

2nd Semester MCA02005 Internet and Web Programming

L-T-P 3-0-0 3 CREDITS

Module I (8 Periods)

Internet Architecture: Internet overview, evolution of internet. Internet components: Local Area Networks, Access Networks, Core Networks, Routers, Transmission infrastructure, ISPs. TCP/IP model, TCP/IP vs OSI model. HTML: HTML Overview, Structure of HTML Documents, Document Types, HTML Elements and attributes. Anchor Attributes, Image Tag and its attributes, Image and Anchors, Table.

Module II (8 Periods)

Image Map: Attributes, Client Side Image Maps and Server Side Maps.

HTML Layout: Background, colors and text, Tables, Frames, Layers, Page content Division <Div>, . CSS: Style Sheet Basic, Properties, Positioning with Style Sheet.

Forms: <FORM> Elements, Form controls. Dynamic HTML.

Module III (8 Periods)

Java Script: Introduction, Client-Side JavaScript, Server-Side JavaScript, JavaScript Objects, JavaScript Security. Operators: Assignment Operators, Comparison Operators, Arithmetic Operators, Increment, Decrement, Unary Negation, Logical Operators, String Operators, Special Operators, Conditional operator, Comma operator, delete, new, this, void.

Statements: Break, comment, continue, delete, do ... while, export, for, for...in, function, if...else, import, labelled, return, switch, var, while.

Module IV (8 Periods)

JavaScript (Properties and Methods of Each) :Array, Boolean, Date, Function, Math, Number, Object, String, regExp. Document and its associated objects, document, Link, Area, Anchor, Image, Applet, Layer.

Events and Event Handlers: General Information about Events, Defining Event Handlers, event.

Module V (8 Periods)

Server Side Programming: Common Gateway Interface (CGI), Active Server Pages.

Internet applications: FTP, Telnet, Email, Chat. World Wide Web: HTTP protocol. Search Engines. E-commerce and security issues including symmetric and asymmetric key, encryption and digital signature, and authentication. Emerging trends, Internet telephony, and virtual reality over the web, etc. Intranet and extranet, firewall.

Books:

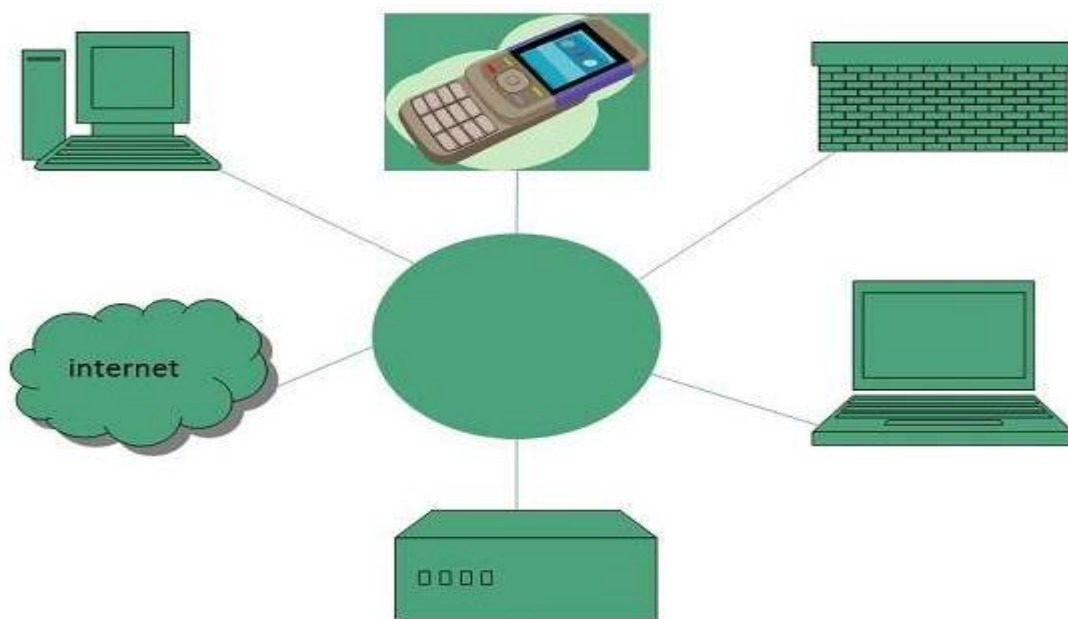
1. Computer Networking: A Top-Down Approach Featuring the Internet by Kurose and Ross, Pearson.
2. Web Design the Complete Reference by Thomas Powell, Tata McGrawHill.
3. HTML The Complete Reference by Thomas Powell, Tata McGrawHill.

Module I (8 Periods)

Internet

Internet is defined as an Information super Highway, to access information over the web. However, It can be defined in many ways as follows:

- Internet is a world-wide global system of interconnected computer networks.
- Internet uses the standard Internet Protocol (TCP/IP).
- Every computer in internet is identified by a unique IP address.
- IP Address is a unique set of numbers (such as 110.22.33.114) which identifies a computer location.
- A special computer DNS (Domain Name Server) is used to give name to the IP Address so that user can locate a computer by a name.
- For example, a DNS server will resolve a name <http://www.tutorialspoint.com> to a particular IP address to uniquely identify the computer on which this website is hosted.
- Internet is accessible to every user all over the world.
-



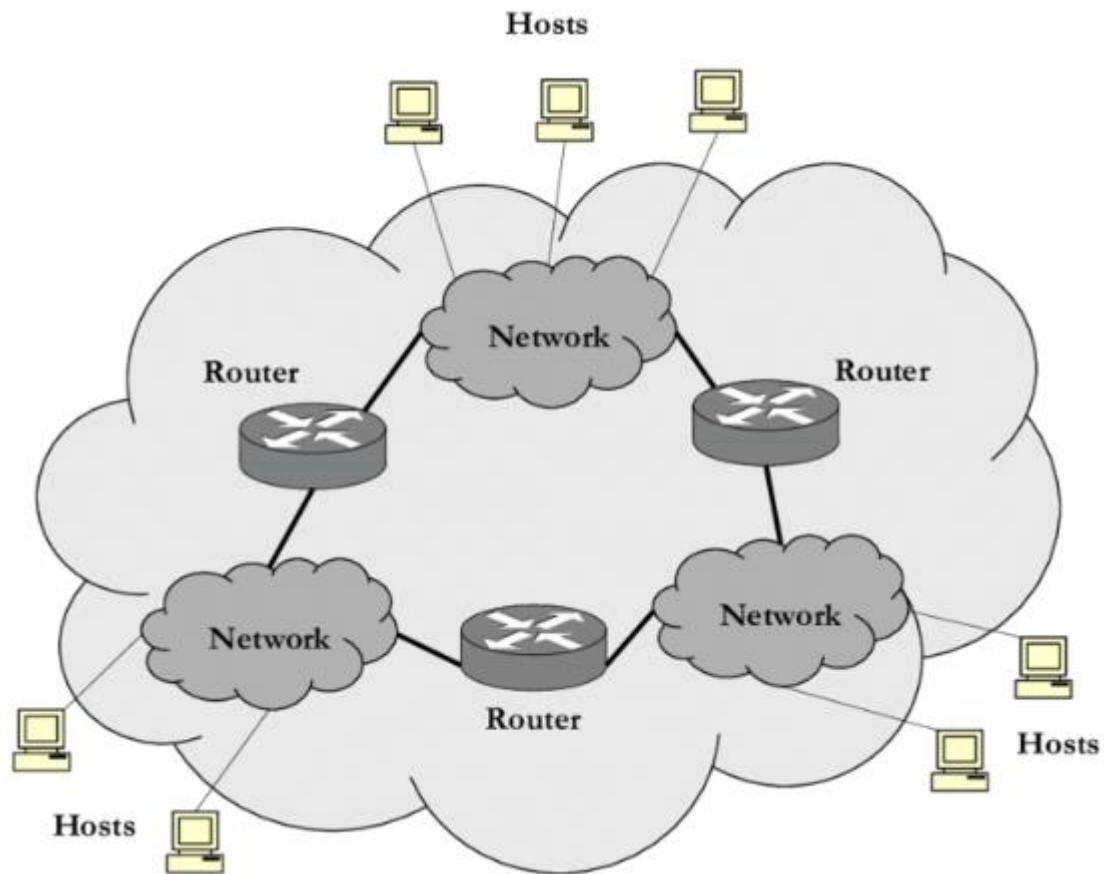
Evolution

The concept of Internet was originated in 1969 and has undergone several technological & Infrastructural changes as discussed below:

- 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.
- Initially, there were only four nodes, formally called Hosts.
- 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 components

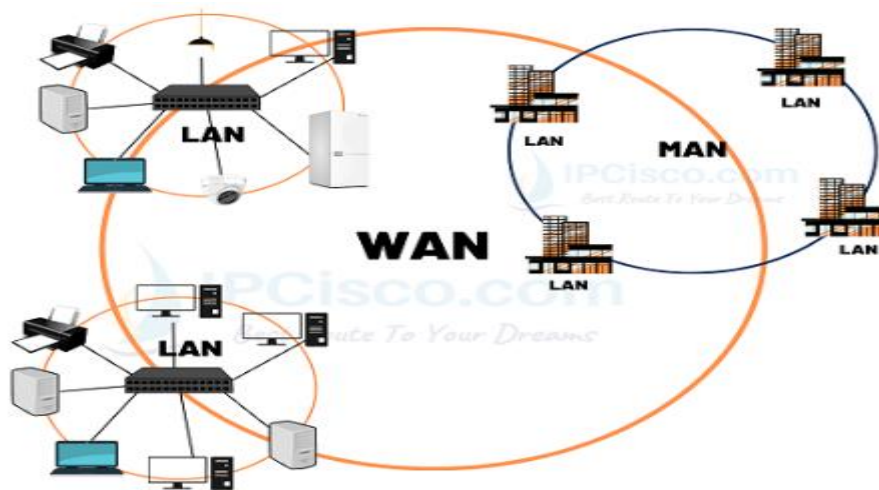


Types of Networks

- LAN stands for **Local Area Network**.
- MAN stands for **Metropolitan Area Network**.
- WAN stands for **Wide Area Network**.

BASIS	LAN	MAN	WAN
Full Form	Local Area Network	Metropolitan Area Network	Wide Area Network
Range	A communication network linking a number of stations in same local area. Range is 1 to 10 km	This network shares the characteristics of packet broadcasting networks. Range is 100 km	A communication network distinguished from a Local Area Network. Range is Beyond 100 km
Media Used	Uses guided media	Uses guided as well as unguided media	Uses unguided media
Speed	A high speed i.e. 100kbps to 100mbps	Optimized for a large geographical area than LAN.	Long distance communications, which may or may not be provided by public packet network.
Cost	cheaper	costly	expensive
Equipment needed	NIC, switch and hub	Modem and router	Microwave, radio, infra-red laser
protocols	Attached Resource computer network (ARCNET), Token ring	Frame relay and asynchronous transfer mode(ATM)	ATM, FDDI, SMDS





Access Networks

An access network is a type of network which physically connects an end system to the immediate router (also known as the “edge router”) on a path from the end system to any other distant end system. Examples of access networks are ISP, home networks, enterprise networks, ADSL, mobile network, FITH etc.

Types of access networks:

Ethernet –

It is the most commonly installed wired LAN technology and it provides services on the Physical and Data Link Layer of OSI reference model. Ethernet LAN typically uses coaxial cable or twisted pair wires.

DSL –

DSL stands for Digital Subscriber Line and DSL brings a connection into your home through telephone lines and a DSL line can carry both data and voice signals and the data part of the line is continuously connected. In DSL you are able to use the Internet and make phone calls simultaneously. DSL modem uses the telephone lines to exchange data with digital subscriber line access multiplexer (DSLAMs). In DSL we get 24 Mbps downstream and 2.5 Mbps upstream.

FTTH –

Fiber to the home (FTTH) uses optical fiber from a central Office (CO) directly to individual buildings and it provides high-speed Internet access among all access networks. It ensures high initial investment but lesser future investment and it is the most expensive and most future-proof option amongst all these access networks.

Wireless LANs –

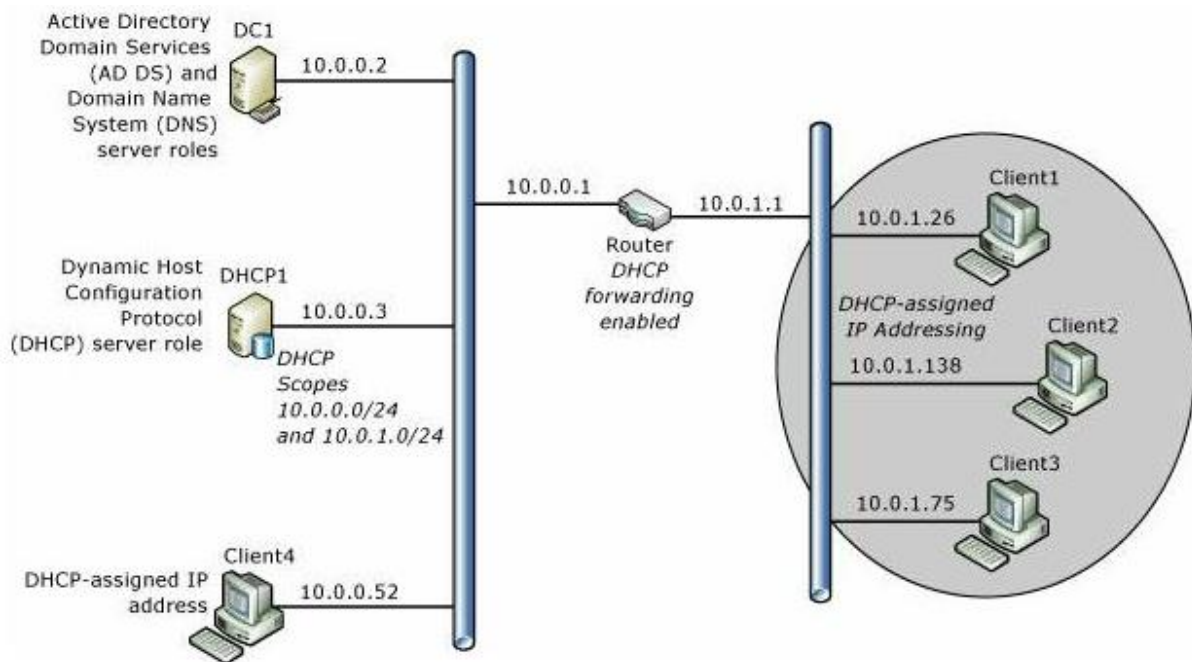
It links two or more devices using wireless communication within a range. It uses high-frequency radio waves and often include an access point for connecting to the Internet.

3G and LTE –

It uses cellular telephony to send or receive packets through a nearby base station operated by the cellular network provider. The term “3G internet” refers to the third generation of mobile phone standards as set by the International Telecommunications Union (ITU). Long Term Evolution (LTE) offers high-speed wireless communication for mobile devices and increased network capacity.

Core Network Overview

The following illustration shows the Windows Server Core Network topology.



Core Network Components

Following are the components of a core network.

Router

This deployment guide provides instructions for deploying a core network with two subnets separated by a router that has DHCP forwarding enabled. You can, however, deploy a Layer 2 switch, a Layer 3 switch, or a hub, depending on your requirements and resources. If you deploy a switch, the switch must be capable of DHCP forwarding or you must place a DHCP server on each subnet. If you deploy a hub, you are deploying a single subnet and do not need DHCP forwarding or a second scope on your DHCP server.

Static TCP/IP configurations

The servers in this deployment are configured with static IPv4 addresses. Client computers are configured by default to receive IP address leases from the DHCP server.

Active Directory Domain Services global catalog and DNS server DC1

Both Active Directory Domain Services (AD DS) and Domain Name System (DNS) are installed on this server, named DC1, which provides directory and name resolution services to all computers and devices on the network.

DHCP server DHCP1

The DHCP server, named DHCP1, is configured with a scope that provides Internet Protocol (IP) address leases to computers on the local subnet. The DHCP server can also be configured with additional scopes to provide IP address leases to computers on other subnets if DHCP forwarding is configured on routers.

Client computers

Computers running Windows client operating systems are configured by default as DHCP clients, which obtain IP addresses and DHCP options automatically from the DHCP server.

Routers

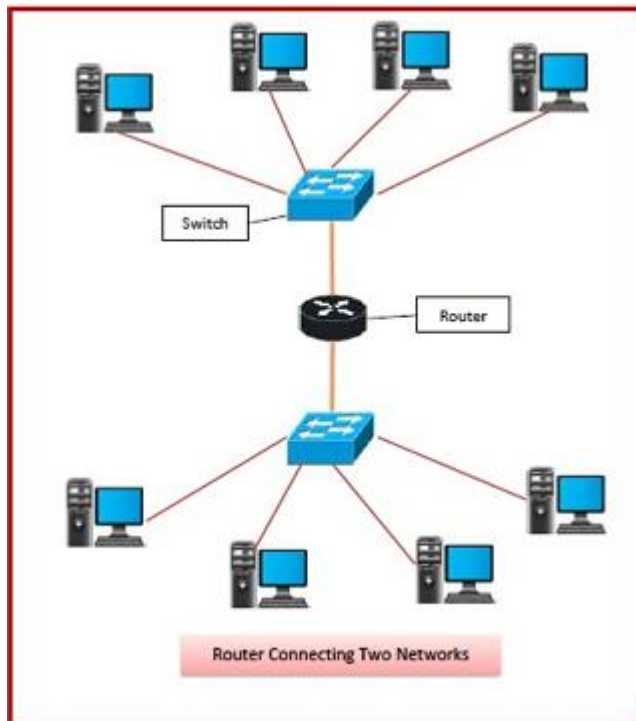
Routers are networking devices operating at layer 3 or a network layer of the OSI model. They are responsible for receiving, analysing, and forwarding data packets among the connected computer networks. When a data packet arrives, the router inspects the destination address, consults its routing tables to decide the optimal route and then transfers the packet along this route.

Features of Routers

A router is a layer 3 or network layer device.

- It connects different networks together and sends data packets from one network to another.
- A router can be used both in LANs (Local Area Networks) and WANs (Wide Area Networks).
- It transfers data in the form of IP packets. In order to transmit data, it uses IP address mentioned in the destination field of the IP packet.
- Routers have a routing table in it that is refreshed periodically according to the changes in the network. In order to transmit data packets, it consults the table and uses a routing protocol.
- In order to prepare or refresh the routing table, routers share information among each other.
- Routers provide protection against broadcast storms.
- Routers are more expensive than other networking devices like hubs, bridges and switches.
- Routers are manufactured by some popular companies like –
 - Cisco
 - D-Link

HP
3Com
Juniper
Nortel



Types of Transmission Technology

Transmission is actually the process of sending and propagating analog or signals of digital information. Transmission technology generally refers to physical layer protocol duties like modulation, demodulation, line coding, and many more. It might also include higher-level protocol duties such as digitizing analog signals, data compression, etc.

Types of Transmission Technology:

Transmission media is basically divided into two categories: Broadcast Networks, Point-to-Point Networks. These are explained as following below.

I. Broadcast Networks:

Broadcast networks are also known as terrestrial networks. It is basically a group of radio stations, television stations, or any other electronic media outlets that simply generate agreement to air, or broadcast, content generally from a centralized source. Broadcasting is simply a method of transferring messages to all the recipients simultaneously.

In this network, a message that is sent by a node is received by all the other nodes connected to the network and share a common medium of communication. Broadcast networks also avoid procedures of complex routing of switched network by simply confirming and ensuring that each transmission of nodes is basically received by all the other nodes in the network. This is the reason why the broadcast network has single communications channel.

In this network, each receiving station just receives all signals that are sent by transmitters. Even routing of signals is highly affected passively. These networks generally have single communication that is shared by all machines present on the network. Short messages also are known as packets that are sent by any of the machines present are received by all of the others present over there. Some of the systems of broadcast also support transmission to subset of machines also known as multicasting. It just links, in contrast, communication channel that is basically shared by all of machines in network.

Advantages of Broadcast Networks –

In this network, packets are generally transmitted and received by all of computers.

- It allows multicasting in the network.
- It has no limit. Even events can also run as long as required.
- It ensures better utilization of all resources available.

Disadvantages of Broadcast Networks –

- It cannot accommodate huge number of devices.
- It doesn't allow personalization of message.

II.Point-to-Point Networks:

Point-to-Point Networks or Point-to-Point Connection is type of private data connection that is connecting securely two or more locations for private data services. It might also be configured to usually carry voice, internet, and data services together all over same point-to-point network. It simply refers to type of communication connection among two endpoints or nodes of communication. It is connection among pairs of machines. Transmission from point-to-point with one sender and receiver is commonly known as unicasting.

This network is generally used for two locations that are required to securely send data that is very sensitive and confidential among each of locations. A point-to-point or P2P (Data Link) also gives or provides path from one point that is fixed to other point being fixed. It is very closed network data transport service that does not travel through public Internet. This network includes various connections

among individual pairs of machine. A packet present on these types of networks might be needed to go through intermediate computers before they reach desired or destination computer. The packets also need to follow multiple routes of different length sizes.

Therefore, routing algorithms are very essential and important in point-to-point connection. This network is generally available in range of bandwidth speeds along with point-to-point T1, point-to-point Ethernet, or many more.

Advantages of Point-to-Point Networks –

- It increases productivity.
- It generally uses leased lines so that speeds are guaranteed.
- It provides better security so that data can be transferred securely with confidence.

Disadvantages of Point-to-Point Networks –

- With this network, we can only connect two sites.
- It is very expensive for distant locations.

ISP: Internet Service Provider

ISP stands for Internet Service Provider. It is a company that provides access to the internet and similar services such as Website designing and virtual hosting. For example, when you connect to the Internet, the connection between your Internet-enabled device and the internet is executed through a specific transmission technology that involves the transfer of information packets through an Internet Protocol route.



Data is transmitted through different technologies, including cable modem, dial-up, DSL, high speed interconnects. Accordingly, based on the method of data transmission, the Internet access provided by ISPs can be divided into many types, some of which are as follows:

Dial-up Internet access: It is the oldest technology to provide Internet access by modem to modem connection using telephone lines. In this method, the user's computer is connected to a modem with a telephone line. This method has become outdated today due to slow connection speed. However, in remote areas, this method can be used where the broadband network is not available.

DSL: DSL, which stands for 'digital subscriber line' is an advanced version of the dial-up Internet access method. It uses high frequency to execute a connection over the telephone network and allows the internet and the phone connection to run on the same telephone line. This method offers an Asymmetric Digital Subscriber (ADSL), where the upload speed is less than the download speed, and a Symmetric Digital Subscriber Line (SDSL), which offers equal upload and download speeds. Out of these two, ADSL is more popular among users and is popularly known as DSL.

