Chapter 1

Introduction to Computer Network and it's Applications

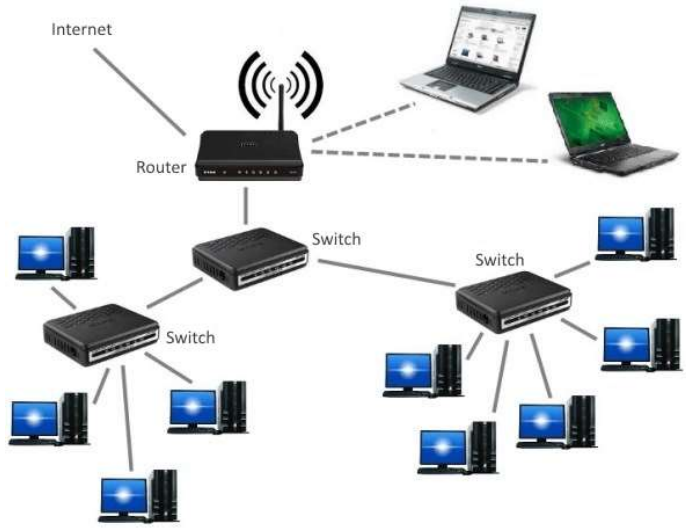
CREATIVE

❖ Computer Network :

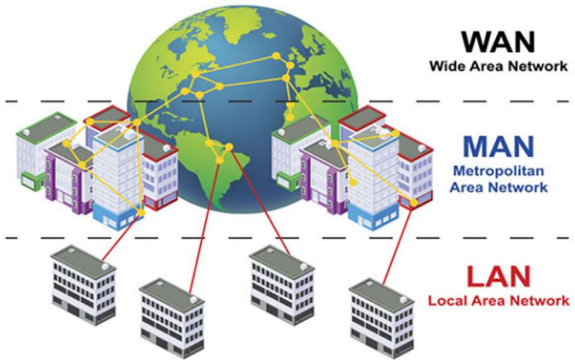
- A computer network is a collection of two or more computer systems that are linked together.
- In other words, a computer network is a system that connects numerous independent computers in order to share information (data) and resources.
- The integration of computers and other different devices allows users to communicate more easily.
- A network connection can be established using either cable or wireless media. Hardware and software are used to connect computers and tools in any network.
- A computer network consists of various kinds of nodes. Servers, networking hardware, personal computers, and other specialized or general-purpose hosts can all be nodes in a computer network. Hostnames and network addresses are used to identify them.

❖ Need of Networking:

- **Sharing informations and resources:** Programs do not have to execute on a single system because of resource and load sharing. We can shares resources like printers, storage devices etc. over the computer network.
- **Reduced costs** – Multiple machines can share printers, tape drives, and other peripherals.
- **Reliability** – If one machine fails, another can take its place.
- **Scalability** - It's simple to add more processors or computers.
- **Communication and mail** - People living apart can work together.
- **Information Access** - Remote information access, access to the internet, e-mail, video conferencing, and online shopping.
- **Entertainment**: - That is interactive (online games, videos, etc.)
- **Social Networking:** We can access various social media platform.



❖ Type of Computer Network:

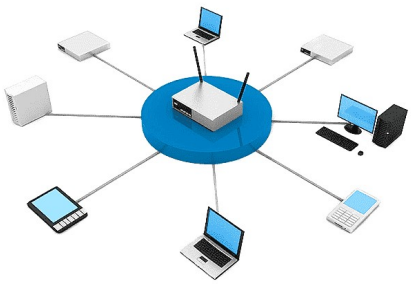


→ A computer network can be categorized by their size. A computer network is mainly of four types:

1. **LAN**(Local Area Network)
2. **MAN**(Metropolitan Area Network)
3. **WAN**(Wide Area Network)
4. **PAN**(Personal Area Network)

⊙ LAN(Local Area Network):

→ Local Area Network is a group of computers connected to each other in a small area such as building, office, compound etc.



→ LAN is used for connecting two or more personal computers through a communication medium such as twisted pair, coaxial cable, etc.

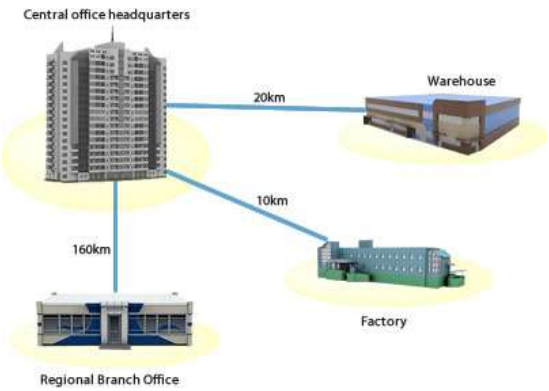
→ It is less costly as it is built with inexpensive hardware such as hubs, network adapters, and ethernet cables.

→ The data is transferred at an extremely faster rate in Local Area Network.

→ Local Area Network provides higher security.

→ A LAN is a network that covers an area of around 10 kilometers. Depending upon the needs of the organization, a LAN can be a single office, building, or Campus. We can have two PCs and one printer in-home office or it can extend throughout a company and include audio and video devices.

⊙ MAN(Metropoliten Area Network):



→ A metropolitan area network is a network that covers a larger geographic area by interconnecting a different LAN to form a larger network.

→ MAN refers to a network that covers an entire city. For example: consider the cable television network.

→ Government agencies use MAN to connect to the citizens and private industries.

→ In MAN, various LANs are connected to each other through a telephone exchange line.

CREATIVE

→ It has a higher range than Local Area Network(LAN).

⊙ WAN (Wide Area Network):

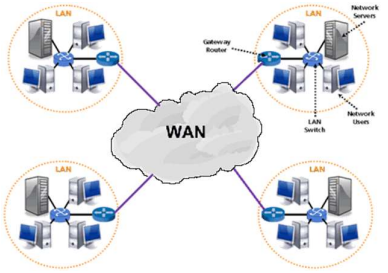
→ WAN refers to a network that connects countries or continents. For example, the Internet allows users to access a distributed system called www from anywhere around the globe.

→ A Wide Area Network is quite bigger network than the LAN and MAN.

→ A Wide Area Network is not limited to a single location, but it spans over a large geographical area through a telephone line, fibre optic cable or satellite links.

→ The internet is one of the biggest WAN in the world.

→ A Wide Area Network is widely used in the field of Business, government, and education.



⊙ PAN (Personal Area Network):

→ Personal Area Network is a network arranged within an individual person, typically within a range of 10 meters.

→ Personal Area Network is used for connecting the computer devices of personal use is known as Personal Area Network.

→ Thomas Zimmerman was the first researcher scientist to bring the idea of the Personal Area Network.

→ Personal Area Network covers an area of 30 feet.

→ Personal computer devices that are used to develop the personal area network are the laptop, mobile phones, media player and play stations.

❖ Internet Terminology:

→ The Internet made up of two words; “Inter” and “Net” . Where Inter means Internally connected and Net means Network.

→ Hence, by combining two words, it becomes one word “Interconnected Networks” called in short Internet.

→ In other words, internet is a collection of multiple networks. Hence internet is a huge network of several small computer networks like PAN, LAN, MAN, WAN.

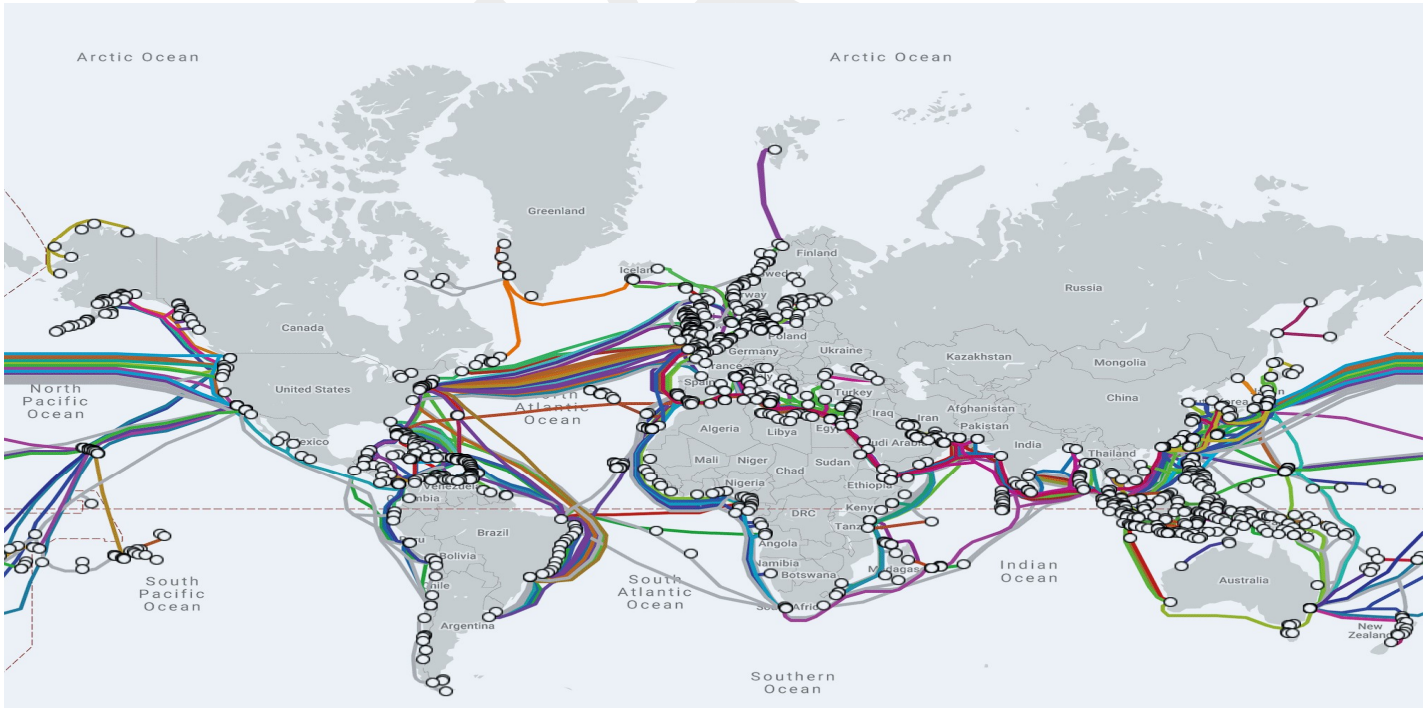
→ We can say that internet is a network of network or meta-network that can be expanded upto the entire globe.

→ The Internet is a global network connecting millions of computers. More than 100 countries are linked into exchanges of data and informations. According to Internet World

Stats, as of December 31, 2011 there was an estimated 2,267,233,742 Internet users worldwide. This represents 32.7% of the world's population.

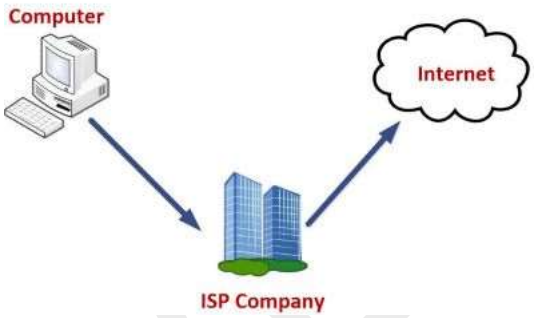
❖ History of Internet :

- → The very first computer network was developed in ARPANET in 1969. It was developed by Advanced Research Projects Agency (APRA) of the Department of Defence, U.S.
- → Next step was commercialising the usage and making the transistors and transmitters fit in smaller devices for convenient Internet usage for the general public. This was introduced in the 1970s.
- → Moving forward, satellites and wireless communication was the main target. Defence Advanced Research Projects Agency (formerly ARPA), supported satellite-based radio packets for mobile usage of networks.
- → The next was the development of TCP. This enabled different machines and networks across the world to assemble data packets. It was in the 1980s that the TCP/IP approach was adapted by researchers and technologists.
- → In 1993, the web browser was introduced, which followed the point-and-click approach and is now a widely used operation for Internet users.
- → The late 1990s was the time when thousands of Internet Service Providers has taken up the market and most of them were from the U.S.
- → And then the 21st century brings in an integration of technology and wireless Internet accessibility for its users. Wherein, wireless broadband services came for internet connections.



Internet Service Provider (ISP):

- 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 device and the internet is executed through a specific transmission technology that involves the transfer of information packets using some Netwotk routing Devices and Protocols.
- In short, ISPs are the medium throug which we can interect with the internet.
- Just like a TV cable network operator who provide us a cable connection through which we can watch the bunch of TV channels in our TV. As we have to pay some charges against the appropriate channel packages.



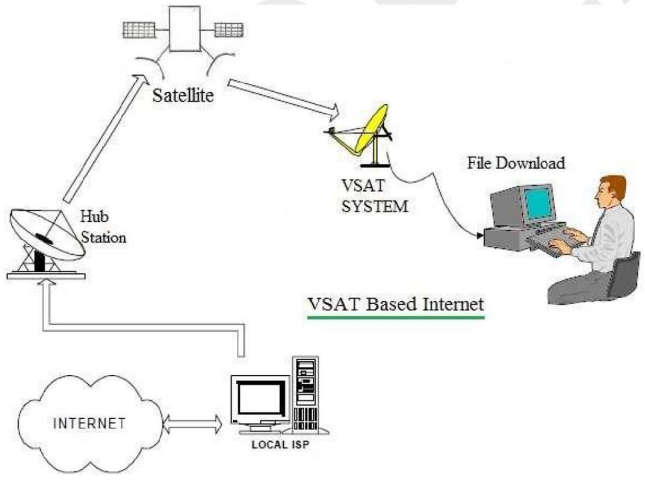
- As per above example, ISPs provide us a service that allows us to connect our devices to the entire network. Against these services we have to pay some charges or we have to subcribe that payable services. To obtain such services we have to register our identity to them.
- Data is transmitted through 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. This method has become outdated today due to slow connection speed.
- **DSL:** DSL, which stands for 'digital subscriber line' is an advanced version of the dial-up Internet access method.
- **Wireless Broadband (WiBB):** It is a modern broadband technology for Internet access. It allows high-speed wireless internet within a large area.
- **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.
- **ISDN:** It is a short form of Integrated Services Digital Network. 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.
- There are three types of ISPs are as under:
- **Tier-1 ISPs:** Tier-I ISPs are those who established their network at globally. They connects networks of other countries.
- **Tier-2 ISPs:** Tier-2 ISPs are those who established their network country level. They connects networks of entire country within a country. They transfers their traffic towards the Tier-1 ISPs.
- **Tier 3 ISPs:** These ISPs connect customers to the internet using by forwarding traffics towards Tier-2 ISP's network.

Intranet:

- The intranet is a private network that belongs to a particular organization.
- It is designed for the exclusive use of an organization and its associates, such as employees, customers, and other authorized people.
- It offers a secure platform to convey information and share data with authorized users.
- Confidential information, database, links, forms, and applications can be made available to the staff through the intranet.
- So, it is like a private internet or an internal website that is operating within an organization to provide its employees access to its information and records.
- Each computer in intranet is identified by a unique IP Address.
- It is based on internet protocols (TCP/IP) and is protected from unauthorized access with firewalls and other security systems.
- So, users on the intranet can access the internet, but the internet users can't access the intranet if they are not authorized for it.
- Furthermore, to access the intranet, the authorized user is required to be connected to its LAN.

VSAT (Very Small Aperture Terminal):

- VSAT (Very Small Aperture Terminal) is a technology that is used for effective communication in remote areas using Satellites and components located on ground.
- The concept of geostationary orbit was first proposed by Russian theorist Konstantin Tsiolkovsky.
- Development of active satellite communication was carried out in the 1960s by NASA. The first of these was the Syncom 1-3 satellites of which Syncom 3 transmitted live coverage of the 1964 Tokyo Olympics to viewers around the world.
- VSAT is of importance, especially in remote areas such as the hilly mountain regions, where Internet connectivity cannot be directly provided.



→There is no option left in such places through which the Internet can be accessed anyway. These places could be the Ocean region and the Sea region, where the citizens have a lack of utilities.

→In such cases, VSAT is one of the technologies that make Internet Connectivity possible so that people can access it. All the work is carried out in a workstation group where the sending and receiving of signals takes place. The data such as audio, video, etc, can be sent and received.

- Therefore, VSAT is considered to be a private Earth Station. This Earth Station is designed in a way such that the signals are transmitted and received with the help of satellites.
- The term 'Very Small' in VSAT (Very Small Aperture Terminal) refers to the antenna size that is very small.

How the Signals are Sent and Received:

- A satellite transponder will receive or send signals to the transceiver located on earth.
- There will be a hub station located on earth which will send or receive signals from satellites.
- Satellite will help in connecting all the users via the hub station.
- Hence for any communication to work, the signal will go from one user to the hub, which will be transmitted to the satellite, which will in turn be transmitted to the intended user.
- In such a way Satellite become a hub for their users and hence it makes the Star Topological Network connections among Satellite and connected devices.

Characteristics of VSAT:

- VSAT is a two-way communication earth satellite station that has an antenna of less than 3 meters.
- The size of the VSAT antenna may vary from 75 cm to 1.2 meters.
- The data ranges between 4 kilobytes per second to 16 megabytes per second in VSAT.

Working for VSAT:

- The end user of the set requires a box that acts as an interface between the user and its system.
- The box consists of an antenna and a transceiver.
- The work of this transceiver is to send and receive signals to and from the transponder located in space.
- Every workstation works like a hub. The satellite sends or receives a signal from it.
- Each user is connected to each of the interconnected hub stations and operates in the form of a Star Topology (There is a server in between and all the nodes are connected to it, forminga star-like figure).
- With this medium, we can transmit data, voice, and video signals.

- **Advantages of VSAT:**
- VSAT Terminals and its Hardware can be installed in vehicles such as Trucks or Vans and can also be used in situations where mobility is a need.
- Audio, Video, and data signals can be transmitted and received efficiently.
- Internet Access: A VSAT Network also serves to provide internet access in addition to pointing to a WAN link.
- It is set in the consumer broadband industry to make it the leading major wave. VSAT network provides an “Always on” broadband Internet Service.
- Information from remote locations can be accessed using satellites.
- It is more used for connectivity in rural areas, ships, and coastal regions.
- Mobile access is another conventional strength of satellite networks. For example, we can do online surfing, watch TV, use applications, and much more.
- VSAT Networks are not affected by earthquakes, Cyclones, and other natural calamities.

→ VSAT Networks, with a low-cost architecture and extra powerful systems, share digital information.

Disadvantages of VSAT:

→ Delay: VSAT Technology uses satellites in Geosynchronous orbit. This type of data transmission has an approximate delay of about 500 milliseconds for each round trip. It introduces a problem with the application that requires a consistent transmission.

→ Environmental conditions: Like other satellite systems, the VSAT network may also get affected by the weather and other environmental conditions. The signal strength may be weak at times, although it depends upon the size of the antenna, frequency band, and the power of the transmitters.

→ Clear view: Since VSAT requires an external antenna, therefore to contact the satellite, the location must have a clear view of the southern sky.

⊙ VSAT / WLL (Wireless Local Loop)

→ Wireless Local loop is a circuit line from a subscriber's phone to the local central office (LCO). But the implementation of local loop of wires is risky for the operators, especially in rural and remote areas due to less number of users and increased cost of installation. Hence, the solution for it is the usage of wireless local loop (WLL) which uses wireless links rather than copper wires to connect subscribers to the local central office.

→ Features:
- Data transmission rates from 2 to 155Mbit/s
- Internet access or telephony support
- No engineering work needed
- The aerial is moved if you move to another location.

Advantages of WLL:

→ Low cost due to no use of conventional copper wires.

→ Much more secure due to digital encryption techniques used in wireless communication.

→ Highly scalable as it doesn't require the installation of more wires for scaling it.

❖ WWW (World Wide Web):

→ A particular collection of web pages that belong to a specific URL is called a website, e.g., www.facebook.com, www.google.com, etc.

→ So, the World Wide Web is like a huge electronic book whose pages are stored on multiple servers across the world.

→ So, the web provides a communication platform for users to retrieve and exchange information over the internet.

→ Some people thinks 'internet' and 'World Wide Web' are same thing. but it is not so.

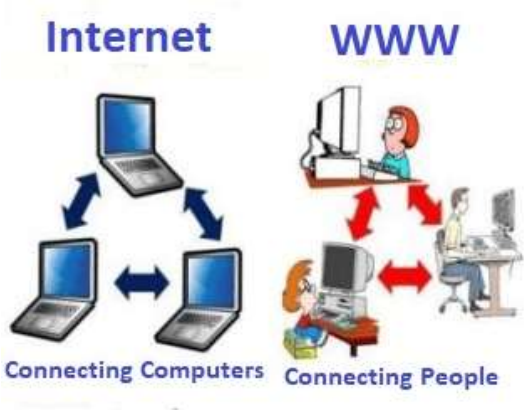
→ Internet is entirely different from WWW. It is a worldwide network of devices like computers, laptops, tablets, etc. It enables users to communicate users each others online. For example, when you chatting with someone online, you are using the internet. But,

when you have opened a website like google.com for information, you are using the WWW; a network of servers over the internet.

❖ History of WWW :

- The WWW was invented in a CERN by a British scientist, **Tim Berners-Lee** in 1989.
- Originally, it was developed by him to fulfill the need of automated information sharing between scientists across the world, so that they could easily share the data and results of their experiments and studies with each other.

- CERN, where Tim Berners worked, is a community of more than 1700 scientists from more than 100 countries. These scientists spend some time on CERN site, and rest of the time they work at their universities and national laboratories in their home countries, so there was a need for reliable communication tools so that they can exchange information.
- Internet and Hypertext were available at this time, but no one thought how to use the internet to link or share one document to another. Tim focused on three main technologies that could make computers understand each other, HTML, URL, and HTTP. So, the objective behind the invention of WWW was to combine recent computer technologies, data networks, and hypertext into a user-friendly and effective global information system and now it is very popular.
- Originally, it was developed by him to fulfill the need of automated
- information sharing between scientists across the world, so that they could easily share the data and results of their experiments and studies with each other.
- CERN, where Tim Berners worked, is a community of more than 1700 scientists from more than 100 countries. These scientists spend some time on CERN site, and rest of the time they work at their universities and national laboratories in their home countries, so there was a need for reliable communication tools so that they can exchange information.
- Internet and Hypertext were available at this time, but no one thought how to use the internet to link or share one document to another. Tim focused on three main technologies that could make computers understand each other, HTML, URL, and HTTP. So, the objective behind the invention of WWW was to combine recent computer technologies, data networks, and hypertext into a user-friendly and effective global information system and now it is very popular.



⊙ Search Engine:

- As the name given, Search Engine is one type of Engine or Machine that helps us to search some appropriate information from WWW according to the keywords that we are types into it.
- In simple words, Search Engine is Online Answering Engine or Machine which answering you based on your question you asked as a keywords.
- Here, engine or machine means a software that is available online any time whenever we searche using any of the browsers.
- Search Engine is a type of software that not required to be installed on your device, it works online using any web browser.
- Google, Yahoo, Bing, Yandex, DuckDuckGo, Baidu, Ask, AOL etc. are the most popular Search Engines.
- Search engines are generally working on three parts that are crawling, indexing, and ranking.

