
    - It specifies that any type of media content can be sent by HTTP as long as both the server and the client can handle the data content.
- **Stateless**
    - HTTP is a stateless protocol as both the client and server know each other only during the current request.
    - Due to this nature of the protocol, both the client and server do not retain the information between various requests of the web pages.

HTTP Request

- HTTP Requests are messages which are sent by the client or user to initiate an action on the server.
- In general, an HTTP request is divided into 3 parts:
    1. A request line
    2. A set of header fields
    3. A body, which is optional

Syntax

Request-Line = Method SP Request-URI SP HTTP-Version CRLF

Request Line

The Request-Line starts with

- A method token, which is followed by
- The Request-URI, the protocol version, and
- Ending with CRLF (Carriage Return Line Feed, used to terminate a line).

Using the SP characters, the elements are separated.

1) Method

The method token is used to indicate the method which was performed on the resource identified by the Request-URI. The method is case sensitive.

Syntax

```
Method = "OPTIONS"
       | "GET"
       | "HEAD"
       | "POST"
       | "PUT"
       | "DELETE"
       | "TRACE"
       | "CONNECT"
```

- A resource is allowed a list of methods and that methods can be specified in an Allow header field.
- The response's return code always notifies the client whether a method is currently allowed on a resource.

Method and Description:

i) GET	This method retrieves information from the given server using a given URI. It cannot apply other effects on the data.
ii) HEAD	Same as the GET method, It is used to transfer the status line and header section only.
iii) POST	Sends the data to the server. For example, file upload, customer information, etc. using the HTML forms.
iv) PUT	Used to replace all the current representations of the target resource with the uploaded content.
v) DELETE	Used to remove all the current representations of the target resource, which is given by URI.
vi) CONNECT	Establishes a tunnel to the server, which is identified by a given URI

.

2) Request-URI

- The Request-URI is a Uniform Resource Identifier. It is used to identify the resource upon which to apply the request.

  Syntax

  Request-URI = "*" | absoluteURI | abs_path | authority

On the nature of the request, these four options for Request-URI depend.

a) The asterisk "*" is used to show that the request does not apply to a particular resource, but it will apply to the server itself. It is allowed only when the method used does not necessarily apply to a resource.

Ex: OPTIONS * HTTP/1.1

b) The **absoluteURI** form is used only when the request is being made to a proxy. The requested proxy is used to forward the request and return the response.

Example

GET http://www.javatpoint.com/WWW/TheProject.html HTTP/1.1

c) The **absolute path** can't be empty. If in the original URI, none is present, it must be given as "/".

d) The **authority** form is only used by the CONNECT method.

The Resource Identified by a Request

Using the examination of Request-URI and the Host header field, we can determine the exact resource identified by the Internet request.

An origin server must use the following rules for determining the requested resource on an HTTP/1.1 request if the origin server does differentiate based on the host requested.

1. The host will be part of the Request-URI if Request-URI is an absoluteURI.
2. The host will be determined by the Host header field value if the Request-URI is not an absoluteURI, and the request includes a header field of the host.
3. The response MUST be a 400 (Bad Request) error message if the host as determined by rule 1 or 2 is not a valid host on the server.

Request Header Fields

The request-header fields are used to allow the client to pass additional information to the server like the request and the client itself. The request header fields act as request modifiers, with semantics equivalent to the parameters on a programming language method invocation.

Syntax

```
request-header = Accept
               | Accept-Charset
               | Accept-Encoding
               | Accept-Language
               | Host
```

The name of the request-header field can be extended reliably only in combination with a change in the version of the protocol.

Example of a HTTP reqest:

```
POST /api/authors HTTP/1.1
Host: myWebApi.com
Content-Type: application/json
Cache-Control: no-cache
```

```
{
    "Name": "Felipe Gavilán",
    "Age": 999
}
```

Response

- HTTP Response sent by a server to the client.
- The response is used to provide the client with the resource it requested.
- It is also used to inform the client that the action requested has been carried out.
- It can also inform the client that an error occurred in processing its request.

Similar to the structure of the request. These parts are:
An HTTP response contains the following things:

1. Status Line
2. Response Header Fields or a series of HTTP headers
3. Message Body (Optional)

Syntax:

Status-Line = HTTP-Version SP Status-Code SP Reason-Phrase CRLF

Status Line

In the response message, the status line is the first line. The status line contains three items:

- **HTTP Version Number**
  - It is used to show the HTTP specification to which the server has tried to make the message comply.