Wireless Broadband (WiBB): It is a modern broadband technology for Internet access. It allows high-speed wireless internet within a large area. To use this technology, you are required to place a dish on the top of your house and point it to the transmitter of your Wireless Internet Service Provider (WISP).

Wi-Fi Internet: It is the short form for "wireless fidelity," which is a wireless networking technology that provides wireless high-speed Internet connections using radio waves. To use the internet, you are required to be within the range of wi-fi network. It is commonly used in public places such as hotels, airports, restaurants to provide internet access to customers.

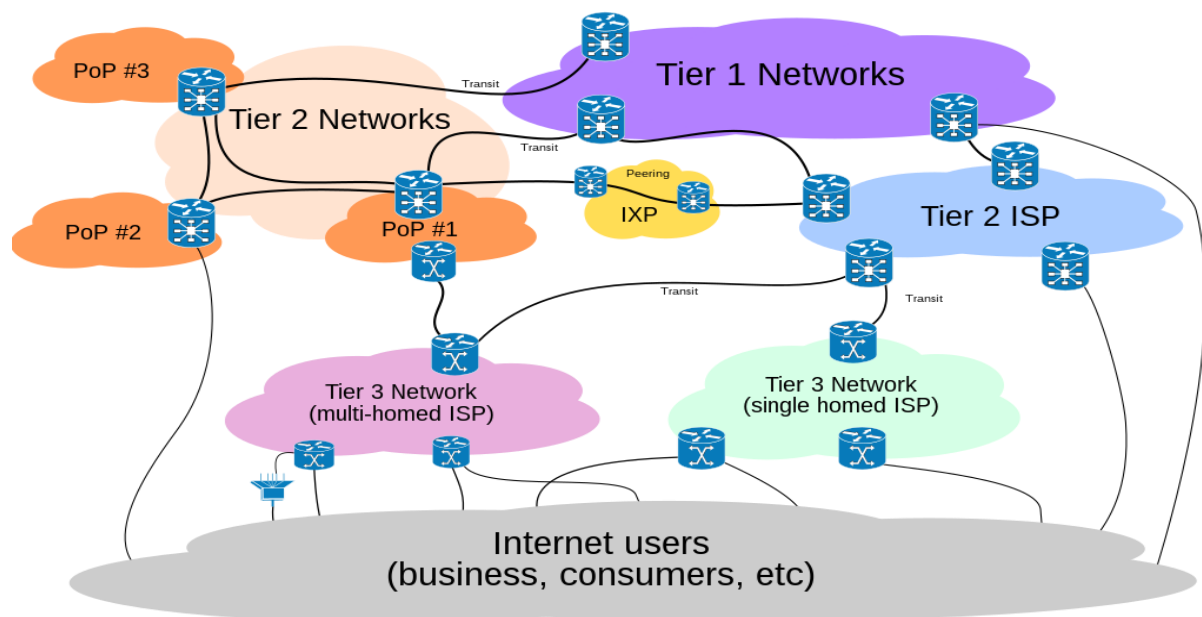
ISDN: It is a short form of Integrated Services Digital Network. It is a telephone system network which integrates a high-quality digital transmission of voice and data over the same standard phone line. It offers a fast upstream and downstream Internet connection speed and allows both voice calls and data transfer.

Ethernet: It is a wired LAN (Local Area Network) where computers are connected within a primary physical space. It enables devices to communicate with each other via a protocol (a set of rules or common network language). It may provide different speeds such as 10 Mbps, 100 Mbps and 10 Gbps.

Internet Backbone

The Internet backbone may be defined by the principal data routes between large, strategically interconnected computer networks and core routers of the Internet. These data routes are hosted by commercial, government, academic and other high-capacity network centers, as well as the Internet exchange points and network access points, that exchange Internet traffic between the countries, continents, and across the oceans. Internet service providers, often Tier 1 networks, participate in Internet backbone traffic by privately negotiated interconnection agreements, primarily governed by the principle of settlement-free peering.

Tier 1 internet service providers (ISP) mesh their high-speed fiber-optic networks together to create the internet backbone, which moves traffic efficiently among geographic regions.



What is POP in Internet?

On the Internet, a *point-of-presence (POP)* is **an access point from one place to the rest of the Internet**. A POP necessarily has a unique Internet Protocol (IP) address.

How traffic gets on the backbone

- Below the Tier 1 ISPs are smaller Tier 2 and Tier 3 ISPs.
- Tier 3 providers provide businesses and consumers with access to the internet. These providers have no access of their own to the internet backbone, so on their own would not be able to connect their customers to all of the billions of internet-attached computers.
- Buying access to Tier 1 providers is expensive.
- So often Tier 3 ISPs contract with Tier 2 (regional) ISPs that have their own networks that can deliver traffic to a limited geographic area but not to all internet-attached devices.
- In order to do that, Tier 2 ISPs contract with Tier 1 ISPs for access to the global backbone, and in that way make the entire internet accessible to their customers.
- This arrangement makes it possible for traffic from a computer on one side of the world to connect to one on the other side. That traffic goes from a source computer to a Tier 3 ISP that routes it to a Tier 2 ISP that routes it to a Tier 1 backbone provider that routes it to the appropriate Tier 2 ISP that routes it to a Tier 3 access provider that delivers it to the destination computer.

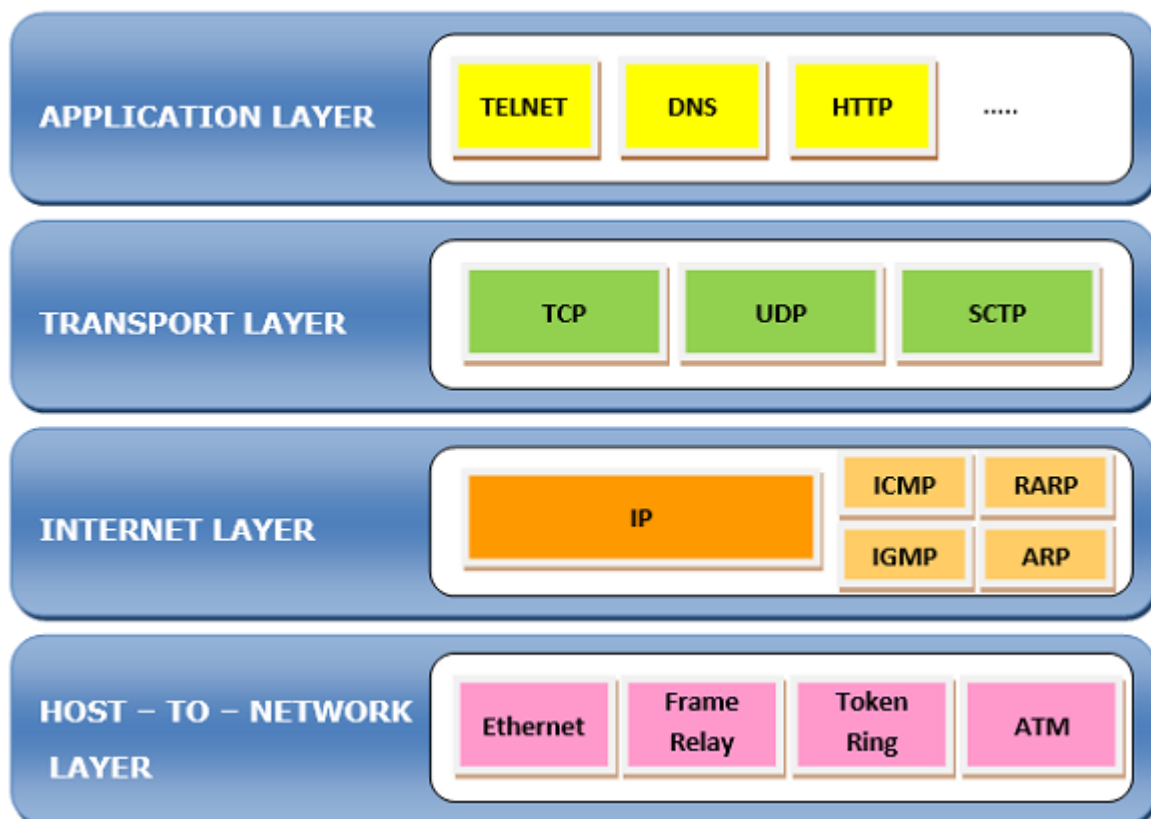
The TCP/IP Reference Model

TCP/IP Reference Model is a four-layered suite of communication protocols. It was developed by the DoD (Department of Defence) in the 1960s. It is named after the two main protocols that are used in the model, namely, TCP and IP. TCP stands for Transmission Control Protocol and IP stands for Internet Protocol.

The four layers in the TCP/IP protocol suite are –

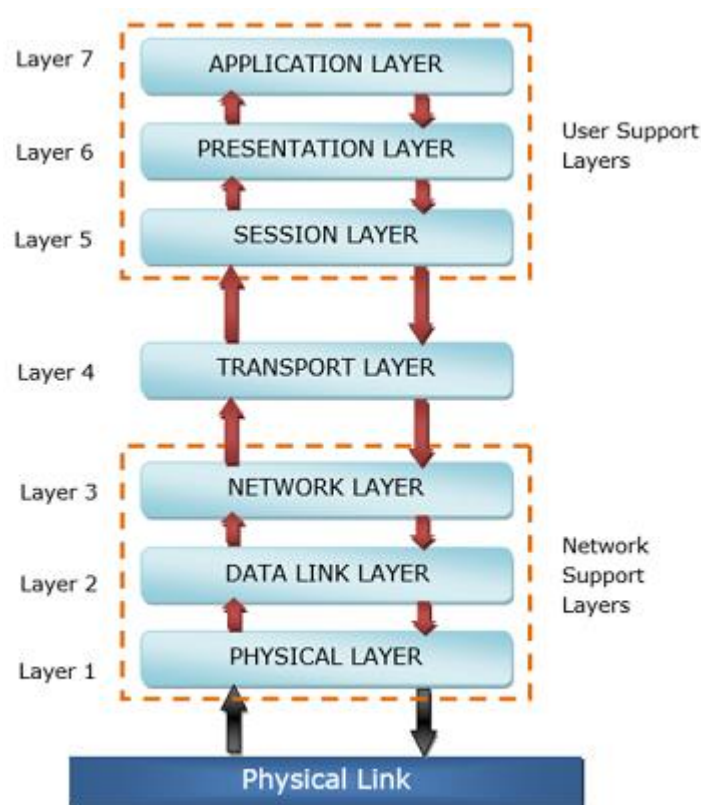
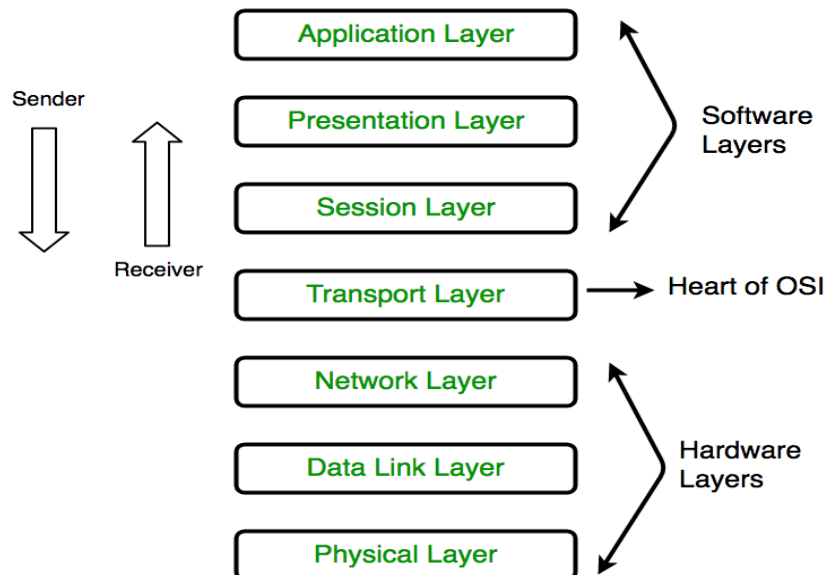
- **Host-to- Network Layer** –It is the lowest layer that is concerned with the physical transmission of data. TCP/IP does not specifically define any protocol here but supports all the standard protocols.
- **Internet Layer** –It defines the protocols for logical transmission of data over the network. The main protocol in this layer is Internet Protocol (IP) and it is supported by the protocols ICMP, IGMP, RARP, and ARP.
- **Transport Layer** – It is responsible for error-free end-to-end delivery of data. The protocols defined here are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).
- **Application Layer** – This is the topmost layer and defines the interface of host programs with the transport layer services. This layer includes all high-level protocols like Telnet, DNS, HTTP, FTP, SMTP, etc.

The following diagram shows the layers and the protocols in each of the layers –



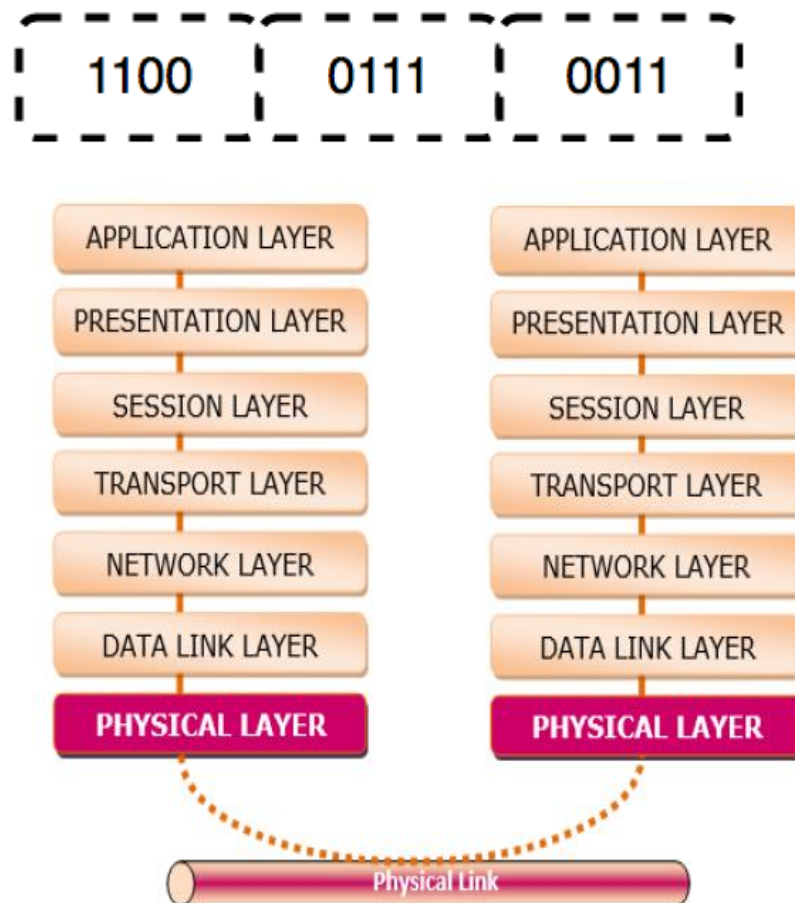
Open Systems Interconnection (OSI) Model

OSI stands for **Open Systems Interconnection**. It has been developed by ISO – ‘**International Organization for Standardization**’, in the year 1984. It is a 7 layer architecture with each layer having specific functionality to perform. All these 7 layers work collaboratively to transmit the data from one person to another across the globe.



Physical Layer (Layer 1):

The lowest layer of the OSI reference model is the physical layer. It is responsible for the actual physical connection between the devices. The physical layer contains information in the form of **bits**. It is responsible for transmitting individual bits from one node to the next. When receiving data, this layer will get the signal received and convert it into 0s and 1s and send them to the Data Link layer, which will put the frame back together.



The functions of the physical layer are as follows:

- **Bit synchronization:** The physical layer provides the synchronization of the bits by providing a clock. This clock controls both sender and receiver thus providing synchronization at bit level.
- **Bit rate control:** The Physical layer also defines the transmission rate i.e. the number of bits sent per second.
- **Physical topologies:** Physical layer specifies the way in which the different, devices/nodes are arranged in a network i.e. bus, star, or mesh topology.
- **Transmission mode:** Physical layer also defines the way in which the data flows between the two connected devices. The various transmission modes possible are Simplex, half-duplex and full-duplex.

* Hub, Repeater, Modem, Cables are Physical Layer devices.

** Network Layer, Data Link Layer, and Physical Layer are also known as **Lower Layers** or **Hardware Layers**.

Data Link Layer (DLL) (Layer 2):

The data link layer is responsible for the node-to-node delivery of the message. The main function of this layer is to make sure data transfer is error-free from one node to another, over the physical layer. When a packet arrives in a network, it is the responsibility of DLL to transmit it to the Host using its MAC address. Data Link Layer is divided into two sublayers:

- Logical Link Control (LLC)
- Media Access Control (MAC)

The packet received from the Network layer is further divided into frames depending on the frame size of NIC(Network Interface Card). DLL also encapsulates Sender and Receiver's MAC address in the header.

The Receiver's MAC address is obtained by placing an ARP(Address Resolution Protocol) request onto the wire asking "Who has that IP address?" and the destination host will reply with its MAC address.



The functions of the Data Link layer are:

- **Framing:** Framing is a function of the data link layer. It provides a way for a sender to transmit a set of bits that are meaningful to the receiver. This can be accomplished by attaching special bit patterns to the beginning and end of the frame.
- **Physical addressing:** After creating frames, the Data link layer adds physical addresses (MAC address) of the sender and/or receiver in the header of each frame.
- **Error control:** Data link layer provides the mechanism of error control in which it detects and retransmits damaged or lost frames.
- **Flow Control:** The data rate must be constant on both sides else the data may get corrupted thus, flow control coordinates the amount of data that can be sent before receiving acknowledgement.
- **Access control:** When a single communication channel is shared by multiple devices, the MAC sub-layer of the data link layer helps to determine which device has control over the channel at a given time.
 - * *Packet in Data Link layer is referred to as **Frame**.*
 - ** Data Link layer is handled by the NIC (Network Interface Card) and device drivers of host machines.
 - *** Switch & Bridge are Data Link Layer devices.

Network Layer (Layer 3):

The network layer works for the transmission of data from one host to the other located in different networks. It also takes care of packet routing i.e. selection of the shortest path to transmit the packet, from the number of routes available. The sender & receiver's IP addresses are placed in the header by the network layer.

The functions of the Network layer are:

- **Routing:** The network layer protocols determine which route is suitable from source to destination. This function of the network layer is known as routing.
- **Logical Addressing:** In order to identify each device on internetwork uniquely, the network layer defines an addressing scheme. The sender & receiver's IP addresses are placed in the header by the network layer. Such an address distinguishes each device uniquely and universally.

* *Segment* in Network layer is referred to as **Packet**.



** Network layer is implemented by networking devices such as routers.

Transport Layer (Layer 4):

The transport layer provides services to the application layer and takes services from the network layer. The data in the transport layer is referred to as *Segments*. It is responsible for the End to End Delivery of the complete message. The transport layer also provides the acknowledgement of the successful data transmission and re-transmits the data if an error is found.

- **At sender's side:** Transport layer receives the formatted data from the upper layers, performs **Segmentation**, and also implements **Flow & Error control** to ensure proper data transmission. It also adds Source and Destination port numbers in its header and forwards the segmented data to the Network Layer.

Note: The sender needs to know the port number associated with the receiver's application.

Generally, this destination port number is configured, either by default or manually. For example, when a web application makes a request to a web server, it typically uses port number 80, because this is the default port assigned to web applications. Many applications have default ports assigned

- **At receiver's side:** Transport Layer reads the port number from its header and forwards the Data which it has received to the respective application. It also performs sequencing and reassembling of the segmented data.

The functions of the transport layer are as follows:

- **Segmentation and Reassembly:** This layer accepts the message from the (session) layer, and breaks the message into smaller units. Each of the segments produced has a header associated with it. The transport layer at the destination station reassembles the message.
- **Service Point Addressing:** In order to deliver the message to the correct process, the transport layer header includes a type of address called service

point address or port address. Thus by specifying this address, the transport layer makes sure that the message is delivered to the correct process.

The services provided by the transport layer:

- **Connection-Oriented Service:** It is a three-phase process that includes
 - Connection Establishment
 - Data Transfer
 - Termination / disconnection

In this type of transmission, the receiving device sends an acknowledgement, back to the source after a packet or group of packets is received. This type of transmission is reliable and secure.

- **Connectionless service:** It is a one-phase process and includes Data Transfer. In this type of transmission, the receiver does not acknowledge receipt of a packet. This approach allows for much faster communication between devices. Connection-oriented service is more reliable than connectionless Service.

** Data in the Transport Layer is called as **Segments**.*

*** Transport layer is operated by the Operating System. It is a part of the OS and communicates with the Application Layer by making system calls.*

*Transport Layer is called as **Heart of OSI model**.*

Session Layer (Layer 5):

This layer is responsible for the establishment of connection, maintenance of sessions, authentication, and also ensures security.

The functions of the session layer are:

- **Session establishment, maintenance, and termination:** The layer allows the two processes to establish, use and terminate a connection.
- **Synchronization:** This layer allows a process to add checkpoints which are considered synchronization points into the data. These synchronization points help to identify the error so that the data is re-synchronized properly, and ends of the messages are not cut prematurely and data loss is avoided.
- **Dialog Controller:** The session layer allows two systems to start communication with each other in half-duplex or full-duplex.

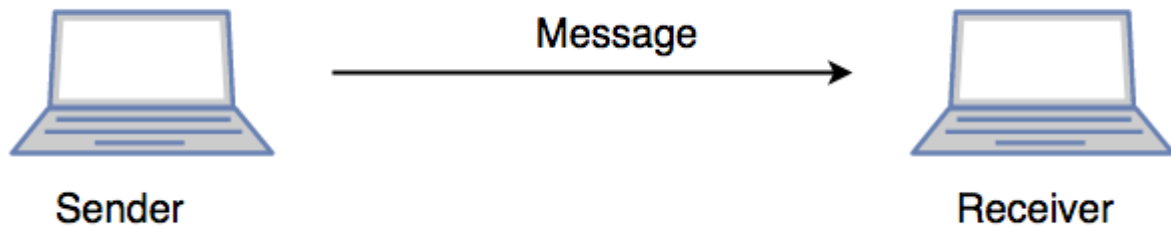
***All the below 3 layers (including Session Layer) are integrated as a single layer in the TCP/IP model as “Application Layer”.*

***Implementation of these 3 layers is done by the network application itself. These are also known as **Upper Layers** or **Software Layers**.*

Scenario:

Let us consider a scenario where a user wants to send a message through some Messenger application running in his browser. The “Messenger” here acts as the application layer which provides the user with an interface to create the data. This

message or so-called Data is compressed, encrypted (if any secure data), and converted into bits (0's and 1's) so that it can be transmitted.



Presentation Layer (Layer 6):

The presentation layer is also called the **Translation layer**. The data from the application layer is extracted here and manipulated as per the required format to transmit over the network.