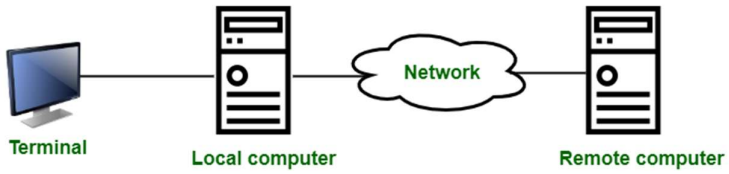
⊙ Remote Login:

- Remote Login is a process in which user can login into remote site i.e. computer and use services that are available on the remote computer. With the help of remote login a user is able to understand result of transferring, result of processing from the remote computer to the local computer.
- Remote Login is the ability for an authorized person to access a computer or network from a geographical distance through a network connection.
- Remote Login enables users to connect to the systems they need when they are physically far away. This is especially important for employees who work at branch offices, are traveling or telecommute.
- Remote Login enables remote users to access files and other system resources on any devices or servers that are connected to the network at any time.
- Technical support professionals can use remote access to connect to users' computers from remote locations to help them resolve issues with their systems or software.
- Remote Login can connect remote PC/Device using two protocols:
  - Rlogin
  - Telnet

❒ RLogin:

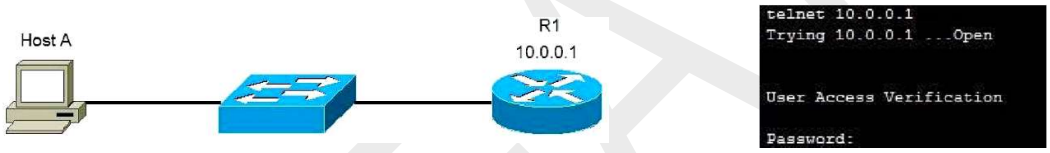
- RLogin is a protocol from Berkeley Unix Operating System. It was developed to work between UNIX operating systems only. It means this protocol is developed to remotely communicate those computers, whose operating system is based on UNIX.
- Rlogin communicates with other PC's over a TCP/IP connection.
- After some time, it has been ported to other operating systems also.
- rlogin is a Unix command that allows an authorized user to login to other Unix machines on a network and to interact as if the user were physically at the host computer.
- Once logged into the host, the user can do anything that host has given permission for, such as read, edit, or delete files.

CREATIVE



❒ Telnet:

- Telnet protocol is a standardapplication protocol that almost every TCP/IP implementation provides.
- It is oldest protocol. Itworks between hosts that use different operating systems.
- Telnet is an application protocol that allows a user to communicate with a remote device. A user on a client machine can use a software (known as a Telnet client) to access a command-line interface of another, remote machine that is running a Telnet server program.
- Telnet is often used by network administrators to access and manage remote devices.
- The network administrator wants to use his computer (Host A) to access and manage the router (R1). The administrator will start a Telnet client program on Host A and enter the IP address of the router R1 (telnet 10.0.0.1)
- The administrator can now manage the remote device (R1) from his own computer.



⊙ E-Mail (Electronic Mail):

- Email stands for Electronic Mail. It is a method to sends messages from one computer to another computer through the internet.
- It is mostly used in business, education, technical communication, document interactions.
- It allows communicating with people all over the world without troubling them.
- It is the information sent electronically between two of more people over a network. It involves a sender and receiver/s.
- The age of email services is older than ARPANET and the Internet. Email services started in 1971 by Ray Tomlinson and now It becomes a professional way to communicate.
- Email services are used in various sectors, organizations, either personally, or among a large group of people.

CREATIVE

- **Advantages of Email Services**

→ Easy and Fast: Composing an email is very simple and one of the fast ways to communicate. We can send an email within a minute just by clicking the mouse.
→ Mass Sending: We can easily send a message to many people at a time through email. Suppose, a company wants to send holiday information to all employees than using email, it can be done easily.
→ Multimedia Email: Email offers to send multimedia, documents, images, audio files, videos, and various types of files.
→ Disadvantages of Email Services
→ Malicious Use: hackers can send viruses through email because sometimes the spam feature unable to classify suspicious emails.
→ Spam: To improve spam feature sometimes some important email is transferred into spam without any notification.
→ Time Consuming: Responding through an email takes more time rather than other message services like WhatsApp, Telegram, etc. Email is good for professional discussion but not good for casual chatting.

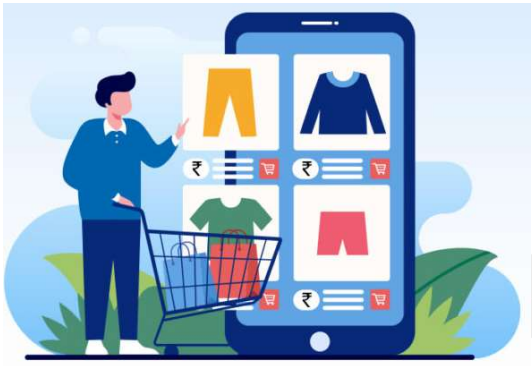
⊙ **E-Commerce:**

→ E-Commerce stands for electronic commerce is the trading of goods and services on the internet.
→ In other words, E-Commerce is nothing but a buying and selling activity of goods and services without visiting the shop or place from where it is available or manufactured.
→ Rather than phisical visit we can avail commercial services via the commercial websites through the help of internet.
→ An ecommerce website is your digital storefront on the internet. It facilitates the transaction between a buyer and seller. It is the virtual space where you showcase your products, and your online customers make their selections.
→ Businesses might create a branded store experience on a store like Amazon, build their own commerce site on a dedicated domain.

- Many types of E-Commerce are as under:
- B2C – Business to Consumer. Businesses sell to individual consumers (end-users). The most common model with many variations.
- B2B – Business to Business. Businesses sell to other businesses. Often the buyer resells products to the consumer.
- C2B – Consumers sell to businesses. C2B businesses allow customers to sell to other companies.
- C2C – Consumer to Consumer. Consumers sell to other consumers. Businesses create online marketplaces that connect consumers.
- B2G – Businesses sell to governments or government agencies.
- C2G – Consumers sell to governments or government agencies.
- G2B – Governments or government agencies sell to businesses.
- G2C - Governments or government agencies sell to consumers.

M-Commerce:

- E-Commerce stands for Mobile Commerce is the trading of goods and services on the internet using wireless handheld (mobile) devices.
- For Ex: We are purchasing laptop online from Amazon / Flipkart's application from our mobile device, such an activity is kind of E-commerce activity but we done it using our mobile device, that's why it is called M-Commerce.
- Another example of m-commerce is mobile banking.
- It is subcategory of ecommerce which does the same this via mobile devices.
- The emerging technology behind m-commerce, which is based on the Wireless Application Protocol (WAP), has made far greater strides in Europe, where mobile devices equipped with Web-ready micro-browsers are much more common than in the United States.



E-Business:

- E-Business refers to performing all type of business activities through internet.
- It includes activities like procurement of raw materials/goods, customer education, supply activities, buying and selling product, making financial transactions etc over internet.
- Internet, intranet, extranet are used in e-business.
- To do the business online or over the internet websites, apps, ERP, CRM etc are required.
- **Activities / Components of E-Business are :**
  - Online store setup.
  - Customer education.
  - Buying and selling product.
  - Financial business transaction.
  - Supply Chain Management.
  - E-mail marketing.

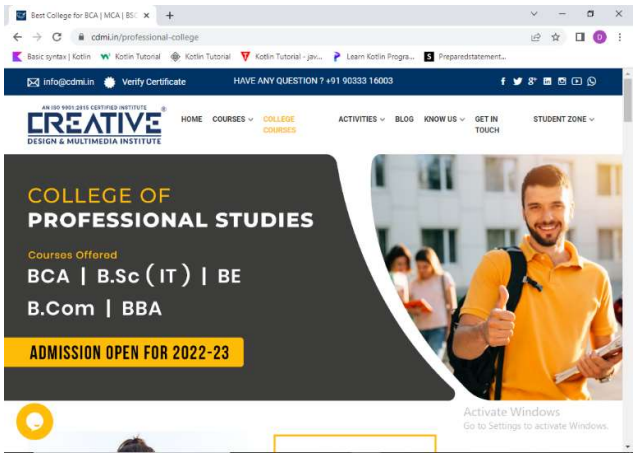
⊙ E-Governance:

- Electronic Governance or E-Governance is the application of Information and Communication Technology (ICT) for providing government services.
- In short, The various services provided by the various governments like central, states, municipal etc. These services served Electronically i.e. online using internet is known as E-Governance. For Ex: Online Aadhar issuence.
- Through the means of e-governance, government services are made available to citizens in a suitable, systematic, and transparent mode.
- All the services that available at certain phisical place is bothering citizens because they consumes time as well as it affect economically as well.
- Through the E-governance platform, governments provide some application to avail the services from there. Hence, citizen can take advantage of these government services from their devices like PC, Mobile etc.
- E-Governance is an initiative towards the Digital India, due to this much more data that were available only in phisical files, now available digitally as well.

❖ Website Basics:

⊙ Web Pages:

- Web Page is a one type of page that can stores relative infromation of website.
- As we know that WWW is a collection of information, the information that stored into WWW is arranged in specific format that is Web Page. Hence whenever we want some information from WWW, it is displayed as web page in the browsers like google chrome, mozilla firefox, internet explorer etc.
- In other words, A webpage is a document written in HTML and can be viewed on any web browser. It is contained within the web server, which can be accessed by entering the URL for that web page, and once it is loaded, it appears on the user's web browser.
- Each webpage is linked with a unique URL; hence two pages cannot have the same URL.
- A webpage may contain text, links for other pages, graphics, audio, videos, animations etc. Moreover, it is mainly used to provide information to the user in text, images, etc. Webpage is main building block of the WWW.

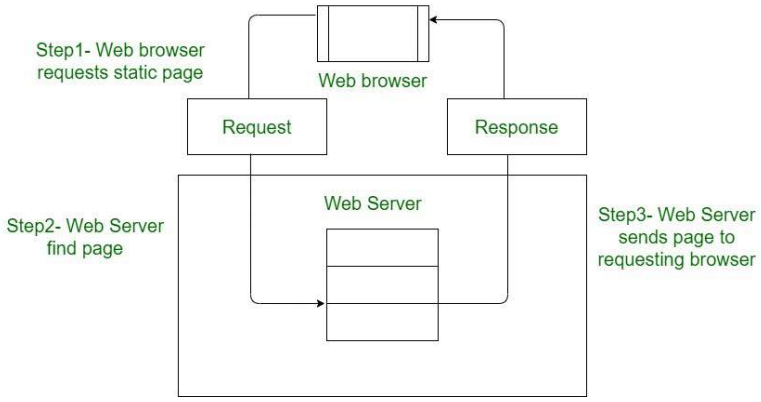


⦿ Website:

- A website is a collection of multiple webpages that are made by HTML. These webpages are internally connected with one page called home page using hyper links and this now becomes a website.
- For Ex: Website is like a textbook. As textbook is a collections of related pages. The information that we want to read is written in the page of book. Same way web pages that contains information of our need are collectively becomes a website.
- To make your website available to every person in the world, it must be stored or hosted on a computer connected to the Internet. Such computers are known as a Web Servers.
- In other words, As the name suggests, a website is a "site" on the "web" where you can post information about yourself, your company, or any other subject that internet users can access.
- There are mainly three types of websites:
  - Static Website
  - Dynamic Website
  - Responsive Website

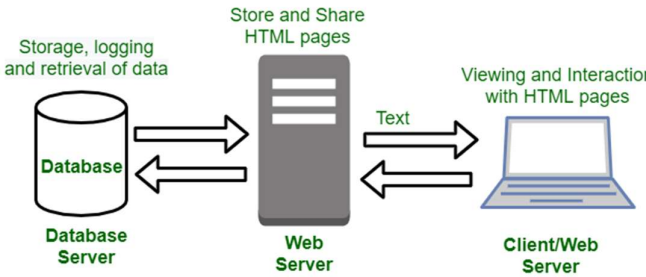
❒ Static Website:

- Static website is only informative website.
- A static website's information does not change automatically, which means it remains static (fixed) or the same for all visitors of the site.
- Static website is created using HTML. Some websites offers attractive designs and structures that can be implemented by CSS. A simple JavaScript also used in development and designing of static website.
- Web pages are returned by the server with no change therefore, static Websites are fast. There is no interaction with databases. Also, they are less costly as the host does not need to support server-side processing with different languages.
- Static does not mean that it will not respond to user actions, These Websites are called static because these cannot be modified on the server or interact with databases.
- There is no Server side coding required in static website.
- Ex. wikipedia.org



❒ Dynamic Website:

- Dynamic website is the website whose content / information is dynamic (not fixed) but changed frequently.
- A dynamic website or webpage has information, which is generated in real-time and changed on the basis of:
- Website visited by which user based on Login Id and Password.
- Website visited by user from which time zone i.e. From which geographical area.
- Website visited at which time of the day.
- Website visited from which platform i.e. from browser or any other application etc.
- For example, Facebook is dynamic website. Because you can't access Facebook without logged in. Based on your login-id and password, server will provide you only your Facebook page and this page changes automatically based on new things added by one of your friend.
- Dynamic website is developed using HTML, XML, CSS, JavaScript as frontend or client-side and PHP, WordPress, JSP, ASP, ASP.Net, Python, J2EE, Node JS, ect. as a backend or server-side.
- Dynamic websites needs to access an external file or a database to get information. For Ex. When you visit the Facebook, you will be asked either for login and password or sign up for creating new account. If you already a registered user, your email and password is saved in the database of the Facebook.
- Ex: Facebook, Twitter, Shopping sites, banking sites etc.



❒ Responsive Website:

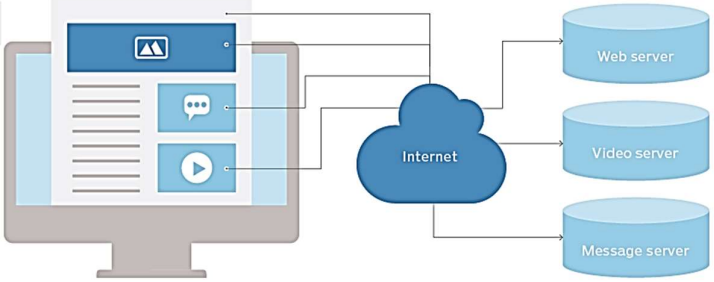
- Nowadays, each and every user and clients use mobile phones, different desktop/laptops and therefore wants a mobile version and different resolution virsions of website. But it's difficult and sometime impossible to create website versions for each new resolutions and devices. That's where responsive design comes in.
- Responsive web design is an approach that simply reflows, adjust, reposition, resize overall content and images according to width of browser or screen size.
- In simple words, responsive websites are designed to be accessed across all devices regardless of size of device screen.
- Some screen sizes that are kept in mind while creating responsive websites are desktop, laptop, mobile phones, and tablet.

⦿ HTTP (Hypertext Transfer Protocol):

- It is a protocol used to access the data on the World Wide Web (www).
- The HTTP protocol can be used to transfer the data in the form of plain text, hypertext, audio, video, and so on.
- HTTP can only works in client-server architecture.
- As soon as a user opens their web browser, they are indirectly using HTTP. HTTP is an application protocol that runs on top of the TCP/IP suite of protocols, which forms the foundation of the internet.
- This protocol is known as HyperText Transfer Protocol because of its efficiency that allows us to use in a hypertext environment where there are rapid jumps from one document to another document.
- HTTP is used to carry the data in the form of MIME-like (Multi-purpose Internet Mail Extensions) format.

- **How HTTP works:**

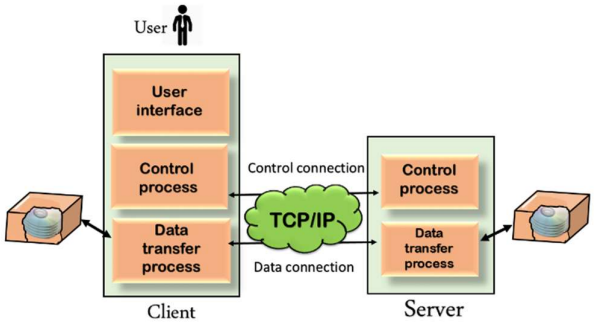
- Through the HTTP protocol, resources are exchanged between client devices and servers over the internet.
- Client devices send requests to servers for the resources needed to load a web page; the servers send responses back to the client to fulfill the requests.



- In addition to the web page files it can serve, a web server contains a program that waits for HTTP requests and handles them when they arrive.
- A web browser is an HTTP client that sends requests to servers. When user enters file requests by either "opening" a web file by typing in a URL or clicking on a hypertext link, the browser builds an HTTP request and sends it to the Internet Protocol address (IP address) indicated by the URL.
- The HTTP Server in the destination server receives the request and sends back the requested file or files associated with the request.

FTP (File Transfer Protocol):

- FTP is a standard internet protocol provided by TCP/IP used for transmitting the files from one host to another.
- It is mainly used for transferring the web page files from their creator to the computer that acts as a server for other computers on the internet.
- It is also used for downloading the files to computer from other servers, It provides the sharing of files.
- Transferring files from one system to another is very simple but sometimes it can cause problems. For example, two systems may have different file standards. Two systems may have different ways to represent text and data. Two systems may have different directory structures.
- FTP protocol overcomes these problems by establishing two connections between hosts. One connection is used for data transfer, and another connection is used for the control connection.
- There are mainly two types of FTP are as under:
- **Anonymous FTP**. This is the most basic form of FTP. It provides support for data transfers without encrypting data or using a username and password. It's most commonly used for download of material that is allowed for unrestricted distribution.
- **No-Anonymous / Password-protected FTP.**
- This is also a basic FTP service, but it requires the use of a username and password, though the service might not be encrypted or secure. It also works on port 21.



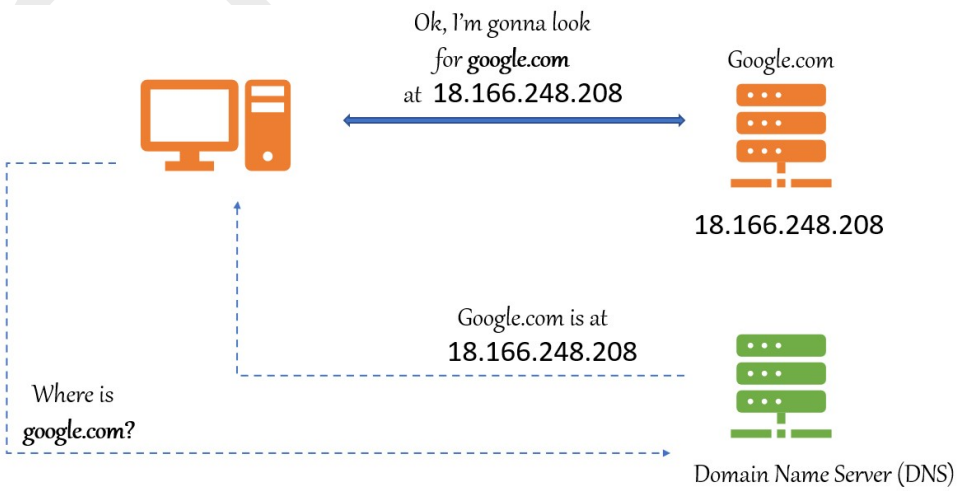
Internet Protocol Address (IP-Address):

- An IP address is a string of numbers separated by periods. IP addresses are expressed as a set of four numbers — an example address might be 192.168.1.38. Each number in the set can range from 0 to 255. So, the full IP addressing range goes from 0.0.0.0 to 255.255.255.255.
- IP addresses are not random. They are mathematically produced and allocated by the Internet Assigned Numbers Authority (IANA), a division of the Internet Corporation for Assigned Names and Numbers (ICANN).
- ICANN is a non-profit organization that was established in the US in 1998 to help maintain the security of the internet and allow it to be usable by all.
- Each time anyone registers a domain on the internet, they go through a domain name registrar, who pays a small fee to ICANN to register the domain.
- Your IP address never be consistent, it can be changed. For example, turning your router on or off can change your IP Address.

- For Ex: When you are out from your home location, your home IP address doesn't go with you. It changes as you change the network of your device. Whenever you change your location with your device every time your IP address changed.
- Whenever you change your location you will be accessing the different networks to connect your device with the internet. Therefore, your device will be allocated a different (temporary) IP address by the ISP (router) of that particular access point.
- It's a responsibility of a router to allot you an IP address to whom you are connected.
- IP address can be of:
  - Private IP
  - Public IP
  - Static IP
  - Dynamic IP

DNS (Domain Name System / Server):

- DNS is a service that translates the domain name into IP addresses. This allows the users of networks to utilize user-friendly names when looking for other hosts instead of remembering the IP addresses.
- In short DNS is a Server that converts the URL into IP address and IP address to URL vise-versa.
- For example, suppose the Creative's website has an IP address of 132.147.165.50, most people would reach this site by specifying www.cdmi.in Therefore, the domain name is more reliable than IP address.
- Whenever user writes url into browser or directly click on the hyperlink, very first the request will forwarded to the DNS. DNS identifies the axact match and converts the URL request into appropriate IP address.
- Now request would go to only the specific router whose IP address is mentioned into the request.
- The domain name space is divided into three different sections: generic domains, country domains, and inverse domain:
  - Generic Domain
  - Country Domain
  - Inverse Domain



❒ Generic Domain:

→ It uses three-character labels, and these labels describe the organization type.
→ Above all are examples of generic Domains, which are used for general purpose.

Label	Description
.com	Commercial Organizations
.org	Nonprofit Organizations
.edu	Educational institutions
.gov	Government institutions
.mil	Military groups
.net	Network Support centers

❒ Country Domain:

→ It uses two-character labels, and these labels describe the abrivation for country. For Ex: if domain is Indian than the domain is .in
→ Above all are examples of country domains,
→ which are normally a sub-domains.
→
→ The format of country domain is same as
→ a generic domain, but it uses two-character
→ country abbreviations.

Label	Description
.in	Indian domain
.uk	United Kingdom (England)
.us	United States
.au	Australia

❒ Inverse Domain:

→ The inverse domain is used for mapping an address to a name. When the server has received a request from the client, and the server contains the files of only authorized clients. To determine whether the client is on the authorized list or not, it sends a query to the DNS server and ask for mapping an address to the name.

⊙ URL (Uniform Resource Locator):

→ A URL (Uniform Resource Locator) is a unique identifier used to locate a resource on the Internet. It is also referred to as a web address.
→ URLs consist of multiple parts -- including a protocol and domain name -- that tell a web browser how and where to retrieve a resource.
→ End users use URLs by typing them directly into the address bar of a browser or by clicking a hyperlink found on a webpage, bookmark list, in an email or from another application.
→ For example, to visit the Creative's website, you will go to the URL https://www.cdmi.in, which is the URL for the Creative Design and Multimedia website.
→ The URL sends users to a specific resource online such as video, webpage, or other resources. When you search any query on Google, it will display the multiple URLs of the resource that are all related to your search query. The displayed URLs are the hyperlink to access the webpages.

Parts of a URL: https://www.techtarget.com/whatis/search/query?q=URL

- Using the URL **https://www.cdmi.in/desktop-development-course-surat** as an example, components of a URL can include:
- **The protocol or scheme.** Used to access a resource on the internet. Protocols include http, https, ftps, mailto and file. The resource is reached through the domain name system name. In this example, the protocol is https.
- **Host name or domain name.** The unique reference that represents a webpage. For this example, **cdmi.in**.
- **Path.** A path refers to a file or location on the web server. For this example, **desktop-development-course-surat**.
- Other examples of parts of a URL can include: The URL **mailto:info@cdmi.in**

initiates a new email addressed to the mailbox info in the domain cdmi.in

Portal:

→ Portal is a specialized or customized website that serves some specific kind of services.
→ Web portals are usually server side web based online platforms that enable user to manage and update information. They are mostly simple in structure and hence were used in early days now replaced by apps.
→ It is a private location on the internet, accessible with a unique URL and Username-password as well. It is 'Back-end' site for designated set of users.
→ A portal is accessed through a unique URL, unique username and password, i.e. apart from URL, personal login is required to see the content on a portal. Some of the popular portals are facebook.com, gmail.com and twitter.com.
→ At this access point, the traffic is limited, and is given to a specific set of users only. It is user-centric.
→ Portals can be categorized as: horizontal (generic) or vertical (specific)

Horizontal (Generic)Portal:

→ The portals that are serves general category of services are known as horizontal or generic portal.
→ Horizontal portal target the entire Internet community.
→ Horizontal portals have "something for everyone" and appeal to a wide range of interests. Google, Yahoo, Bing are prime examples, giving the user a convenient gateway to access popular email, news, weather and other information that might be valuable.
→ These sites often referred to as "mega portals" usually contains search engine and provide the ability for user to personalize the page by offering various channels.