  Example

  HTTP-Version = HTTP/1.1
- **Status Code**
  - It is a three-digit number that indicates the result of the request.
  - The first digit defines the class of the response.
  - The last two digits do not have any categorization role.
  - There are five values for the first digit, which are as follows:

Code and Description

- **1xx: Information -** It shows that the request was received and continuing the process.
- **2xx: Success -** It shows that the action was received successfully, understood, and accepted.
- **3xx: Redirection -** It shows that further action must be taken to complete the request.
- **4xx: Client Error -** It shows that the request contains incorrect syntax, or it cannot be fulfilled.
- **5xx: Server Error -** It shows that the server failed to fulfil a valid request.

c) Reason Phrase

- It is also known as the status text. It is a human-readable text that summarizes the meaning of the status code.

Ex: HTTP/1.1 200 OK

Here,

- HTTP/1.1 is the HTTP version.
- 200 is the status code.
- OK is the reason phrase.

Response Header Fields

- The HTTP Headers contain the information about server and response.
- Used to displaying the response, storing the response for future.

```
response-header = Accept-Ranges
                | Age
                | Location
                | Proxy-Authenticate
                | Retry-After
                | Server
```

The name of the Response-header field can be extended reliably only in combination with a change in the version of the protocol.

Message Body

- The body of the message is used for most responses.
  - For a response to a **successful** request, the body of the message contains
    - The status of the action requested or
    - The resource requested
  - For the response to an **unsuccessful** request, the body of the message contains
    - Action the client needs to take to complete the request successfully or
    - The reason for the error.

Example of an HTTP response:

```
HTTP/1.1 200 OK
Date: Thu, 03 Jan 2019 23:26:07 GMT
Server: gws
Accept-Ranges: bytes
Content-Length: 68894
Content-Type: text/html; charset=UTF-8

<!doctype html><html …
```

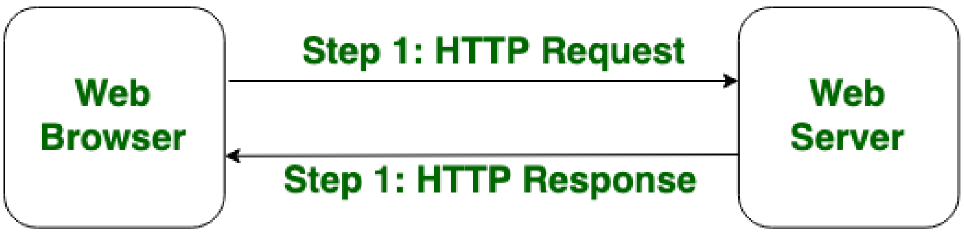
Generation of Dynamic Web Pages

- A web page can contain huge information including text, graphics, audio, video and hyper links.
- These hyper links are the link to other web pages.

Static Web page

- Static web pages are also known as flat or stationary web page.
- They are loaded on the client's browser as exactly they are stored on the web server.

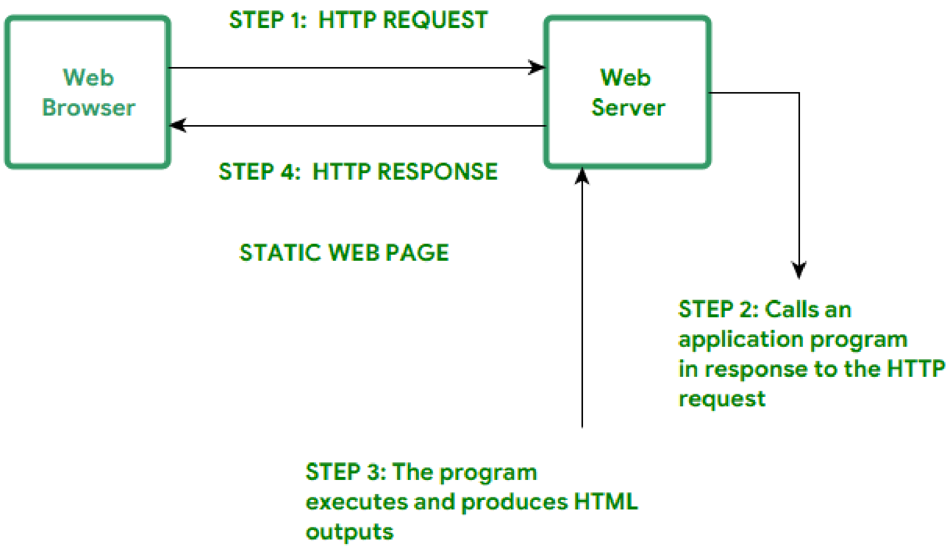
- User can only read the information but can't do any modification or interact with the information.
- Static web pages are only used when the information is no more required to be modified.