The functions of the presentation layer are :

- **Translation:** For example, ASCII to EBCDIC.
- **Encryption/ Decryption:** Data encryption translates the data into another form or code. The encrypted data is known as the ciphertext and the decrypted data is known as plain text. A key value is used for encrypting as well as decrypting data.
- **Compression:** Reduces the number of bits that need to be transmitted on the network.

Application Layer (Layer 7):

At the very top of the OSI Reference Model stack of layers, we find the Application layer which is implemented by the network applications. These applications produce the data, which has to be transferred over the network. This layer also serves as a window for the application services to access the network and for displaying the received information to the user.

Example: Application – Browsers, Skype Messenger, etc.

***Application Layer is also called Desktop Layer.*



The functions of the Application layer are :

- Network Virtual Terminal
- FTAM-File transfer access and management
- Mail Services
- Directory Services

OSI model in a nutshell

No.	Layer Name	Responsibility	Information Form (Data Unit)	Device
7	Application Layer	Helps in identifying the client and synchronize communication	Message	-
6	Presentation Layer (Translation Layer)	Data from application layer is extracted and manipulated as required format for transmission	Message	-
5	Session Layer	Establishes connection, maintenance, authentication and ensure security	Message	Gateway
4	Transport Layer (HEART of OSI)	Take service from network layer and provide it to application layer	Segment	Firewall
3	Network Layer	Transmission of data from one host to other. Located in different network	Packet	Router
2	Data Link Layer	Node to node delivery of messages	Frame	Switch, Bridge
1	Physical Layer	Establishing physical connection between devices	Bits	Hub, Repeater, Modem, Cables

OSI model summarized (table form)

OSI vs. TCP/IP Reference Model

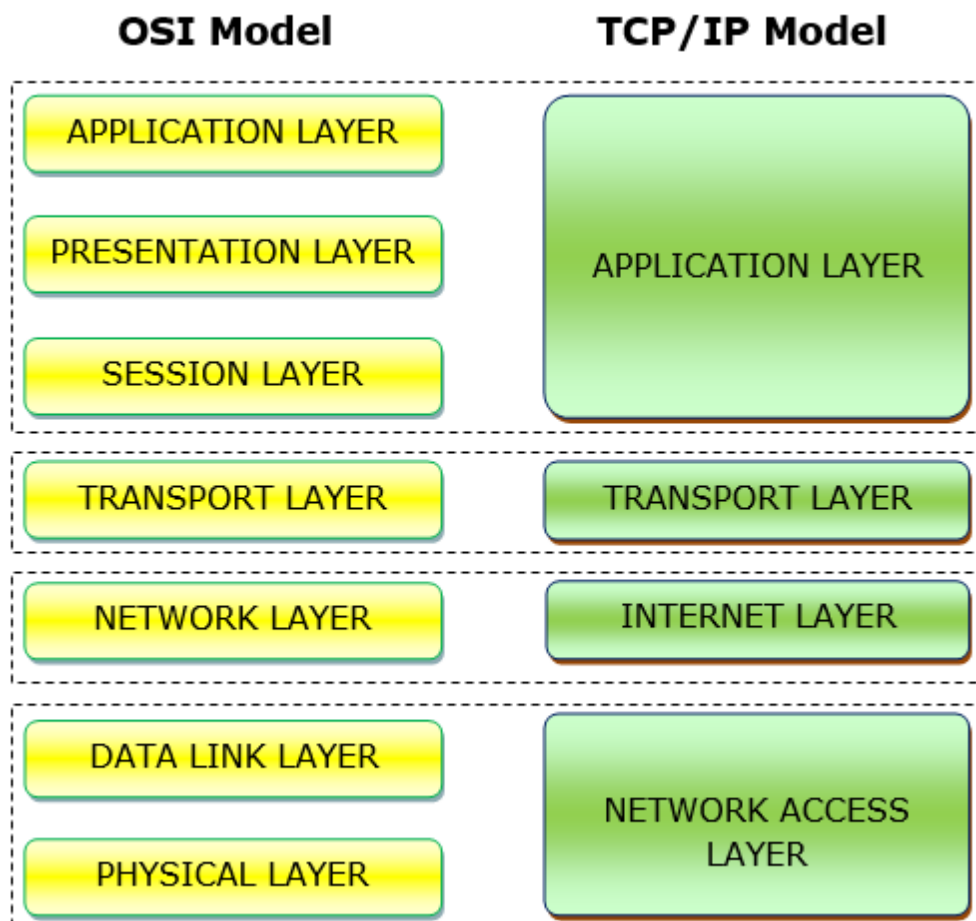
Similarities between OSI and TCP / IP Reference Models

- Both the reference models are based upon layered architecture.
- The layers in the models are compared with each other. The physical layer and the data link layer of the OSI model correspond to the link layer of the TCP/IP model. The network layers and the transport layers are the same in both the models. The session layer, the presentation layer and the application layer of the OSI model together form the application layer of the TCP/IP model.
- In both the models, protocols are defined in a layer-wise manner.
- In both models, data is divided into packets and each packet may take the individual route from the source to the destination.

Differences between OSI and TCP / IP Reference Models

- OSI model is a generic model that is based upon functionalities of each layer. TCP/IP model is a protocol-oriented standard.
- OSI model distinguishes the three concepts, namely, services, interfaces, and protocols. TCP/IP does not have a clear distinction between these three.
- OSI model gives guidelines on how communication needs to be done, while TCP/IP protocols layout standards on which the Internet was developed. So, TCP/IP is a more practical model.
- In OSI, the model was developed first and then the protocols in each layer were developed. In the TCP/IP suite, the protocols were developed first and then the model was developed.
- The OSI has seven layers while the TCP/IP has four layers.

The following diagram shows the corresponding layers of OSI and TCP/IP models



N.B.OSI model acts as a reference model and is not implemented on the Internet because of its late invention. The current model being used is the **TCP/IP model**. TCP/IP is the communication protocol suite that connects network devices to the Internet, while OSI is a reference model that outlines the functions of a networking system.

HTML - Overview

HTML stands for **Hyper Text Markup Language**, which is the most widely used language on Web to develop web pages. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML stands for **Hypertext Markup Language**, and it is the most widely used language to write Web Pages.

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

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

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

Why to Learn HTML?

Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers. Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

Some of the key advantages of learning HTML:

- **Create Web site –**
You can create a website or customize an existing web template if you know HTML well.
- **Become a web designer –**
If you want to start a career as a professional web designer, HTML and CSS designing is a must skill.
- **Understand web –**
If you want to optimize your website, to boost its speed and performance, it is good to know HTML to yield best results.
- **Learn other languages –**
Once you understand the basic of HTML then other related technologies like javascript, php, or angular are become easier to understand.

Hello World using HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>CIME,Bhubaneswar</title>
  </head>
  <body>
    <h1>MCA 4th Semester</h1>
    <p> Internet and Web Programming(MCA02005)</p>
  </body>
</html>
```

Applications of HTML

- **Web pages development –**

HTML is used to create pages which are rendered over the web. Almost every page of web is having html tags in it to render its details in browser.

- **Internet Navigation –**

HTML provides tags which are used to navigate from one page to another and is heavily used in internet navigation.

- **Responsive UI –**

HTML pages now-a-days works well on all platform, mobile, tabs, desktop or laptops owing to responsive design strategy.

- **Offline support**

HTML pages once loaded can be made available offline on the machine without any need of internet.

- **Game development-**

HTML5 has native support for rich experience and is now useful in gaming development arena as well.

Basic HTML Document

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

```
<!DOCTYPE html>
<html>

  <head>
    <title>This is document title</title>
  </head>

  <body>
    <h1>This is a heading</h1>
    <p>Document content goes here.....</p>
  </body>

</html>
```

HTML Tags

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

Above example of HTML document uses the following tags –

Sr.No	Tag & Description
1	<!DOCTYPE...> This tag defines the document type and HTML version.
2	<html> This tag encloses the complete HTML document and mainly comprises of document header which is represented by <head>...</head> and document body which is represented by <body>...</body> tags.
3	<head> This tag represents the document's header which can keep other HTML tags like <title> , <link> etc.
4	<title> The <title> tag is used inside the <head> tag to mention the document title.
5	<body> This tag represents the document's body which keeps other HTML tags like <h1> , <div> , <p> etc.
6	<h1> This tag represents the heading.
7	<p> This tag represents a paragraph.

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

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

HTML Document Structure

A typical HTML document will have the following structure –

```
<html>

    <head>
        Document header related tags
    </head>

    <body>
        Document body related tags
    </body>

</html>
```

The <!DOCTYPE> Declaration

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

```
<!DOCTYPE html>
```

There are many other declaration types which can be used in HTML document depending on what version of HTML is being used.

HTML Elements

An HTML element is defined by a start tag, some content, and an end tag.

HTML Elements

The HTML **element** is everything from the start tag to the end tag:

<tagname>Content goes here...</tagname>

Examples of some HTML elements:

<h1>My First Heading</h1>

<p>My first paragraph.</p>

Start tag	Element content	End tag
<h1>	My First Heading	</h1>
<p>	My first paragraph.	</p>
 	<i>none</i>	<i>none</i>

Nested HTML Elements

HTML elements can be nested (this means that elements can contain other elements).

All HTML documents consist of nested HTML elements.

The following example contains four HTML elements (<html>, <body>, <h1> and <p>):

Example

<!DOCTYPE html>

<html>

<body>

<h1>My First Heading</h1>

<p>My first paragraph. </p>

</body>

</html>

O/P

My First Heading

My first paragraph.

Example Explained

The `<html>` element is the root element and it defines the whole HTML document.

It has a start tag `<html>` and an end tag `</html>`.

Then, inside the `<html>` element there is a `<body>` element:

```
<body>
<h1>My First Heading</h1>
<p>My first paragraph.</p>
</body>
```

The `<body>` element defines the document's body.

It has a start tag `<body>` and an end tag `</body>`.

Then, inside the `<body>` element there are two other elements: `<h1>` and `<p>`:

```
<h1>My First Heading</h1>
<p>My first paragraph.</p>
```

The `<h1>` element defines a heading.

It has a start tag `<h1>` and an end tag `</h1>`:

```
<h1>My First Heading</h1>
```

The `<p>` element defines a paragraph.

It has a start tag `<p>` and an end tag `</p>`:

```
<p>My first paragraph.</p>
```

Never Skip the End Tag

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

Example

```
<html>
<body>
<p>This is a paragraph
<p>This is a paragraph
</body>
</html>
```

O/P

This is a paragraph.

This is a paragraph.

However, never rely on this! Unexpected results and errors may occur if you forget the end tag!

Empty HTML Elements

HTML elements with no content are called empty elements.

The `
` tag defines a line break, and is an empty element without a closing tag:

Example

```
<p>This is a <br> paragraph with a line break.</p>
```

O/P

**This is a
paragraph with a line break.**

HTML is Not Case Sensitive

HTML tags are not case sensitive: `<P>` means the same as `<p>`.

The HTML standard does not require lowercase tags, it is recommended lowercase in HTML, and demands lowercase for stricter document types like XHTML.

HTML Tag Reference

Tag	Description
<code><html></code>	Defines the root of an HTML document
<code><body></code>	Defines the document's body
<code><h1></code> to <code><h6></code>	Defines HTML headings

HTML Attributes

HTML attributes provide additional information about HTML elements.

HTML Attributes

All HTML elements can have attributes

Attributes provide additional information about elements

Attributes are always specified in the start tag

Attributes usually come in name/value pairs like: `name="value"`

The href Attribute

The `<a>` tag defines a hyperlink. The href attribute specifies the URL of the page the link goes to.

Example-

```
<html>
<body>
<h1>The href Attribute</h1>
<a href="https://www.bput.ac.in">Visit BPUT</a>
</body>
</html>
```

(hypertext reference) HTML code used to create a link to another page.

The src Attribute

The tag is used to embed an image in an HTML page. The **src** attribute specifies the path to the image to be displayed:

Example

```

```

Ex

```
<!DOCTYPE html>
<html>
<body>
<h2>The src Attribute</h2>
<p>HTML images are defined with the img tag, and the filename of the image
source is specified in the src attribute:</p>

</body>
</html>
```

The alt Attribute

The required alt attribute for the tag specifies an alternate text for an image, if the image for some reason cannot be displayed. This can be due to slow connection, or an error in the src attribute, or if the user uses a screen reader.

Example

```

```

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

<p>If we try to display an image that does not exist, the value of the alt attribute
will be displayed instead. </p>
</body>
</html>
```

The style Attribute

The style attribute is used to add styles to an element, such as color, font, size, and more.

Example

```
<p style="color:red;">This is a red paragraph.</p>
<!DOCTYPE html>
<html>
<body>
<h2>The style Attribute</h2>
<p>The style attribute is used to add styles to an element, such as color:</p>
<p style="color:red;">This is a red paragraph.</p>
</body>
</html>
```

O/P

The style Attribute

The style attribute is used to add styles to an element, such as color:

This is a red paragraph.

The lang Attribute

You should always include the **lang** attribute inside the <html> tag, to declare the language of the Web page. This is meant to assist search engines and browsers.

The following example specifies English as the language:

```
<!DOCTYPE html>
<html lang="en">
<body>
...
</body>
</html>
```

Country codes can also be added to the language code in the lang attribute. So, the first two characters define the language of the HTML page, and the last two characters define the country.

The following example specifies English as the language and United States as the country:

```
<!DOCTYPE html>
<html lang="en-US">
<body>
...
```

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

The title Attribute

The title attribute defines some extra information about an element.

The value of the title attribute will be displayed as a tooltip when you mouse over the element:

Ex

```
<!DOCTYPE html>
<html>
<body>
<h2 title="I'm a header">The title Attribute</h2>
<p title="I'm a tooltip">Mouse over this paragraph, to display the title attribute as a
tooltip.</p>
</body>
</html>
```

O/P

The title Attribute

Mouse over this paragraph, to display the title attribute as a tooltip.

<a>: The Anchor element

The HTML anchor tag defines a hyperlink that links one page to another page. It can create hyperlink to other web page as well as files, location, or any URL. The "href" attribute is the most important attribute of the HTML a tag. and which links to destination page or URL.

The <a> HTML element (or anchor element), with its href attribute, creates a hyperlink to web pages, files, email addresses, locations in the same page, or anything else a URL can address.

Content within each <a> should indicate the link's destination.

If the href attribute is present, pressing the enter key while focused on the <a> element will activate it.

The <a> tag defines a hyperlink, which is used to link from one page to another.

The most important attribute of the <a> element is the href attribute, which indicates the link's destination.

By default, links will appear as follows in all browsers:

- An unvisited link is underlined and blue

- A visited link is underlined and purple

- An active link is underlined and red

HTML Table

HTML table tag is used to display data in tabular form (row * column). There can be many columns in a row.

We can create a table to display data in tabular form, using `<table>` element, with the help of `<tr>`, `<td>`, and `<th>` elements.

In Each table, table row is defined by `<tr>` tag, table header is defined by `<th>`, and table data is defined by `<td>` tags.

HTML tables are used to manage the layout of the page e.g. header section, navigation bar, body content, footer section etc. But it is recommended to use div tag over table to manage the layout of the page .

HTML Table Tags

Tag	Description
<code><table></code>	It defines a table.
<code><tr></code>	It defines a row in a table.
<code><th></code>	It defines a header cell in a table.
<code><td></code>	It defines a cell in a table.
<code><caption></code>	It defines the table caption.
<code><colgroup></code>	It specifies a group of one or more columns in a table for formatting.
<code><col></code>	It is used with <code><colgroup></code> element to specify column properties for each column.
<code><tbody></code>	It is used to group the body content in a table.
<code><thead></code>	It is used to group the header content in a table.
<code><tfooter></code>	It is used to group the footer content in a table.

HTML Table Example

Let's see the example of HTML table tag. Its output is shown above.

Problem : Create a table to describe

Header name :	First_Name	Last_Name	Marks
Rows	Sonoo	Jaiswal	60
	James	William	80
	Swati	Sironi	82
	Chetna	Singh	72

Ans:

```
<html>
<body>
<table>
<tr><th>First_Name</th><th>Last_Name</th><th>Marks</th></tr>
<tr><td>Sonoo</td><td>Jaiswal</td><td>60</td></tr>
<tr><td>James</td><td>William</td><td>80</td></tr>
<tr><td>Swati</td><td>Sironi</td><td>82</td></tr>
<tr><td>Chetna</td><td>Singh</td><td>72</td></tr>
</table>
</body>
</html>
```

O/P

First_Name	Last_Name	Marks
------------	-----------	-------

Sonoo	Jaiswal	60
James	William	80
Swati	Sironi	82
Chetna	Singh	72

HTML Table with Border

HTML Border attribute

The HTML `<table>` border Attribute is used to specify the border of a table. It sets the border around the table cells.

Syntax:

`<table border="1|0">`

Attribute Values:

1: It sets the border around the table cells.

0: It removes (not set) the border around the table cells.

Problem

First_Name	Last_Name	Marks
Sonoo	Jaiswal	60
James	William	80
Swati	Sironi	82
Chetna	Singh	72

Answer

```
<html>
<body>
<table border="1">
<tr><th>First_Name</th><th>Last_Name</th><th>Marks</th></tr>
<tr><td>Sonoo</td><td>Jaiswal</td><td>60</td></tr>
<tr><td>James</td><td>William</td><td>80</td></tr>
<tr><td>Swati</td><td>Sironi</td><td>82</td></tr>
<tr><td>Chetna</td><td>Singh</td><td>72</td></tr>
</table>
</body>
</html>
```

O/P:

First_Name	Last_Name	Marks
Sonoo	Jaiswal	60
James	William	80
Swati	Sironi	82
Chetna	Singh	72

HTML Table Sizes

HTML Table Width

To set the width of a table, add the style attribute to the <table> element:

Example

Set the width of the table to 100%:

```
<html>
<body>
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
</body>
</html>
```


Example

Set the width of the first column to 70%:

```
<table style="width:100%">
  <tr>
    <th style="width:70%">Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

HTML Table Row Height

To set the height of a specific row, add the style attribute on a table row element:

Example Set the height of the second row to 200 pixels:

```
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr style="height:200px">
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

HTML Table - Colspan

To make a cell span over multiple columns, use the colspan attribute:

Example

```
<table>
  <tr>
    <th colspan="2">Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>43</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>57</td>
  </tr>
</table>
```

Note: The value of the colspan attribute represents the number of columns to span.

HTML Table - Rowspan

To make a cell span over multiple rows, use the rowspan attribute:

Example

```
<table>
  <tr>
    <th>Name</th>
    <td>Jill</td>
  </tr>
  <tr>
    <th rowspan="2">Phone</th>
    <td>555-1234</td>
  </tr>
  <tr>
    <td>555-8745</td>
  </tr>
</table>
```

HTML Form

An **HTML form** is a section of a document which contains controls such as text fields, password fields, checkboxes, radio buttons, submit button, menus etc.

An HTML form facilitates the user to enter data that is to be sent to the server for processing such as name, email address, password, phone number, etc.

Why use HTML Form

HTML forms are required if you want to collect some data from of the site visitor.

For example: If a user want to purchase some items on internet, he/she must fill the form such as shipping address and cred7t/debit card details so that item can be sent to the given address.

HTML Form Syntax

```
<form action="server url" method="get|post">
```

```
//input controls e.g. textfield, textarea, radiobutton, button
```

```
</form>
```

HTML Form Tags

Let's see the list of HTML 5 form tags.

Tag	Description
<form>	It defines an HTML form to enter inputs by the used side.
<input>	It defines an input control.
<textarea>	It defines a multi-line input control.
<label>	It defines a label for an input element.
<fieldset>	It groups the related element in a form.
<legend>	It defines a caption for a <fieldset> element.
<select>	It defines a drop-down list.
<optgroup>	It defines a group of related options in a drop-down list.
<option>	It defines an option in a drop-down list.
<button>	It defines a clickable button.

<datalist>	It specifies a list of pre-defined options for input control.
<keygen>	It defines a key-pair generator field for forms.
<output>	It defines the result of a calculation.

HTML <form> element

The HTML <form> element provide a document section to take input from user. It provides various interactive controls for submitting information to web server such as text field, text area, password field, etc.

The <form> element does not itself create a form but it is container to contain all required form elements, such as <input>, <label>, etc.

Syntax:

```
<form>
//Form elements
</form>
```

Get vs. Post

There are many differences between the Get and Post request. Let's see these differences:

GET	POST
1) In case of Get request, only limited amount of data can be sent because data is sent in header.	In case of post request, large amount of data can be sent because data is sent in body.
2) Get request is not secured because data is exposed in URL bar.	Post request is secured because data is not exposed in URL bar.
3) Get request can be bookmarked.	Post request cannot be bookmarked.
4) Get request is idempotent . It means second request will be ignored until response of first request is delivered	Post request is non-idempotent.
5) Get request is more efficient and used more than Post.	Post request is less efficient and used less than get.

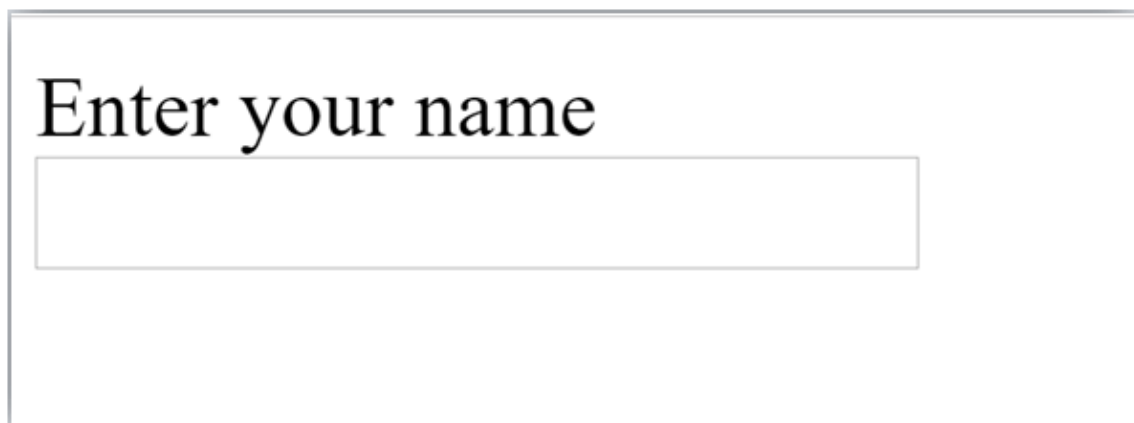
HTML <input> element

The HTML <input> element is fundamental form element. It is used to create form fields, to take input from user. We can apply different input field to gather different information from user. Following is the example to show the simple text input.

Example:

```
<html>
<body>
  <form>
    Enter your name <br>
    <input type="text" name="username">
  </form>
</body>
</html>
```

Output:

A screenshot of a web browser displaying the output of the provided HTML code. It shows a rectangular form container. Inside the container, the text "Enter your name" is displayed in a large, black, serif font. Below the text, there is a single-line text input field with a light gray border and a white background.