Vertical (Specific)Portal:

→ The portals that are serves special kind of services are known as vertical or specific portal.
→ Vertical portal target the limited services and focus on a particular industry, such as payment portal, news and media, corporate portal, web portal.
→ Vertical portals have "something for some-one" and appeal to a limited services.

- **Types of portals:**

Web-Portal	Education Portal
Private Portal	Stock Portal
Corporate Portal	Tender Portal
Government Portal	Search Portal
Payment Portal	News/Media Portal

⊙ Web-Browser:

→ A software application used to access information from the WWW is called a Web Browser.
→ When a user requests some information, the web browser fetches the data from a web server and then displays the webpage on the user's screen.
→ "World Wide Web" was the first web browser created by Tim Berners Lee in 1990. This is completely different from the World Wide Web we use today.
→ In 1993, the "Mosaic" web browser was released. It had the feature of adding images and an innovative graphical interface. It was the "the world's first popular browser"
→ After this, in 1994, Marc Andreessen (leader of Mosaic Team) started working on a new web browser, which was released and was named "Netscape Navigator"
→ In 1995, "Internet Explorer" was launched by Microsoft. It soon overtook as the most popular web browser.

- **How Web-Browser works?**

→ A web browser helps us find information anywhere on the internet. It is installed on the client computer and requests information from the webserver such a type of working model is called a client-server model.
→ It works as a compiler to render HTML which is used to design a webpage.
→ Whenever we search anything on the internet, the browser loads a web page written in HTML, including text, links, images, and other items such as style sheets and JavaScript functions.

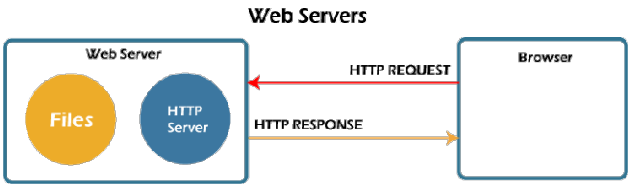
- **Some popular Web Browsers**

→ Google Chrome
→ Microsoft Edge
→ Mozilla Firefox
→ Safari

CREATIVE

Web-Server:

- A web server is dedicated software that runs on the server-side.
- In other words a Web-server is a specialized computer with dedicated server program intalled on it. It is mainly design and develop to accept and process on the user request and send back response to the user.
- When any user requests their web browser to run any web page, the webserver places all the data materials together into an organized web page and forwards them back to the web browser with the help of the Internet.
- This intercommunication of a web server with a web browser is done with the help of a protocol named HTTP (Hypertext Transfer Protocol).



- **How do web servers work?**

- The term web server can denote server hardware or server software, or in most cases, both hardware and software might be working together.
- Whenever any web browser, such as Google Chrome, Microsoft Edge or Firefox, requests for a web page hosted on a web server, the browser will process the request forward with the help of HTTP. At the server end, when it receives the request, the HTTP server will accept the request and immediately start looking for the requested data and forwards it back to the web browser via HTTP.
- The software side is having several components, with at least an HTTP server. The HTTP server is able to understand HTTP and URLs request.
- As hardware, a web server is a computer that stores web server software and other files related to a website, such as HTML documents, images and JavaScript files and website related other contents.

Web-Hosting:

- Web hosting is an online service that enables you to publish your website or web application on the internet.
- Once you created your website, you want that your site would be available in internet to every people who wants to visit it. For that purpose you need to have a some space on the internet where your website can reside.
- This space on internet is a known as domain, where you deploy / host / publish your website.
- When you sign up for a web hosting service, you basically rent some space on a physical server where you can store all the files and data necessary for your website to work properly.
- Web hosts provide the hosting technology and resources required for the effective and secure operation of your website. They are responsible for keeping the server up and

CREATIVE

running, implementing security measures, and ensuring that data such as texts, photos, and other files are transferred successfully to the visitors' browsers.

- **How Does Web Hosting Work?**

→ The server that hosts your website is a physical computer that runs continuously to make the site available for visitors all the time. Buying servers for web hosting will alow you to store all the data of your website in the those servers of your provider.
→ Once a user enters your domain name into their browser's address bar, the web host's server will transfer all the files necessary to load your website.
→ You can host a website yourself, but it requires extensive technical skills as well as including the equipment, infrastructure, hardware, and software.
→ Furthermore, you will also have to handle all the ongoing maintenance.
→ A web hosting service provider ensures that your website performs optimally and with better security protocols. In addition, it simplifies the many complex aspects of hosting a website – from software installation to technical support.

- **Types of Web-Hosting:**

❒ **Shared Hosting:**

→ With shared hosting, multiple users share the same server resources, including memory, processing power, and storage space.
→ Because of its simplicity and affordability, shared web hosting is an excellent solution for small businesses and personal websites that do not require advanced configuration or higher bandwidth.

❒ **Dedicated Hosting:**

→ Dedicated hosting provides a physical server for each website. You can configure the server, choose your desired operating system and software, and customize the entire hosting environment to your specifications.
→ Renting a dedicated server is just as powerful as having your own on-site server, but with the added benefit of getting professional support from your web host. Thus, dedicated hosting is ideal for large online businesses that deal with heavy traffic.

❒ **Cloud Hosting:**

→ This web hosting solution uses several virtual servers to host sites. Thus, if one server experiences high traffic or a problem, the remaining ones will take over and maintain the website operating.

❒ **Virtual Private Server (VPS) Hosting:**

→ With this web hosting type, your website also shares a physical server with other users, but the web host creates a virtual partition for each user. Thus, a site hosted on a virtual private server get an allocated amount of resources.
→ VPS web hosting is a great option for medium-sized sites, eCommerce shops, and large blogs with a rapidly growing number of visitors.

CREATIVE

Chapter 2

Basic of HTML & Advance HTML5

What is HTML?

- HTML stands for **Hyper Text Markup Language.**
- HTML was invented by **Berners-Lee.**
- HTML is the standard markup language for creating Web pages.
- HTML describes the structure of **Web pages** using markup.
- HTML elements tell the browser how to display the content
- HTML elements are represented by **Tags.**
- HTML tags label pieces of content such as "heading", "paragraph", "table", and so on…

HTML Structure

```
<!DOCTYPE html>
<html>
<head>
        <title>Page Title</title>
</head>
<body>
</body>
</html>
```

- The <!DOCTYPE html> declaration defines this document to be HTML5.
- The <html> element is the root element of an HTML page.
- The <head> element contains meta information about the document.
- The <title> element specifies a title for the document.
- The <body> element contains the visible page content.

HTML Versions

Version	Year
HTML	**1991**
HTML 2.0	**1995**
HTML 3.2	**1997**
HTML 4.01	**1999**
XHTML	**2000**
HTML5	**2014**
HTML5.1	**2016**

CREATIVE

HTML5.2	**2017**

HTML Editors

- ✓ Notepad
- ✓ Notepad++
- ✓ SubLime Text
- ✓ Dream Viewer
- ✓ NetBeans etc..

Browsers

- ✓ Google Chrome
- ✓ Mozilla Firefox
- ✓ Opera
- ✓ Internet Explorer
- ✓ Safari etc.

HTML Tags

- There are two types of tags..
  - **Container Tag / Pair Tag**
  - **Singular Tag / Unpair Tag**

Syntax :

<tagname>content goes here...**</tagname>**

- HTML tags normally come **in pairs** like and
- The first tag in a pair is the **start tag,** the second tag is the **end tag**
- The end tag is written like the start tag, but with a **forward slash** inserted before the tag name

HTML Attributes

- All HTML elements can have **attributes**
- Attributes provide **additional information** about an element
- Attributes are always specified in **the start tag**
- Attributes usually come in name/value pairs like: **name="value"**

```
<!DOCTYPE html>
<html>
<head>
        <title>Page Title</title>
</head>
<body [bgcolor="color-name"] => Set Background color for body
        [text="color-name"] => Set Text / Font Color in body
        [link="color-name"] => Set Link color in body
```

CREATIVE

[alink="color-name"] => Set **Active Link** color in **body**

[vlink="color-name"] => Set **Visited Link** color in **body**

[background="url"] > => Set a **Background Image** in **body**

</body>

</html>

Heading Tag

- The <h1> element defines a large heading.

Syntax:

<**h**x [title="text"] => add tips for any Heading tags

[align="center|left|right"] => align text left, right or center in a page >

</**h**x>

X = 1, 2, 3, 4, 5, 6

Example:

```
<!DOCTYPE html>
<html>
<head>
        <title>Page Title</title>
</head>
<body>

        <h1>Creative Design & Multimedia Institute</h1>
        <h2>Creative Design & Multimedia Institute</h2>
        <h3>Creative Design & Multimedia Institute</h3>
        <h4>Creative Design & Multimedia Institute</h4>
        <h5>Creative Design & Multimedia Institute</h5>
        <h6>Creative Design & Multimedia Institute</h6>


</body>
</html>
```

CREATIVE

Font Size :

Tag	Font Size
Normal Text	**16px (Times New Roman)**
<h1>	**32px**
<h2>	**24px**
<h3>	**18px**
<h4>	**16px**
<h5>	**13px**
<h6>	**10px**

HTML Comment Text

- ✓ With comments you can place notifications and reminders in your HTML code.

Syntax :

```
<!-- Write your comments here -->
```

BR Tag :

**
 => Break Row**

- ✓ The
 tag inserts a single line break.
- ✓ The
 tag is useful for writing addresses or poems.
- ✓ The
 tag is an empty tag which means that it has no end tag.

Example:

```
<!DOCTYPE html>
<html>
<head>
        <title>Page Title</title>
</head>
<body>
                Be not afraid of greatness.<br>
                Some are born great,<br>
                some achieve greatness,<br>
                and others have greatness thrust upon them.
 </body>
 </html>
```

CREATIVE

HTML Formatting Elements

- In the previous chapter, you learned about the HTML **style attribute**.
- HTML also defines special **elements** for defining text with a special **meaning**.
- HTML uses elements like and for formatting output, like **bold** or *italic* text.
- Formatting elements were designed to display special types of text:
  - - **Paragraph** text
    - The tag defines a paragraph.
    - Example:
      - This is some text in a paragraph. .
    - Output:
      - This is some text in a paragraph.
  - - **Bold** text
    - To display text with BOLD effect we can use <B> tag.
    - Example:
      - This is normal text - and this is bold text.
    - Output:
      - This is normal text - **and this is bold text**.
  - - **Important** text
    - The tag is used to define text with strong importance.
    - Example:
      - This is normal text - and this is Important text .
    - Output:
      - This is normal text - **and this is Important text**.
  - - **Italic** text
    - The content inside is typically displayed in *italic*.
    - Example:
      - This is normal text - and this is Italic text .
    - Output:
      - This is normal text - *and this is Italic text*.

CREATIVE

- - **Emphasized** text
  - The tag is used to define emphasized text.
  - Example:
    - This is normal text - and this is Emphasized text .
  - Output:
    - This is normal text - *and this is Emphasized text*.
- <address>
  - The <address> tag defines the contact information for the author/owner of a document or an article.
  - The contact information can be an email address, URL, physical address, phone number, social media handle, etc.
  - Example:
    - <address>

      Written by Creative.

      Visit us at:

      Example.com

      Box 564, Disneyland

      USA

      </address>
  - Output:
    - *Written by Creative .*
      Visit us at:
      Example.com
      Box 564, Disneyland
      USA
- <cite>
  - The <cite> tag defines the title of a creative work (e.g. a book, a poem, a song, a movie, a painting etc.).
  - The text in the <cite> element usually renders in italic.
  - Example:
    - <cite>The Scream</cite> by Edward Munch. Painted in 1893.
  - Output:

CREATIVE

- *The Scream* by Edward Munch. Painted in 1893.

❖ <dfn>

- The <dfn> tag stands for the "definition element", and it specifies a term that is going to be defined within the content.
- Example:
  - <dfn>HTML</dfn> is the standard markup language for creating web pages.
- Output:
  - *HTML* is the standard markup language for creating web pages.

❖ <code>

- The <code> tag is used to define a piece of computer code.
- The content inside is displayed in the browser's default monospace font.
- Example:
  - The HTML <code>button</code> tag defines a clickable button.
- Output:
  - The HTML `button` tag defines a clickable button.

❖ <samp>

- The <samp> tag is used to define sample output from a computer program.
- The content inside is displayed in the browser's default monospace font.
- Example:
  - <samp>File not found.
Press F1 to continue</samp>
- Output:

```
File not found.
Press F1 to continue
```

❖ <kbd>

- The <kbd> tag is used to define keyboard input.
- The content inside is displayed in the browser's default monospace font.
- Example:
  - Press <kbd>Ctrl</kbd> + <kbd>C</kbd> to copy text (Windows).
  - Press <kbd>Cmd</kbd> + <kbd>C</kbd> to copy text (Mac OS).

CREATIVE

- Output:
  - Press **Ctrl + C** to copy text (Windows).
  - Press **Cmd + C** to copy text (Mac OS).

❖ <strike> / <del> / <s>

- The <strike> tag was used in HTML 4 to define strikethrough text.
- The <del> tag defines text that has been deleted from a document.
- The <s> tag specifies text that is no longer correct, accurate or relevant.
- Example:
  - Price <strike>350$</strike> 300$
  - Price ~~350$~~ 300$
  - Price <s>350$</s> 300$
- Output:
  - Price ~~350$~~ 300$

❖ <sup>

- The <sup> tag defines superscript text.
- Superscript text appears half a character above the normal line, and is sometimes rendered in a smaller font.
- Example:
  - X ²
- Output:
  - X^2

❖ <sub>

- The <sub> tag defines subscript text.
- Subscript text appears half a character below the normal line, and is sometimes rendered in a smaller font.
- Example:
  - H ₂O
- Output:
  - H_2O

CREATIVE

- <big>
  - The <big> tag was used in HTML 4 to define bigger text.
  - Example:
    - This is a normal paragraph.
    - <big> This is a bigger paragraph. </big>
  - Output:
    - This is a normal paragraph.
    - This is a bigger paragraph.
- <small>
  - The <small> tag was used in HTML 4 to define smaller text.
  - Example:
    - This is a normal paragraph.
    - <big> This is a smaller paragraph. </big>
  - Output:
    - This is a normal paragraph.
    - This is a smaller paragraph.
- <pre>
  - The <pre> tag defines preformatted text.
  - The text will be displayed exactly as written in the HTML source code.
  - Example:

```
<pre> Text in a pre element
       is displayed in a fixed-width
       font, and it preserves
       both     spaces and
       line breaks</pre>
```

  - Output:

```
Text in a pre element
is displayed in a fixed-width
font, and it preserves
both      spaces and
line breaks
```

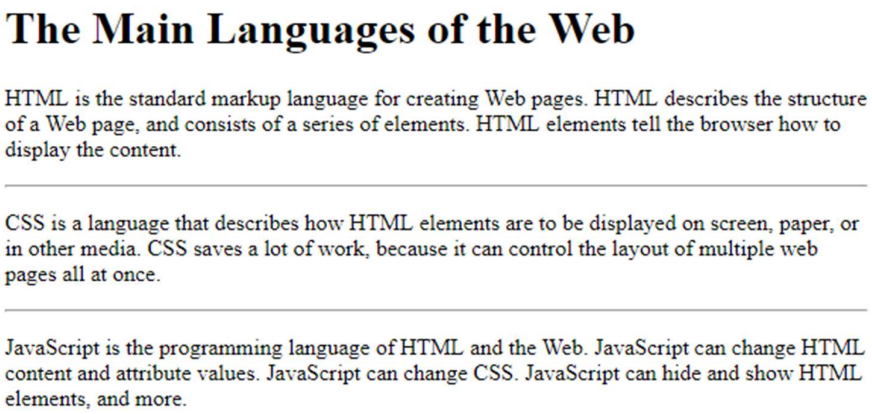
CREATIVE

-
  - The tag was used in HTML 4 to specify the font face, font size, and color of text.
  - Syntax:
    - …Content Here…
  - Example:
    - Hellooo…! Good Afternoon
  - Output:
    - Hellooo…! Good Afternoon
- <hr>
  - The <hr> element is most often displayed as a horizontal rule that is used to separate content (or define a change) in an HTML page.
  - Syntax:
    - <hr [align="left | center | right"]
      [size="Pixel"] => Height
      [width="Pixel or Percent"]
      [color="color-name"]>
  - Example:

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

<h1>The Main Languages of the Web</h1>

<p>HTML is the standard markup language for creating Web pages. HTML
describes the structure of a Web page, and consists of a series of
elements. HTML elements tell the browser how to display the content.</p>

<hr>

<p>CSS is a language that describes how HTML elements are to be displayed
on screen, paper, or in other media. CSS saves a lot of work, because it
can control the layout of multiple web pages all at once.</p>

<hr>

<p>JavaScript is the programming language of HTML and the Web. JavaScript
can change HTML content and attribute values. JavaScript can change CSS.
JavaScript can hide and show HTML elements, and more.</p>

</body>
</html>
```

CREATIVE

- Output:



The Main Languages of the Web

HTML is the standard markup language for creating Web pages. HTML describes the structure of a Web page, and consists of a series of elements. HTML elements tell the browser how to display the content.

CSS is a language that describes how HTML elements are to be displayed on screen, paper, or in other media. CSS saves a lot of work, because it can control the layout of multiple web pages all at once.

JavaScript is the programming language of HTML and the Web. JavaScript can change HTML content and attribute values. JavaScript can change CSS. JavaScript can hide and show HTML elements, and more.

❖ <marquee>

- The <marquee> tag is a container tag of HTML is implemented for creating scrollable text or images within a web page from either left to right or top to bottom.
- Syntax:

```
<marquee [behavior="Scroll | Slide | Alternate"]
         [direction="Left | Right | Up | Down"]
         [loop="1 | 2 | ...N"]
         [scrollamount="number"]
         [bgcolor="Color Name"]
         [height="pixels"]
         [width="pixels"] >.....</marquee>
```

- Example:
  - <marquee behavior="scroll" direction="up" scrollamount="1">Slow Scrolling</marquee>
  - <marquee behavior="scroll" direction="right" scrollamount="12">Little Fast Scrolling</marquee>
  - <marquee behavior="scroll" direction="left" scrollamount="20">Fast Scrolling</marquee>
  - <marquee behavior="scroll" direction="right" scrollamount="50">Very Fast Scrolling</marquee>

CREATIVE

❖ <a>

- 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.
- Syntax:

```
<a [href="URL"]
    [target="_blank"] >...</a>
```

- Example:
  - Visit cdmi.in!
- Output:
  - Visit cdmi.in

❖ <img>

- The <img> tag is used to embed an image in an HTML page.
- The <img> tag has two required attributes:
  - src - Specifies the path to the image
  - alt - Specifies an alternate text for the image, if the image for some reason cannot be displayed
- Syntax:

```
<img [alt="AlternativeText]
    [border="border-width"]
    [height="pixels"]
    [width="pixels"]
    [hspace="pixels"] => Horizontal Margin
    [vspace="pixels"] > => Vertical Margin
```

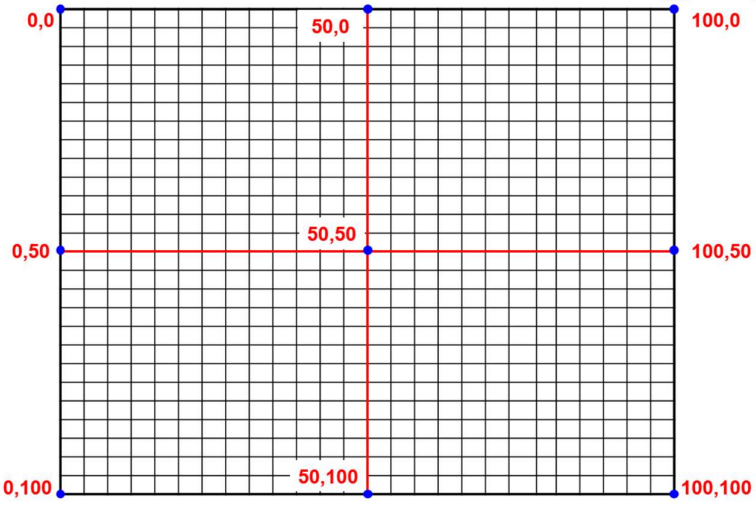
- Example:

```
<img src="your image path here" width="500" border="5" alt="Creative College" >
```

CREATIVE

Image Map:

- The HTML <map> tag defines an image map.
- An image map is an image with clickable areas.
- The areas are defined with one or more <area> tags.
- To map an image we have to follow some special steps.
  1. Set #mapname to image by <IMG>
  2. Use MAP tag <MAP>
  3. Specify AREA with LINK using <AREA>



- Syntax :

```
<map name="[mapname]">
    <area shape="[rect, circle,poly]" coords="x1,y1,x2,y2"  href="URL
">
</map>
```

- Example :

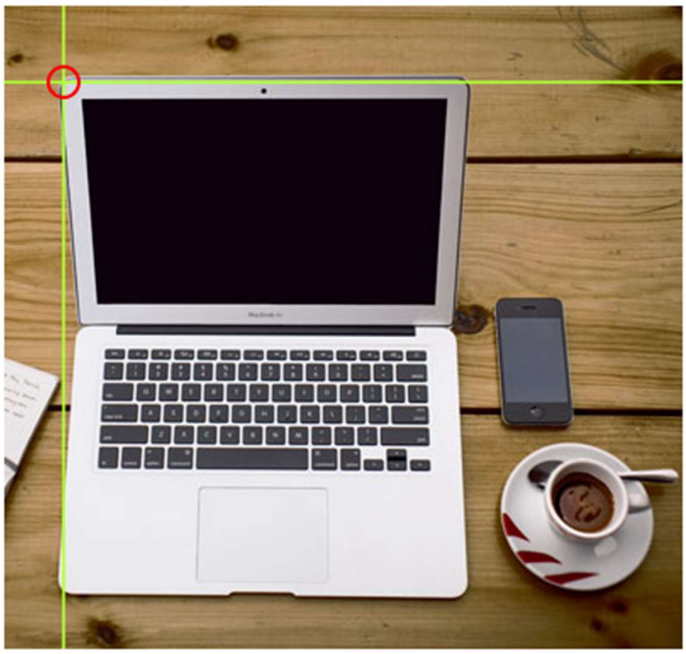
```
<img src="workplace.jpg" alt="Workplace" usemap="#workmap">
<map name="workmap">
    <area shape="rect" coords="34,44,270,350"  href="computer.html">
    <area shape="rect" coords="290,172,333,250"  href="phone.html">
     <area shape="circle" coords="337,300,44" href="coffee.html">
</map>
```

How Does it work ?

✓ Shape=”rect”

- The coordinates for shape="rect" come in pairs, one for the x-axis and one for the y-axis.

- So, the coordinates 34,44 is located 34 pixels from the left margin and 44 pixels from the top:



- The coordinates 270,350 is located 270 pixels from the left margin and 350 pixels from the top:

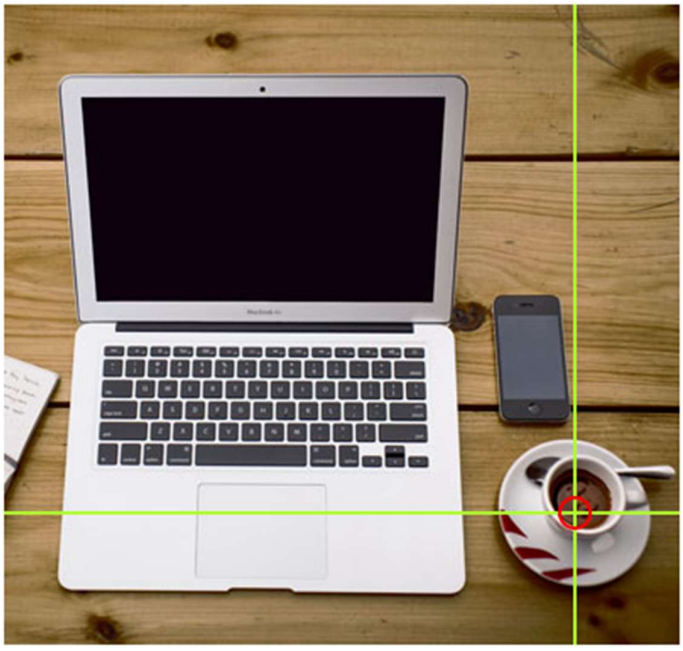
- Now we have enough data to create a clickable rectangular area:

```
<area shape="rect" coords="34, 44, 270, 50" href="computer.html">
```

CREATIVE

✓ Shape="circle" :

- To add a circle area, first locate the coordinates of the center of the circle:
  `337,300`



- Then specify the radius of the circle:
  `44` pixels

- Now you have enough data to create a clickable circular area:

```
<area shape="circle" coords="337, 300, 44" href="coffee.html">
```

CREA

List Creation Tags

- - Ordered List
  - - List Item
- - Unrdered List
  - - List Item
- <dl> - Description List
  - <dt> - Defination Term
  - <dd> - Defination Desription

 - Ordered List :

- The <ol> tag defines an ordered list. An ordered list can be numerical or alphabetical.
- The <li> tag is used to define each list item.
- Syntax :

```
<ol [start="value"] [type="A | a | I | I | 1"] [reversed]>
    <li>.....</li>
</ol>
```

- Example :

```
<ol start="5" type="A">
    <li>Coffee</li>
    <li>Tea</li>
    <li>Milk</li>
</ol>
```

- Output :

E. Coffee

F. Tea

G. Milk

CREATIVE

❖ - Unordered List :

- The <ul> tag defines an unordered (bulleted) list.
- Use the <ul> tag together with the <li> tag to create unordered lists.
- Syntax :

```
<ul [type="disc | circle | square"] >
    <li>……..</li>
</ul>
```

- Example :

```
<ul type="square">
    <li>Coffee</li>
    <li>Tea</li>
    <li>Milk</li>
</ul>
```

- Output :
  - Coffee
  - Tea
  - Milk

❖ <dl> - Description List :

- The <dl> tag defines a description list.
- The <dl> tag is used in conjunction with <dt> (defines terms/names) and <dd> (describes each term/name)..
- Syntax :

```
<dl >
    <dt>……..</dt>
    <dd>……..</dd>
</dl>
```

- Example :

```
<dl>
    <dt>Coffee</dt>
    <dd>Black hot drink</dd>
    <dt>Milk</dt>
```

CREATIVE

```
        <dd>White cold drink</dd>
    </dl>
```

- Output :

```
Coffee
        Black hot drink
Milk
        White cold drink
```

Embed:

- The <embed> tag defines a container for an external resource, such as a web page, a picture, a media player, or a plug-in application.
- Syntax :

```
<embed [type="image/jpg | text/html | video/webm"]
        src="URL" width="px" height="px">
```

- Example :

```
<embed type="image/jpg" src="pic_trulli.jpg" width="300" height="200">
```

- Output :

- Example :

```
<embed type="text/html" src="snippet.html" width="500" height="200">
<embed type="video/webm" src="video.mp4" width="400" height="300">
```

Inserting Special Characters

- Many mathematical, technical, and currency symbols, are not present on a normal keyboard.
- To add such symbols to an HTML page, you can use the entity name or the entity number (a decimal or a hexadecimal reference) for the symbol.
- **Ex:**

```
<p>I will display &euro;</p>
<p>I will display &#8364;</p>
<p>I will display &#x20AC;</p>
```

Output:

I will display €
I will display €
I will display €

Description	Special Character	Code
Acute Accent	´	´
Ampersand	&	&
Cent Sign	¢	¢
Copyright	©	©
Dagger	†	†
Degree Sign	°	º
Division Sign	÷	÷
Euro	€	₫
Fraction (one-half)	½	½
Fraction (one-fourth)	¼	¼
Fraction (three-fourths)	¾	¾
Greater-than sign	>	>
Left-angle quote	«	«
Less-than sign	<	<
Multiply sign	x	×
Plus or minus sign	±	±
Quotation Mark (left)	“	“
Quotation Mark (right)	”	”
Registered Trademark	®	®
Right-angel quote	»	»
Superscript one	¹	¹
Trademark	™	™

Tables in HTML

- The <table> tag defines an HTML table.
- An HTML table consists of one <table> element and one or more <tr>, <th>, and <td> elements.
- The **<caption>** tag defines a **Table Caption.**
- The **<tr>** element defines a **Table Row**.
- The **<th>** element defines a **Table Header**.
- The **<td>** element defines a **Table Cell**.

❖ <table> :

- Syntax:

```
<table align="[left | center | right]"
        border="value"
        width="value/percent"
        height=" value/percent"
        bordercolor="color"
        bgcolor="color"
        cellpadding="value"
        cellspacing="value">.....</table>
```

❖ <caption> :

- Syntax:

```
< caption>.....</caption>
```

❖ <tr> :

- To add a new row in a table we must use TR tag
- Syntax:

```
<tr bgcolor="color">.....</tr>
```

CREATIVE

❖ <th> :

- The text in <th> elements are bold and centered.
- Each table header is defined with a <th> tag.
- Syntax:

```
<th align="[left | center | right]"
    valign="[bottom | middle | top]"
    width="pixel"
    height=" pixel"
    bgcolor="color"
    colspan="value"
    rowsapn="value">.....</th>
```

❖ <td> :

- Each table data/cell is defined with a <td> tag.
- Syntax:

```
<td align="[left | center | right]"
    valign="[bottom | middle | top]"
    width="pixel"
    height=" pixel"
    bgcolor="color"
    colspan="value"
    rowsapn="value">.....</td>
```

❖ **Example :** Design a web page to display TABLE as given…

No.	Full Name	Position	Salary
1	Bill Gates	Founder of Microsoft	$1000
2	Steve Jobs	Founder of Apple	$1200
3	Larry Page	Founder Google	$1100
4	Mark Zuckerberg	Founder of Facebook	$1300

CREATIVE

Time Table					
Hours	Mon	Tue	Wed	Thu	Fri
	Science	Maths	Science	Maths	Arts
	Social	History	English	Social	Sports
	Lunch				
	Science	Maths	Science	Maths	Project
	Social	History	English	Social	

A test table with merged cells

	Average		Red eyes
	height	weight	
Males	1.9	0.003	40%
Females	1.7	0.002	43%

HTML Table Example

Student Name		Labs			Avg
First	Last	Lab1	Lab2	Lab3	
Sol	Rosenberg	100	100	100	100.0
Frank	Rizzo	50	60	90	66.6
Ali	Kam	90	80	60	76.6
Summary	Avg	80.0	80.0	83.3	81.1
	Min	50	60	60	66.6

Country	State	City	Street	Male	Female	Others
1	Kerala	Cochin	New Street	500	600	6
			Main Street	300	288	2
		Trivandrum	Guru Street	500	600	10
			TVK Street	500	600	2
	Maharastra	Mumbai	Krisha Street	700	850	1
			Main Street	500	600	2
		Surath	New Street	500	600	4
			Billa Street	500	600	2
2	Alaska	AKA Central	New Street	200	210	2
			Cross Street	1000	1050	10

RESULT : BSC SEM5 [CBCS]

Certificate showing the number of marks obtained by ID NO.
Shri/Smt./Kumari
of **SIR P. T. SARVAJANIK COLLEGE OF SCIENCE, SURAT**
in each header of passing at the **BACHELOR OF SCIENCE (FIFTH SEMESTER) held in APRIL-2016**

Subject Name	Passing Mark			Mark Obtained					
	External	Internal	Total	External	Internal	Total	GR.	GP.	Credit
LANGUAGE THROUGH LITERATURE	50/18	20	70/25	19	(09)	28	E	5	2
PAPER-VI									
PAPER-VII									
PAPER-VIII									
PAPER-IX									
PAPER-X									
PAPER-XI									
PHYSICS	300/108	120	420/151	(143)	(60)	(203)	E	5	12
PHYSICS PRACTICAL	120/43	60	180/65	(91)	(44)	(135)	B	8	6
ELECTRONICS	50/18	20	70/25	(26)	(10)	(36)	D	6	2
NATIONAL SERVICE SCHEME									2
Aggregate Total	**520**		**740**	**279**		**402**			
Result	**PASS**								

Frames in HTML

- HTML frames are used to divide your browser window into multiple sections where each section can load a separate HTML document.
- When we want to use more then one HTML files in a web page, at that time we have to use FRAMES.
- A collection of frames in the browser window is known as a frameset.
- The window is divided into frames in a similar way the tables are organized: into rows and columns.
- **Disadvantages of Frames**
  - ✓ There are few drawbacks with using frames, so it's never recommended to use frames in your webpages –
    - Some smaller devices cannot cope with frames often because their screen is not big enough to be divided up.
    - Sometimes your page will be displayed differently on different computers due to different screen resolution.
    - The browser's back button might not work as the user hopes.
    - There are still few browsers that do not support frame technology.

Creating Frames

- ✓ To use frames on a page we use <frameset> tag instead of <body> tag.
- ✓ The <frameset> tag defines, how to divide the window into frames.
- ✓ The rows attribute of <frameset> tag defines horizontal frames and cols attribute defines vertical frames.
- ✓ Each frame is indicated by <frame> tag and it defines which HTML document shall open into the frame.
- ✓ Note – The <frame> tag deprecated in HTML5. Do not use this element.

Syntax :

```
<frameset rows="height in % or px" cols="width in % or px" frameborder="0 or 1"
border="0 to n">
        <frame name = "frame name" src = "File Path" noresize="noresize"
scrolling="yes or no" frameborder="0 or 1" />
        <noframes>
            <body>Your browser does not support frames.</body>
        </noframes>
</frameset>
```

- ✓ **Example 1**
  - Following is the example to create three horizontal frames –

```
<!DOCTYPE html>
<html>
    <head>
      <title>HTML Frames</title>
    </head>
    <frameset rows = "10%,80%,10%">
      <frame name = "top" src = "https://cdmi.in/" />
      <frame name = "main" src = "https://cdmi.in/" />
      <frame name = "bottom" src = "https://cdmi.in/" />
      <noframes>
          <body>Your browser does not support frames.</body>
      </noframes>
    </frameset>
</html>
```

CREATIVE

✓ **Example 2**

- Let's put the above example as follows, here we replaced rows attribute by cols and changed their width. This will create all the three frames vertically –

```
<!DOCTYPE html>
<html>
    <head>
      <title>HTML Frames</title>
    </head>
    <frameset cols = "25%,50%,25%">
        <frame name = "left" src = "https://cdmi.in/" />
        <frame name = "center" src = "https://cdmi.in/" />
        <frame name = "right" src = " https://cdmi.in/" />
        <noframes>
                <body>Your browser does not support frames.</body>
        </noframes>
    </frameset>
</html>
```

- <iframe>:

✓ The <iframe> tag specifies an inline frame.

✓ An inline frame is used to embed another document within the current HTML document.

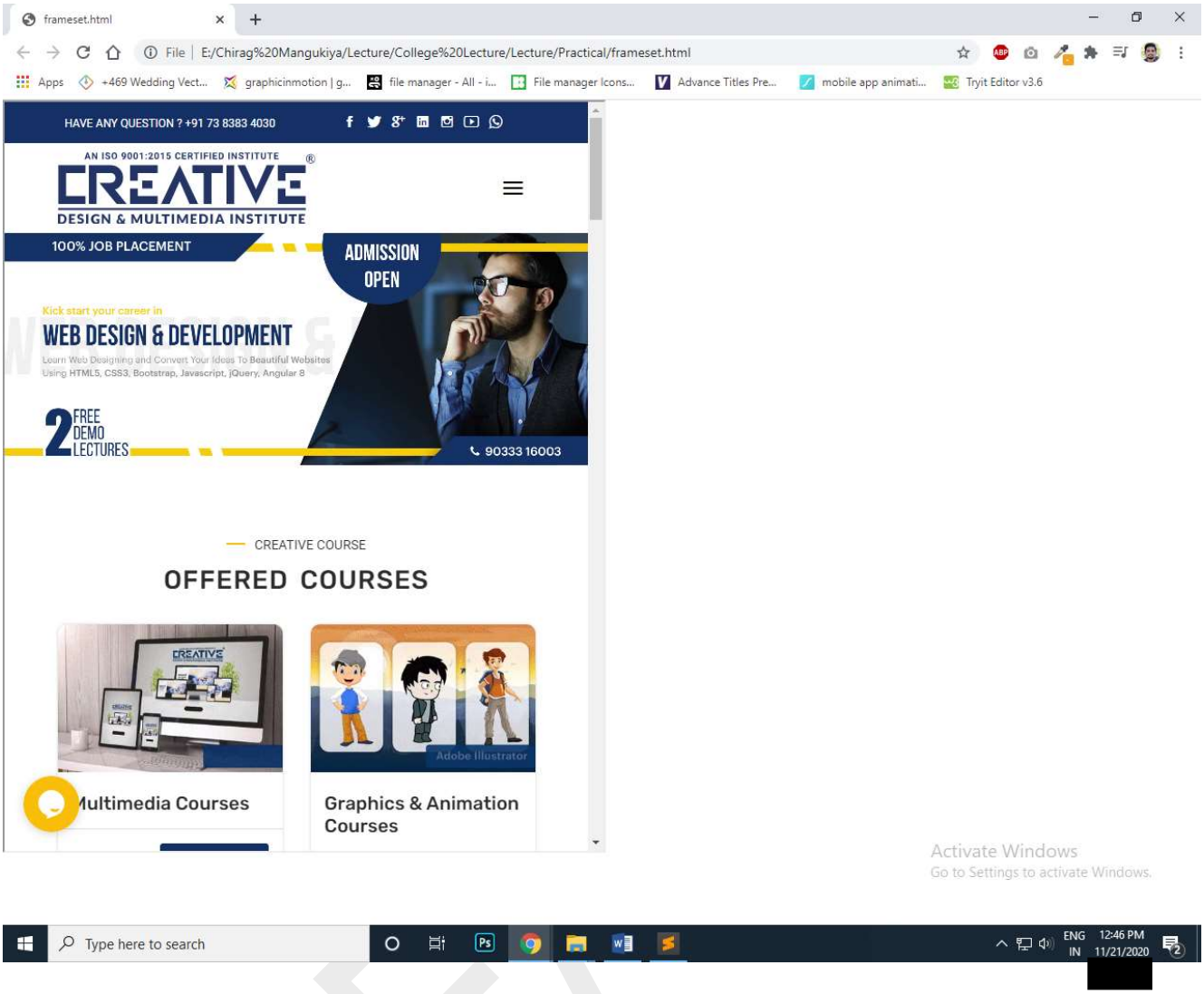
✓ Syntax:

```
<iframe src="URL" width="pixel/percent" height="pixel">
</iframe>
```

✓ Example:

```
<iframe src="https://www.cdmi.in/" width="50%" height="700">
</iframe>
```

CREATIVE



HTML Form Elements

- An HTML form is used to collect user input.
- The user input is most often sent to a server for processing.
- Syntax:

```
<form method="[ get | post ] "  action="URL" name="form name">
        form elements
</form>
```

❖ The <form> element is a container for different types of input elements, such as: **text fields, checkboxes, radio buttons, submit buttons**, etc.

❖ **Form Methods :**

- **GET :**
  - Get method will DISPLAY all the information of page in to URL.
  - Note : Some of Browser will note display information on URL by GET...
- **POST :**
  - POST method will HIDE all the information of page in to URL.

<input> Elements

❖ The HTML <input> element is the most used form element.

❖ An <input> element can be displayed in many ways, depending on the type attribute.

Type	Description
<input type="text">	Displays a single-line text input field
<input type="password">	Displays text as a password
<input type="radio">	Displays a radio button (for selecting one of many choices)
<input type="checkbox">	Displays a checkbox (for selecting zero or more of many choices)
<input type="submit">	Displays a submit button (for submitting the form)
<input type="reset">	Reset form controls
<input type="button">	Displays a clickable button

- Syntax:

<input type="***Control Type***"
[Name="Name of Control"]
[Value="DefaultValue"]
[MaxLength="IntLen"]
[Size="InputBoxLen"]
[ReadOnly][Disabled]
[Align="Left|Right"]
[Checked="Checked"] >

CREATIVE

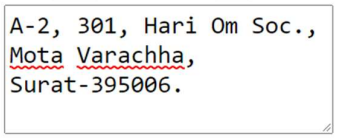
❖ <textarea> :

- The <textarea> tag defines a multi-line text input control.
- The <textarea> element is often used in a form, to collect user inputs like comments or reviews.
- The size of a text area is specified by the <cols> and <rows> attributes (or with CSS).
- Syntax :

```
< textarea [Name="Control Name"]
            [Cols="IntColumns"]
            [Rows="IntRows"]
            [Disabled] [ReadOnly]
            [Wrap="Off | Physical"] >
                Default Text...
</ textarea >
```

Example :

```
< textarea cols="22" rows="4" placeholder="Address"></ textarea >
```



❖ <select> :

- The <select> element is used to create a drop-down list.
- The <select> element is most often used in a form, to collect user input.
- Syntax:

```
< select [Name="Control Name"]
          [Disabled] [Multiple]
          [Size="Input Num of Character"] >
              <option>....</option>
</select>
```

❖ <option> :

- The <select> element is used to create a drop-down list.
- The <select> element is most often used in a form, to collect user input.
- Syntax:

```
<option [Name="Control Name"]
        [Value="Initial Value"]
        [Selected]>.....</option>
```

Example :

```
< select >
    <option>--Select City--</option>
    <option>Surat</option>
    <option>Vapi</option>
    <option>Baroda</option>
    <option>Valsad</option>
    <option>Navsari</option>
</ select >
```

Student Detail Form

Roll No. :	Roll no.
Name :	Enter Name
Address :	Address
Date of Birth :	Day Month Year
Gender :	Male Female
Hobby :	Cricket Reading Travelling Dancing Singing
Username :	Enter Username
Password :	Enter Password
Button Submit Reset	

Design a Student Data Entry Form using form and input elements.

HTML5 – Overview

- HTML5 is not only a new version of HTML language enriched with new elements and attributes, but a set of technologies for building more powerful and diverse web sites and applications, that support multimedia, interact with software interfaces, etc.

HTML5 Benefits

The main benefits of HTML5 are listed below:

- HTML5 is supported by all modern browsers.
- HTML5 is more device friendly. It is easy for use.
- HTML5 can help creating attractive websites with CSS, JavaScript, etc.
- HTML5 supports SVG (Scalable Vector Graphics - Wikipedia) and other virtual graphics. In earlier versions, the use of vector graphics was possible only in combination with such technologies as Flash, VML(Vector Markup Language), etc.
- The code becomes cleaner mainly due to replacing div tags with Semantic elements, which help better structure content of the web page and improve readability.
- HTML5 supports geolocation, which makes it possible to determine the users' location.
- The new standards were specified for playing multimedia (animation, audio, video) directly in the browser without having to install additional plug-ins.

Disadvantages of HTML5:

- It is supported only by modern browsers.
- You must write long codes which is time-consuming.

CREATIVE

- Syntax:

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
</head>
<body>
    <header>
        Your Content Here...
    </header>
    <nav>
        Your Content Here...
    </nav>
    <section>
        <article>
            Your Content Here...
        </article>
        <article>
            Your Content Here...
        </article>

        <aside>
            Your Content Here...
        </aside>
    </section>
    <footer>
        Your Content Here...
    </footer>
</body>
</html>
```

- **HTML5 comes with a lot of flexibility and it supports the following features** :
  - Uppercase tag names.
  - Quotes are optional for attributes.
  - Attribute values are optional.
  - Closing empty elements are optional

CREATIVE

- **The following tags have been introduced for better structure :**
  - <header>
    - This tag represents the header of a section.
  - <nav>
    - This tag represents a section of the document intended for navigation.
  - <section>
    - This tag represents a generic document or application section.
  - <article>
    - This tag represents an independent piece of content of a document, such as a blog entry or newspaper article.
  - <aside>
    - This tag represents a piece of content that is only slightly related to the rest of the page.
  - <footer>
    - This tag represents a footer for a section and can contain information about the author, copyright information, et cetera.

HTML5 Web Form

HTML5 Form Elements	
<input type="color">	Displays a color picker box.
<input type="date">	Displays a date box.
<input type="time">	Displays a time box.
<input type="datetime-local">	Displays a date with time box.
<input type="month">	Displays a month box.
<input type="week">	Displays a week box.
<input type="number">	Displays a number box.
<input type="range">	Displays a range bar.
<input type="email">	Displays a email box.
<input type="url">	Displays a url box.
<input type="file">	Displays a Choose File Button.

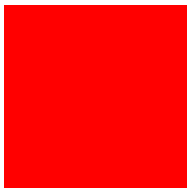
CREATIVE

What is Canvas ?

- The <canvas> tag is used to draw graphics, on the fly, via scripting (usually JavaScript).
- It can be used to draw graphs, make photo compositions or do simple (and not so simple) animations.
- Here is a simple <canvas> element which has only two specific attributes **width** and **height**.
- Example :

```
<!DOCTYPE HTML>
<html>
<head>
    <title>My Canvas</title>
</head>
<body>
        <canvas id="demo" width="100“ height="100"></canvas>

        <script>
            var canvas = document.getElementById("demo");
            var ctx = canvas.getContext("2d");
            ctx.fillStyle = "#FF0000";
            ctx.fillRect(0, 0, 80, 80);
        </script>
 </body>
</html>
```

- Output :



CREATIVE

Audio:

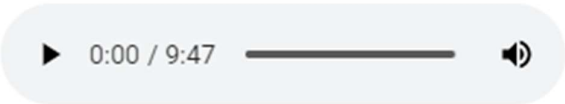
- The <audio> tag is used to embed sound content in a document, such as music or other audio streams.
- There are three supported audio formats in HTML: MP3, WAV, and OGG.
- To play an audio file in HTML, use the <audio> element:
- Syntax :

```
<audio controls>
    <source src="Your path here"
        type="[audio/mpeg | audio/ogg | audio/wav] ">
</audio>
```

- Example :

```
<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
</audio>
```

- Output :



Video:

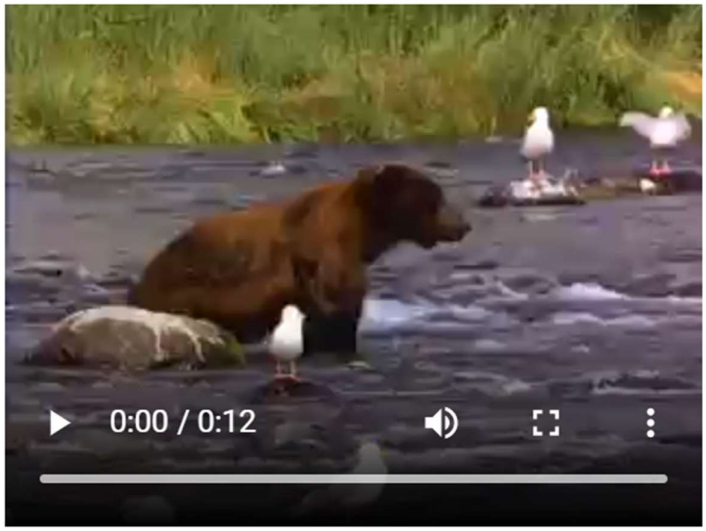
- The <video> tag is used to embed video content in a document, such as a movie clip or other video streams.
- The <video> tag contains one or more <source> tags with different video sources. The browser will choose the first source it supports.
- Syntax :

```
<video controls autoplay loop muted width height>
    <source src="Your path here" type="[video/mp4 | video/ogg |
video/webM] ">
</video>
```

- Example :

```
<video width="320" height="240" controls>
    <source src="movie.mp4" type="video/mp4">
    <source src="movie.mp4" type="video/ogg">
</video>
```

- Output :



CREATIVE

Chapter 3

Cascading Style Sheet & CSS3

CREATIVE

What is CSS?