Static Web Page

Dynamic Web page

- Dynamic web page shows different information at different point of time.
- It is possible to change a portion of a web page without loading the entire web page.
- Usually written in CGI, AJAX, ASP or ASP.NET,
- Take more time to load than simple static pages.
- e.g., weather updates or stock prices, live scores.


STEP 1: HTTP REQUEST
Web Browser
Web Server
STEP 4: HTTP RESPONSE
STATIC WEB PAGE
STEP 2: Calls an application program in response to the HTTP request
STEP 3: The program executes and produces HTML outputs

Dynamic web page

Two types of dynamic web pages

Server-side dynamic web page

- It is created by using server-side scripting.
- A web page that changes when it's loaded or visited.
- When the pages are loaded, server-side content is generated.
- Ex: login pages, shopping carts and submission forms..

Client-side dynamic web page

- It is processed using client side scripting such as JavaScript.
- These scripts generate "client-side content" on the user's computer, rather than the webserver.
- And then passed in to Document Object Model (DOM).

Scripting Languages

- Scripting languages are like programming languages that allow us to write programs in form of script.
- These scripts are interpreted not compiled and executed line by line.

Client-side Scripting

- Client-side scripting refers to the programs that are executed on client-side.
- Client-side scripts contains the instruction for the browser to be executed in response to certain user's action.
- Client-side scripting programs can be embedded into HTML files or also can be kept as separate files.
- Commonly used client side scripting languages are listed below

  JavaScript
  - It is a prototype based scripting language.
  - It inherits its naming conventions from java.
  - All java script files are stored in file having .js extension.

  VBScript
  - It is an open source web programming language developed by Microsoft.
  - It is superset of JavaScript and adds object oriented programming.

  ActionScript
  - It is an object oriented programming language used for the development of websites and software targeting Adobe flash player.

  Dart
  - It is an open source web programming language developed by Google. It relies on source-to-source compiler to JavaScript.

Server-side Scripting

- **Server-side scripting** acts as an interface for the client and also limit the user access the resources on web server.
- It can also collect the user's characteristics in order to customize response.
- Some of the server side scripting languages are

  PHP
  - PHP is by far the most used server-side scripting language.
  - Above 80% of websites are running on PHP.
  - It was the first programming language designed specifically for the web

  Java
  - Many large enterprise applications in banking and insurance and mainframes use java.
  - Spring is one of the most popular frameworks for Java web applications

  Ruby
  - Ruby is mostly popular for small applications as it is suitable for rapid web development. Rails is the most popular framework to use with Ruby.

Node.js (JavaScript)

- Node.js is the newest in the list (released in 2009) and the quickest growing today.
- It provides the ability to run JavaScript code server-side.
- Express is a popular web framework that can be used develop Node.js web application

Python

- Python is one of the best programming languages for web development,
- Flask, Django are some of the framework for Python to create web applications

Mark up Language (HTML5):

Basics of Html

- All HTML documents must start with a document type declaration: <!DOCTYPE html>.
- The HTML document itself begins with <html> and ends with </html>.
- The visible part of the HTML document is between <body> and </body>.
- Anatomy of an HTML document

```
<!DOCTYPE html>
<html>
      <head>
            <meta charset="utf-8">
            <title>My test page</title>
      </head>
      <body>
            <img src="images/firefox-icon.png" alt="My test image">
      </body>
</html>
```

<!DOCTYPE html> — doctype.

- It is a required preamble.
- It must only appear once, at the top of the page.
- doctypes act as links to a set of rules that the HTML page had to follow.
- Had automatic error checking and other useful things.
- However these days, they don't do much and are basically just needed to make sure the document behaves correctly.

<html></html> — the <html> element.

- This element wraps all the content on the entire page and is sometimes known as the root element.

<head></head> — the <head> element.

- This element acts as a container for all the stuff we want to include on the HTML page that isn't the content we are showing to our page's viewers.
- This includes things like keywords and a page description that you want to appear in search results, CSS to style our content, character set declarations, and more.

<meta charset="utf-8">

- This element sets the character set the document should use to UTF-8 which includes most characters from the vast majority of written languages.

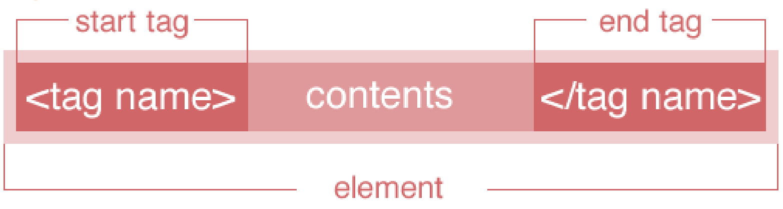
<title></title> — the <title> element.