HTML TextField Control

The type="text" attribute of input tag creates textfield control also known as single line textfield control. The name attribute is optional, but it is required for the server side component such as JSP, ASP, PHP etc.

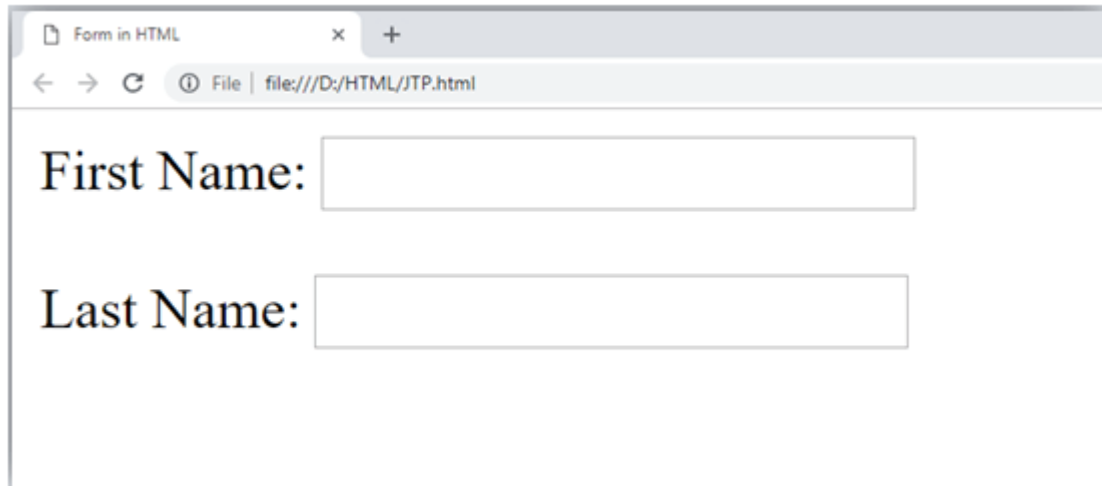
<form>

First Name: <input type="text" name="firstname"/>

Last Name: <input type="text" name="lastname"/>

</form>

Output:

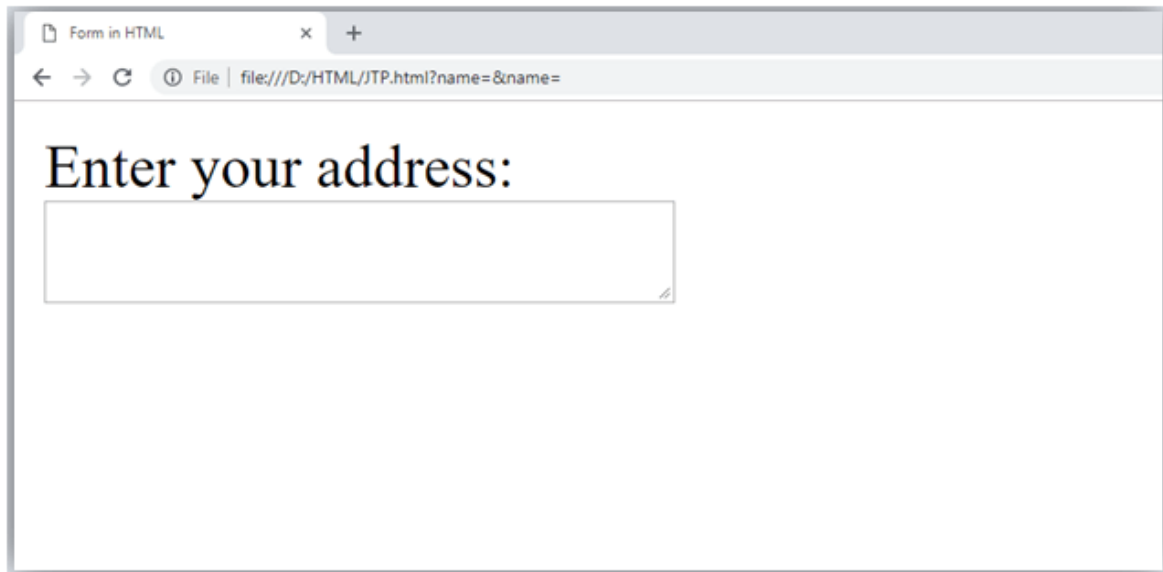
A screenshot of a web browser window. The title bar says "Form in HTML". The address bar shows "File | file:///D:/HTML/JTP.html". The main content area contains two text input fields. The first field is preceded by the label "First Name:" and the second field is preceded by the label "Last Name:". Both labels are in a serif font.

HTML <textarea> tag in form

The <textarea> tag in HTML is used to insert multiple-line text in a form. The size of <textarea> can be specify either using "rows" or "cols" attribute or by CSS.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>Form in HTML</title>
</head>
<body>
  <form>
    Enter your address:<br>
    <textarea rows="2" cols="20"></textarea>
  </form>
</body>
</html>
```



Label Tag in Form

It is considered better to have label in form. As it makes the code parser/browser/user friendly.

If you click on the label tag, it will focus on the text control. To do so, you need to have for attribute in label tag that must be same as id attribute of input tag.

```
<form>
  <label for="firstname">First Name: </label> <br/>
    <input type="text" id="firstname" n
="firstname"/> <br/>
  <label for="lastname">Last Name: </label>
    <input type="text" id="lastname" name="lastname"/> <br/>
</form>
```

Output:

First Name:

Last Name:

HTML Password Field Control

The password is not visible to the user in password field control.

```
<form>  
  <label for="password">Password: </label>  
    <input type="password" id="password" name="password"/> <br/>  
</form>
```

Output:

Password:

HTML 5 Email Field Control

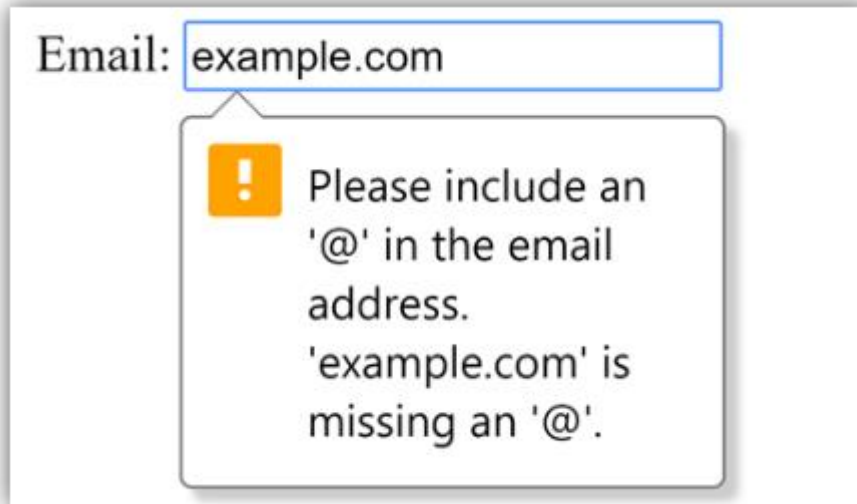
The email field is new in HTML 5. It validates the text for correct email address. You must use @ and . in this field.

```
<form>  
  <label for="email">Email: </label>  
    <input type="email" id="email" name="email"/> <br/>  
</form>
```

It will display in browser like below:

Email:

Note: If we will not enter the correct email, it will display error like:



Radio Button Control

The radio button is used to select one option from multiple options. It is used for selection of gender, quiz questions etc.

If you use one name for all the radio buttons, only one radio button can be selected at a time.

Using radio buttons for multiple options, you can only choose a single option at a time.

```
<form>
<!DOCTYPE html>
<html>
<body>
<h2>Radio Buttons</h2>
<p>Choose your favorite Web language:</p>
</form>
```

```

<input type="radio" id="html" name="fav_language1" value="HTML">
<label for="html">HTML</label><br>
<input type="radio" id="css" name="fav_language2" value="CSS">
<label for="css">CSS</label><br>
<input type="radio" id="javascript" name="fav_language3" value="JavaScript">
<label for="javascript">JavaScript</label>
</form>
</body>
</html>
</form>

```

Output:

Radio Buttons

Choose your favorite Web language:

- ☐ HTML
- ☐ CSS
- ☐ JavaScript

Checkbox Control

The checkbox control is used to check multiple options from given checkboxes.

```

<form>
Hobby:<br>
    <input type="checkbox" id="cricket" name="cricket" value="cricket"/>
    <label for="cricket">Cricket</label> <br>
    <input type="checkbox" id="football" name="football" value="football"/>
    <label for="football">Football</label> <br>
    <input type="checkbox" id="hockey" name="hockey" value="hockey"/>
    <label for="hockey">Hockey</label>
</form>

```

Note: These are similar to radio button except it can choose multiple options at a time and radio button can select one button at a time, and its display.

Output:

Hobby:

☒ Cricket

☒ Football

☐ Hockey

The <select> Element

The <select> element defines a drop-down list. The <option> elements defines an option that can be selected.

By default, the first item in the drop-down list is selected.

```
<form>
  <label for="cars">Choose a car:</label>
  <select id="cars" name="cars">
    <option value="volvo">Volvo</ option>
    <option value="saab">Saab</option>
    <option value="fiat">Fiat</option>
    <option value="audi">Audi</option>
  </select>
</form>
```

Visible Values:

Use the size attribute to specify the number of visible values.

Syntax: <select id="cars" name="cars" size="3">

Allow Multiple Selections:

Use the multiple attribute to allow the user to select more than one value.

Syntax:<select id="cars" name="cars" size="4" multiple>

Submit button control

HTML <input type="submit"> are used to add a submit button on web page. When user clicks on submit button, then form get submit to the server.

Syntax:

```
<input type="submit" value="submit">
```

The type = submit , specifying that it is a submit button

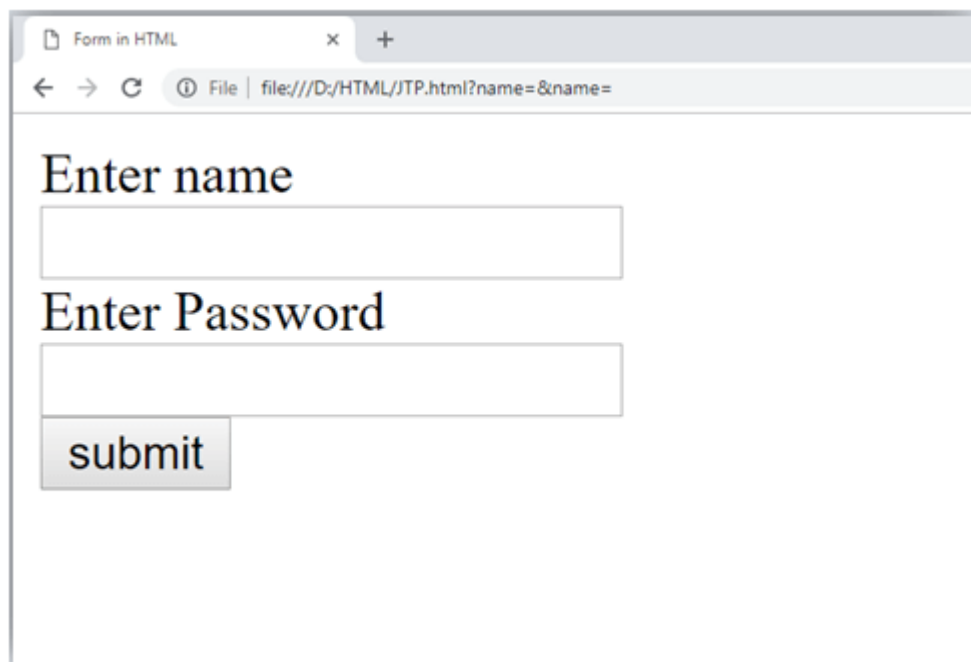
The value attribute can be anything which we write on button on web page.

The name attribute can be omit here.

Example:

```
<form>
  <label for="name">Enter name</label><br>
  <input type="text" id="name" name="name"><br>
  <label for="pass">Enter Password</label><br>
  <input type="Password" id="pass" name="pass"><br>
  <input type="submit" value="submit">
</form>
```

Output:



HTML <fieldset> element:

The <fieldset> element in HTML is used to group the related information of a form. This element is used with <legend> element which provide caption for the grouped elements.

Example:

```
<form>
  <fieldset>
    <legend>User Information:</legend>
    <label for="name">Enter name</label><br>
    <input type="text" id="name" name="name"><br>
    <label for="pass">Enter Password</label><br>
    <input type="Password" id="pass" name="pass"><br>
    <input type="submit" value="submit">
  </fieldset>
</form>
```

Output:



HTML Form Example1

The form element

First name:

Last name:

Click the "Submit" button and the form-data will be sent to a page on the server called "action_page.php".

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

<h1>The form element</h1>

```
<form action="/action_page.php">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="submit" value="Submit">
</form>
```

<p>Click the "Submit" button and the form-data will be sent to a page on the server called "action_page.php".</p>

</body>

</html>

HTML Form Example2

Registration form

User personal information

Enter your full name

Enter your email

Enter your password

confirm your password

Enter your gender

☐ Male

☐ Female

☐ others

Enter your Address:

<!DOCTYPE html>

<html>

<head>

<title>Form in HTML</title>

</head>

<body>

```

<h2>Registration form</h2>
<form>
  <fieldset>
    <legend>User personal information</legend>
    <label>Enter your full name</label><br>
    <input type="text" name="name"><br>
    <label>Enter your email</label><br>
    <input type="email" name="email"><br>
    <label>Enter your password</label><br>
    <input type="password" name="pass"><br>
    <label>confirm your password</label><br>
    <input type="password" name="pass"><br>
    <br><label>Enter your gender</label><br>
    <input type="radio" id="gender" name="gender" value="male"/>Male <br>
    <input type="radio" id="gender" name="gender" value="female"/>Female
  <br/>
    <input type="radio" id="gender" name="gender" value="others"/>others
  <br/>
    <br>Enter your Address:<br>
    <textarea></textarea><br>
    <input type="submit" value="sign-up">
  </fieldset>
</form>
</body>
</html>

```

HTML Form Example3

Enter name:	<input type="text"/>
Enter password:	<input type="password"/>
Enter Email:	<input type="email"/>
Enter Gender:	<input type="radio"/> male <input type="radio"/> female
Select Country:	<input type="text" value="india"/> ▼
<input type="button" value="register"/>	

```
<!DOCTYPE>
<html>
<body>
<form action="http://www.javatpoint.com/javascriptpages/valid.jsp">
<table>
<tr>
    <td class="tdLabel"><label for="register_name" class="label">Enter
name:</label></td>
    <td><input type="text" name="name" value="" id="register_name"
style="width:160px"/></td>
</tr>
<tr>
    <td class="tdLabel"><label for="register_password" class="label">Enter
password:</label></td>
    <td><input type="password" name="password" id="register_password"
style="width:160px"/></td>
</tr>
<tr>
    <td class="tdLabel"><label for="register_email" class="label">Enter
Email:</label></td>
    <td>
<input type="email" name="email" value="" id="register_email"
style="width:160px"/></td>
</tr>
<tr>
    <td class="tdLabel"><label for="register_gender" class="label">Enter
Gender:</label></td>
    <td>
<input type="radio" name="gender" id="register_gendermale" value="male"/>
<label for="register_gendermale">male</label>
<input type="radio" name="gender" id="register_genderfemale" value="female"/>
<label for="register_genderfemale">female</label>
</td>
</tr>
<tr>
    <td class="tdLabel"><label for="register_country" class="label">Select
Country:</label></td>
    <td><select name="country" id="register_country" style="width:160px">
<option value="india">india</option>
```



```

    <option value="pakistan">pakistan</option>
    <option value="africa">africa</option>
    <option value="china">china</option>
    <option value="other">other</option>
</select>
</td>
</tr>
<tr>
    <td colspan="2"><div align="right"><input type="submit" id="register_0"
value="register"/>
</div></td>
</tr>
</table>
</form>
</body>
</html>

```

HTML <frame> Tag

Not Supported in HTML5.

The <frame> tag was used in HTML 4 to define one particular window (frame) within a <frameset>.

What to Use Instead?

Example

Use the <iframe> tag to embed another document within the current HTML document:

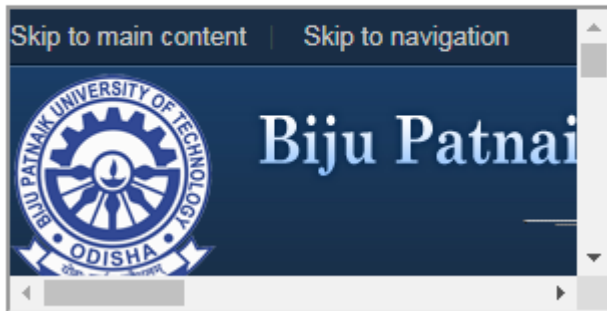
```

<!DOCTYPE html>
<html>
<body>
<h1>The iframe element</h1>
<iframe src="https://www.bput.ac.in" title="Biju
Patnaik University of Technology">
</iframe>
</body>
</html>

```

O/P

The iframe element



<div> and Tag in HTML

Both the tags (<div> and) are used to represent the part of the web page, <div> tag is used as a block part of the webpage and tag is used as an inline part of the webpage like below:

```
<div>An IT institute <span>CIME</span></div>
```

HTML <div> tag: The div tag is known as Division tag. The div tag is used in HTML to make divisions of content on the web page like (text, images, header, footer, navigation bar, etc). Div tag has both opening(<div>) and closing (</div>) tags and it is mandatory to close the tag. As we know Div tag is a block-level tag. In this example, the div tag contains the entire width. It will be displayed div tag each time on a new line, not on the same line.

- **Example:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Div tag</title>
    <style>
      div {
        color: white;
        background-color: #009900;
        margin: 2px;
        font-size: 25px;
      }
    </style>
```

```

</head>
<body>
  <div> div tag </div>
  <div> div tag </div>
  <div> div tag </div>
  <div> div tag </div>
</body>
</html>

```

Output:



TML tag:

The HTML span element is a generic inline container for inline elements and content. It used to group elements for styling purposes (by using the class or id attributes). A better way to use it when no other semantic element is available. The span tag is very similar to the div tag, but div is a block-level tag and span is an inline tag.

Example

```

<!DOCTYPE html>
<html>
<head>
  <title>span tag</title>
</head>
<body>
  <h2>Welcome To CIME</h2>
  <!-- Inside paragraph applying span tag
  with different style -->
  <p><span style="background-color:lightgreen">
    CIME</span> is An IT institute
    where you can<span style="color:blue;">
    study</span> Post Graduate Programmes <span
    style="background-color:lightblue;"> likes MCA or M.Tech(CSE)</span>

```

and build a professional career in IT field.

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

Output:

Welcome To CIME

CIME is An IT institute where you can study Post Graduate Programmes likes MCA or M.Tech(CSE) and build a professional career in IT field.

Differences between <div> and tag:

<div>	
The <div> tag is a block level element.	The tag is an inline element.
It is best to attach it to a section of a web page.	It is best to attach a CSS to a small section of a line in a web page.
It accepts align attribute.	It does not accept align attribute.
This tag should be used to wrap a section, for highlighting that section.	This tag should be used to wrap any specific word that you want to highlight in your webpage.

Module II (8 Periods)

CSS (Cascading Style Sheets)

CSS (Cascading Style Sheets) is a stylesheet language used to design the webpage to make it attractive. The reason of using CSS is to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page.

Why we learn CSS?

Styling has been an essential property for any website since many decades. It increases the standards and overall look of the website which makes it easier for the user to interact with it. A website cannot be made without CSS, as styling is **MUST** since no user would want to interact with a dull and shabby website. So for knowing Web Development, learning CSS is must.

Base for web development: HTML and CSS is the basic skill that every web developer should know. It is the basic skill that is required for building a website.

Makes your website look attractive: A website that's dull and plain will not attract the user most probably, so adding some style would surely make your website presentable to the user.

Makes the design come live: A web developer is responsible in making the design given to him as a live product. CSS is used for styling to develop the design of the website.

Increases user experience of website: A website with a simple yet beautiful UI would help the users to go through the website easily. CSS is used to make the user interface better.

More career opportunities: Since CSS is a basic requirement while learning Web Development, therefor there are abundant career opportunities for it. As a freelancer too you can land up to many projects.

Basic Format: It is the basic structure of HTML webpage and we use CSS style inside webpage. In a web page, we use internal CSS (i.e. adding CSS code inside <head> tag of HTML code).

Cascading Style Sheets, fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page.

CSS is easy to learn and understood, but it provides powerful control over the presentation of an HTML document.

WHY CSS?

CSS saves time: You can write CSS once and reuse the same sheet in multiple HTML pages.

Easy Maintenance: To make a global change simply change the style, and all elements in all the webpages will be updated automatically.

Search Engines: CSS is considered a clean coding technique, which means search engines won't have to struggle to "read" its content.

Superior styles to HTML: CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.

Offline Browsing: CSS can store web applications locally with the help of an offline cache. Using this we can view offline websites.

```
<!DOCTYPE html>
<html>

<head>
  <!-- Head section of web page -->
  <title></title>

  <!-- Stylesheet of web page -->
  <style></style>
</head>

<body>
  <!-- Body section of web page -->
</body>

</html>
```

Example:

Let's see a small example of HTML webpage with CSS styles. Here, we use CSS styles to set the alignment and text color in a webpage.

```
<!DOCTYPE html>
<html>
<head>
  <title>
    Simple HTML webpage with CSS style
  </title>

  <!-- Stylesheet of web page -->
  <style>
    body {
```

```
        text-align: center;
    }

    h1 {
        color: green;
    }
</style>
</head>
<body>
    <h1>Welcome to CIME</h1>
    <p>An IT Institute in Bhubaneswar,Odisha.</p>
</body>
</html>
O/P
```

Welcome to CIME

An IT Institute in Bhubaneswar,Odisha.

What and where to learn in 2021?

CSS is a web development technology that stands behind the look-and-feel of every web page. It is proving to be capable to do so much more with its most recent evolution. The versions of CSS just keep getting better and better with time, which basically means an improved programming platform for developers, resulting in more efficient and faster web designs.

As we make our way through 2021, the most in-demand and important visual language of the web is CSS3. It helps you to engage your website users with fast loading web pages across all browsers.

CSS3 lets you gain control over the visual appearance of your website by means of layout, layering, typography, and special effects. You can easily customize any theme for WordPress or can create the visual for your app or site with CSS3.

You can find the programming community-recommended best CSS3 tutorials [here](#).

CSS 1

The first CSS specification to become an official W3C Recommendation is CSS level 1, published in December 1996. Among its capabilities are support for

- Font properties such as typeface and emphasis
- Color of text, backgrounds, and other elements
- Text attributes such as spacing between words, letters, and lines of text
- Alignment of text, images, tables and other elements
- Margin, border, padding, and positioning for most elements
- Unique identification and generic classification of groups of attributes
- The W3C no longer maintains the CSS 1 Recommendation.