- **CSS** stands for **Cascading Style Sheets.**
- CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.
- **CSS** describes how HTML elements are to be displayed on screen, paper, or in other media.

Benefits of CSS

- **CSS saves time** :
  - You can write CSS once and then 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 web pages will be updated automatically.
- **Global web standards**:
  - It's a good idea to start using CSS in all the HTML pages to make them compatible with future browsers.
- **Platform Independence**:
  - The Script offer consistent platform independence and can support latest browsers as well.

CSS Comments

- Comments are used to explain the code, and may help when you edit the source code at a later date.
- Comments are ignored by browsers.
- A CSS comment is placed inside the <style> element, and starts with /* and ends with */
- Syntax:

```
/* This is CSS Comment Text  */
```

Types Of CSS / Stylesheet

- CSS can be added to HTML documents in 3 ways:
  1. **Inline** - by using the style attribute inside HTML elements
  2. **Internal** - by using a <style> element in the <head> section
  3. **External** - by using a <link> element to link to an external CSS file
  4. **Multiple CSS**
- The most common way to add CSS, is to keep the styles in external CSS files.

1. Inline CSS :

- An inline CSS is used to apply a unique style to a single HTML element.
- An inline CSS uses the style attribute of an HTML element.
- Example:

```
<h1 style="color:blue;">A Blue Heading</h1>

<p style="color:red;">A red paragraph.</p>
```

2. Internal CSS :

- An internal CSS is used to define a style for a single HTML page.
- An internal CSS is defined in the <head> section of an HTML page, within a <style> element.
- Example :

```
<!DOCTYPE html>
<html>
<head>
    <style>
        body {background-color: powderblue;}
        h1   {color: blue;}
        p    {color: red;}
    </style>
```

```
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
</html>
```

3. External CSS :

- An external style sheet is used to define the style for many HTML pages.
- To use an external style sheet, add a link to it in the <head> section of each HTML page:
- Example:

```
<!DOCTYPE html>
<html>
<head>
        <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
        <h1>This is a heading</h1>
        <p>This is a paragraph.</p>
</body>
</html>
```

style.css

```
body
{
        background-color: powderblue;
}
h1
{
        color: blue;
}
p
{
        color: red;
}
```

CREATIVE

4. Multiple CSS :

- Actually Multiple Style Sheet is not a type of Style Sheets but it is a combination of
  - External Style Sheet,
  - Internal Style Sheet and
  - Inline Style Sheet
    - as per designers requirement.
- We can use more then one style sheet in a HTML document file.
- This method of working with more then one style sheet that's why it is known as Multiple Style Sheet.
- Example:

```
<!DOCTYPE html>
<html>
<head>
        <link rel="stylesheet" type="text/css" href="style.css">
        <style>
                h1   {color: blue;}
                p    {color: red;}
        </style>
</head>
<body>
        <h1>This is a heading</h1>
        <p style="font-size:20px;" >This is a paragraph.</p>
</body>
</html>
```

style.css

```
body
{
        background-color: powderblue;
}
p
{
        background-color: yellow;
}
```

CREATIVE

CSS Slectors

- There are two types of selector:
    1. Class
    2. ID

1. Class :

- The .class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character, followed by the name of the class.
- HTML elements can also refer to more than one class (look at Example 2 below).
- Syntax :

```
.class {
    css declarations;
}
```

- Example :

```
<html>
<head>
    <style>
        p.center {
            text-align: center;
            color: red;
        }
        p.large {
            font-size: 30px;
        }
    </style>
</head>
<body>
    <p class="center large">This paragraph refers to two classes.</p>
</body>
</html>
```

2. ID :

- The #id selector styles the element with the specified id.
- Syntax :

```
#id {
    css declarations;
}
```

- Example :

```
<html>
<head>
    <style>
        #firstname {
            font-size: 25px;
            color: blue;
        }
    </style>
</head>
<body>
    <p id="firstname">This paragraph refers to two classes.</p>
</body>
</html>
```

CREATIVE

CSS Font Properties

- The CSS font properties allow you to change the font family, boldness, size and the style of a text.

[1] Font-family Property

[2] Font-style Property

[3] Font-weight Property

[4] Font-size Property

[5] Font-variant Property

1. Font-family :

- The font-family property specifies the font for an element.
- There are two types of font family names:
  - **family-name** - The name of a font-family, like "times", "courier", "arial", etc.
  - **generic-family** - The name of a generic-family, like "serif", "sans-serif", "cursive", "fantasy", "monospace".
- Syntax :

font-family: *family-name*|*generic-family*

- Example :

```
<html>
<head>
    <style>
        p {
            font-family: "calibri", Times, serif;
        }
    </style>
</head>
<body>
    <h1>The font-family Property</h1>
    <p>This is a paragraph, shown in the Times New Roman font.</p>
</body>
</html>
```

CREATIVE

2. Font-style :

- The font-style property specifies the font style for a text.
- Syntax :

```
font-style: normal | italic | oblique;
```

- Example :

```
<html>
<head>
    <style>
        p {
            font-style: italic;
        }
    </style>
</head>
<body>
    <h1>The font-family Property</h1>
    <p>This is a paragraph, shown in the Times New Roman font.</p>
</body>
</html>
```

3. Font-weight :

- The font-weight property sets how thick or thin characters in text should be displayed.
- Syntax :

```
font-weight: normal | bold | bolder | lighter | number
```

- Example :

```
<html>
<head>
    <style>
        p {
            font-weight: bold;
        }
    </style>
</head>
<body>
    <p>This is a paragraph.</p>
</body>
</html>
```

CREATIVE

4. **Font-size :**

- The font-size property sets the size of a font.
- Syntax :

```
font-size : medium | xx-small | x-small | small | large | x-large |
            xx-large | smaller | larger | length | % | px
```

- Example :

```
<html>
<head>
    <style>
        p { font-size: xx-large; }
        font { font-size: xx-small; }
    </style>
</head>
<body>
    <p>This is a xx-large paragraph.</p>
    <font>This is a xx-small paragraph.</font>
</body>
</html>
```

5. **Font-variant :**

- In a small-caps font, all lowercase letters are converted to uppercase letters.
- Syntax :

```
font-variant : normal | small-caps
```

- Example :

```
<html>
<head>
    <style>
        h1 { font-variant: small-caps; }
    </style>
</head>
<body>
    <h1> Play with the two different font variants!</h1>
</body>
</html>
```

CREATIVE

CSS Text Properties

- The CSS text properties allow you to change the appearance of text.
- It is possible to change the color of text, increase or decrease the space between characters in a text, align a text, decorate a text, indent the first line in a text and more.

[1] Color Property

[2] Text-align Property

[3] Text-decoration Property

[4] Text-transform Property

[5] Word-spacing Property

[6] Letter-spacing Property

1. Color :

- The color property specifies the color of text.
- Syntax :

```
Color : colorname;
```

- Example :

```
<html>
<head>
    <style>
        body {  color: red;  }
        h1 { color: #00ff00;  }
        font { color: rgb(0,0,255);  }
    </style>
</head>
<body>
    <h1>This is heading 1</h1>
    <p>This is an ordinary paragraph. Notice that this text is red. The default text-color for a page is defined in the body selector.</p>
    <font>This is a paragraph with font. This text is blue.</font>
</body>
</html>
```

CREATIVE

2. Text-align :

- The text-align property specifies the horizontal alignment of text in an element.
- Syntax :

```
text-align: left | right | center | justify ;
```

- Example :

```
<html>
<head>
    <style>
        p { text-align: center; }
    </style>
</head>
<body>
    <p>This is an ordinary paragraph. Notice that this text is red. The
    default text-color for a page is defined in the body selector.</p>
</body>
</html>
```

3. Text-decoration :

- The text-decoration property specifies the decoration added to text, and is a shorthand property for.
  - text-decoration-line (required)
  - text-decoration-color
  - text-decoration-style
- Syntax :

  text-decoration: [text-decoration-line] [text-decoration-color] [text-decoration-style];

- Example :

```
<html>
<head>
    <style>
        h1 { text-decoration: overline; }
        h2 { text-decoration: line-through; }
        h3 { text-decoration: underline; }
        h4 { text-decoration: underline overline; }
    </style>
```

CREATIVE

```
</head>
<body>

    <h1>This is heading 1</h1>
    <h2>This is heading 2</h2>
    <h3>This is heading 3</h3>
    <h4>This is heading 4</h4>

</body>
</html>
```

4. Text-transform :

- The text-transform property controls the capitalization of text.
- Syntax :

  text- transform : none | capitalize | uppercase | lowercase

- Example :

```
<html>
<head>
    <style>
        h1 {  text-transform : capitalize; }
        h2 {  text-transform : uppercase; }
        h3 {  text-transform : lowercase; }
    </style>
</head>
<body>

    <h1>This is capitalize text.</h1>
    <h2>This is uppercase text.</h2>
    <h3>This is lowercase.</h3>

</body>
</html>
```

CREATIVE

5. Word-spacing :

- The word-spacing property increases or decreases the white space between words.
- Syntax :

  text- spacing : normal | *length*

- Example :

```
<html>
<head>
    <style>
        p {  word-spacing: 30px; }
    </style>
</head>
<body>
    <p>This is some text. This is some text.</p>
</body>
</html>
```

6. letter-spacing :

- The letter-spacing property increases or decreases the space between characters in a text.
- Syntax :

  Letter-spacing : normal | *length*

- Example :

```
<html>
<head>
    <style>
        p {  letter-spacing: 5px; }
    </style>
</head>
<body>
    <p>This is some text. This is some text.</p>
</body>
</html>
```

CREATIVE

CSS Background Properties

- The CSS Background properties allow you to control the background color of an element.
- Set an image as the background, repeat a background image, vertically or horizontally and position an image on a page.

[1] Background-color Property

[2] Background-image Property

[3] Background-repeat Property

[4] Background-attachment Property

[5] Background-position Property

CREATIVE

1. Background-color :

- The background-color property sets the background color of an element.
- Syntax :

```
background-color: color | transparent;
```

- Example :

```
<html>
<head>
    <style>
        body {  background-color: orange;  }  OR
        body {  background-color: #ff9900;  }  OR
        body {  background-color: rgb(255,130,255);  }
    </style>
</head>
<body>

    <h1>The background-color Property</h1>
    <p>The background color can be specified with a color name.</p>

</body>
</html>
```

2. **Background-image :**

- The background-image property sets one or more background images for an element.
- By default, a background-image is placed at the top-left corner of an element, and repeated both vertically and horizontally.
- Syntax :

```
background-image: URL | none;
```

- Example :

```
<html>
<head>
    <style>
        body {  background-color: url("nature.jpg");  }
    </style>
</head>
<body>

    <h1>The background-image Property</h1>
    <p>Hello World!</p>

</body>
</html>
```

3. **Background-repeat :**

- The background-repeat property sets if/how a background image will be repeated.
- By default, a background-image is repeated both vertically and horizontally.
- Syntax :

```
background-repeat: repeat | repeat-x | repeat-y | no-repeat;
```

Value	Description
repeat	The background image is repeated both vertically and horizontally
repeat-x	The background image is repeated only horizontally
repeat-y	The background image is repeated only vertically
no-repeat	The background-image is not repeated. The image will only be shown once

CREATIVE

- Example :

```
<html>
<head>
    <style>
        body {
            background-color: url("pattern.jpg");
            background-repeat: no-repeat;
            }
    </style>
</head>
<body>

    <h1>The background-repeat Property</h1>
    <p> Here, the background image is repeated only vertically.</p>

</body>
</html>
```

4. Background-attachment :

- The background-attachment property sets whether a background image scrolls with the rest of the page, or is fixed.
- Syntax :

```
background-attachment: scroll | fixed;
```

Value	Description
scroll	The background image will scroll with the page. This is default
fixed	The background image will not scroll with the page

- Example :

```
<html>
<head>
    <style>
        body {
            background-color: url("pattern.jpg");
```

```
                    background-repeat: no-repeat;
                    background-attachment: fixed;
                    }
        </style>
</head>
<body>

        <h1>The background-attachment Property</h1>
        <p> The background-image is fixed. Try to scroll down the
page..</p>
        .
        .
        .
        <p> The background-image is fixed. Try to scroll down the
page..</p>

</body>
</html>
```

5. **Background-position :**

- The background-position property sets the starting position of a background image.
- Syntax :

```
background-position: value;
```

Value	Description
left top **left center** **left bottom** **right top** **right center** **right bottom** **center top** **center center** **center bottom**	If you only specify one keyword, the other value will be "center"
x% y%	The first value is the horizontal position and the second value is the vertical. The top left corner is 0% 0%.
xpos ypos	The first value is the horizontal position and the second value is the vertical. The top left corner is 0 0.

- Example :

```
<html>
<head>
    <style>
        body {
            background-color: url("pattern.jpg");
            background-repeat: repeat;
            background-position: center bottom;
            }
    </style>
</head>
<body>

    <h1>The background-repeat Property</h1>
    <p> Here, the background image is repeated only vertically.</p>

</body>
</html>
```

CSS List-style Properties

- The list-style property is a shorthand for the following properties:
  - list-style-type
  - list-style-position
  - list-style-image
- Syntax :

  list-style: *list-style-type list-style-position list-style-image;*

1. List-style-type :

- The list-style-type specifies the type of list-item marker in a list.
- Syntax :

  list-style-type: value;

Value	Description
none	No marker is shown
disc	Default value. The marker is a filled circle
circle	The marker is a circle

CREATIVE

square	The marker is a square
decimal	The marker is a number
lower-alpha	The marker is lower-alpha (a, b, c, d, e, etc.)
lower-greek	The marker is lower-greek
lower-latin	The marker is lower-latin (a, b, c, d, e, etc.)
lower-roman	The marker is lower-roman (i, ii, iii, iv, v, etc.)
upper-alpha	The marker is upper-alpha (A, B, C, D, E, etc.)
upper-greek	The marker is upper-greek
upper-latin	The marker is upper-latin (A, B, C, D, E, etc.)
upper-roman	The marker is upper-roman (I, II, III, IV, V, etc.)
almenian	The marker is traditional Armenian numbering

- Example :

```
<html>
<head>
    <style>
        ul {
            list-style-type: decimal;
        }
    </style>
</head>
<body>
    <ul>
        <li>Coffee</li>
        <li>Tea</li>
        <li>Coca Cola</li>
    </ul>
</body>
</html>
```

2. **List-style-position :**

- The list-style-position property specifies the position of the list-item markers (bullet points).
- Syntax :

```
list-style-position: inside | outside;
```

CREATIVE

1. inside : The bullet points will be inside the list item
2. outside : The bullet points will be outside the list item. This is default

- Example :

```
<html>
<head>
    <style>
        ul {
            list-style-position: inside;
        }
    </style>
</head>
<body>
    <ul>
        <li>Coffee</li>
        <li>Tea</li>
        <li>Coca Cola</li>
    </ul>
</body>
</html>
```

3. List-style-image :

- The list-style-image property replaces the list-item marker with an image.
- Syntax :

```
list-style-image: none | URL;
```

- Example :

```
<html>
<head>
    <style>
        ul {
            list-style-image: url('square.png');
        }
    </style>
</head>
<body>
    <ul>
        <li>Coffee</li>
```

CREATIVE

```
            <li>Tea</li>
            <li>Coca Cola</li>
        </ul>
    </body>
    </html>
```

CSS Padding Properties

- An element's padding is the space between its content and its border.
- This property can have from one to four values.
    - **If the padding property has four values:**
        - padding:10px 5px 15px 20px;
            - top padding is 10px
            - right padding is 5px
            - bottom padding is 15px
            - left padding is 20px
    - **If the padding property has three values:**
        - padding:10px 5px 15px;
            - top padding is 10px
            - right and left padding are 5px
            - bottom padding is 15px
    - **If the padding property has two values:**
        - padding:10px 5px;
            - top and bottom padding are 10px
            - right and left padding are 5px
    - **If the padding property has one value:**
        - padding:10px;
            - all four paddings are 10px
    - padding-top: 10px;
    - padding-bottom: 10px;
    - padding-left: 10px;

CREATIVE

- padding-right: 10px;

- **Note:** Negative values are not allowed.
- Syntax:

```
padding: length;
```

  - *length* - specifies a padding in px, pt, cm, etc.
  - *%* - specifies a padding in % of the width of the containing element.

- Example:

```
<html>
<head>
    <style>
        p {
            padding: 70px;
        }
    </style>
</head>
<body>
    <p>
        This element has a padding of 70px.
    </p>
</body>
</html>
```

CSS Margin Properties

- The CSS margin properties are used to create space around elements, outside of any defined borders.
- This property can have from one to four values.
  - **If the margin property has four values:**
    - margin :10px 5px 15px 20px;
      - top margin is 10px
      - right margin is 5px
      - bottom margin is 15px
      - left margin is 20px

CREATIVE

- **If the margin property has three values:**
  - margin :10px 5px 15px;
    - top margin is 10px
    - right and left margin are 5px
    - bottom margin is 15px
- **If the margin property has two values:**
  - margin :10px 5px;
    - top and bottom margin are 10px
    - right and left margin are 5px
- **If the margin property has one value:**
  - margin :10px;
    - all four margin are 10px
- margin-top: 10px;
- margin-bottom: 10px;
- margin-left: 10px;
- margin-right: 10px;

- **Note:** Negative values are allowed.
- Syntax:

  margin: length | auto;

  - *length* - specifies a padding in px, pt, cm, etc.
  - *%* - specifies a padding in % of the width of the containing element.
  - *Auto* - The browser calculates a margin

- Example:

```
<html>
<head>
    <style>
        p {
            margin: 30px;
        }
    </style>
```

```
</head>
<body>
    <p>
        This element has a margin of 30px.
    </p>
    <p>
        This element has a margin of 30px.
    </p>
</body>
</html>
```

CSS Border Properties

- The CSS border properties allow you to specify the style, width, and color of an element's border.
- The border property is a shorthand property for:
  - border-width
  - border-style (required)
  - border-color
- Syntax :

  border: border-width border-style border-color;

1. **Border-width :**
   - The border-width property sets the width of an element's four borders. This property can have from one to four values.
   - The width can be set as a specific size (in px, pt, cm, em, etc) or by using one of the three pre-defined values: thin, medium, or thick.
   - Syntax:

     border-width: medium | thin | thick | *length*;

   - **If the border-style property has four values:**
     - border-width: thin medium thick 10px;

- top border is thin
- right border is medium
- bottom border is thick
- left border is 10px

- **If the border-style property has three values:**
  - border-width: thin medium thick;
    - top border is thin
    - right and left borders are medium
    - bottom border is thick

- **If the border-style property has two values:**
  - border-width: thin medium;
    - top and bottom borders are thin
    - right and left borders are medium
- **If the border-style property has one values:**
  - border-width: thin;
    - all four borders are thin
- border-top-width: 10px;
- border-bottom-width: 10px;
- border-left-width: 10px;
- border-right-width: 10px;
- Example : **HTML Code :**

```
<html>
<head>
    <style>
        p {
            border-style: solid;
            border-width: thin 20px medium;
        }
    </style>
</head>
```

```
<body>

    <p>Some Text Here.</p>

</body>
</html>
```

2. Border-style :

- The border-style property sets the style of an element's four borders. This property can have from one to four values.
- Syntax:

  border-style: none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset;

dotted	Defines a dotted border
dashed	Defines a dashed border
solid	Defines a solid border
double	Defines a double border
groove	Defines a 3D grooved border. The effect depends on the border color value
ridge	Defines a 3D ridged border. The effect depends on the border color value
inset	Defines a 3D inset border. The effect depends on the border color value
outset	Defines a 3D outset border. The effect depends on the border color value
none	Defines no border
hidden	Defines a hidden border

- **If the border-style property has four values:**
  - border-style: dotted solid double dashed;
    - top border is dotted
    - right border is solid
    - bottom border is double
    - left border is dashed
- **If the border-style property has three values:**
  - border-style: dotted solid double;
    - top border is dotted

- right and left borders are solid
- bottom border is double

- **If the border-style property has two values:**
  - border-style: dotted solid;
    - top and bottom borders are dotted
    - right and left borders are solid
- **If the border-style property has one values:**
  - border-style: dotted;
    - all four borders are dotted
- border-top-style: dotted;
- border-bottom- style: dashed;
- border-left- style: double;
- border-right- style: solid;

- Example : **HTML Code :**

```
<html>
<head>
    <style>
        p {
            border-style: dotted;
        }
    </style>
</head>
<body>

    <p>Some Text Here.</p>

</body>
</html>
```

3. **Border-color :**

- The border-color property is used to set the color of the four borders.
- The color can be set by:
  - ✓ name - specify a color name, like "red"

- ✓ HEX - specify a HEX value, like "#ff0000"
- ✓ RGB - specify a RGB value, like "rgb(255,0,0)"
- ✓ transparent

- Syntax:

  border-color: color | transparent;

- **If the border-color property has four values:**
  - border-color: red green blue pink;
    - top border is red
    - right border is green
    - bottom border is blue
    - left border is pink
- **If the border-color property has three values:**
  - border-color: red green blue;
    - top border is red
    - right and left borders are green
    - bottom border is blue
- **If the border-color property has two values:**
  - border-color: red green;
    - top and bottom borders are red
    - right and left borders are green
- **If the border-color property has one value:**
  - border-color: red;
    - all four borders are red
- border-top-color: red;
- border-bottom- color: green;
- border-left- color: blue;
- border-right- color: cyan;
- Example : **HTML Code :**

```
<html>
<head>
    <style>
        p {
            border-color: red;
        }
    </style>
</head>
<body>
    <p>Some Text Here.</p>
</body>
</html>
```

CSS3 Backgound-Gradients Properties

- CSS3 gradients let you display smooth transitions between two or more specified colors.
- CSS3 defines two types of gradients:
    - **Linear Gradients** (goes down/up/left/right/diagonally)
    - **Radial Gradients** (defined by their center)

1. **Linear Gradients:**
    - To create a linear gradient you must define at least two color stops.
    - Color stops are the colors you want to render smooth transitions among.
    - You can also set a starting point and a direction (or an angle) along with the gradient effect.
    - Syntax :

      background: linear-gradient(direction, color-stop1, color-stop2, ...);

    - Example :

```
<html>
<head>
    <style>
        div { height: 300px; }
```

```
            div.simple {  background: linear-gradient(red, yellow);  }
            div.side {  background: linear-gradient(to right, red , yellow); }
            div.angle {  background: linear-gradient(45deg, red , yellow); }
        </style>
    </head>
    <body>
        <div class="simple">Some Text Here.</div>
        <div class="side">Some Text Here.</div>
        <div class="angle">Some Text Here.</div>
    </body>
    </html>
```