- This sets the title of the page, which is the title that appears in the browser tab the page is loaded in.
- It is also used to describe the page when we bookmark/favorite it.

<body></body> — the <body> element.

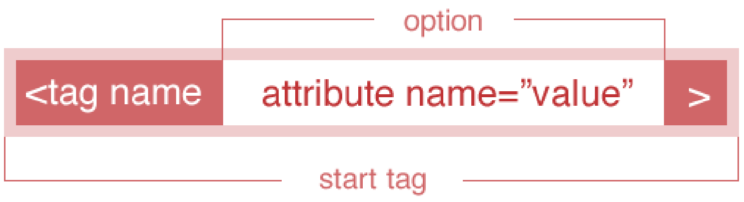
- This contains all the content that we want to show to web users when they visit the page, whether that's text, images, videos, games, playable audio tracks, or whatever else.

Syntax and Tags of Html



The main parts of our element are as follows:

1. **The opening tag:** This consists of the name of the element (in this case, p), wrapped in opening and closing **angle brackets**. This states where the element begins or starts to take effect — in this case where the paragraph begins.
2. **The closing tag:** This is the same as the opening tag, except that it includes a *forward slash* before the element name. This states where the element ends — in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.
3. **The content:** This is the content of the element, which in this case, is just text.
4. **The element:** The opening tag, the closing tag, and the content together comprise the element.



An attribute should always have the following:

1. A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
2. The attribute name followed by an equal sign.
3. The attribute value wrapped by opening and closing quotation marks.

Nesting elements

- We can put elements inside other elements too — this is called nesting.
  <p> My cat is <strong>very</strong> grumpy.</p>
- We do however need to make sure that the elements are properly nested.

Empty elements

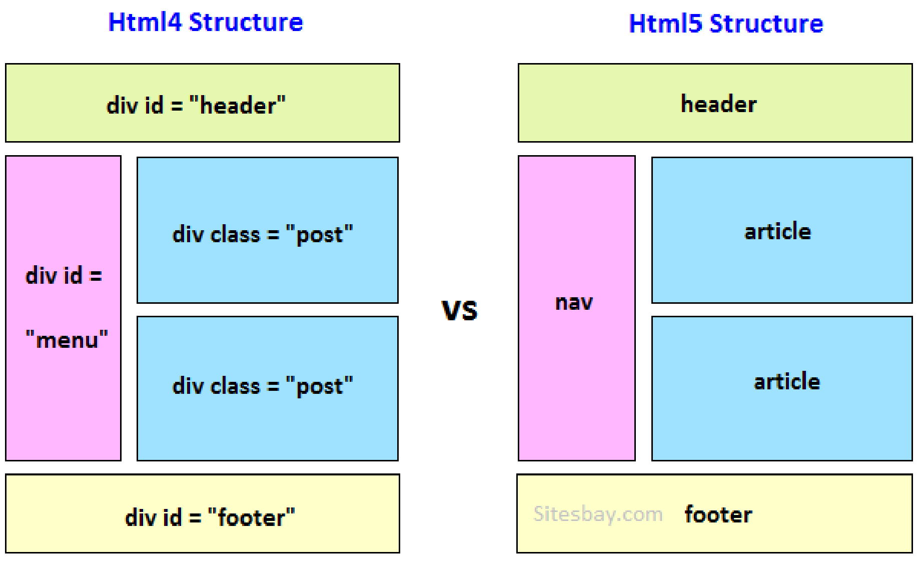
- Some elements have no content and are called **empty elements**.
- Some of the empty tags are <img>, <input>, <link>, <meta>, <hr>, etc.

Example:

```
<p>This paragraph contains <br> a line break.</p>
<img src="images/sky.jpg" alt="Cloudy Sky">
<input type="text" name="username">
```

Introduction to HTML5

- It is enriched with advance features which makes it easy and interactive for designer/developer and users.
- It facilitates you to design better forms and build web applications that work offline.
- It provides you advance features.
- HTML5 supports both **audio** and **video** while none of them were part of
- **HTML5** provides full support for running JavaScript.
- In **HTML5**, inline **mathML** and **SVG** can be used in a text.
- HTML5 supports new types of form controls, such as **date** and **time, email, number, category, title, Url, search, etc.**
- Many elements have been introduced in HTML5. Some of the most important are **time, audio, description, embed, fig, shape, footer, article, canvas, navy, output, section, source, track, video**, etc.