CSS 2

- CSS level 2 specification was developed by the W3C and published as a Recommendation in May 1998.
- A superset of CSS 1, CSS 2 includes a number of new capabilities like absolute, relative, and fixed positioning of elements and z-index, the concept of media types, support for aural style sheets and bidirectional text, and new font properties such as shadows.
- The W3C no longer maintains the CSS 2 recommendation.
- CSS level 2 revision 1 or CSS 2.1 fixes errors in CSS 2, removes poorly-supported or not fully interoperable features and adds already-implemented browser extensions to the specification.

CSS 3

- Instead of defining all features in a single, large specification like CSS 2, CSS 3 is divided into several separate documents called "modules".
- Due to the modularization, different modules have different stability and are in different status. As of March 2011, there are over 40 CSS modules published from the CSS Working Group. Some modules such as Selectors, Namespaces, Color and Media Queries are considered stable and are either in Candidate Recommendation or Proposed Recommendation status.

W3.CSS Versions

- W3.CSS was released in 2017.
- The latest version is W3.CSS 4.15 December 2020.

Conclusion

CSS3 is the latest version of CSS. It is only compatible with IE9 and not with older versions of browsers. The more you code, the more you will learn about CSS3 but there is one thing to note -- you cannot master CSS3 unless you know about CSS. CSS3 took the properties of CSS and developed them to include new features to

provide ease of use to the designers. CSS3 is capable of supporting responsive designs and can also handle media queries as compared to CSS, which does not support responsive design and is not capable of handling media queries. CSS3 is very important for web designers because it provides a vast range of options and it helps in creating more enhanced opportunities for designing a webpage. Through web designing, marketers can increase their product awareness in the market easily.

While CSS and CSS2 had ‘simple selectors’, CSS3 calls the components as ‘a sequence of simple selectors’.

CSS3 came up with some key web design considerations like rounded borders that help in rounding up the borders without any hassle. This turned out to be a huge plus point for developers who were struggling with initial versions of CSS borders. CSS3 has the capability to split text sections into multiple columns so that it can be read like a newspaper. In CSS2, the developers had difficulty because the standard was not equipped with automatically breaking the text so that it fits within a box.

Types of CSS Styles

There are three types of CSS which are given below:

- Inline CSS
- Internal or Embedded CSS
- External CSS

There are some distinct ways of implementing the CSS code. These are known as CSS styles. Three types of CSS styles are available; inline CSS, external CSS, and internal CSS.

Although each of the CSS styles serves the same goal i.e. styling HTML code, how it is done is different for each. We will explore each of the CSS styles in the following section:

Internal CSS

Also known as embedded CSS, internal CSS refers to the practice of adding the CSS code to the HTML document pertaining to the web page where we wish to add the CSS styling.

For adding internal CSS, one needs to add the <style> tag inside the <head> section of the HTML file. Internal CSS is extremely useful for styling a single web page.

How to Use Internal CSS

- **Step #01** - Open the HTML file and go to the <head> tag.
- **Step #02** - Add <style type="text/css"> here.
- **Step #03** - Now, add the CSS code starting from the following line.
- **Step #04** - Close the <style> tag using </style>.
- **Step #05** - Save the HTML file for the changes to take effect.

Pros

- No need to upload multiple files as the CSS code is added to the same HTML file corresponding to the web page.
- Class and ID selectors can be used.

Cons

- Adding CSS code to the HTML file results in increasing the page size and therefore, reducing the loading speed.
- Using it for multiple web pages is ineffective as it is required to add the same CSS rules for every web page.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
    h1 {
        color: red;
        margin-left: 40px;
    }
</style>
</head>
<body>
<h1>This is a heading</h1>
</body>
</html>
```

Output

This is a heading

External CSS

To qualify for the external CSS style, a web page is required to be linked to an external file containing the CSS code. External CSS is a super-effective CSS styling method when developing a big website.

One can create the external .css file using some text editor, such as Notepad and Rapid CSS Editor. Here, the CSS code resides in a document other than the one containing the HTML code for the concerned web page, hence the name.

All minor and major changes for a website leveraging external CSS can be made merely by editing the single external .css file.

How to Use External CSS

- **Step #01** - Open a text editor and create a new file. Add the CSS code here that you wish to apply to the HTML web page(s).
- **Step #02** - Save the file as .css file and exit.
- **Step #03** - Open the HTML document where you wish to apply the CSS code.
- **Step #04** - Navigate to the <head> section in the HTML file and insert a reference to the external CSS file just after the <title> tag.
- **Step #05** - Save the HTML file.

Pros

- A single external CSS file can be used for styling several web pages.
- HTML files leveraging external CSS have a cleaner structure and are smaller in size.

Cons

- Linking to or uploading several external CSS files might decrease a website's download speed and affect its performance.
- Web pages requiring the external CSS file might not be rendered accurately until the same is fully loaded.

Example

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="mystyle.css">
</head>
<body>
<h1>CIME,BHUBANESWAR</h1>
</body>
</html>
```

Output



mystyle.css

```
body {
    background-color: lightblue;
}
h1 {
    color: navy;
    margin-left: 20px;
}
```

N.B: Do not add a space between the property value and the unit.

Incorrect (space): margin-left: 20 px;

Correct (nospace): margin-left: 20px;

```
<head>
  <link type = "text/css" href = "mystyle.css" media = " all" />
</head>
```

Imported CSS - @import Rule

@import is used to import an external stylesheet in a manner similar to the <link> element. Here is the generic syntax of @import rule.

```
<head>
  @import "URL";
</head>
```

Here URL is the URL of the style sheet file having style rules. You can use another syntax as well –

```
<head>
  @import url("URL");
</head>
```

Example

Following is the example showing you how to import a style sheet file into HTML document –

```
<head>
  @import "mystyle.css";
</head>
```

Inline CSS

- The inline CSS style is used for styling a particular HTML element and not the entire HTML code. For implementing inline CSS, one needs to add the style attribute to every HTML tag that requires styling.
- Selectors aren't used here.
- Maintaining a website only by using inline CSS is impractical. This is so because following the inline CSS styling every HTML tag must be styled separately. Hence, using it isn't recommended.
- Inline CSS, however, is quite useful in some particular scenarios. For instance, situations in which:
 - ✓ The CSS style must be applied only to one element, or when access to CSS files isn't available.
 - ✓ This type of CSS styling is used mainly for previewing/testing changes as well as applying quick fixes to a web page/website.

How to Use Inline CSS

- **Step #01** - Open the HTML file where you need to add the inline CSS.
- **Step #02** - Now, navigate to the element(s) where you want to insert the inline CSS.
- **Step #03** - Add style="code" to the tag(s) you wish to use the inline CSS at. Here, code is the CSS code that you need to add. For example, if we wish to add inline CSS to <h1> tag, it will look something like this:

<h1 style="code">

Pros

- Allows instantly inserting CSS code to any HTML file.
- There is no need for creating and uploading a separate file for adding the CSS code.

Cons

- Adding CSS code to each and every HTML element wastes time.
- Styling several elements affects page size and download speed.
- Too much inline CSS can result in a messy HTML structure.

Example

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color:blue;text-align:center;">CIME,BHUBANESWAR</h1>
</body>
</html>
```

Output

CIME,BHUBANESWAR

What if I Use All the 3 CSS Styles in a Single Web Page?

It is possible to use each of the 3 CSS styles in one web page. What will result, however, is that the inline CSS style will override the other two CSS styles i.e. inline CSS code will take effect and not the other two. The priority for the 3 types of CSS styles follows this particular order:

inline CSS > internal CSS > external CSS

As you can use, when internal and external CSS styles are used for a web page, the internal CSS style will override the external CSS style.

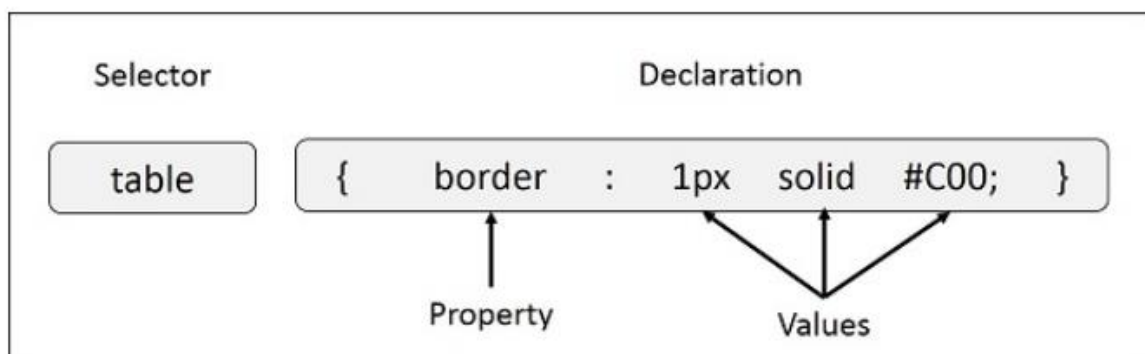
CSS - Syntax

A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts.

You can put CSS Style Rule Syntax as follows –

selector { property: value }

- **Selector** – A selector is an HTML tag at which a style will be applied. This could be any tag like <h1> or <table> etc.
- **Property** – A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be color, border etc.
- **Value** – Values are assigned to properties. For example, color property can have value either red or #F1F1F1 etc.



Example – You can define a table border as follows –

```
table{ border :1px solid #C00;}
```

Here table is a selector border is a property given value 1px solid #C00 is the value of that property.

Example- To give colour to level 1 heading

```
h1 {color : #36CFFF;}
```

Example-

Multiple Style Rules

You may need to define multiple style rules for a single element. You can define these rules to combine multiple properties and corresponding values into a single block as defined in the following example –

Example-

```
h1 {  
  color: #36C;  
  font-weight: normal;  
  letter-spacing: .4em;  
  margin-bottom: 1em;  
  text-transform: lowercase;  
}
```

Here all the property and value pairs are separated by a semicolon (;). You can keep them in a single line or multiple lines. For better readability, we keep them in separate lines.

Grouping Selectors

You can apply a style to many selectors if you like. Just separate the selectors with a comma.

Example-

```
h1, h2, h3 {  
    color: #36C;  
    font-weight: normal;  
    letter-spacing: .4em;  
    margin-bottom: 1em;  
    text-transform: lowercase;  
}
```

This define style rule will be applicable to h1, h2 and h3 element as well. The order of the list is irrelevant. All the elements in the selector will have the corresponding declarations applied to them.

CSS Rules Overriding

We have discussed four ways to include style sheet rules in a an HTML document. Here is the rule to override any Style Sheet Rule.

- Any inline style sheet takes highest priority. So, it will override any rule defined in <style>...</style> tags or rules defined in any external style sheet file.
- Any rule defined in <style>...</style> tags will override rules defined in any external style sheet file.
- Any rule defined in external style sheet file takes lowest priority, and rules defined in this file will be applied only when above two rules are not applicable.

Handling old Browsers

There are still many old browsers who do not support CSS. So, we should take care while writing our Embedded CSS in an HTML document. The following snippet shows how you can use comment tags to hide CSS from older browsers –

```
<style type = "text/css">  
  <!--  
    body, td {  
      color: blue;  
    }  
  -->  
</style>
```

CSS Comments

- Many times, you may need to put additional comments in your style sheet blocks. So, it is very easy to comment any part in style sheet.
- You can simply put your comments inside `/*.....this is a comment in style sheet.....*/`.
- You can use `/**/` to comment multi-line blocks in similar way you do in C and C++ programming languages.

Example

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      p {
        color: red;
        /* This is a single-line comment */
        text-align: center;
      }
      /* This is a multi-line comment */
    </style>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</html>
```

Output

Hello World!

CSS - Colors

CSS uses color values to specify a color. Typically, these are used to set a color either for the foreground of an element (i.e., its text) or else for the background of the element. They can also be used to affect the color of borders and other decorative effects.

You can specify your color values in various formats. Following table lists all the possible formats –

Format	Syntax	Example
Hex Code	#RRGGBB	p{color:#FF0000;}
Short Hex Code	#RGB	p{color:#6A7;}
RGB %	rgb(rrr%,ggg%,bbb%)	p{color:rgb(50%,50%,50%);}
RGB Absolute	rgb(rrr,ggg,bbb)	p{color:rgb(0,0,255);}
keyword	aqua, black, etc.	p{color:teal;}

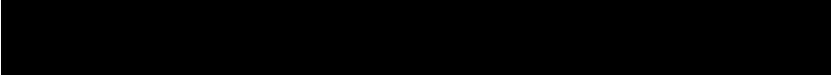







These formats are explained in more detail in the following sections –

CSS Colors - Hex Codes

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

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Jasc Paintshop Pro, or even using Advanced Paint Brush.

Each hexadecimal code will be preceded by a pound or hash sign '#'. Following are the examples to use Hexadecimal notation.

Color	Color HEX
	#000000
	#FF0000
	#00FF00
	#0000FF
	#FFFF00
	#00FFFF
	#FF00FF
	#C0C0C0

	#FFFFFF
--	---------

CSS Colors - Short Hex Codes

This is a shorter form of the six-digit notation. In this format, each digit is replicated to arrive at an equivalent six-digit value. For example: #6A7 becomes #66AA77.

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Jasc Paintshop Pro, or even using Advanced Paint Brush.

Each hexadecimal code will be preceded by a pound or hash sign '#'. Following are the examples to use Hexadecimal notation.

Color	Color HEX
	#000
	#F00
	#0F0
	#0FF
	#FF0
	#0FF
	#F0F
	#FFF

CSS Colors - RGB Values

CSS Colors - RGB Values

This color value is specified using the **rgb()** property. This property takes three values, one each for red, green, and blue. The value can be an integer between 0 and 255 or a percentage.

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

Following is the example to show few colors using RGB values.

Color	Color RGB
	rgb(0,0,0)
	rgb(255,0,0)
	rgb(0,255,0)
	rgb(0,0,255)
	rgb(255,255,0)

	rgb(0,255,255)
	rgb(255,0,255)
	rgb(192,192,192)
	rgb(255,255,255)

Building Color Codes

You can build millions of color codes using our Color Code Builder. Check our **HTML Color Code Builder** (<https://htmlcolorcodes.com/>). To use this tool, you would need a Java Enabled Browser.

Browser Safe Colors

Here is the list of 216 colors which are supposed to be most safe and computer independent colors. These colors vary from hexa code 000000 to FFFFFFFF. These colors are safe to use because they ensure that all computers would display the colors correctly when running a 256 color palette –

000000	000033	000066	000099	0000CC	0000FF
003300	003333	003366	003399	0033CC	0033FF
006600	006633	006666	006699	0066CC	0066FF
009900	009933	009966	009999	0099CC	0099FF
00CC00	00CC33	00CC66	00CC99	00CCCC	00CCFF
00FF00	00FF33	00FF66	00FF99	00FFCC	00FFFF
330000	330033	330066	330099	3300CC	3300FF
333300	333333	333366	333399	3333CC	3333FF
336600	336633	336666	336699	3366CC	3366FF
339900	339933	339966	339999	3399CC	3399FF
33CC00	33CC33	33CC66	33CC99	33CCCC	33CCFF
33FF00	33FF33	33FF66	33FF99	33FFCC	33FFFF
660000	660033	660066	660099	6600CC	6600FF
663300	663333	663366	663399	6633CC	6633FF
666600	666633	666666	666699	6666CC	6666FF

669900	669933	669966	669999	6699CC	6699FF
66CC00	66CC33	66CC66	66CC99	66CCCC	66CCFF
66FF00	66FF33	66FF66	66FF99	66FFCC	66FFFF
990000	990033	990066	990099	9900CC	9900FF
993300	993333	993366	993399	9933CC	9933FF
996600	996633	996666	996699	9966CC	9966FF
999900	999933	999966	999999	9999CC	9999FF
99CC00	99CC33	99CC66	99CC99	99CCCC	99CCFF
99FF00	99FF33	99FF66	99FF99	99FFCC	99FFFF
CC0000	CC0033	CC0066	CC0099	CC00CC	CC00FF
CC3300	CC3333	CC3366	CC3399	CC33CC	CC33FF
CC6600	CC6633	CC6666	CC6699	CC66CC	CC66FF
CC9900	CC9933	CC9966	CC9999	CC99CC	CC99FF
CCCC00	CCCC33	CCCC66	CCCC99	CCCCCC	CCCCFF
CCFF00	CCFF33	CCFF66	CCFF99	CCFFCC	CCFFFF
FF0000	FF0033	FF0066	FF0099	FF00CC	FF00FF
FF3300	FF3333	FF3366	FF3399	FF33CC	FF33FF
FF6600	FF6633	FF6666	FF6699	FF66CC	FF66FF
FF9900	FF9933	FF9966	FF9999	FF99CC	FF99FF
FFCC00	FFCC33	FFCC66	FFCC99	FFCCCC	FFCCFF
FFFF00	FFFF33	FFFF66	FFFF99	FFFFCC	FFFFFF

CSS - Measurement Units

CSS supports a number of measurements including absolute units such as inches, centimeters, points, and so on, as well as relative measures such as percentages and em units. You need these values while specifying various measurements in your Style rules e.g. border = "1px solid red".

We have listed out all the CSS Measurement Units along with proper Examples –

Unit	Description	Example
%	Defines a measurement as a percentage relative to another value, typically an enclosing element.	p {font-size: 16pt; line-height: 125%;}
cm	Defines a measurement in centimeters.	div {margin-bottom: 2cm;}
em	A relative measurement for the height of a font in em spaces. Because an em unit is equivalent to the size of a given font, if you assign a font to 12pt, each "em" unit would be 12pt; thus, 2em would be 24pt.	p {letter-spacing: 7em;}
ex	This value defines a measurement relative to a font's x-height. The x-height is determined by the height of the font's lowercase letter x.	p {font-size: 24pt; line-height: 3ex;}
in	Defines a measurement in inches.	p {word-spacing: .15in;}
mm	Defines a measurement in millimeters.	p {word-spacing: 15mm;}
pc	Defines a measurement in picas. A pica is equivalent to 12 points; thus, there are 6 picas per inch.	p {font-size: 20pc;}
pt	Defines a measurement in points. A point is defined as 1/72nd of an inch.	body {font-size: 18pt;}
px	Defines a measurement in screen pixels.	p {padding: 25px;}

CSS - Backgrounds

You can set the following background properties of an element –

The **background-color** property is used to set the background color of an element.

The **background-image** property is used to set the background image of an element.

The **background-repeat** property is used to control the repetition of an image in the background.

The **background-position** property is used to control the position of an image in the background.

The **background-attachment** property is used to control the scrolling of an image in the background.

The background property is used as a shorthand to specify a number of other background properties.

Set the Background Color

Following is the example which demonstrates how to set the background color for an element.

```
<html>
  <head>
  </head>
  <body>
    <p style = "background-color:yellow;">
      This text has a yellow background color.
    </p>
  </body>
</html>
```

This will produce following result –

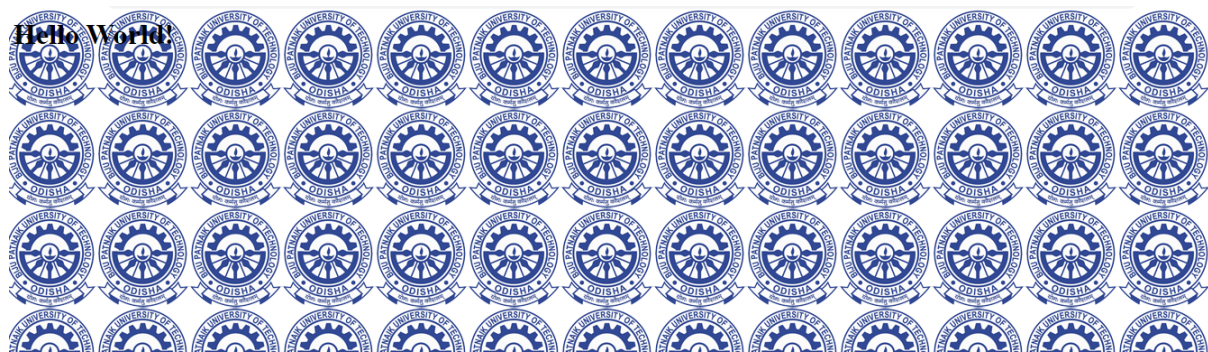
This text has a yellow background color.

Set the Background Image

We can set the background image by calling local stored images as shown below

```
<html>
<head>
<style>
body {
background-image: url("https://www.bput.ac.in/images/bput-logo.png");
background-color: #cccccc;
}
</style>
</head>
<body>
<h1>Hello World!</h1>
</body>
</html>
```

O/P



Repeat the Background Image

The following example demonstrates how to repeat the background image if an image is small. You can use *no-repeat* value for *background-repeat* property if you don't want to repeat an image, in this case image will display only once.

By default *background-repeat* property will have *repeat* value.

```
<html>
  <head>
    <style>
      body {
        background-image: url("https://www.bput.ac.in/images/bput-logo.png");
        background-repeat: repeat;
      }
    </style>
  </head>

  <body>
    <p>Hello World</p>
  </body>
</html>
```

O/P



```

<html>
<head>
<style>
  body {
    background-image: url("https://www.bput.ac.in/images/bput-logo.png");
    background-repeat: no-repeat;
  }
</style>
</head>

<body>
  <p>Hello World</p>
</body>
</html>

```

O/P



```

<html>
<head>
<style>
  body {
    background-image: url("https://www.bput.ac.in/images/bput-logo.png");
    background-repeat: repeat-y;
  }
</style>
</head>
<body>
  <p>Hello World</p>
</body>
</html>

```

O/P



```

<html>
<head>
  <style>
    body {
      background-image: url("https://www.bput.ac.in/images/bput-logo.png");
      background-repeat: repeat-x;
    }
  </style>
</head>
<body>
  <p>Hello World</p>
</body>
</html>
O/P

```



Set the Background Image Position

The following example demonstrates how to set the background image position 100 pixels away from the left side.

```

<html>
<head>
  <style>

```

```
    body {
        background-image: url("https://www.bput.ac.in/images/bput-logo.png ");
        background-position:100px;
    }
</style>
</head>

<body>
    <p>Hello World</p>
</body>
</html>
```

The following example demonstrates how to set the background image position 100 pixels away from the left side and 200 pixels down from the top.

```
<html>
<head>
    <style>
        body {
            background-image: url("https://www.bput.ac.in/images/bput-logo.png ");
            background-position:100px 200px;
        }
    </style>
</head>

<body>
    <p>Hello World</p>
</body>
</html>
```

CSS - Fonts

This chapter teaches you how to set fonts of a content, available in an HTML element. You can set following font properties of an element –

The **font-family** property is used to change the face of a font.

The **font-style** property is used to make a font italic or oblique.

The **font-variant** property is used to create a small-caps effect.