2. **Radial Gradients:**

- A radial gradient is defined by its center.
- To create a radial gradient you must also define at least two color stops.
- Syntax :

  background: radial-gradient(*shape size* at *position, start-color, ..., last-color*);

- Example :

```
<html>
<head>
    <style>
        div { height: 300px; }
        div.simple {  background: radial-gradient(red, yellow, green);  }
        div.stops {  background: radial-gradient(red 5%, yellow 15%, green
        60%) ; }
        div.shape {  background: radial-gradient(circle, red , yellow, green);
        }
    </style>
</head>
<body>
    <div class="simple">Some Text Here.</div>
    <div class="stops">Some Text Here.</div>
    <div class="shape">Some Text Here.</div>
</body>
</html>
```

CREATIVE

CSS3 Shadow Properties

- With CSS3 you can add shadow to text and to elements.
- In this chapter you will learn about the following properties:
    1. **text-shadow**
    2. **box-shadow**

1. **text-shadow:**
    - The CSS3 text-shadow property applies shadow to text.
    - Syntax :

```
text-shadow: x  y  blur  color;
```

    - Example :

```
<html>
<head>
    <style>
        div.simple {  text-shadow: 0 0 3px #FF0000;  }
        div.multiple {  text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF; }
    </style>
</head>
<body>
    <div class=”simple”>Some Text Here.</div>
    <div class=”multiple”>Some Text Here.</div>
</body>
</html>
```

2. **box-shadow:**
    - The CSS3 box-shadow property applies shadow to elements.
    - Syntax :

```
box-shadow: x  y  blur  color;
```

    - Example :

```
<html>
<head>
    <style>
        div {
            height: 300px; box-shadow: 10px 10px 5px gray;  }
    </style>
```

```
</head>
<body>
    <div>Some Text Here.</div>
</body>
</html>
```

CSS3 2D/3D Transforms Properties

- CSS3 transforms allow you to translate, rotate, scale, and skew elements.
- A transformation is an effect that lets an element change shape, size and position.
- CSS3 supports 2D and 3D transformations.
    1. **translate()**
    2. **rotate() => rotateX(),rotateY()**
    3. **scale()**
    4. **skew()**

1. **translate:**
    - The translate() method moves an element from its current position (according to the parameters given for the X-axis and the Y-axis).
    - Syntax :

      ```
      transform: translate(x, y);
      ```

    - Example :

```
<html>
<head>
    <style>
        div {
            width: 200px;
            height: 200px;
            background-color: red;
            transform: translate(50px, 100px); }
    </style>
</head>
<body>
    <div>Transformed Element</div>
```


CREATIVE

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

2. **rotate:**

- The rotate() method rotates an element clockwise or counter-clockwise according to a given degree.
- Syntax :

```
transform: rotate(deg);
```

- Example :

```
<html>
<head>
    <style>
        div {
            width: 200px;
            height: 200px;
            background-color: red;
            transform: rotate(45deg); }
    </style>
</head>
<body>
    <div>Transformed Element</div>
</body>
</html>
```

3. **Scale:**

- The scale() method increases or decreases the size of an element (according to the parameters given for the width and height).
- Syntax :

```
transform: scale(x, y);
```

- Example :

```
<html>
<head>
    <style>
        div {
            width: 200px;
```

CREATIVE

```
                height: 200px;
                background-color: red;
                transform: scale(2, 3); }
        </style>
    </head>
    <body>
        <div>Transformed Element</div>
    </body>
    </html>
```

4. **skew:**

- The skew() method skews an element along the X-axis and Y-axis by the given angle.
- Syntax :

```
transform: skew(x, y);
```

- Example :

```
<html>
<head>
    <style>
        div {
            width: 200px;
            height: 200px;
            background-color: red;
            transform: skew(20deg, 10deg); }
    </style>
</head>
<body>
    <div>Transformed Element</div>
</body>
</html>
```

CREATIVE

CSS3 Transition Properties

- To create a transition effect, you must specify two things:
  - the CSS property you want to add an effect to
  - the duration of the effect
- Example :

```
<!DOCTYPE html>
<html>
<head>
<style>
        .div1 {
          width: 100px;
          height: 100px;
          background: red;
          transition: width 2s;
        }
        .div1:hover {
          width: 300px;
        }
</style>
</head>
<body>

        <div class="div1"> Hover over the div element above, to see the
        transition effect.</div>

</body>
</html>
```

CREATIVE

CSS3 Position Properties

- The position property specifies the type of positioning method used for an element.
- There are four different position values:
  - **static**
  - **relative**
  - **fixed**
  - **absolute**
  - **sticky**
- **Example:**
  - HTML elements are positioned static by default.
  - Static positioned elements are not affected by the top, bottom, left, and right properties.

```
<!DOCTYPE html>
<html>
<head>
<style>
        div.static {
                border: 2px green solid;
                position: static;
        }
        div.relative {
                border: 2px green solid;
                position: relative;
                left: 50px;
        }
        div.fixed {
                border: 2px green solid;
                position: fixed;
                right: 50px;
                bottom: 50px;
        }


        div.absolute {
```


CREATIVE

```
            border: 2px green solid;
            position: absolute;
            top: 100px;
            left: 200px;
        }
        div.sticky{
            height: 600px;
            background-color: red;
        }
        div.sticky h1{
            padding: 30px;
            background-color: yellow;
            position: sticky;
            top: 0px;
        }


</style>
</head>
<body>

    <div class="static"> This div element has position: static;</div>
    <div class="relative"> This div element has position: relative;</div>
    <div class="fixed"> This div element has position: fixed;</div>
    <div class="absolute"> This div element has position: absolute;</div>
    <div class="sticky">
        <h1>This div element has position: sticky; </h1>
    </div>

</body>
</html>
```

CREATIVE

CSS Pseudo Classes

- A CSS ***pseudo-class*** is a keyword added to a selector that specifies a special state of the selected element(s).
- A pseudo-class consists of a colon (:) followed by the pseudo-class name (e.g., :hover).
- A functional pseudo-class also contains a pair of parentheses to define the arguments (e.g., :dir()).
- The element that a pseudo-class is attached to is defined as an anchor element (e.g., button in case button:hover).

Selector	Example	Example description
:hover	a:hover	Selects links on mouse over
:first-child	p:first-child	Selects every <p> elements that is the first child of its parent
:first-of-type	p:first-of-type	Selects every <p> element that is the first <p> element of its parent
:last-child	p:last-child	Selects every <p> elements that is the last child of its parent
:last-of-type	p:last-of-type	Selects every <p> element that is the last <p> element of its parent
:nth-child(n)	p:nth-child(2)	Selects every <p> element that is the second child of its parent
:nth-child(odd)	p:nth-child(odd)	Selects every <p> element that is the odd child of its parent
:nth-child(even)	p:nth-child(even)	Selects every <p> element that is the even child of its parent
:nth-of-type(n)	p:nth-of-type(2)	Selects every <p> element that is the second <p> element of its parent
:nth-last-child(n)	p:nth-last-child(2)	Selects every <p> element that is the second child of its parent, counting from the last child
:nth-last-of-type(n)	p:nth-last-of-type(2)	Selects every <p> element that is the second <p> element of its parent, counting from the last child
:only-child	p:only-child	Selects every <p> element that is the only child of its parent
:only-of-type	p:only-of-type	Selects every <p> element that is the only <p> element of its parent
:not(selector)	:not(p)	Selects every element that is not a <p> element
:is(selector)	:is(p)	Selects every element that is a <p> element

All CSS Pseudo Elements

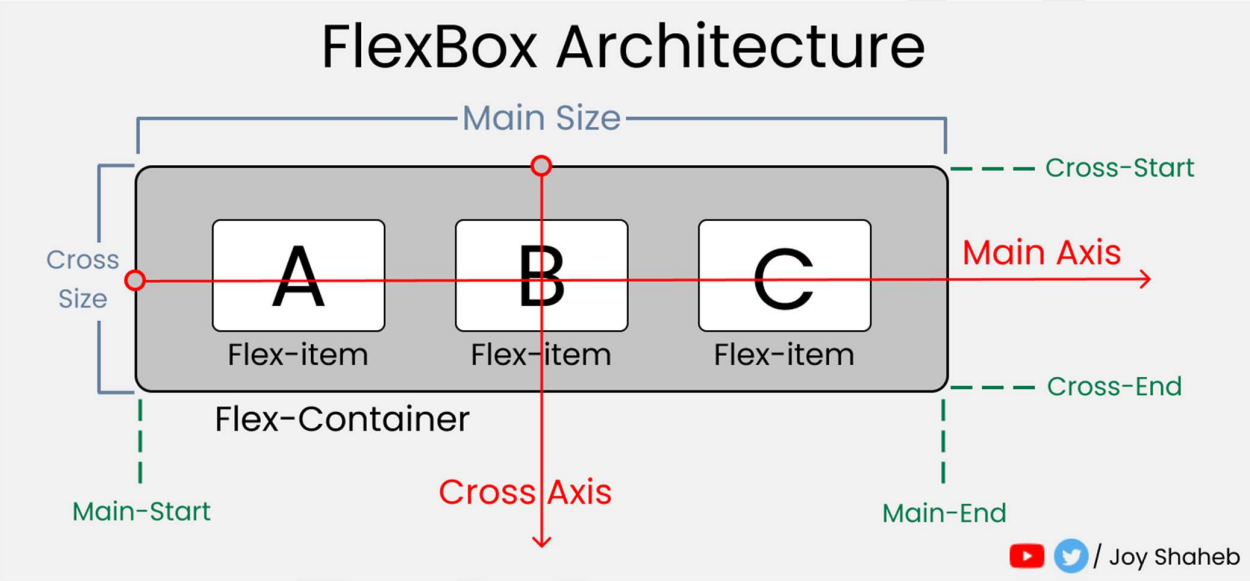
Selector	Example	Example description
::first-letter	p::first-letter	Selects the first letter of each <p> element
::first-line	p::first-line	Selects the first line of each <p> element
::selection	p::selection	Selects the portion of an element that is selected by a user
::placeholder	Input::placeholder	Change placeholder color

CREATIVE

CSS Flexbox Properties

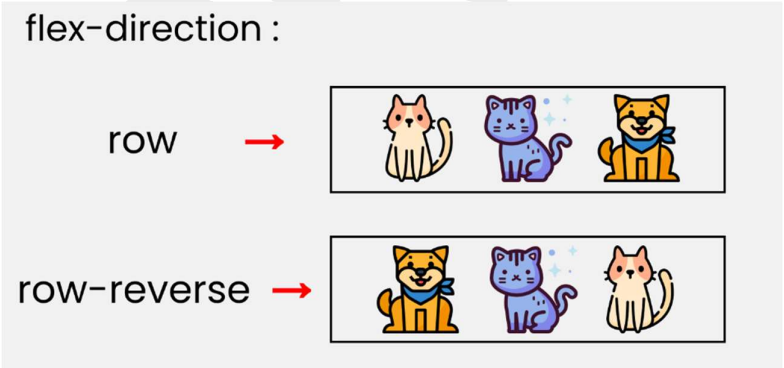
Flexbox Architecture

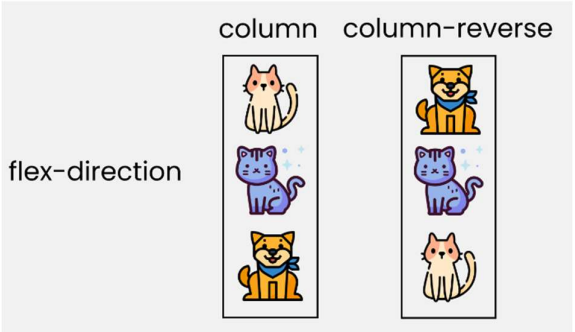
- So how does Flexbox architecture work? The flex-items [Contents] are distributed along the main axis and cross axis. And, depending on the flex-direction property, the layout position changes between rows and columns.



Flex-direction Property :

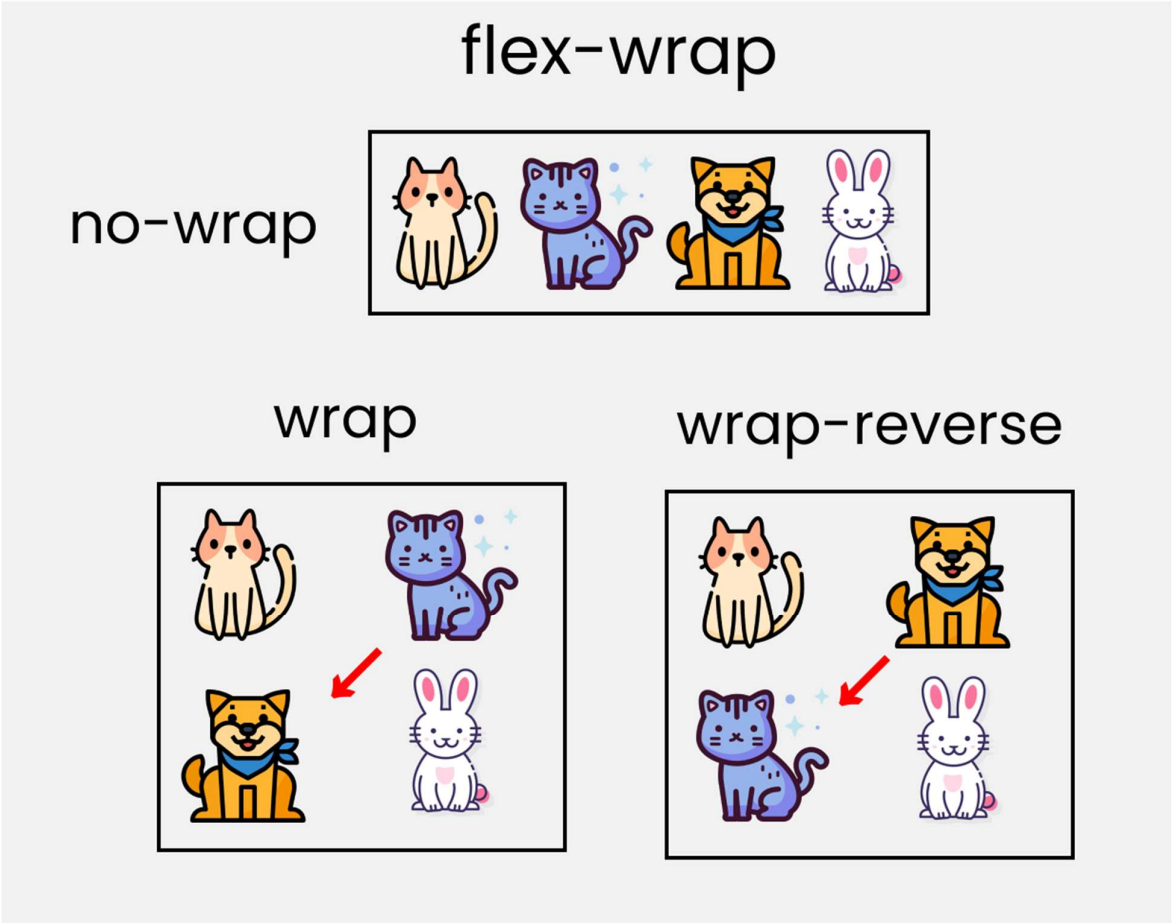
- This property allows us to set the direction and orientation in which our flex-items should be distributed inside the flex-container.





flex–wrap :

- This property helps you set the number of flex-items you want in a line or row.

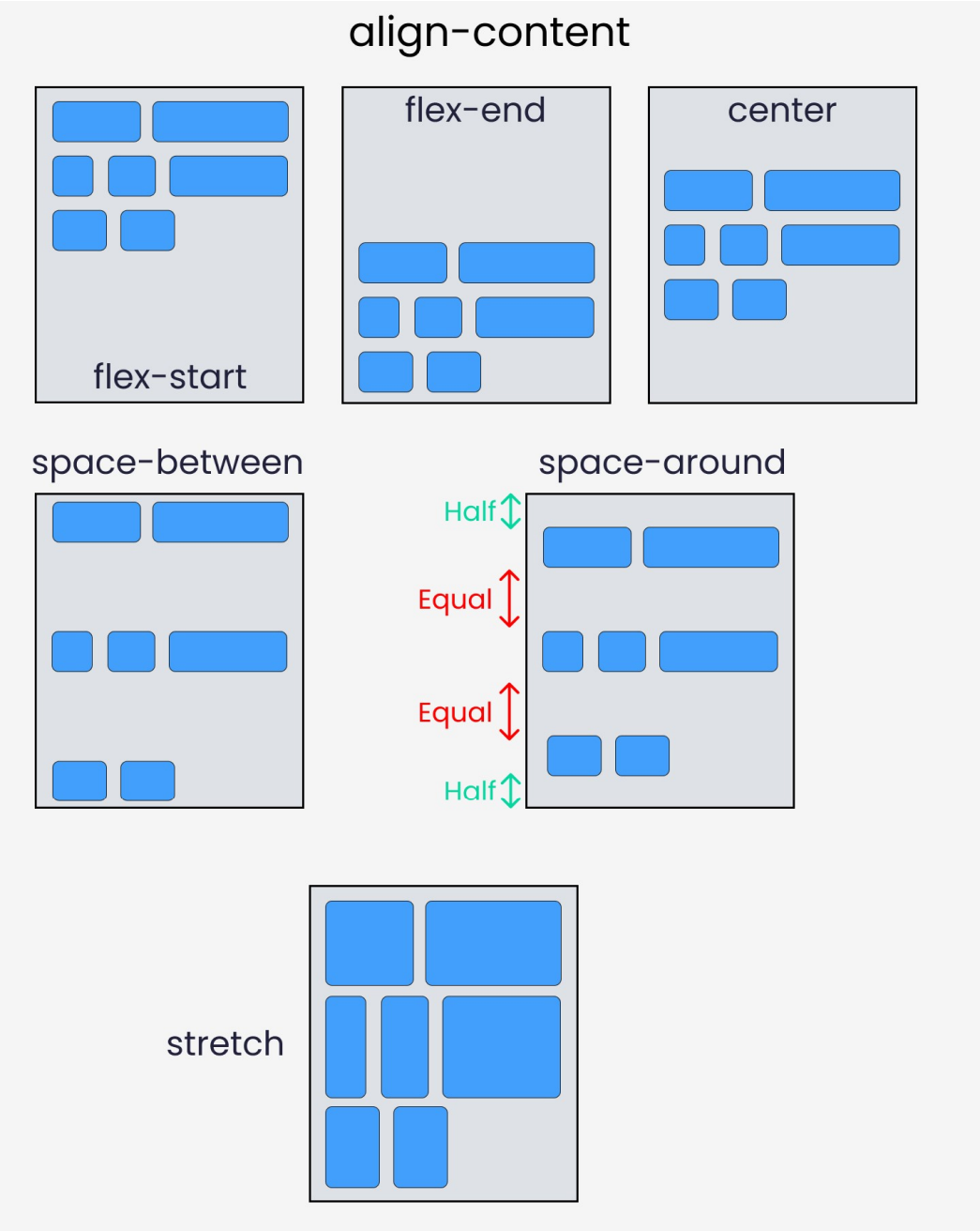


Justify-content Property :

- This property arranges flex-items along the **MAIN AXIS** inside the flex-container.

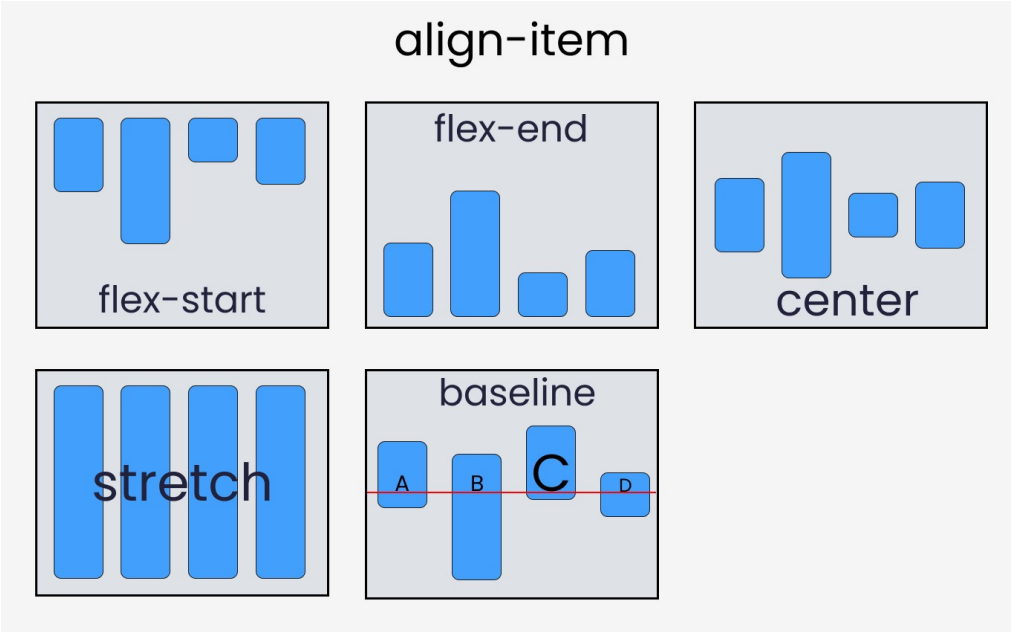
Align-content Property :

- This property arranges flex-items along the CROSS AXIS inside the flex-container. This is similar to justify-content.



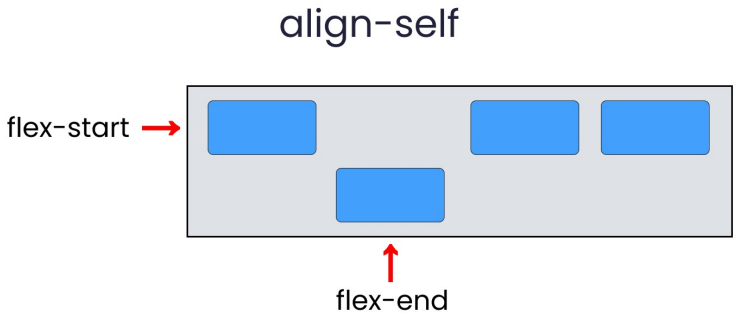
Align-items Property :

- This property distributes Flex-items along the Cross Axis.



Align-self Property :

- This property works on the child classes. It positions the selected item along the **Cross Axis**.



In total we have 6 values:

flex-start	center	stretch
flex-end	baseline	auto

Reference Links:

- https://yoksel.github.io/flex-cheatsheet/
- https://flexbox.malven.co/
- https://flexboxfroggy.com/

CSS Advance Properties

CSS Filter Properties

Filter	Description
none	Default value. Specifies no effects
blur(px)	Applies a blur effect to the image. A larger value will create more blur. **If no value is specified, 0 is used.**
brightness(%)	Adjusts the brightness of the image. **0% will make the image completely black.** **100% (1) is default and represents the original image.** **Values over 100% will provide brighter results.**
contrast(%)	Adjusts the contrast of the image. **0% will make the image completely black.** **100% (1) is default, and represents the original image.** **Values over 100% will provide results with more contrast.**
drop-shadow(h-shadow v-shadow blur spread color)	Applies a drop shadow effect to the image.
grayscale(%)	Converts the image to grayscale. **0% (0) is default and represents the original image.** **100% will make the image completely gray (used for black and white images).** Note: Negative values are not allowed.
hue-rotate(deg)	Applies a hue rotation on the image. The value defines the number of degrees around the color circle the image samples will be adjusted. 0deg is default, and represents the original image. **Note: Maximum value is 360deg.**
invert(%)	Inverts the samples in the image. **0% (0) is default and represents the original image.** **100% will make the image completely inverted.** Note: Negative values are not allowed.
opacity(%)	Sets the opacity level for the image. The opacity-level describes the transparency-level, where: **0% is completely transparent.** **100% (1) is default and represents the original image (no transparency).** Note: Negative values are not allowed. Tip: This filter is similar to the opacity property.
saturate(%)	Saturates the image. **0% (0) will make the image completely un-saturated.** **100% is default and represents the original image.** **Values over 100% provides super-saturated results.** Note: Negative values are not allowed.
sepia(%)	Converts the image to sepia. **0% (0) is default and represents the original image.** **100% will make the image completely sepia.** Note: Negative values are not allowed.