Features	Html	Html5
Definition	A hypertext markup language (HTML) is the primary language for developing web pages.	HTML5 is a new version of HTML with new functionalities with markup language with Internet technologies.
Multimedia support	Language in **HTML** does not have support for video and audio.	**HTML5** supports both video and audio.
Storage	The HTML browser uses cache memory as temporary storage.	HTML5 has the storage options like:**application cache, SQL database,** and **web storage**.

Browser compatibility	HTML is compatible with almost all browsers because it has been present for a long time, and the browser made modifications to support all the features.	In HTML5, we have many new tags, elements, and some tags that have been **removed/modified**, so only some browsers are fully compatible with **HTML5**.
Threading	In HTML, the browser interface and JavaScript running in the same thread.	The HTML5 has the JavaScript Web Worker API, which allows the browser interface to run in multiple threads.
Storage	Uses cookies to store data.	Uses local storage instead of cookies
Vector and Graphics	Vector graphics are possible with the help of technologies like **VML, Silverlight, Flash,etc**.	Vector graphics is an integral part of **HTML5, SVG** and **canvas**.
Shapes	It is not possible to create shapes like **circles, rectangles, triangles**.	We can draw shapes like **circles, rectangles, triangles**.
Doc type	Doctype declaration in html is too long <! DOCTYPE HTML PUBLIC "- // W3C // DTD HTML 4.01 // EN" "http://www.w3.org/TR/html4/strict.dtd">	The DOCTYPE declaration in html5 is very simple "<! DOCTYPE html>
Character Encoding	Character encoding in HTML is too long. <! DOCTYPE HTML PUBLIC "- // W3C // DTD HTML 4.0 Transitional // EN">	Character encoding declaration is simple <meta charset = "UTF-8">

Semantic/Structural Elements

- A semantic element clearly describes its meaning to both the browser and the developer.
- Non-semantic elements – Tells nothing about its content. Ex: <div> and <span>.
- Semantic elements - Clearly defines its content. Ex: <form>, <table>, and <article>.

Index	Semantic Tag	Description
1.	<article>	Defines an article

2.	<aside>	Defines content aside from the page content
3.	<details>	Defines additional details that the user can view or hide
4.	<figcaption>	Defines a caption for a <figure> element
5.	<figure>	Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.
6.	<footer>	Defines a footer for a document or section
7.	<header>	Specifies a header for a document or section
8.	<main>	Specifies the main content of a document
9.	<mark>	Defines marked/highlighted text
10.	<nav>	Defines navigation links
11.	<section>	Defines a section in a document
12.	<summary>	Defines a visible heading for a <details> element
13.	<time>	Defines a date/time

Few example of sematic and structural elements are given below

HTML5 <mark> Element

- The <mark> tag defines text that should be marked or highlighted.

Example:

```
<!DOCTYPE html>
 <html>
   <head>
        <style>
          mark {
            background-color: pink
            color: red
          }
         </style>
   </head>

 <body>
    <p>A mark element is displayed like this:</p>
    <mark>Highlighted text!!</mark>
    <p>Change the default CSS settings to see the effect.</p>
  </body>
</html>
```

HTML5 <summary> Element

- The <summary> tag defines a visible heading for the <details> element.
- The heading can be clicked to view/hide the details.
- The <summary> element should be the first child element of the <details> element.

Example:

```
<!DOCTYPE html>
<html>
 <body>
   <h1>The summary element</h1>
 <details>
   <summary>Epcot Center</summary>
   <p>Epcot is a theme park at Walt Disney World Resort featuring exciting attractions,
international pavilions, award-winning fireworks and seasonal special events.
  </p>
 </details>
 </body>
</html>
```

HTML5 <article> Element

HTML <article> element defines article content within a document, page, application, or a website. It can be used to represent a forum post, a magazine, a newspaper article, or a big story.

Example:

```
<article>
 <h2>Today's highlights</h2>
 <p>First story</p>
 <p>Second story</p>
 <p>Third story</p>
</article>
```

HTML5 <aside> Element

The <aside> element represent the content which is indirectly giving information to the main content of the page. It is frequently represented as a sidebar.

Example:

```
<body>
  <h2>My last year memories</h2>
  <p>I have visited Paris with my friends last month. This was the memorable journey
and   I wish to go there again.</p>
 <aside>
   <h4>Paris</h4>
   <p>Paris, France's capital, is a major European city and a global center for art, fashion,
gastronomy and culture</p>
 </aside>
</body>
```