The **font-weight** property is used to increase or decrease how bold or light a font appears.

The **font-size** property is used to increase or decrease the size of a font.

The **font property** is used as shorthand to specify a number of other font properties.

Set the Font Family

Following is the example, which demonstrates how to set the font family of an element. Possible value could be any font family name.

```
<html>
  <head>
  </head>
  <body>
    <p style = "font-family:georgia,garamond,serif;">
      This text is rendered in either georgia, garamond, or the
      default serif font depending on which font you have at your system.
    </p>
  </body>
</html>
```

This will produce following result –

O/P

This text is rendered in either georgia, garamond, or the default serif font depending on which font you have at your system.

Set the Font Style

Following is the example, which demonstrates how to set the font style of an element. Possible values are normal, italic and oblique.

```
<html>
  <head>
  </head>
  <body>
    <p style = "font-style:italic;">
      This text will be rendered in italic style
    </p>
```

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

This will produce following result –

O/P

This text will be rendered in italic style

Set the Font Variant

The following example demonstrates how to set the font variant of an element. Possible values are *normal* and *small-caps*.

```
<html>
  <head>
  </head>

  <body>
    <p style = "font-variant:small-caps;">
      This text will be rendered as small caps
    </p>
  </body>
</html>
```

O/P

This text will be rendered as small caps

Set the Font Weight

The following example demonstrates how to set the font weight of an element. The font-weight property provides the functionality to specify how bold a font is. Possible values could be normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900.

```
<html>
  <head>
  </head>

  <body>
    <p style = "font-weight:bold;">
      This font is bold.
    </p>

    <p style = "font-weight:bolder;">
      This font is bolder.
    </p>
```

```
<p style = "font-weight:500;">
  This font is 500 weight.
</p>
</body>
</html>
```

This will produce following result –

This font is bold.

This font is bolder.

This font is 500 weight.

Set the Font Size

The following example demonstrates how to set the font size of an element. The font-size property is used to control the size of fonts. Possible values could be xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, size in pixels or in %.

```
<html>
<head>
</head>
<body>
  <p style = "font-size:20px;">
    This font size is 20 pixels
  </p>

  <p style = "font-size:small;">
    This font size is small
  </p>

  <p style = "font-size:large;">
    This font size is large
  </p>
</body>
</html>
```

This will produce following result –

This font size is 20 pixels

This font size is small

This font size is large

Set the Font Size Adjust

The following example demonstrates how to set the font size adjust of an element. This property enables you to adjust the x-height to make fonts more legible. Possible value could be any number.

```
<html>
  <head>
  </head>

  <body>
    <p style = "font-size-adjust:0.61;">
      This text is using a font-size-adjust value.
    </p>
  </body>
</html>
```

This will produce following result –

This text is using a font-size-adjust value.

Set the Font Stretch

The following example demonstrates how to set the font stretch of an element. This property relies on the user's computer to have an expanded or condensed version of the font being used.

Possible values could be normal, wider, narrower, ultra-condensed, extra-condensed, condensed, semi-condensed, semi-expanded, expanded, extra-expanded, ultra-expanded.

```
<html>
  <head>
  </head>
  <body>
    <p style = "font-stretch:ultra-expanded;">
      If this doesn't appear to work, it is likely that your computer
      doesn't have a <br>condensed or expanded version of the font being used.
    </p>
  </body>
</html>
```

This will produce following result –

If this doesn't appear to work, it is likely that your computer doesn't have a condensed or expanded version of the font being used.

Shorthand Property

You can use the font property to set all the font properties at once. For example –

```
<html>
  <head>
  </head>
  <body>
    <p style = "font:italic small-caps bold 15px georgia;">
      Applying all the properties on the text at once.
    </p>
  </body>
</html>
```

This will produce following result –

APPLYING ALL THE PROPERTIES ON THE TEXT AT ONCE.

CSS - Text

You can set following text properties of an element –

- ✓ The **color** property is used to set the color of a text.
- ✓ The **direction** property is used to set the text direction.
- ✓ The **letter-spacing** property is used to add or subtract space between the letters that make up a word.
- ✓ The **word-spacing** property is used to add or subtract space between the words of a sentence.
- ✓ The **text-indent** property is used to indent the text of a paragraph.
- ✓ The **text-align** property is used to align the text of a document.
- ✓ The **text-decoration** property is used to underline, overline, and strikethrough text.
- ✓ The **text-transform** property is used to capitalize text or convert text to uppercase or lowercase letters.
- ✓ The **white-space** property is used to control the flow and formatting of text.
- ✓ The **text-shadow** property is used to set the text shadow around a text.

Set the Text Color

The following example demonstrates how to set the text color. Possible value could be any color name in any valid format.

```
<html>
<head>
</head>
<body>
  <p style = "color:red;">
    CIME,Bhubaneswar.
  </p>
</body>
</html>
```

It will produce the following result –

CIME,Bhubaneswar.

Set the Text Direction

The following example demonstrates how to set the direction of a text. Possible values are ltr or rtl.

```
<html>
<head>
</head>
<body>
  <p style = "direction:rtl;">
    This text will be rendered from right to left
  </p>
</body>
</html>
```


It will produce the following result –

This text will be rendered from right to left

Set the Space between Characters

The following example demonstrates how to set the space between characters. Possible values are normal or a number specifying space..

```
<html>
  <head>
  </head>
  <body>
    <p style = "letter-spacing:5px;">
      This text is having space between letters.
    </p>
  </body>
</html>
```

It will produce the following result –

T h i s t e x t i s h a v i n g s p a c e b e t w e e n l e t t e r s .

Set the Space between Words

The following example demonstrates how to set the space between words. Possible values are normal or a number specifying space.

```
<html>
  <head>
  </head>
  <body>
    <p style = "word-spacing:5px;">
      This text is having space between words.
    </p>
  </body>
</html>
```

This will produce following result –

This text is having space between words.

Set the Text Indent

The following example demonstrates how to indent the first line of a paragraph. Possible values are % or a number specifying indent space.

```
<html>
  <head>
  </head>
  <body>
    <p style = "text-indent:1cm;">
      This text will have first line indented by 1cm and this line will remain at
```

its actual position this is done by CSS text-indent property.

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

It will produce the following result –

This text will have first line indented by 1cm and this line will remain at its actual position this is done by CSS text-indent property.

Set the Text Alignment

The following example demonstrates how to align a text. Possible values are left, right, center, justify.

```
<html>
<head>
</head>
<body>
  <p style = "text-align:right;">
    This will be right aligned.
  </p>
  <p style = "text-align:center;">
    This will be center aligned.
  </p>
  <p style = "text-align:left;">
    This will be left aligned.
  </p>
</body>
</html>
```

This will produce following result –

This will be right aligned.

This will be center aligned.

This will be left aligned.

Decorating the Text

The following example demonstrates how to decorate a text. Possible values are none, underline, overline, line-through, blink.

```
<html>
<head>
</head>
<body>
  <p style = "text-decoration:underline;">
    This will be underlined
  </p>
  <p style = "text-decoration:line-through;">
    This will be striked through.
  </p>
```

```
<p style = "text-decoration:underline;">
  This will have a over line.
</p>
<p style = "text-decoration:blink;">
  This text will have blinking effect
</p>
</body>
</html>
```

This will produce following result –

This will be underlined

~~This will be striked through.~~

This will have a over line.

This text will have blinking effect

Set the Text Cases

The following example demonstrates how to set the cases for a text. Possible values are none, capitalize, uppercase, lowercase.

```
<html>
<head>
</head>
<body>
  <p style = "text-transform:capitalize;">
    This will be capitalized
  </p>
  <p style = "text-transform:uppercase;">
    This will be in uppercase
  </p>
  <p style = "text-transform:lowercase;">
    This will be in lowercase
  </p>
</body>
</html>
```

This will produce following result –

This Will Be Capitalized

THIS WILL BE IN UPPERCASE

this will be in lowercase

Set the White Space between Text

The following example demonstrates how white space inside an element is handled. Possible values are normal, pre, nowrap.

```
<html>
  <head>
  </head>
  <body>
    <p style = "white-space:pre;">
      This text has a line break and the white-space pre setting
      tells the browser to honor it just like the HTML pre tag.
    </p>
  </body>
</html>
```

This will produce following result –

This text has a line break and the white-space pre setting
tells the browser to honor it just like the HTML pre tag.

Set the Text Shadow

The following example demonstrates how to set the shadow around a text. This may not be supported by all the browsers.

```
<html>
  <head>
  </head>
  <body>
    <p style = "text-shadow:4px 4px 8px blue;">
      If your browser supports the CSS text-shadow property,
      this text will have a blue shadow.
    </p>
  </body>
</html>
```

It will produce the following result –

If your browser supports the CSS text-shadow property, this text will have a blue shadow.

CSS - Borders

The *border* properties allow you to specify how the border of the box representing an element should look. There are three properties of a border you can change –

- The **border-color** specifies the color of a border.
- The **border-style** specifies whether a border should be solid, dashed line, double line, or one of the other possible values.
- The **border-width** specifies the width of a border.

Now, we will see how to use these properties with examples.

The border-color Property

The border-color property allows you to change the color of the border surrounding an element. You can individually change the color of the bottom, left, top and right sides of an element's border using the properties –

- **border-bottom-color** changes the color of bottom border.
- **border-top-color** changes the color of top border.
- **border-left-color** changes the color of left border.
- **border-right-color** changes the color of right border.

The following example shows the effect of all these properties –

```
<html>
<head>
  <style type = "text/css">
    p.example1 {
      border:1px solid;
      border-bottom-color:#009900; /* Green */
      border-top-color:#FF0000; /* Red */
      border-left-color:#330000; /* Black */
      border-right-color:#0000CC; /* Blue */
    }
    p.example2 {
      border:1px solid;
      border-color:#009900; /* Green */
    }
  </style>
</head>

<body>
  <p class = "example1">
    This example is showing all borders in different colors.
  </p>

  <p class = "example2">
    This example is showing all borders in green color only.
  </p>
```

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

It will produce the following result –

This example is showing all borders in different colors.

This example is showing all borders in green color only.

The border-style Property

The border-style property allows you to select one of the following styles of border –

- **none** – No border. (Equivalent of border-width:0;)
- **solid** – Border is a single solid line.
- **dotted** – Border is a series of dots.
- **dashed** – Border is a series of short lines.
- **double** – Border is two solid lines.
- **groove** – Border looks as though it is carved into the page.
- **ridge** – Border looks the opposite of groove.
- **inset** – Border makes the box look like it is embedded in the page.
- **outset** – Border makes the box look like it is coming out of the canvas.
- **hidden** – Same as none, except in terms of border-conflict resolution for table elements.

You can individually change the style of the bottom, left, top, and right borders of an element using the following properties –

- **border-bottom-style** changes the style of bottom border.
- **border-top-style** changes the style of top border.
- **border-left-style** changes the style of left border.
- **border-right-style** changes the style of right border.

The following example shows all these border styles –

```
<html>
<head>
</head>

<body>
  <p style = "border-width:4px; border-style:none;">
    This is a border with none width.
  </p>

  <p style = "border-width:4px; border-style:solid;">
    This is a solid border.
  </p>

  <p style = "border-width:4px; border-style:dashed;">
    This is a dashed border.
  </p>
```

```
<p style = "border-width:4px; border-style:double;">
```

This is a double border.

```
</p>
```

```
<p style = "border-width:4px; border-style:groove;">
```

This is a groove border.

```
</p>
```

```
<p style = "border-width:4px; border-style:ridge">
```

This is a ridge border.

```
</p>
```

```
<p style = "border-width:4px; border-style:inset;">
```

This is a inset border.

```
</p>
```

```
<p style = "border-width:4px; border-style:outset;">
```

This is a outset border.

```
</p>
```

```
<p style = "border-width:4px; border-style:hidden;">
```

This is a hidden border.

```
</p>
```

```
<p style = "border-width:4px;
```

```
border-top-style:solid;
```

```
border-bottom-style:dashed;
```

```
border-left-style:groove;
```

```
border-right-style:double;">
```

This is a a border with four different styles.

```
</p>
```

```
</body>
```

```
</html>
```

It will produce the following result –

This is a border with none width.

This is a solid border.

This is a dashed border.

This is a double border.

This is a groove border.

This is a ridge border.

This is a inset border.

This is a outset border.

This is a hidden border.

This is a border with four different styles.

The border-width Property

The border-width property allows you to set the width of an element borders. The value of this property could be either a length in px, pt or cm or it should be set to *thin*, *medium* or *thick*.

You can individually change the width of the bottom, top, left, and right borders of an element using the following properties –

- **border-bottom-width** changes the width of bottom border.
- **border-top-width** changes the width of top border.
- **border-left-width** changes the width of left border.
- **border-right-width** changes the width of right border.

The following example shows all these border width –

```
<html>
  <head>
  </head>

  <body>
    <p style = "border-width:4px; border-style:solid;">
      This is a solid border whose width is 4px.
    </p>

    <p style = "border-width:4pt; border-style:solid;">
      This is a solid border whose width is 4pt.
    </p>
```



```
<p style = "border-width:thin; border-style:solid;">
```

This is a solid border whose width is thin.

```
</p>
```

```
<p style = "border-width:medium; border-style:solid;">
```

This is a solid border whose width is medium;

```
</p>
```

```
<p style = "border-width:thick; border-style:solid;">
```

This is a solid border whose width is thick.

```
</p>
```

```
<p style = "border-bottom-width:4px;border-top-width:10px;
border-left-width: 2px;border-right-width:15px;border-style:solid;">
```

This is a a border with four different width.

```
</p>
```

```
</body>
```

```
</html>
```

It will produce the following result –

This is a solid border whose width is 4px.

This is a solid border whose width is 4pt.

This is a solid border whose width is thin.

This is a solid border whose width is medium;

This is a solid border whose width is thick.

This is a a border with four different width.

Border Properties Using Shorthand

The border property allows you to specify color, style, and width of lines in one property –

The following example shows how to use all the three properties into a single property. This is the most frequently used property to set border around any element.

```
<html>
```

```
<head>
```

```
</head>
```

```
<body>
```

```
<p style = "border:4px solid red;">
```

This example is showing shorthand property for border.

</p>

</body>

</html>

It will produce the following result –

This example is showing shorthand property for border.

CSS - Margins

The *margin* property defines the space around an HTML element. It is possible to use negative values to overlap content.

The values of the margin property are not inherited by the child elements. Remember that the adjacent vertical margins (top and bottom margins) will collapse into each other so that the distance between the blocks is not the sum of the margins, but only the greater of the two margins or the same size as one margin if both are equal.

We have the following properties to set an element margin.

- The **margin** specifies a shorthand property for setting the margin properties in one declaration.
- The **margin-bottom** specifies the bottom margin of an element.
- The **margin-top** specifies the top margin of an element.
- The **margin-left** specifies the left margin of an element.
- The **margin-right** specifies the right margin of an element.

Now, we will see how to use these properties with examples.

The Margin Property

The margin property allows you set all of the properties for the four margins in one declaration. Here is the syntax to set margin around a paragraph –

Here is an example –<html>

```
<head>
```

```
</head>
```

```
<body>
```

```
<p style = "margin: 15px; border:1px solid black;">
```

```
    all four margins will be 15px
```

```
</p>
```

```
<p style = "margin:10px 2%; border:1px solid black;">
```

```
    top and bottom margin will be 10px, left and right margin will be 2%  
    of the total width of the document.
```

```
</p>
```

```
<p style = "margin: 10px 2% -10px; border:1px solid black;">
```

```
    top margin will be 10px, left and right margin will be 2% of the  
    total width of the document, bottom margin will be -10px
```

```
</p>
```

```
<p style = "margin: 10px 2% -10px auto; border:1px solid black;">
```

```
    top margin will be 10px, right margin will be 2% of the total  
    width of the document, bottom margin will be -10px, left margin  
    will be set by the browser
```

```
</p>
```

```
</body>
```

</html>

It will produce the following result –

all four margins will be 15px
top and bottom margin will be 10px, left and right margin will be 2% of the total width of the document.
top margin will be 10px, left and right margin will be 2% of the total width of the document, bottom margin will be -10px
top margin will be 10px, right margin will be 2% of the total width of the document, bottom margin will be -10px, left margin will be set by the browser

The margin-bottom Property

The margin-bottom property allows you set bottom margin of an element. It can have a value in length, % or auto.

Here is an example –

```
<html>
  <head>
  </head>

  <body>
    <p style = "margin-bottom: 15px; border:1px solid black;">
      This is a paragraph with a specified bottom margin
    </p>

    <p style = "margin-bottom: 5%; border:1px solid black;">
      This is another paragraph with a specified bottom margin in percent
    </p>
  </body>
</html>
```

It will produce the following result –

This is a paragraph with a specified bottom margin
This is another paragraph with a specified bottom margin in percent

The margin-top Property

The margin-top property allows you set top margin of an element. It can have a value in length, % or auto.

Here is an example –

```
<html>
  <head>
  </head>

  <body>
    <p style = "margin-top: 15px; border:1px solid black;">
      This is a paragraph with a specified top margin
```

</p>

<p style = "margin-top: 5%; border:1px solid black;">

This is another paragraph with a specified top margin in percent

</p>

</body>

</html>

It will produce the following result –

This is a paragraph with a specified top margin

This is another paragraph with a specified top margin in percent

The margin-left Property

The margin-left property allows you set left margin of an element. It can have a value in length, % or auto.

Here is an example –

<html>

<head>

</head>

<body>

<p style = "margin-left: 15px; border:1px solid black;">

This is a paragraph with a specified left margin

</p>

<p style = "margin-left: 5%; border:1px solid black;">

This is another paragraph with a specified top margin in percent

</p>

</body>

</html>

It will produce the following result –

This is a paragraph with a specified left margin

This is another paragraph with a specified top margin in percent

The margin-right Property

The margin-right property allows you set right margin of an element. It can have a value in length, % or auto.

Here is an example –

<html>

<head>

</head>

<body>

<p style = "margin-right: 15px; border:1px solid black;">

This is a paragraph with a specified right margin

</p>

<p style = "margin-right: 5%; border:1px solid black;">

This is another paragraph with a specified right margin in percent

</p>

</body>

</html>

It will produce the following result –

This is a paragraph with a specified right margin

This is another paragraph with a specified right margin in percent

CSS - Paddings

The *padding* property allows you to specify how much space should appear between the content of an element and its border –

The value of this attribute should be either a length, a percentage, or the word *inherit*. If the value is *inherit*, it will have the same padding as its parent element. If a percentage is used, the percentage is of the containing box.

The following CSS properties can be used to control lists. You can also set different values for the padding on each side of the box using the following properties –

- The **padding-bottom** specifies the bottom padding of an element.
- The **padding-top** specifies the top padding of an element.
- The **padding-left** specifies the left padding of an element.
- The **padding-right** specifies the right padding of an element.
- The **padding** serves as shorthand for the preceding properties.

Now, we will see how to use these properties with examples.

The padding-bottom Property

The *padding-bottom* property sets the bottom padding (space) of an element. This can take a value in terms of length or %.

Here is an example –

```
<html>
<head>
</head>

<body>
  <p style = "padding-bottom: 15px; border:1px solid black;">
    This is a paragraph with a specified bottom padding
  </p>

  <p style = "padding-bottom: 5%; border:1px solid black;">
    This is another paragraph with a specified bottom padding in percent
  </p>
</body>
</html>
```

It will produce the following result –

This is a paragraph with a specified bottom padding

This is another paragraph with a specified bottom padding in percent

The padding-top Property

The *padding-top* property sets the top padding (space) of an element. This can take a value in terms of length or %.

Here is an example –

```
<html>
  <head>
  </head>

  <body>
    <p style = "padding-top: 15px; border:1px solid black;">
      This is a paragraph with a specified top padding
    </p>

    <p style = "padding-top: 5%; border:1px solid black;">
      This is another paragraph with a specified top padding in percent
    </p>
  </body>
</html>
```

It will produce the following result –

The padding-left Property

The *padding-left* property sets the left padding (space) of an element. This can take a value in terms of length or %.

Here is an example –

```
<html>
  <head>
  </head>

  <body>
    <p style = "padding-left: 15px; border:1px solid black;">
      This is a paragraph with a specified left padding
    </p>

    <p style = "padding-left: 15%; border:1px solid black;">
      This is another paragraph with a specified left padding in percent
    </p>
  </body>
</html>
```

It will produce the following result –

This is a paragraph with a specified left padding

This is another paragraph with a specified left padding in percent
--

The padding-right Property

The *padding-right* property sets the right padding (space) of an element. This can take a value in terms of length or %.

Here is an example –<html>

```
<head>
```

```
</head>
```

```
<body>
```

```
<p style = "padding-right: 15px; border:1px solid black;">
```

This is a paragraph with a specified right padding

```
</p>
```

```
<p style = "padding-right: 5%; border:1px solid black;">
```

This is another paragraph with a specified right padding in percent

```
</p>
```

```
</body>
```

```
</html>
```

It will produce the following result –

This is a paragraph with a specified right padding

This is another paragraph with a specified right padding in percent

The Padding Property

The *padding* property sets the left, right, top and bottom padding (space) of an element. This can take a value in terms of length or %.

Here is an example –<html>

```
<head>
```

```
</head>
```

```
<body>
```

```
<p style = "padding: 15px; border:1px solid black;">
```

all four padding will be 15px

```
</p>
```

```
<p style = "padding:10px 2%; border:1px solid black;">
```

top and bottom padding will be 10px, left and right padding will be 2% of the total width of the document.

```
</p>
```

```
<p style = "padding: 10px 2% 10px; border:1px solid black;">
```

top padding will be 10px, left and right padding will be 2% of the total width of the document, bottom padding will be 10px

```
</p>
```

```
<p style = "padding: 10px 2% 10px 10px; border:1px solid black;">  
    top padding will be 10px, right padding will be 2% of  
    the total width of the document, bottom padding and top padding will be  
10px  
</p>  
</body>  
</html>
```

It will produce the following result –

all four padding will be 15px

top and bottom padding will be 10px, left and right padding will be 2% of the total width of the document.

top padding will be 10px, left and right padding will be 2% of the total width of the document
padding will be 10px

top padding will be 10px, right padding will be 2% of the total width of the document, bottom
and top padding will be 10px

CSS - Cursors

The *cursor* property of CSS allows you to specify the type of cursor that should be displayed to the user.

One good usage of this property is in using images for submit buttons on forms. By default, when a cursor hovers over a link, the cursor changes from a pointer to a hand. However, it does not change form for a submit button on a form. Therefore, whenever someone hovers over an image that is a submit button, it provides a visual clue that the image is clickable.