CREATIVE

backdrop-filter :

- The backdrop-filter property is used to apply a graphical effect to the area behind an element.
  - **Syntax :**

    backdrop-filter : none | filter ;
  - Example :

Cursor :

- The cursor property specifies the mouse cursor to be displayed when pointing over an element.

- Syntax :

  cursor : value ;

cursor	alias | alll-scroll | auto | cell | context-menu | col-resize | copy | crosshair | default | e-resize | ew-resize | grab | grabbing | help | move | n-resize | ne-resize | nesw-resize | ns-resize | nw-resize | nwse-resize | no-drop | none | not-allowed | pointer | progress | row-resize | s-resize | se-resize | sw-resize | text | url | w-resize | wait | zoom-in | zoom-out

object-fit :

- The object-fit property is used to specify how an <img> or <video> should be resized to fit its container.

- Syntax :

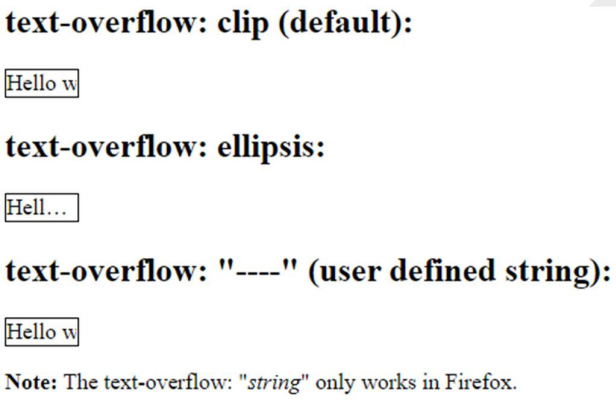
  object-fit : fill | contain | cover | scale-down | none ;

CREATIVE

text-overflow :

- The text-overflow property specifies how overflowed content that is not displayed should be signaled to the user. It can be clipped, display an ellipsis (...), or display a custom string.
- Syntax :

  text-overflow : clip | ellipsis | string ;
- Example :



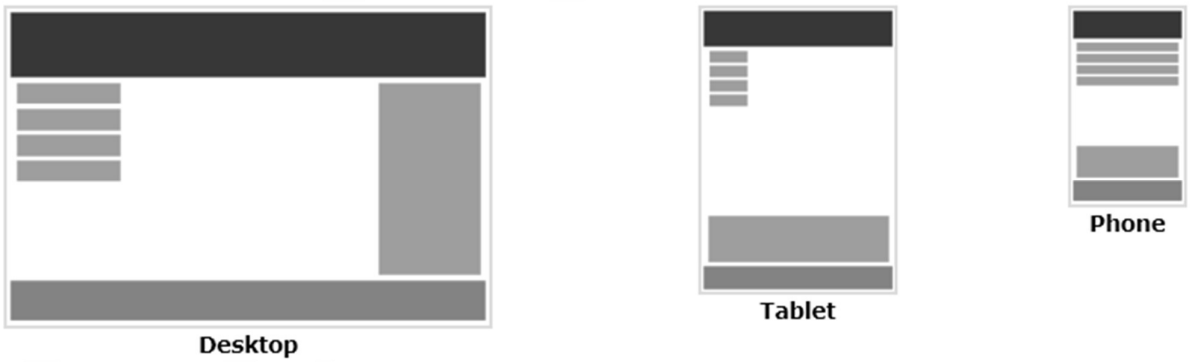
visibility :

- The visibility property specifies whether or not an element is visible.
- Syntax :

  visibility : visible | hidden | collapse ;

Media Query

- The @media at-rule specifies a set of styles that are applied only to certain media types.
- Media queries are a popular technique for delivering a responsive web design to desktops, laptops, tablets, and mobile phones.
- Besides media types, there are media features which have names and accept certain values like properties.
- But there are differences between properties and media features:
  - Properties cannot display without values, and features do not require values.
  - Features accept only single value unlike properties.
- Media queries are used to check the following:
  - width and height of the viewport
  - width and height of the device
  - orientation
  - resolution



Syntax :

- A media query consists of a media type and can contain one or more expressions, which resolve to either true or false.

```
@media not|only mediatype and (expressions) {
        CSS-Code;
}
```

CSS3 Media Types

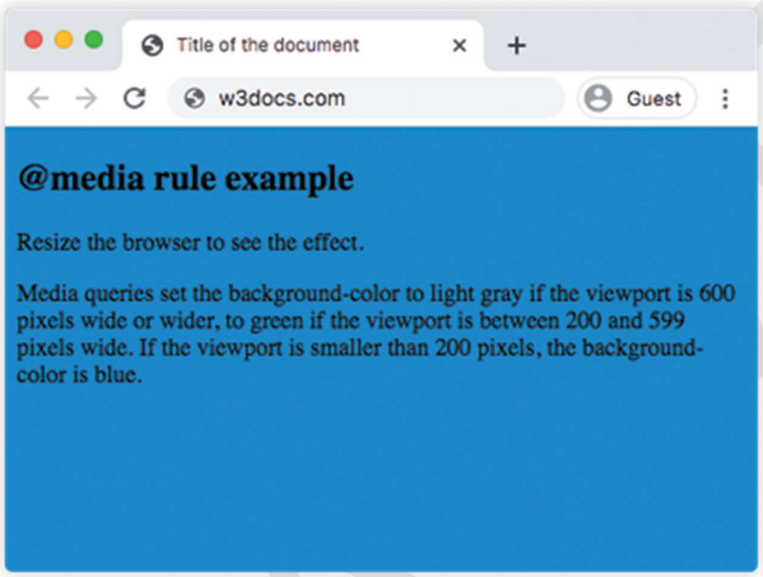
Value	Description
all	Used for all media type devices
print	Used for printers
screen	Used for computer screens, tablets, smart-phones etc.
speech	Used for screenreaders that "reads" the page out loud

Example of the @media rule:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Title of the document</title>
        <style>
         body {
                background-color: #1c87c9;
         }
         @media screen and (max-width: 411px) {
          body {
                background-color: #cce5ff;
          }
         }
         @media screen and (min-width: 768px) {
          body {
                 background-color: #eee;
          }
         }
         @media screen and (max-width: 962px) and (min-width: 450px),
         (min-width: 1366px) {
                background-color: #333;
         }
        </style>
    </head>
  <body>
   <h2>@media rule example</h2>
   <p>Resize the browser to see the effect.</p>
```

```
<p>Media queries set the background-color to light gray if the viewport is 600 pixels wide or wider, to green if the viewport is between 200 and 599 pixels wide. If the viewport is smaller than 200 pixels, the background-color is blue.</p>
</body>
</html>
```

Output :



CREATIVE

Google Font Conectivity (Offline / Online)

- The CSS @font-face is a rule which allows web designers to define online fonts to display text on their website. So, web designers won't need to use "web-safe fonts" anymore.
- We must first define a name for the font (like myFirstFont) in the new @font-face rule and then point to the font file.
- Each browser has its own format. We use ttf, otf, eot, svg, svgz, woff, woff2 formats.

  - **OTF / TTF :**

    The WOFF was created for the reaction of OpenType Font and TrueType Font because these formats could easily (and illegally) be copied. However, OpenType capabilities may interest lots of designers (such as ligatures)

  - **EOT :**

    Embedded Open Type format, created by Microsoft (the original innovators of @font-face) over 15 years ago, is the only format that IE8 and below will recognize when using @font-face.

  - **SVG/SVGZ :**

    Scalable Vector Graphics (Font) is a vector re-creation of the font making it much lighter in file size and ideal for mobile use. It is the only format that is allowed by version 4.1 and below of Safari for iPhone. SVGZ is the SVG's zipped version.

  - **WOFF/WOFF2 :**

    Web Open Font Format, created for use on the web and developed by Mozilla together with other organizations, WOFF fonts often load faster than other formats because they use a compressed version of the structure used by the OpenType (OTF) and TrueType (TTF) fonts. WOFF2 is the new generation of WOFF and it has better compression.

Syntax :

```
@font-face {
    font-properties
}
```

CREATIVE

Example of the @font-face rule :

```
<style>
        @font-face {
        font-family: 'myFont';
         src: url('webfont.eot');
                        /* IE9 Compat Modes */
        src: url('webfont.eot?#iefix') format('embedded-opentype'), /* IE6-IE8 */
         url('webfont.woff2') format('woff2'), /* Super Modern Browsers */
         url('webfont.woff') format('woff'), /* Pretty Modern Browsers */
        url('webfont.ttf') format('truetype'), /* Safari, Android, iOS */
        url('webfont.svg#svgFontName') format('svg');
         /* Legacy iOS */
         }
        div {
                font-family: myFont, sans-serif;
         }
 </style>
```

CREATIVE

Chapter 4
Javascript

CREATIVE

Introduction to JavaScript

- Basically, Java is a full programming language developed by Sun Microsystems with formal structure.
- But, JavaScript is a scripting language developed by Netscape that is used to modify web pages.
- The JavaScript must be written in the HTML document between container tag <SCRIPT>….</SCRIPT>
- Special Attribute For <script> tag is, type="text/javascript".

Use to JavaScript

- JavaScript provides programming tool to HTML designer.
- JavaScript can put dynamic text in HTML page.
- JavaScript can react to events.
- JavaScript can read and write HTML elements.
- JavaScript can be used to validate data.
- JavaScript can be used to detect the visitor's browse.
- JavaScript can be used to create cookies.

Features of JavaScript

- **JavaScript is case sensitive**
  - JavaScript is case sensitive so the functions which have lowercase should be written like that.
  - If we want to use write() function of JavaScript then we cannot use it as Write().
  - In JavaScript variables are also case sensitive so variable "a" is not same as "A".
- **The Symbols must be closed if opened**
  - In JavaScript, symbols like ({" ' , Such symbols if used then it must closed like, ' " }).
- **White space is ignored**
  - White space are space, tab and new line.
  - These all kind of white spaces are ignored by the JavaScript.
- **Comments**
  - We can put comments in JavaScript
    - Single Line "//" slash is used.
    - For multi line comment /* */

CREATIVE

- **Ending statements with semicolon**
  - In JavaScript the semicolons are optional.
  - But if we want to put new statement in same line we can use **Semicolon.**
  - **Example of Semicolon..**
    - document.write("Hello")**;** document.write("Test");

- **Script Tag in HEAD section.**
  - The script tag is used in head section.
  - Mostly User Defined Function will paste in head section
  - Syntax:

```
<head>
    <script type="text/javascript">
            ……
    </script>
</head>
```

- **Script Tag in BODY section.**
  - To display something in the document page we can use Script Tag in Body Section.
  - Syntax:

```
<body>
    <script type="text/javascript">
            ……
    </script>
</body>
```

- **Insert Special Characters**
  - We can insert special characters like " ' \ by giving backslash (****) in document.
    - **For Example,** document.write("Hello! \'Friends\'")
  - Output :

    Hello 'Friends'

Variables

- Human mind has memory cells inside the brain to remembering various values.
- Similar to human brain computer has also memory Cells.
  - Each memory cell in the computer has unique address.
  - It is not possible to remember the address of each memory cells.
  - That's why cell is introduced with a name which is known as **"Variable".**

CREATIVE

Variable Declaration

- To declare a variable in JavaScript we can use following syntax.

  Syntax : var <variable name> = value;

  Example : var a=10;

  - Here, the **var** is a keyword for defining the variable.
  - But this **var** keyword is not compulsory.
  - A variable can defined without using **var**. a=10;

Rules For Variable Naming

- The first letter of variable-name should be alphabet. Like, s1
- The variable-name can consist of digits or alphabets.
- There should be no space between the variable-name.
- No special symbols other than underscore(_) used.
- Variable-name cannot be a keyword.
- The max-length of variable-name is 31 characters.

Life Time of Variable

- **Local Variable**
  - When the variable is declared inside of the function, the lifetime of that variable is for that function only.
  - It cannot be accessed outside the function and so such variable is know as local variable.

- **Global variable**
  - When the variable is declared outside of the function, the lifetime of the variable is for all the functions.
  - We can use that variable for each function in the same page. It is called as global variable.

JavaScript Operator & Operand

- **What is Operator?**
  - The symbols which are used to perform mathematical or logical operation are known as operator.
  - Like,
    - +, -, *, /, &&, ||, !=, <, >, etc.
- **What is Operand?**
  - Operand means the variable or value which is used before and after the operator.
  - Like,
    - A+B (Here, **A** and **B** are operand and **+** is operator)

CREATIVE

JavaScript Operator

- In JavaScript there are different types of operators available which are given below.
    1. **Arithmetic Operator**
    2. **Relational Operator**
    3. **Logical Operator**
    4. **Assignment Operator**
    5. **String Operator**
- **Arithmetic Operator:**
    - Arithmetic operators are used when any mathematical operation is to be done.

Sign	Introduction
+	Used for addition
-	Used for subtraction
*	Used for multiplication
/	Used for division
%	Used for finding the remainder. It is known as modulus operator.

- **Logical Operator:**
    - When more than two conditions are to be checked at the same time, we can use logical.

Sign	Introduction
&&	And operator. Returns true if all the condition are true
||	Or operator. Returns true if any of the conditions is true.
!	Not operator. It reverses the condition. That means if the result of the condition is true then it converts into false.

- **Relational Operator:**
    - When variables are related to some value at that the relational operator is used.

Sign	Introduction
==	This operator is used to compare the two variables or the values to check the equality. It checks only the value and not the data type.
===	This operator is used for comparing the equality of the variables or the values to check the equality. It will check value and data types also.
!=	This operator is for not equality. It will return true if the values are not equal regardless the data type.
!==	This operator is also for not equality. It will compare two values and data types too.
<	Used for less than
>	Used for greater than
>=	Used for greater than equal to
<=	Used for less than equal to

CREATIVE

- **Assignment Operator:**
  - There are different assignment operators as given.

Sign	Introduction
=	Assigning the value a=5
*=	a*=10 is similar to a=a*10
/=	a/=10 is similar to a=a/10
+=	a+=10 is similar to a=a+10
-=	a-=10 is similar to a=a-10
%=	a%=10 is similar to a=a%10

- **String Operator:**
  - The operator + is used for the concatenation of the two strings and so + is also known as String Operator.
  - **Syntax** : document.write("Text"+varList);
  - **Purpose** : To writing output to a page.
  - **Example** : document.write("Monarch");
    document.write(10+10);
  - Note : We can put required HTML tags inside inverted commas.

Conditional Statements

- While we working with JavaScript we can use such kind of control structure to get particular output with branching.

1. **If (Statements) :**
   - To check the given condition. If condition is TURE then it execute given statements.
   - **Syntax** :
```
if(condition)
{
    Statements when condition is true;
}
```
   - **Purpose**: It will process only if the condition is true otherwise nothing.
   - **Example** :
```
<script type="text/javascript">
    var a=5;
    if(a==1)
    {
        document.write("It is 1");
    }
</script>
```
   - **Design a web page using JavaScript display as given. (Simple if…)**

****

A=10

B=20

B is Greater

****

CREATIVE

- **If…Else :**

Syntax :

```
if(condition)
{
        Statements when condition is true;
}
else
{
        Statements when condition is false;
}
```

Purpose :

- It will process first section if the condition is true otherwise it executes the second section if the condition is false.

Example :

```
<script type="text/javascript">
        var age=19
        document.write("<br>Age ="+age);
        if(age>18){
                document.write("<br>You can VOTE");
        }
        else {
                document.write("<br>You cannot VOTE");
        }
</script>
```

- **If…Else if (Ladder If) :**

Syntax :

```
if(condition)
        {   Statements when condition is true;  }
  else if(condition)
        {   Statements when condition is true;  }
  else
        { Statement when all condition is false;}
```

Purpose :

- It will process first section if first condition is true, or it will process second section if second condition is true, otherwise it executes else section if all the condition is false.

CREATIVE

Example :

```
<script type="text/javascript">
        var no=25
        document.write("No = "+no+"<br>");
        if(no>0){
            document.write("Number is Positive");
        }else if(no<0){
            document.write("Number is Negative");
        }else{
            document.write("Number is Zero");
          }
</script>
```

- **Nested If :**

Syntax :

```
if(condition)
{
    if(condition)
    {
        When condition is true;
    }
}
else
{
        Statement when all condition is false;
}
```

Purpose :

- Nested means one **if** inside the other **if**. If first condition will be true then it will check the other **if**, both condition will be true then it will execute given statement.

Example :

```
<script type="text/javascript">
    var age=23;
    var gen="M";
    if(age>21)
    {
        if(gen=='M'){
            document.write("You can Marry");
        }
        else{
            document.write("Sorry! you can not Marry");
        }
    }
</script>
```

CREATIVE

2. Switch Case :

- The switch statement causes a particular group of statements to be chosen from several available groups.
- The selection is tasted upon the current value of an expression that is included within the switch statement.
- Switch case is a built in multi-way decision statement.
- It tests value of given variable (or Text Expression) against a list of case values & block of statements associated with it is executed when a match is found.
- The general form of the switch statement is:
- switch(expression)
- Here expression result is an integer value or char type.
- **Syntax :**

```
switch(TxtExpression)
{
    Case TxtExp1:
        Statements;
    Case TxtExp2:
        Statements;
    Case TxtExpN:
        Statements;
    Default:
        Statements;
}
```

- **Purpose** : Switch case statement will provide us easy features of working it will provide selection based switching in group of statements. After completion of group execution we must put a break; statement to stop execution in each group. We can use either integer number or character for switching from list of group.
- **Example** :

```
<script type="text/JavaScript">
switch(“B")
{
    case “R":
        document.write("Your Choice Red")
        break;
```


CREATIVE

```
        case “G":
                document.write("Your Choice Green")
                break;
        case “B":
                document.write("Your Choice Blue")
                break;
        default:
                document.write("Your Choice is another")
}
</script>
```

LOOPS

- Loops are used when we want to execute a part for a block of statements for specified number of times or depending on some specified condition.
- The main thing about any of the loop is that it executed for the number of times based on the logical expression (condition) that is specified to the particular loop.
- When loop checks for the condition to execute at that time there are two types of loops available.
    - Entry Control Loop
    - Exit Control Loop

❖ **Entry Control Loop :**

- The loop which is checked before executing the statements of loop is called as Entry Controlled loop.
- This loop is first checked and if it is true then only the statements under that loop are executed and if it is not true then it transfers the control at to the end of the loop. They are as follow.
    1. **For( ) loop**
    2. **While( ) loop**

1. For…. loop:

Syntax :

```
for(initialization; condition; increment or decrement)
{
        statements;
}
```

Purpose : To provide a looping while condition will be true we can use for loop.

Initialization : Initialize the value of variable.

Test Condition : Test condition at here.

Increment/decrement: To increase or decrease the value.

CREATIVE

Example :

```
<script type="text/javascript">
    var a=1;
    for(a=1;a<=10;a++)
    {
        document.write("<br>"+a);
    }
</script>
```

OUTPUT :

1 2 3 4 5 6 7 8 9 10

2. **While…. loop:**
   - **Syntax :**
   ```
   while(test condition)
   {
       statements;
   }
   ```
   - **Purpose :**
     1. To provide a looping till condition will be true we can use while loop.
     2. While loop will check the condition first.
     3. If the condition will be true then it will execute the statements given in the loop, after once complete the loop, it will again and again check the condition if condition will false then it will auto exit for the loop.
   - **Example :**

```
<script type="text/javascript">
    var a=1;
    while(a<=10)
    {
        document.write("<br>"+a);
        a++;
    }
</script>
```

❖ **Exit Control Loop :**

- The loop which is checked after executing the statements of loop is called as exit controlled loop.
- This loop is not checked in the began of loop but checks after execution of statement.
- While we want to execute any loop at least once at that time we can use this looping statement. They are as follow.

CREATIVE

- **Do…while( )**

Syntax :

```
do
{
        statements;
}while(test condition);
```

Purpose :

- To provide a looping till condition will be true we can use do…while loop.
- Do…while loop will check the condition after execution of statements provided in to loop.
- If the condition will be true then it will again execute the statements given in the loop, if condition will false then it will auto exit for the loop.

Example :

```
<script type="text/javascript">
        var a=1;
        do
        {
                document.write("<br>"+a);
                a++;
        } while(a<=10)
</script>
```

✓ Break Statements

Syntax : break;
Purpose : The break command will break the loop.
Example :

```
<script type="text/javascript">
        var i=0;
        for(i=0;i<=10;i++)
        {
                if(i==3){
                        break;
                        document.write("<br>Num. is" +i);
                }
        }
</script>
```

CREATIVE

✓ Continue Statements

Syntax : continue;
Purpose : The continue command will break the current loop and continue with the next loop.
Example :

```
<script type="text/javascript">
        for(i=0;i<=10;i++)
        {
            if(i==3){
                continue;
                document.write("Num. is" +i);
            }
        }
</script>
```

Dialog Boxes

- ✓ JavaScript provides us the ability to pickup users input or display small amount of text to the user by using dialog boxes.
- ✓ There are different types of dialog boxes as given.
    1. **Alert Dialog Boxes**
    2. **Conform Boxes**
    3. **Prompt Boxes**

1. Alert Dialog Boxes

Syntax : alert("message");

Purpose : To display some textual information on web browser window we can use alert dialog box.

Example :

```
<script type="text/javascript">
        var name="Creative Design";
        alert("Welcome to "+name);
</script>
```

2. Confirm Dialog Boxes

Syntax : confirm("message");

Purpose : A confirm box is used if we want the user to verify or accept something. A confirm box displays the predefined message and Ok & Cancel button.

Example :
```
<script type="text/javascript">
        if(confirm("Are you Coming?"))
        {
                document.write("Welcome");
        }
        else
        {
                document.write("No Thanks");
        }
</script>
```

3. **Prompt Dialog Boxes**

Syntax : prompt([“Text”],[default value]);

Purpose : To take input a user value we can use this prompt box.

Example :
```
<script type=“text/javascript”>
        var a,b;
        a=prompt("Enter Value for A :",0);
        b=prompt("Enter Value for B :",0);
        document.write("Value of A + B =",parseInt(a)+parseInt(b));
</script>
```

JavaScript Arrays

What is an Array?

- An array is a special variable, which can hold more than one value at a time.

How to create an Array?

Syntax : var array_name = [item1, item2, ...]; **OR** var array_name = new Array(“item1”, “item2”, ...);

Example : var cars = ["Audi", "Volvo", "BMW"]; **OR** var cars = new Array("Audi", "Volvo", "BMW");

How to Access the element of an Array?

- You access an array element by referring to the **index number**.
- Example:

```
var cars = new Array("Audi", "Volvo", "BMW");
document.write(cars[0]);
document.write(cars[1]);
document.write(cars[2]);
        OR
document.write(cars);
```

JavaScript Functions

- There are two types of functions are in JavaScript.
  - Library Functions
  - User Defined Functions

  - **Library Functions**
    - The functions whose definitions are inbuilt in JavaScript is known as JavaScript library functions.
  - **User Defined Functions**
    - Those process which are required again and again those requirement codes in specified format is known as User Defined Functions.
    - Various Kind of UDF.
      - No argument and No Return value.
      - With argument and No Return value.
      - With argument and With Return value.
    - **Syntax :**