The following table shows the possible values for the cursor property –

Sr. No.	Value & Description
1	auto Shape of the cursor depends on the context area it is over. For example an I over text, a hand over a link, and so on...
2	crosshair A crosshair or plus sign
3	default An arrow
4	pointer A pointing hand (in IE 4 this value is hand)
5	move The I bar
6	e-resize The cursor indicates that an edge of a box is to be moved right (east)
7	ne-resize The cursor indicates that an edge of a box is to be moved up and right (north/east)
8	nw-resize The cursor indicates that an edge of a box is to be moved up and left (north/west)
9	n-resize The cursor indicates that an edge of a box is to be moved up (north)
10	se-resize The cursor indicates that an edge of a box is to be moved down and right (south/east)
11	sw-resize The cursor indicates that an edge of a box is to be moved down and left

	(south/west)
12	s-resize The cursor indicates that an edge of a box is to be moved down (south)
13	w-resize The cursor indicates that an edge of a box is to be moved left (west)
14	text The I bar
15	wait An hour glass
16	help A question mark or balloon, ideal for use over help buttons
17	<url> The source of a cursor image file

NOTE – You should try to use only these values to add helpful information for users, and in places, they would expect to see that cursor. For example, using the crosshair when someone hovers over a link can confuse visitors.

Here is an example –

```
<html>
  <head>
  </head>

  <body>
    <p>Move the mouse over the words to see the cursor change:</p>

    <div style = "cursor:auto">Auto</div>
    <div style = "cursor:crosshair">Crosshair</div>
    <div style = "cursor:default">Default</div>

    <div style = "cursor:pointer">Pointer</div>
    <div style = "cursor:move">Move</div>
    <div style = "cursor:e-resize">e-resize</div>
    <div style = "cursor:ne-resize">ne-resize</div>
    <div style = "cursor:nw-resize">nw-resize</div>

    <div style = "cursor:n-resize">n-resize</div>
    <div style = "cursor:se-resize">se-resize</div>
    <div style = "cursor:sw-resize">sw-resize</div>
    <div style = "cursor:s-resize">s-resize</div>
    <div style = "cursor:w-resize">w-resize</div>
```

```
<div style = "cursor:text">text</div>
<div style = "cursor:wait">wait</div>
<div style = "cursor:help">help</div>
</body>
</html>
```

It will produce the following result –

Move the mouse over the words to see the cursor change:

Auto
Crosshair
Default
Pointer
Move
e-resize
ne-resize
nw-resize
n-resize
se-resize
sw-resize
s-resize
w-resize
text
wait
help

CSS - Outlines

Outlines are very similar to borders, but there are few major differences as well –

- An outline does not take up space.
- Outlines do not have to be rectangular.
- Outline is always the same on all sides; you cannot specify different values for different sides of an element.

NOTE – The outline properties are not supported by IE 6 or Netscape 7.

You can set the following outline properties using CSS.

- The **outline-width** property is used to set the width of the outline.
- The **outline-style** property is used to set the line style for the outline.
- The **outline-color** property is used to set the color of the outline.
- The **outline** property is used to set all the above three properties in a single statement.

The outline-width Property

The *outline-width* property specifies the width of the outline to be added to the box. Its value should be a length or one of the values *thin*, *medium*, or *thick*, just like the border-width attribute.

A width of zero pixels means no outline.

Here is an example –

```
<html>
  <head>
  </head>

  <body>
    <p style = "outline-width:thin; outline-style:solid;">
      This text is having thin outline.
    </p>
    <br />

    <p style = "outline-width:thick; outline-style:solid;">
      This text is having thick outline.
    </p>
    <br />

    <p style = "outline-width:5px; outline-style:solid;">
      This text is having 5x outline.
    </p>
  </body>
</html>
```

It will produce the following result –

This text is having thin outline.

This text is having thick outline.

This text is having 5x outline.

The outline-style Property

The *outline-style* property specifies the style for the line (solid, dotted, or dashed) that goes around an element. It can take one of the following values –

- **none** – No border. (Equivalent of outline-width:0;)
- **solid** – Outline is a single solid line.
- **dotted** – Outline is a series of dots.
- **dashed** – Outline is a series of short lines.
- **double** – Outline is two solid lines.
- **groove** – Outline looks as though it is carved into the page.
- **ridge** – Outline looks the opposite of groove.
- **inset** – Outline makes the box look like it is embedded in the page.
- **outset** – Outline makes the box look like it is coming out of the canvas.
- **hidden** – Same as none.

Here is an example –<html>

```
<head>  
</head>
```

```
<body>  
  <p style = "outline-width:thin; outline-style:solid;">  
    This text is having thin solid outline.  
  </p>  
  <br />
```

```
  <p style = "outline-width:thick; outline-style:dashed;">  
    This text is having thick dashed outline.  
  </p>  
  <br />
```

```
  <p style = "outline-width:5px;outline-style:dotted;">  
    This text is having 5x dotted outline.  
  </p>  
</body>
```

```
</html>
```

It will produce the following result –

This text is having thin solid outline.

This text is having thick dashed outline.

This text is having 5x dotted outline.

The outline-color Property

The *outline-color* property allows you to specify the color of the outline. Its value should either be a color name, a hex color, or an RGB value, as with the color and border-color properties.

Here is an example –

```
<html>
  <head>
  </head>

  <body>
    <p style = "outline-width:thin; outline-style:solid;outline-color:red">
      This text is having thin solid red  outline.
    </p>
    <br />

    <p style = "outline-width:thick; outline-style:dashed;outline-color:#009900">
      This text is having thick dashed green outline.
    </p>
    <br />

    <p      style      =      "outline-width:5px;outline-style:dotted;outline-
color:rgb(13,33,232)">
      This text is having 5x dotted blue outline.
    </p>
  </body>
</html>
```

It will produce the following result –

This text is having thin solid red outline.

This text is having thick dashed green outline.

This text is having 5x dotted blue outline.

The outline Property

The *outline* property is a shorthand property that allows you to specify values for any of the three properties discussed previously in any order but in a single statement.

Here is an example –

```
<html>
  <head>
  </head>

  <body>
    <p style = "outline:thin solid red;">
      This text is having thin solid red outline.
    </p>
    <br />

    <p style = "outline:thick dashed #009900;">
      This text is having thick dashed green outline.
    </p>
    <br />

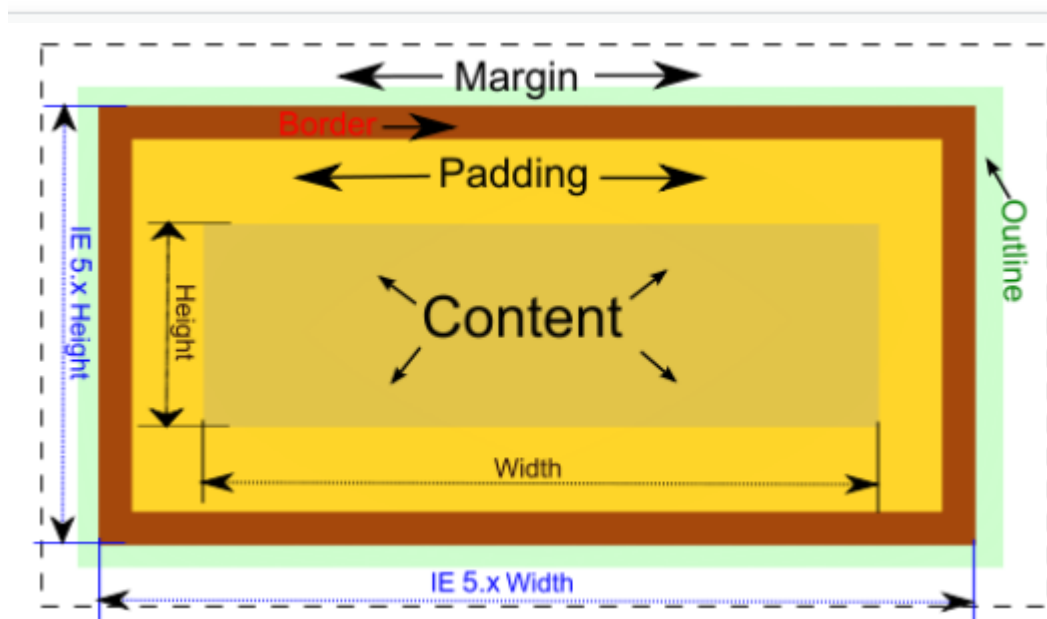
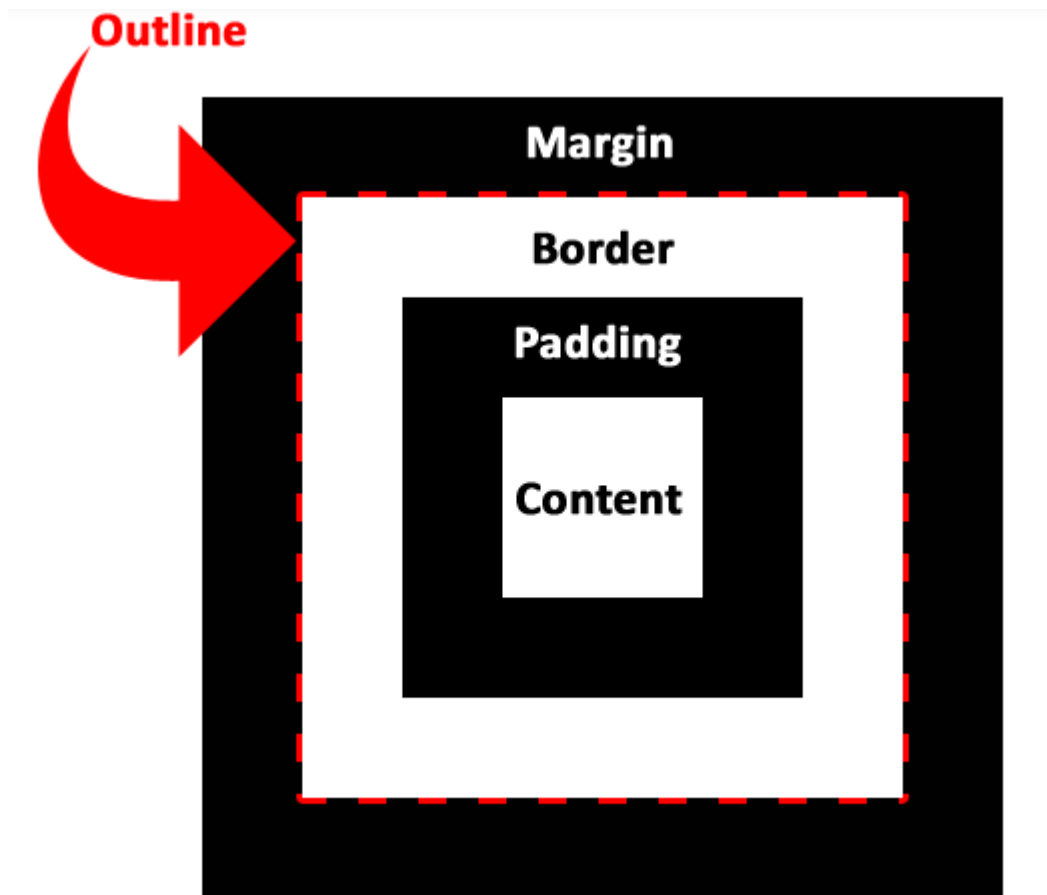
    <p style = "outline:5px dotted rgb(13,33,232);">
      This text is having 5x dotted blue outline.
    </p>
  </body>
</html>
```

It will produce the following result –

This text is having thin solid red outline.

This text is having thick dashed green outline.

This text is having 5x dotted blue outline.



What is difference between margin and outline?

outline only makes a line around any element to make it look different from the other elements. it will not give any space. whereas margin will give space around any element.

What is difference between border and outline ?

Border is created inside the element, where as outline is created outside the element. So border is computed along with the width and height of the element, while outline draws outside the element.

They are very similar properties but not identical. While the border is rectangular, the outline can be non-rectangular. Outline also does not take space.

MARGIN VERSUS PADDING

MARGIN	PADDING
CSS property that is used to create space around the element outside the defined border	CSS property that is used to create space around the element, inside the defined border
The values of margin can be auto, length, % or inherit	The values of padding can be length, % or inherit type

CSS - Dimension

We have the following properties that allow you to control the dimensions of a box.

- The **height** property is used to set the height of a box.
- The **width** property is used to set the width of a box.
- The **line-height** property is used to set the height of a line of text.
- The **max-height** property is used to set a maximum height that a box can be.
- The **min-height** property is used to set the minimum height that a box can be.
- The **max-width** property is used to set the maximum width that a box can be.
- The **min-width** property is used to set the minimum width that a box can be.

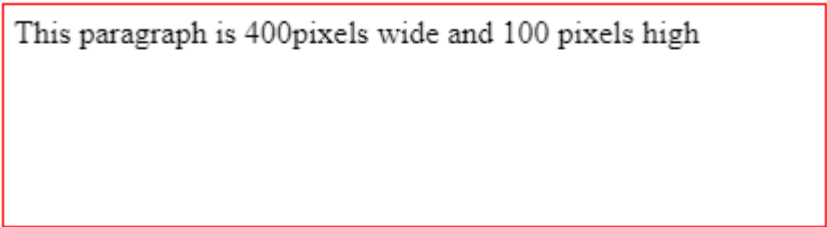
The Height and Width Properties

The height and width properties allow you to set the height and width for boxes. They can take values of a length, a percentage, or the keyword auto.

Here is an example –

```
<html>
  <head>
  </head>
  <body>
    <p style = "width:400px; height:100px; border:1px solid red; padding:5px;
margin:10px;">
      This paragraph is 400pixels wide and 100 pixels high
    </p>
  </body>
</html>
```

It will produce the following result –



This paragraph is 400pixels wide and 100 pixels high

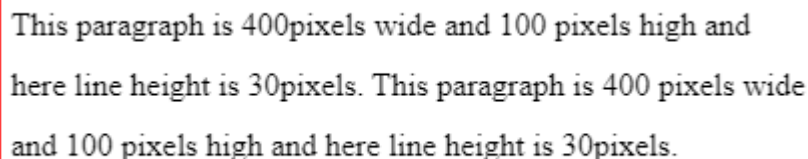
The line-height Property

The line-height property allows you to increase the space between lines of text. The value of the line-height property can be a number, a length, or a percentage.

Here is an example –

```
<html>
  <head>
  </head>
  <body>
    <p style = "width:400px; height:100px; border:1px solid red; padding:5px;
margin:10px; line-height:30px;">
      This paragraph is 400pixels wide and 100 pixels high and here line height is
30pixels.
      This paragraph is 400 pixels wide and 100 pixels high and here line height is
30pixels.
    </p>
  </body>
</html>
```

It will produce the following result –



This paragraph is 400pixels wide and 100 pixels high and here line height is 30pixels. This paragraph is 400 pixels wide and 100 pixels high and here line height is 30pixels.

The max-height Property

The max-height property allows you to specify maximum height of a box. The value of the max-height property can be a number, a length, or a percentage.

NOTE – This property does not work in either Netscape 7 or IE 6.

Here is an example –

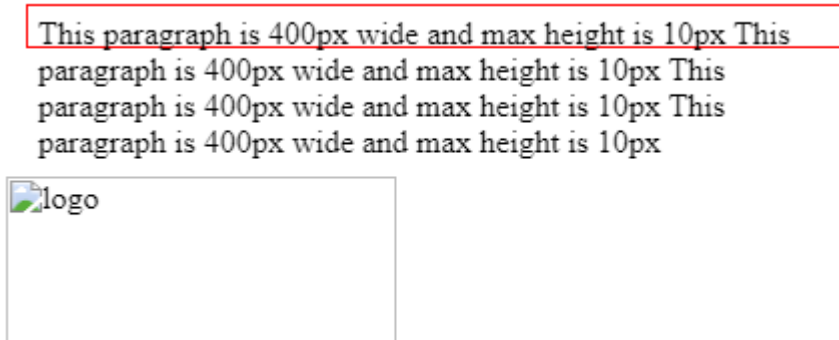
```
<html>
  <head>
  </head>
  <body>
    <p style = "width:400px; max-height:10px; border:1px solid red; padding:5px;
margin:10px;">
      This paragraph is 400px wide and max height is 10px
      This paragraph is 400px wide and max height is 10px
      This paragraph is 400px wide and max height is 10px
      This paragraph is 400px wide and max height is 10px
    </p>
    <br>
    <br>
```

```

<br>
<img alt = "logo" src = "/css/images/logo.png" width = "195" height = "84" />
</body>
</html>

```

It will produce the following result –



The min-height Property

The min-height property allows you to specify minimum height of a box. The value of the min-height property can be a number, a length, or a percentage.

NOTE – This property does not work in either Netscape 7 or IE 6.

Here is an example –

```

<html>
<head>
</head>
<body>
  <p style = "width:400px; min-height:200px; border:1px solid red;
padding:5px; margin:10px;">
    This paragraph is 400px wide and min height is 200px
    This paragraph is 400px wide and min height is 200px
    This paragraph is 400px wide and min height is 200px
    This paragraph is 400px wide and min height is 200px
  </p>
  <img alt = "logo" src = "/css/images/logo.png" width = "95" height = "84" />
</body>
</html>

```

It will produce the following result –

This paragraph is 400px wide and min height is 200px This
paragraph is 400px wide and min height is 200px This
paragraph is 400px wide and min height is 200px This
paragraph is 400px wide and min height is 200px



The max-width Property

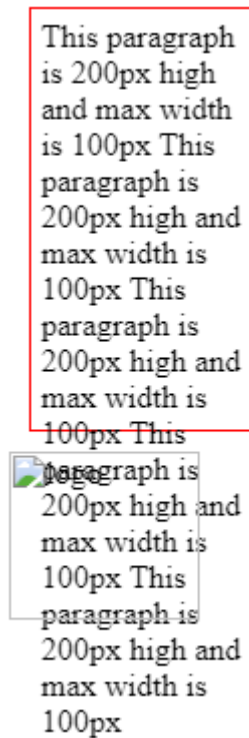
The max-width property allows you to specify maximum width of a box. The value of the max-width property can be a number, a length, or a percentage.

NOTE – This property does not work in either Netscape 7 or IE 6.

Here is an example –

```
<html>
  <head>
  </head>
  <body>
    <p style = "max-width:100px; height:200px; border:1px solid red;
padding:5px; margin:10px;">
      This paragraph is 200px high and max width is 100px
      This paragraph is 200px high and max width is 100px
      This paragraph is 200px high and max width is 100px
      This paragraph is 200px high and max width is 100px
      This paragraph is 200px high and max width is 100px
    </p>
    <img alt = "logo" src = "/images/css.gif" width = "95" height = "84" />
  </body>
</html>
```

This will produce following result –



The min-width Property

The min-width property allows you to specify minimum width of a box. The value of the min-width property can be a number, a length, or a percentage.

NOTE – This property does not work in either Netscape 7 or IE 6.

Here is an example –

```
<html>
  <head>
  </head>
  <body>
    <p style = "min-width:400px; height:100px; border:1px solid red;
padding:5px; margin:10px;">
      This paragraph is 100px high and min width is 400px
      This paragraph is 100px high and min width is 400px
    </p>
    <img alt = "logo" src = "/css/images/css.gif" width = "95" height = "84" />
  </body>
</html>
```

It will produce the following result –

This paragraph is 100px high and min width is 400px This paragraph is 100px high and min 400px

 logo

CSS - Scrollbars

CSS provides a property called ***overflow*** which tells the browser what to do if the box's contents is larger than the box itself. This property can take one of the following values –

Sr. No.	Value & Description
1	visible Allows the content to overflow the borders of its containing element.
2	hidden The content of the nested element is simply cut off at the border of the containing element and no scrollbars is visible.
3	scroll The size of the containing element does not change, but the scrollbars are added to allow the user to scroll to see the content.
4	auto The purpose is the same as scroll, but the scrollbar will be shown only if the content does overflow.

Here is an example –

```
<html>
<head>
  <style type = "text/css">
    .scroll {
      display:block;
      border: 1px solid red;
      padding:5px;
      margin-top:5px;
      width:300px;
      height:50px;
      overflow:scroll;
    }
    .auto {
      display:block;
      border: 1px solid red;
      padding:5px;
      margin-top:5px;
      width:300px;
      height:50px;
      overflow:auto;
```

```
}  
</style>  
</head>
```

```
<body>
```

```
<p>Example of scroll value:</p>
```

```
<div class = "scroll">
```

I am going to keep lot of content here just to show you how
scrollbars works if there is an overflow in an element box.

This provides your horizontal as well as vertical scrollbars.

```
</div>
```

```
<br />
```

```
<p>Example of auto value:</p>
```

```
<div class = "auto">
```

I am going to keep lot of content here just to show you how
scrollbars works if there is an overflow in an element box.

This provides your horizontal as well as vertical scrollbars.

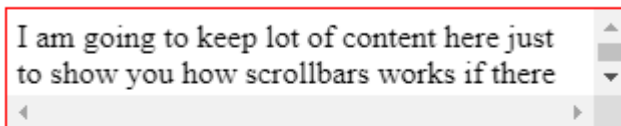
```
</div>
```

```
</body>
```

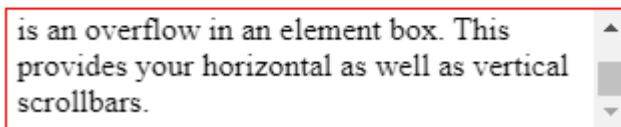
```
</html>
```

It will produce the following result –

Example of scroll value:



Example of auto value:



CSS - Layers

CSS gives you opportunity to create layers of various divisions. The CSS layers refer to applying the z-index property to elements that overlap with each other.

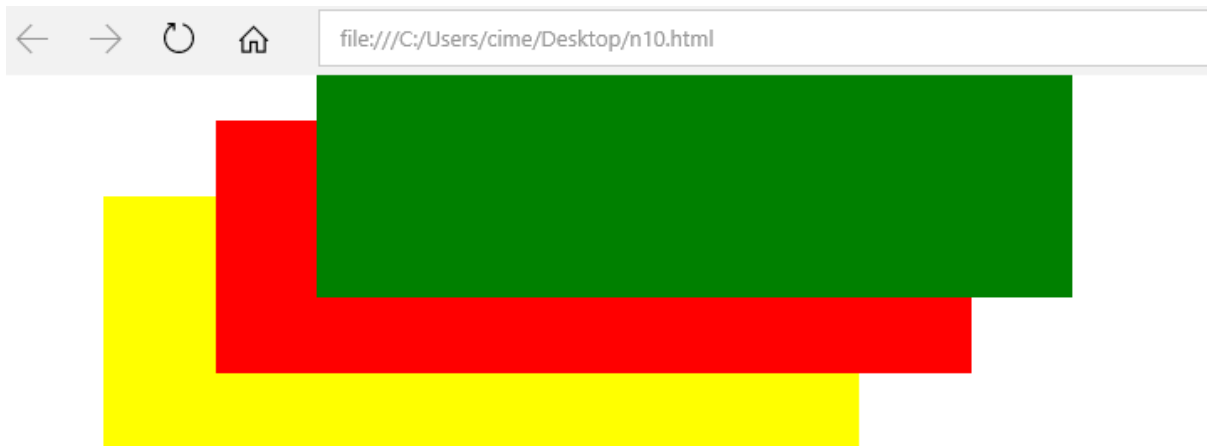
The z-index property is used along with the position property to create an effect of layers. You can specify which element should come on top and which element should come at bottom.

A z-index property can help you to create more complex webpage layouts.

Following is the example which shows how to create layers in CSS.

```
<html>
<head>
</head>
<body>
  <div style = "background-color:red;
    width:300px;
    height:100px;
    position:relative;
    top:10px;
    left:80px;
    z-index:2">
  </div>
  <div style = "background-color:yellow;
    width:300px;
    height:100px;
    position:relative;
    top:-60px;
    left:35px;
    z-index:1;">
  </div>
  <div style = "background-color:green;
    width:300px;
    height:100px;
    position:relative;
    top:-220px;
    left:120px;
    z-index:3;">
  </div>
</body>
</html>
```

It will produce the following result –



CSS id Attribute

- The HTML id attribute is used to specify a unique id for an HTML element.
- You cannot have more than one element with the same id in an HTML document.
- The id attribute specifies a unique id for an HTML element. The value of the id attribute must be unique within the HTML document.
- The id attribute is used to point to a specific style declaration in a style sheet. It is also used by JavaScript to access and manipulate the element with the specific id.
- The syntax for id is: write a hash character (#), followed by an id name. Then, define the CSS properties within curly braces { }.

In the following example we have an `<h1>` element that points to the id name "myHeader". This `<h1>` element will be styled according to the `#myHeader` style definition in the head section:

```
<head>
<style>
#myHeader {
  background-color: yellow;
  color: green;
  padding: 40px;
  text-align: center;
}
</style>
</head>
<body>
<h1 id="myHeader">CIME</h1>
```

</body>

</html>

O/P



CIME

Note:

- The id name is case sensitive!
- The id name must contain at least one character, cannot start with a number, and must not contain whitespaces (spaces, tabs, etc.).

CSS Class Attribute

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
.intro {  
  background-color: yellow;  
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1 class="intro">Demo of the .class selector</h1>
```

```
<div class="intro">
```

```
<p>My name is Donald.</p>
```

```
<p>I live in Duckburg.</p>
```

```
</div>
```

```
<p>My best friend is Mickey.</p>
```

```
<p class="intro">College of IT and Management Education (CIME) (formerly  
known as Centre for IT Education (CITE) came up as an initiative of the Orissa  
State Electronics Development Corporation Ltd. (A State Government Enterprise,  
Odisha) under Electronics and Information Technology Department of  
Government of Odisha on 4th August, 2000 to impart professional educational  
programs in the field of IT, Management and Engineering.</p>
```

</body>
</html>

O/P

Demo of the .class selector

My name is Donald.

I live in Duckburg.

My best friend is Mickey.

College of IT and Management Education (CIME) (formerly known as Centre for IT Education (CITE) came up as an initiative of the Orissa State Electronics Development Corporation Ltd. (A State Government Enterprise, Odisha) under Electronics and Information Technology Department of Government of Odisha on 4th August, 2000 to impart professional educational programs in the field of IT, Management and Engineering.

Difference between Class and ID

A class name can be used by multiple HTML elements, while an id name must only be used by one HTML element within the page:

The following are the important differences between Id and Class.

Sr. No.	Key	Id	Class
1	Syntax	In Html for an element ID name starts with the “#” symbol followed by a unique name assigned to it.	On the other hand class assigned to an element has its name starts with “.” followed by class name.
2	Selector	Only one ID selector can be attached to an element.	Multiple class selectors can be attached to an element.
3	Uniqueness	Id is unique in a page and can only apply to at most one element	The class can be applied to multiple elements so it could be multiple times on a single page.

CSS Layout - The position Property

The position property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).

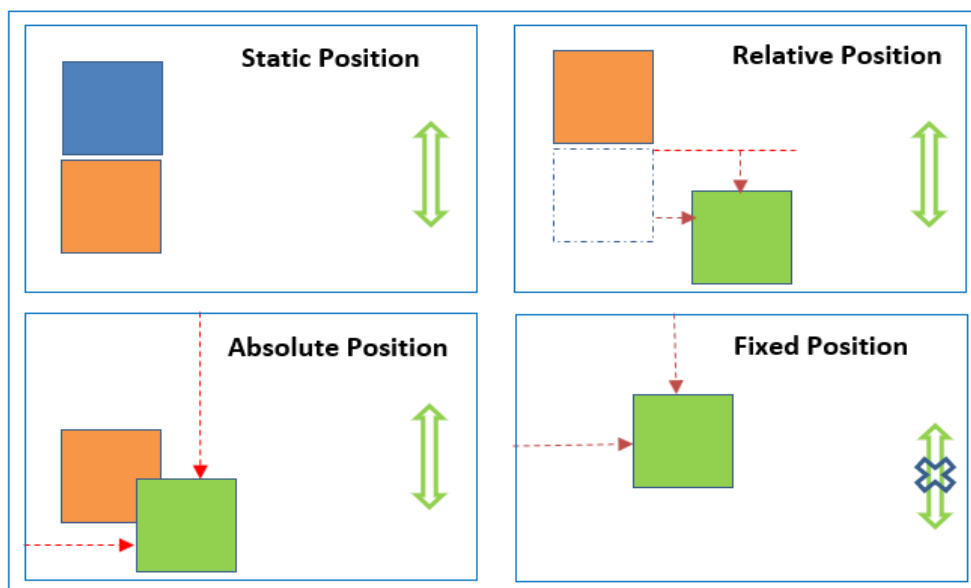
The position Property

The position property specifies the type of positioning method used for an element.

There are five different position values:

- static
- relative
- fixed
- absolute
- sticky

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the position value.



position: static;

HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

This <div> element has position: static;

Example

```
<!DOCTYPE html>
```



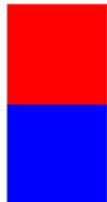
```

<html>
<head>
<style>
.p1 {
    width:50px;
    height:50px;
    background-color:red;
    position:static;
}
.p2 {
    width:50px;
    height:50px;
    background-color:blue;
}

</style>
</head>
<body>
    <div class="p1">
    </div>
    <div class="p2">
    </div>
</body>
</html>

```

O/P



position: relative;

An element with position: relative; is positioned relative to its normal position.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

This <div> element has position: relative;
Here is the CSS that is used:

Example

```
<!DOCTYPE html>
```

```

<html>
<head>
<style>
.p1 {
  width:50px;
  height:50px;
  background-color:red;
}
.p2 {
  width:50px;
  height:50px;
  background-color:blue;
  position:relative;
  left:60px;
  top:20px;
}
</style>
</head>
<body>
  <div class="p1">
  </div>
  <div class="p2">
  </div>
</body>
</html>

```

O/P



position: fixed;

An element with `position: fixed;` is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The `top`, `right`, `bottom`, and `left` properties are used to position the element.

A fixed element does not leave a gap in the page where it would normally have been located.

Notice the fixed element in the lower-right corner of the page. Here is the CSS that is used:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
.p1 {
  width:50px;
  height:50px;
  background-color:red;
  position:fixed;
  left:40px;
  top:40px;
}
</style>
</head>
<body>
  <div class="p1">
    </div>
</body>
</html>
```

O/P



This <div> element has position: fixed;

position: absolute;

An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

Note: Absolute positioned elements are removed from the normal flow, and can overlap elements.

Here is a simple example:

This <div> element has position: relative; This <div> element has position: absolute;

Here is the CSS that is used:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
.p1 {
  width:50px;
  height:50px;
  background-color:red;
}
.p2 {
  width:50px;
  height:50px;
  background-color:green;
}
.p3 {
  width:50px;
  height:50px;
  background-color:blue;
  position:absolute;
  left:60px;
  top:20px;
}
</style>
</head>
<body>
  <div class="p1">
  </div>
  <div class="p2">
  </div>
  <div class="p3">
  </div>
</body>
</html>
```

O/P



position: sticky;

An element with position: sticky; is positioned based on the user's scroll position.

A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed).

Note: Internet Explorer does not support sticky positioning. Safari requires a -webkit- prefix (see example below). You must also specify at least one of top, right, bottom or left for sticky positioning to work.

In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
.p1 {
  width:50px;
  height:50px;
  background-color:red;
  position:relative;
  left:50px;
  top:50px;
}
.p2 {
  width:50px;
  height:50px;
  background-color:green;
  position:absolute;
  right:50px;
  top:50px;
}
</style>
</head>
<body>
  <div class="p1">
  </div>
  <br/>
  <div class="p1">
  </div>
  <br />
```

```
<div class="p1">  
</div>  
<div class="p2">  
</div>  
<br />  
<div class="p2">  
</div>  
<br />  
<div class="p2">  
</div>  
</body>  
</html>
```

O/P



Difference between Relative position and Absolute CSS position (Relative vs Absolute position)

- Relative position places the element relative to its normal position and Absolute position places the element at the exact position specified.
- Relative position elements are placed based on normal document flow, but Absolute position elements are completely removed from normal document flow and place at the exact position.
- Absolute position elements are placed based on either parent element position when parent element position is relative/absolute OR document body (browser viewport).

Difference between Static position and Relative CSS position (Static position vs Relative position)

- Left/right/top/bottom properties are not applicable to static position elements but applicable to Relative position elements.
- Static position is a default position.

Static and Relative position similarities

- Relative position works like Static when left/right/top/bottom properties are not applied.
- Relative and Static position elements are placed based on normal document flow.

Difference between Static and Absolute CSS position (Static vs Absolute position)

- Static position is a default position.
- Static position elements are placed based on normal document flow, but Absolute position elements are completely removed from normal document flow and place at exact position.
- Left/right/top/bottom properties are not applicable to Static position elements but applicable to Absolute position elements.

Difference between Fixed and Absolute CSS position (Fixed vs Absolute position)

- Absolute position places the element at the exact position specified. This position element scrolls with the HTML page.
- Fixed position works like Absolute position, but it is relative to view port. This position element doesn't scroll with the HTML page.

Following different situations explains how CSS position works with parent and child elements when parent and child CSS positions are different.

Note: Irrespective of parent element position, if child element position is Static, Left/right/top/bottom properties are not applicable to child element.

Image Map:

The implementation of image maps requires the processing of the mouse click coordinates in conjunction with knowing the image areas and their associated hypertext links.

The processing can take place in one of two places:

- On the server where the Web pages are stored.
- In the Web browser.

Originally, the only option for processing image maps was to do so on the server. However, later versions of HTML have added features allowing client-side processing.

The basic idea behind an image map is that you combine two different components. i.e.

- A map of defined linked areas
 - An image
- ✓ The map is overlaid on the image, and the clickable areas coincide with portions of the image.
- ✓ In HTML the image and the clickable areas are coded separately. However, from the visitor's perspective, it appears that portions of the image itself are linked to different destination.

Client side image maps

- Client side image maps are generally more popular. With a client side image map, you can specify a list of areas that will be used as the links. This gives the user the chance to see immediately if where they are about to click is somewhere useful.
- There are four types of these areas; rectangles, circles, polygons and default.

Rectangle

✓ **This expects four coordinates.i.e.**

- horizontal position of the top-left corner
- vertical position (from the top of the image) of the top-left corner
- horizontal position of the bottom-right corner
- vertical position of the bottom-right corner

An example would be:

shape="rect" coords="10,20,75,40"

Circle

✓ **This expects three coordinates.i.e.**

- horizontal position of the centre
- vertical position of the centre
- radius of the circle (percentage radii are taken as a percentage of the shorter side of the image)

An example would be:

shape="circle" coords="50,80,20"

Polygon

✓ **This expects as many pairs of coordinates as you need to make your polygon.**

- These can make any polygon shapes you need, and can have sloping lines.
- All coordinates are specified as horizontal position then vertical position, with all of them in a long comma separated list. The last pair of coordinates can optionally match the first.

An example would be:

shape="poly" coords="217,305,218,306,218,306,228,316,243,316,243,325,229,325,229,322,217,310"

- Internet Explorer does not understand the default shape.

HTML Elements Used to Create Client Side Image Maps

There are three HTML elements used to create image maps:

- **img:** specifies the location of the image to be included in the map.
- **map:** is used to create the map of clickable areas.
- **area:** is used within the map element to define the clickable areas.

Steps of Client side image maps

Step 1: Determine the size of an image

```

```

Step 2: Create a map to overlay the image

The map code is quite simple. It looks like this:

```
<map name="map_example">
</map>
```

Step 3: Define the coordinates for the map shapes

We need to create shapes to overlay over the image. i.e.

- rectangle (rect),
- circle,
- polygon(poly)

We can create that shape, or area, in an HTML map by using the following code:

```
<area shape="poly" coords="130,147,200,107,254,219,130,228"
href="https://www.bput.ac.in/"
target="_blank" alt="BPUT" />
```

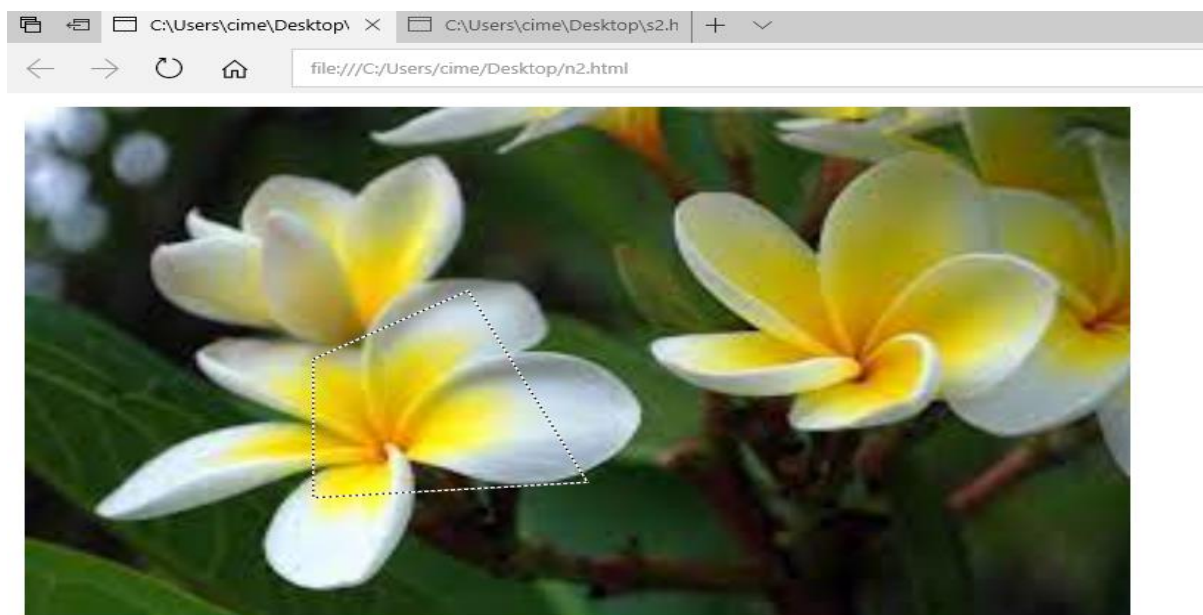
Step 4: Put it all together

We can combine the image, map, and shapes into a single block of code that looks like this:

```
<!DOCTYPE html>
<html>
<body>
<map name="map_example">
<area shape="poly" coords="130,147,200,107,254,219,130,228"
      href="https://www.bput.ac.in/"
      target="_blank" alt="BPUT" />
<area shape="poly" coords="130,147,130,228,6,219,59,107"
      href="http://cime.ac.in/"
      target="_blank" alt="CIME" />
<area shape="poly" coords="130,147,200,107,130,4,59,107"
      href="C:\Users\cime\Desktop\s2.html"
      target="_blank" alt="s2" /> </map>

</body>
</html>
```

O/P



What is the purpose of target _blank?

target="_blank" is a special keyword that will open links in a new tab every time. target="blank" will open the first-clicked link in a new tab, but any future links that share target="blank" will open in that same newly-opened tab.

usemap attribute

- The usemap attribute specifies an image as a client-side image map (an image map is an image with clickable areas).
- The usemap attribute is associated with a <map> element's name attribute, and creates a relationship between the and the <map>.
- The usemap attribute cannot be used if the element is a descendant of an <a> or <button> element.

Server-side image maps

Server-side image mapping uses images, .map files and CGI scripts. Often server-side image maps use the Imagemap programme.

The programme processing image maps needs to be on the server, usually in the cgi-bin/ directory.

The HREF attribute of server side imagemaps is always a hyperlink to the cgi-bin directory:

```
<a href="http://cime.ac.in/course.html">  
    
</a>
```

In fact, the HREF path can sometimes be interpreted as a combination of two paths:

The location of the "imagemap" programme, such as /cgi-bin/imagemap

The location of the (coordinate) map for the image, user_dir/image.map. This describes the various areas of the image.

ismap attribute

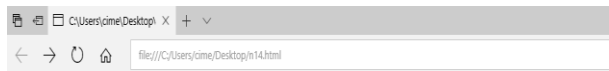
The **ismap attribute** is a boolean attribute. When present, it specifies that the image is part of a server-side image map (an image map is an image with clickable areas). When clicking on a server-side image map, the click coordinates are sent to the server as a URL query string.

A major disadvantage of server-side image maps is that their implementation can differ between different server systems. This means that the author of the website using image maps needs to communicate closely with the Web server's administrator to ensure that the maps work correctly.

Ex

```
<!DOCTYPE html>
<html>
<body>
<h1>The img ismap attribute</h1>
<a href="http://cime.ac.in/course.html">
  
</a>
<p>Click the image, and the click coordinates will be sent to the server as a URL
query string.</p>
</body>
</html>
```

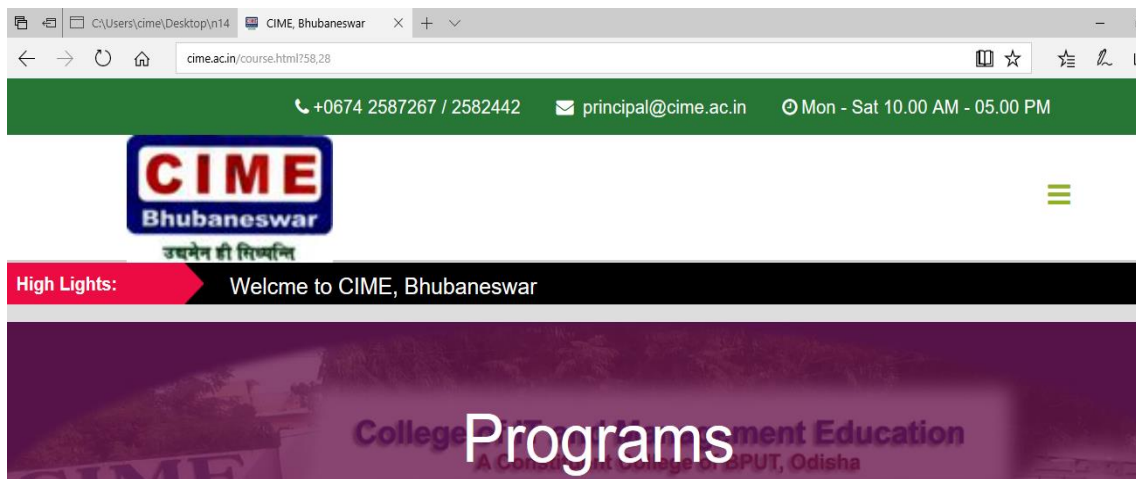
O/P



The img ismap attribute



Click the image, and the click coordinates will be sent to the server as a URL query string.



Client-side image maps vs Server-side image maps

Clients-side image maps have a number of advantages over server-side maps.

- Reduced server load, since the mapping is done on the client.
- There is no need to communicate with server side scripts.
- The user can see the image maps associated hypertext URLs in the status bar of their browser before clicking.

Dynamic Hypertext Markup language (DHTML)

DHTML stands for **Dynamic Hypertext Markup language** i.e., **Dynamic HTML**.

Dynamic HTML is not a markup or programming language but it is a term that combines the features of various web development technologies for creating the web pages dynamic and interactive.

The DHTML application was introduced by Microsoft with the release of the 4th version of IE (Internet Explorer) in 1997.

Components of Dynamic HTML

DHTML consists of the following four components or languages:

- HTML 4.0
- CSS
- JavaScript
- DOM.

HTML 4.0

HTML is a client-side markup language, which is a core component of the DHTML. It defines the structure of a web page with various defined basic elements or tags.

CSS

CSS stands for Cascading Style Sheet, which allows the web users or developers for controlling the style and layout of the HTML elements on the web pages.

JavaScript

JavaScript is a scripting language which is done on a client-side. The various browser supports JavaScript technology. DHTML uses the JavaScript technology for accessing, controlling, and manipulating the HTML elements. The statements in JavaScript are the commands which tell the browser for performing an action.

DOM

DOM is the document object model. It is a w3c standard, which is a standard interface of programming for HTML. It is mainly used for defining the objects and properties of all elements in HTML.

Uses of DHTML

Following are the uses of DHTML (Dynamic HTML):

- It is used for designing the animated and interactive web pages that are developed in real-time.
- DHTML helps users by animating the text and images in their documents.
- It allows the authors for adding the effects on their pages.
- It also allows the page authors for including the drop-down menus or rollover buttons.
- This term is also used to create various browser-based action games.
- It is also used to add the ticker on various websites, which needs to refresh their content automatically.

Features of DHTML

Following are the various characteristics or features of DHTML (Dynamic HTML):

- Its simplest and main feature is that we can create the web page dynamically.
- **Dynamic Style** is a feature, that allows the users to alter the font, size, color, and content of a web page.
- It provides the facility for using the events, methods, and properties. And, also provides the feature of code reusability.
- It also provides the feature in browsers for data binding.
- Using DHTML, users can easily create dynamic fonts for their web sites or web pages.
- With the help of DHTML, users can easily change the tags and their properties.
- The web page functionality is enhanced because the DHTML uses low-bandwidth effect.

Difference between HTML and DHTML

Following table describes the differences between HTML and DHTML:

HTML (Hypertext Markup language)	DHTML (Dynamic Hypertext Markup language)
1. HTML is simply a markup language.	1. DHTML is not a language, but it is a set of technologies of web development.
2. It is used for developing and creating web pages.	2. It is used for creating and designing the animated and interactive web sites or pages.
3. This markup language creates static web pages.	3. This concept creates dynamic web pages.
4. It does not contain any server-side scripting code.	4. It may contain the code of server-side scripting.
5. The files of HTML are stored with the .html or .htm extension in a system.	5. The files of DHTML are stored with the .dhtm extension in a system.
6. A simple page which is created by a user without using the scripts or styles called as an HTML page.	6. A page which is created by a user using the HTML, CSS, DOM, and JavaScript technologies called a DHTML page.
7. This markup language does not need database connectivity.	7. This concept needs database connectivity because it interacts with users.