```
Function function_name([var list])
{
        Statements…
        [return value]
}
```

1. **No argument and No Return value**
   - In this kind of UDF, there are no argument will pass to the function and no Return value to the collar function.
   - **Example** :

```
<html>
<head>
        <title></title>
        <script type="text/javascript">
                function star()
                {
                        var a;
                        document.write("<br>");
                        for(a=1;a<=70;a++){
                                document.write("*");
                        }
                        document.write("<br>");
                }
        </script>
</head>
```

CREATIVE

```
<body>
    <script type="text/javascript">
        star();
        document.write("This is Testing");
        star();
    </script>
</body>
</html>
```

2. With argument and No Return value

- In this kind of UDF, function takes some argument and does not have any return value.
- **Example** :

```
<html>
<head>
    <title></title>
    <script type="text/javascript">
        function star(x)
        {
            var b;
            document.write("<br>");
            for(b=1;b<=x;b++){
                document.write(b);
            }
        }
    </script>
</head>
<body>
    <script type="text/javascript">
        var a;
        for(a=1;a<=5;a++){
            star(a);
        }
    </script>
</body>
</html>
```

- **Output :**

```
1
12
123
1234
12345
```

CREATIVE

3. **With argument and with Return value**
   - In this kind of UDF, this type of functions takes some argument and does return the value.
   - When the value is returned by the function it is required that when the function is called the variable should be there to accept the returned value from the function.
   - **Example** :

```
<html>
<head>
        <title></title>
        <script type="text/javascript">
                function add(a, b)
                {
                        var c;
                        c=a+b;
                        return c;
                }
        </script>
</head>
<body>
        <script type="text/javascript">
                var value;
                value=add(5,6);
                document.write(value);
        </script>
</body>
</html>
```

   - **Output :**

     11

CREATIVE

JavaScript String Function

- **big()**
  - To display text object with BIG look
  - **Syntax :** stringObject.big();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city.big());
    document.write("<br>Welcome to Our College");
</script>
```

- **small()**
  - To display text object with Small look
  - **Syntax :** stringObject.small();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city.small());
    document.write("<br>Welcome to Our College");
</script>
```

- **bold()**
  - To display text object with Bold look
  - **Syntax :** stringObject.bold();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city.bold());
    document.write("<br>Welcome to Our College");
</script>
```

- **italics()**
  - To display text object with italic look
  - **Syntax :** stringObject.italics();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. italics());
    document.write("<br>Welcome to Our College");
</script>
```

CREATIVE

- **strike()**
  - To display text object with strikeline look
  - **Syntax :** stringObject.strike();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. strike());
    document.write("<br>Welcome to Our College");
</script>
```

- **fontcolor()**
  - To display text object with color full font.
  - Here, value of color can be specified Hexa code used in HTML or the name of color.
  - **Syntax :** stringObject.fontcolor();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. fontcolor('red'));
    document.write("<br>Welcome to Our College");
</script>
```

- **fontsize()**
  - To display text object with defined size.
  - **Syntax :** stringObject.fontsize(*integer value 1 to 7*);
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. fontsize(6));
    document.write("<br>Welcome to Our College");
</script>
```

- **link()**
  - To display link for given path/URL.
  - **Syntax :** stringObject.link(*string URL*);
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. link("image/demo.jpg"));
    document.write("<br>Welcome to Our College");
</script>
```

- **length()**
  - This function will count number of character in the string and integer value.
  - **Syntax :** stringObject.length();
  - **Example :**

```
<script type="text/javascript">
    var nm="Good Morning "
    size=nm.length;
    document.write("<br>Size is ",size);
```


CREATIVE

```
</script>
```

❖ **charAt()**

- To specifies the character at the given position.
- The counting of the position start from 0.
- **Syntax :** stringObject.charAt(integer position);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write("<br> ",name.charAt(5));
</script>
```

- **Output :**

e

❖ **concat()**

- This function is used to join two or more than one string with the original string.
- **Syntax :** stringObject.concat(string1, string2,…);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write("<br> ",name.concat(", Always Creative"));
</script>
```

- **Output :**

Be Creative, Always Creative

- **Exercise:**
  - Store Surname, Name and Father name in variable FullName using Concat function…

❖ **indexOf()**

- This function returns the position of first occurrence of the specified string value in the given string.
- If it not found string in same case then it will return -1 else it will display the position of that string.
- **Syntax :** stringObject.indexOf(string, integer from index);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write(name.indexOf("e"));
</script>
```

- **Output :**

1

❖ **lastIndexOf()**

- This function returns the position of last occurrence of the specified string value in the given string.
- If it not found string in same case then it will return -1 else it will display the position of that string.
- **Syntax :** stringObject.lastindexOf(string, integer from index);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write(name.lastIndexOf("e"));
</script>
```

- **Output :**

10

CREATIVE

- **match()**
  - This function is used to search the given string from string object. If matched then it will display the same name or it will display the 'null' as return.
  - **Syntax :** stringObject.match(string search value);
  - **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.match("Creative"));
    if(mystr != null){
            document.write("Your String is Match");
    }
    else{
            document.write("Your String is Not Match");
    }
</script>
```

  - **Output :**

```
Your String is Match
```

- **replace()**
  - This function is used to search the given string from string object. If matched then it will display the same name or it will display the 'null' as return.
  - **Syntax :** stringObject.replace(string search value);
  - **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.replace("Be","Always"));
    document.write(mystr);
</script>
```

  - **Output :**

```
Always Creative
```

- **search()**
  - This function is used to search value from given string.
  - **Syntax :** stringObject.search(string search value);
  - **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.search("Creative"));
    if(mystr >= 0){
            document.write("Text Found");
    }else{
            document.write("Text Not Found");
    }
</script>
```

  - **Output :**

```
Text Found
```

- **slice()**
  - This function slice string form the given integer starting point to ending point.
  - Here, starting point is compulsory but ending point is default end of string.
  - **Syntax :** stringObject.slice(int start, int end);
    - int start mean AFTER THAT number to

- int end mean TILL THAT number.

- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write(name.slice(0,2));
    document.write("<br>"+name.slice(3,11));
</script>
```

- **Output :**

```
Be
Creative
```

❖ **substr()**

- This function substr will return string form the given start position and it will stop on the length or end of string.
- The index of first character is 0.
- **Syntax :** stringObject.substr(start, length);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.substr(3,6));
    document.write(mystr);
</script>
```

- **Output :**

```
Creati
```

❖ **substring()**

- This function substring will display the text from starting index to ending index.
- The index of first character is 0.
- **Syntax :** stringObject.substring(start, end length);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.substring(3,8));
    document.write(mystr);
</script>
```

- **Output :**

```
Creat
```

❖ **toLowerCase()**

- This function will convert given string into lowercase.
- **Syntax :** stringObject.toLowerCase();
- **Example :**

```
<script type="text/javascript">
    var name="Be CreAtivE";
    document.write (name.toLowerCase());
</script>
```

- **Output :**

```
be creative
```

❖ **toUpperCase()**

- This function will convert given string into lowercase.
- **Syntax :** stringObject.toUpperCase();
- **Example :**

```
<script type="text/javascript">
    var name="Be CreAtivE";
    document.write (name.toUpperCase());
</script>
```

- **Output :**

  BE CREATIVE

JavaScript Math Function

- **abs()**
  - To find given number as absolute value. It will convert negative value into positive..
  - **Syntax :** Math.abs();
  - **Example :**

```
<script type="text/javascript">
    for(var a=-3;a<=3;a++)
    {
            document.write("<br>");
            document.write(Math.abs(a));
    }
</script>
```

  - **Output :**

    3
    2
    1
    0
    1
    2
    3

- **ceil()**
  - To display nearest round up integer value.
  - **Syntax :** Math.ceil(float value);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.ceil(3.5));
</script>
```

  - **Output :**

    4

- **floor()**
  - To display nearest round down integer value.
  - **Syntax :** Math.ceil(float value);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.floor(3.5));
</script>
```

  - **Output :**

    3

CREATIVE

- **pow()**
  - To find the power of the number for the given number and exponential.
  - **Syntax :** Math.pow(number, Exponential);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.pow(3,2));
</script>
```

  - **Output :**

```
9
```

- **random()**
  - This function will return random integer value each time the page is refreshed.
  - Here, the return type is integer.
  - **Syntax :** Math.random();
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.random());
</script>
```

  - **Output :**

```
0.8419394853003848
```

- **max()**
  - This function will return maximum value from given integer values.
  - **Syntax :** Math.max(int val1, int val2, int val3, …….);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.max(25,50,30,5,20));
</script>
```

  - **Output :**

```
50
```

- **min()**
  - This function will return minimum value from given integer values.
  - **Syntax :** Math.min(int val1, int val2, int val3, …….);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.min(25,50,30,5,20));
</script>
```

  - **Output :**

```
5
```

- **round()**
  - This function round the floating value as mathematical rules.
  - **Syntax :** Math.round(Float Value);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.round(25.579521));
</script>
```

  - **Output :**

```
26
```

CREATIVE

JavaScript Date Function

- **date()**
  - This function returns the current date and time of system.
  - **Syntax :** Date();
  - **Example :**

```
<script type="text/javascript">
    document.write(Date());
</script>
```

  - **Output :**

    Sat Jan 16 2021 17:16:57 GMT+0530 (India Standard Time)

- **getDay()**
  - This function get only Day from the date object. In the range of 0 to 6.
  - Where, 0 = Sunday and 6= Saturday
  - **Syntax :** dateObject.getDay();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Day:"+cdate.getDay());
</script>
```

  - **Output :**

    Day : 23

- **getMonth()**
  - This function get only month from the date object. In the range of 0 to 11.
  - Where, 0 = Januray and 11= December
  - **Syntax :** dateObject.getMonth();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Month:"+cdate.getMonth());
</script>
```

  - **Output :**

    Month : 1

- **getFullYear()**
  - The getFullYear() method returns the year of a date as a four digit number:
  - **Syntax :** dateObject.getFullYear();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Year : "+cdate.getFullYear());
</script>
```

  - **Output :**

    Year : 2021

CREATIVE

❖ **getHours()**

- The getHours() method returns the hours of a date as a number (0-23):
- **Syntax :** dateObject.getHours();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Hours : "+cdate.getHours());
</script>
```

- **Output :**

Hours : 15

❖ **getMinutes()**

- The getMinutes() method returns the minutes of a date as a number (0-59):
- **Syntax :** dateObject.getMinutes();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Minutes : "+cdate.getMinutes());
</script>
```

- **Output :**

Minutes : 15

❖ **getSeconds()**

- The getSeconds() method returns the Seconds of a date as a number (0-59):
- **Syntax :** dateObject.getSeconds();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Seconds : "+cdate.getSeconds());
</script>
```

- **Output :**

Seconds : 50

❖ **getMilliseconds()**

- The getMilliseconds() method returns the Milli Seconds of a date as a number (0-999):
- **Syntax :** dateObject.getMilliseconds();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Milli Seconds : "+cdate.getMilliseconds());
</script>
```

- **Output :**

Milli Seconds : 600

❖ **getTime()**

- The getTime() method returns the number of milliseconds since January 1, 1970:
- **Syntax :** dateObject.getTime();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Milliseconds : "+cdate.getTime());
```

```
</script>
```

- **Output :** **Milliseconds : 1611395321245**

❖ setDate()

- The setDate() method sets the day of a date object (1-31):
- **Syntax :** dateObject.setDate();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setDate(6)
    document.write("Set Date:"+cdate);
</script>
```

- **Output :**

Set Date : Wed Jan 06 2021 15:42:02 GMT+0530 (India Standard Time)

❖ setMonth()

- The setMonth() method sets the month of a date object (0-11):
- Where, 0 = Januray and 11= December
- **Syntax :** dateObject.setMonth();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setMonth(3);
    document.write("Set Month:"+cdate);
</script>
```

- **Output :**

Set Month : Fri Apr 23 2021 15:41:34 GMT+0530 (India Standard Time)

❖ setFullYear()

- The setFullYear() method sets the year of a date object. In this example to 2020:
- **Syntax :** dateObject.setFullYear();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setFullYear(2025);
    document.write("Set Year : "+cdate);
</script>
```

- **Output :**

Set Year : Thu Jan 23 2025 15:44:19 GMT+0530 (India Standard Time)

❖ setHours()

- The setHours() method sets the hours of a date object (0-23):
- **Syntax :** dateObject.setHours();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setHours(21);
    document.write("Set Hours : "+cdate);
</script>
```

- **Output :**

Set Hours : Sat Jan 23 2021 21:58:43 GMT+0530 (India Standard Time)

CREATIVE

❖ **setMinutes()**

- The setMinutes() method sets the minutes of a date object (0-59):
- **Syntax :** dateObject.setMinutes();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setMinutes(35);
    document.write("Set Minutes : "+cdate);
</script>
```

- **Output :**

Set Minutes : Sat Jan 23 2021 16:35:48 GMT+0530 (India Standard Time)

❖ **setSeconds()**

- The setSeconds() method sets the seconds of a date object (0-59):
- **Syntax :** dateObject.setSeconds();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setMinutes(40);
    document.write("Set Seconds : "+cdate);
</script>
```

- **Output :**

Set Month : Sat Jan 23 2021 16:11:35 GMT+0530 (India Standard Time)

JavaScript Array Function

❖ **concat()**

- This function is used to join two or more then array in a single array.
- **Syntax :** arrayObject.concat(array1, array2,…);
- **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    var vege=new Array("Carrot","Radish","Tomoto");
    fruit=fruit.concat(vege);
    document.write("Mix  : ",fruit);
</script>
```

- **Output :**

Mix : Banana,Apple,Mango,Carrot,Radish,Tomoto

❖ **join()**

- This function is used to Display all the elements of any array with specified separator.
- **Syntax :** arrayObject.join(string separator);
- **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    document.write("<br>Fruits  : ",fruit.join(" * * "));
</script>
```

- **Output :**

Fruits : Banana * * Apple * * Mango

CREATIVE

- **pop()**
  - This function is used to delete the item from the array. Deletion will starting from the end of the existing array element.
  - **Syntax :** arrayObject.pop( );
  - **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    fruit.pop();
    document.write("<br>Fruits  : ",fruit);
</script>
```

  - **Output :**

    Fruits : Banana,Apple

- **push()**
  - This function is used to add the item into array.
  - **Syntax :** arrayObject.push( );
  - **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    fruit.push("Orange");
    document.write("<br>Fruits  : ",fruit);
</script>
```

  - **Output :**

    Fruits : Banana,Apple,Mango,Orange

- **reverse()**
  - This function is used to reverses the order of the elements in an array.
  - **Syntax :** arrayObject.reverse( );
  - **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    var revarray=fruit.reverse();
    document.write("<br>Fruits  : ",revarray);
</script>
```

  - **Output :**

    Fruits : Mango,Apple,Banana

- **shift()**
  - This function is used to remove first element of the array.
  - **Syntax :** arrayObject.shift( );
  - **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    fruit.shift();
    document.write("<br>Fruits  : ",fruit);
</script>
```

  - **Output :**

    Fruits : Apple,Mango

CREATIVE

❖ **sort()**

- The sort() method sorts the items of an array.
- **Syntax :** arrayObject.sort( );
- **Example :**

```
<script type="text/javascript">
    var data=new Array("7","4","2","1","5");
    document.write("<br>Number  : ",data);
    data.sort();
    document.write("<br>Sorted Number  : ",data);
</script>
```

- **Output :**

Number : 7,4,2,1,5
Sorted Number : 1,2,4,5,7

❖ **length()**

- This function is used to count number of element present in the array.
- **Syntax :** arrayObject.length( );
- **Example :**

```
<script type="text/javascript">
    var data=new Array("7","4","2","1","5");
    var elements=data.length;
    document.write("Number of element :",elements);
</script>
```

- **Output :**

Number of element : 5

JavaScript Events

❖ **onclick()**

- This event occurs when some click action is performed on any of the html element.
- **Example :**

```
<!DOCTYPE html>
<html>
<body>
    <h1>The onclick Event</h1>
    <p>The onclick event is used to trigger a function when an element is clicked
on.</p>
    <button onclick="myFunction()">Click me</button>
    <p id="demo"></p>

    <script>
        function myFunction() {
            document.getElementById("demo").innerHTML = "Hello World";
        }
    </script>
</body>
</html>
```

❖ **ondblclick()**

- The ondblclick event occurs when the user double-clicks on an element.
- **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<h1>The onclick Event</h1>
	<p>The onclick event is used to trigger a function when an element is clicked
on.</p>
	<button ondblclick="myFunction()">Click me</button>
	<p id="demo"></p>

	<script>
		function myFunction() {
			document.getElementById("demo").innerHTML = "Hello World";
		}
	</script>
</body>
</html>
```

❖ **onmouseover ()**

- The onmouseover event occurs when the mouse pointer is moved onto an element, or onto one of its children.
- **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<h1 onmouseover="myFunction()" >The onmouseover Event</h1>
	<script>
		function myFunction() {
			alert("Hello World");
		}
	</script>
</body>
</html>
```

❖ **onmouseout ()**

- The onmouseout event occurs when the mouse pointer is moved out of an element, or out of one of its children.
- **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<h1 onmouseout="myFunction()" >The onmouseout Event</h1>
	<script>
		function myFunction() {
			alert("Hello World");
		}
	</script>
</body>
</html>
```

❖ **onkeypress ()**

- The onkeypress event occurs when the user presses a key (on the keyboard).
- **Example :**

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


CREATIVE

```
<body>
    <input type="text" onkeypress="myFunction()" id="txtbox">

    <script>
        function myFunction() {
            document.getElementById("txtbox").value.toUppercase();
        }
    </script>
</body>
</html>
```

- **onkeyup()**
  - The onkeyup event occurs when the user releases a key (on the keyboard).
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
    <input type="text" onkeyup="myFunction()" id="txtbox">

    <script>
        function myFunction() {
            var x = document.getElementById("txtbox");
            x.value = x.value.toUpperCase();
        }
    </script>
</body>
</html>
```

- **onfocus ()**
  - The onfocus event occurs when an element gets focus.
  - The onfocus event is most often used with <input>, <select>, and <a>.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
    <input type="text" onfocus="myFunction()" id="txtbox">

    <script>
        function myFunction() {
            document.getElementById("txtbox").style.background = "yellow";
        }
    </script>
</body>
</html>
```

- **onblur()**
  - The onblur event occurs when an object loses focus.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
    <input type="text" onblur="myFunction()" id="txtbox">

    <script>
        function myFunction() {
            document.getElementById("txtbox").style.background = "red";
```

CREATIVE

```
        }
    </script>
</body>
</html>
```

- **onload()**
  - The onload event occurs when an object has been loaded.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body onload=”myFunction()”>
    <h1>Hello World!</h1>
    <script>
        function myFunction() {
            alert("Page is loaded");
        }
    </script>
</body>
</html>
```

- **onchange()**
  - The onchange event occurs when the value of an element has been changed.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body >
    <p>Select a new car from the list.</p>
    <select id="mySelect" onchange="myFunction()">
        <option value="Audi">Audi</option>
        <option value="BMW">BMW</option>
        <option value="Mercedes">Mercedes</option>
        <option value="Volvo">Volvo</option>
    </select>
    <p id="demo"></p>
    <script>
        function myFunction() {
            var x = document.getElementById("mySelect").value;
            document.getElementById("demo").innerHTML = "You selected: " +
        x;
        }
    </script>
</body>
</html>
```

- **onsubmit()**
  - The onsubmit event occurs when a form is submitted.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body >
    <form onsubmit="myFunction()">
        Enter name: <input type="text" name="fname">
        <input type="submit" value="Submit">
    </form>

    <script>
        function myFunction() {
            alert("The form was submitted");
```


CREATIVE

```
        }
    </script>
</body>
</html>
```

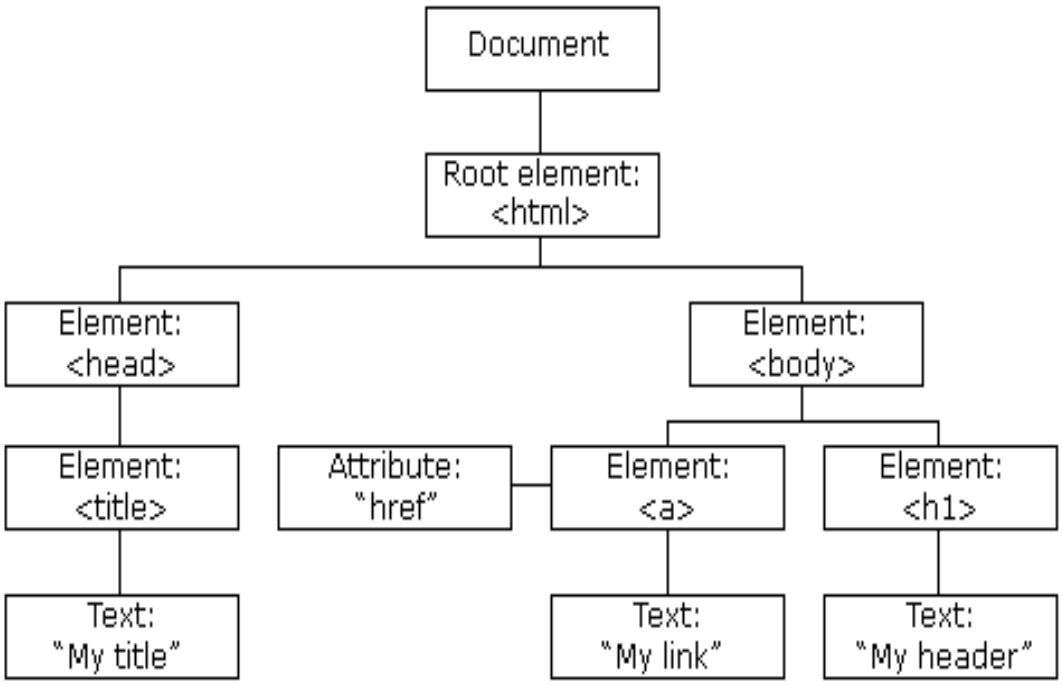
- **onreset()**
  - The onreset event occurs when a form is reset.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body >
    <form onreset="myFunction()">
        Enter name: <input type="text" name="fname">
        <input type="reset" value="Reset">
    </form>

    <script>
    function myFunction() {
        alert("The form was Reset");
    }
    </script>
</body>
</html>
```

DOM (Document Object Model) & History Object

- When a web page is loaded, the browser creates a Document Object Model of the page.
- The HTML DOM model is constructed as a tree of Objects.

The HTML DOM Tree of Objects



- ✓ **With the object model, JavaScript gets all the power it needs to create dynamic HTML:**
  - JavaScript can change all the HTML elements in the page
  - JavaScript can change all the HTML attributes in the page
  - JavaScript can change all the CSS styles in the page
  - JavaScript can add new HTML elements and attributes
  - JavaScript can create new HTML events in the page

History Object

- ✓ The history object contains the URLs visited by the user (within a browser window).
- ✓ The history object is part of the window object and is accessed through the window.history property.
- ✓ **Property of JavaScript history object :**
  - **length:** It returns the length of the history URLs visited by user in that session.
  - **back() :** Loads the previous URL in the history list.
  - **forword()** : Loads the next URL in the history list.
  - **go() :** Loads the next URL in the history list.

What is Validation ?

- ✓ Verification that something is correct or conforms to a certain standard.
- ✓ In data collection or data entry, it is the process of ensuring that the data that are entered fall within the accepted boundaries of the application collecting the data.
- ✓ For example, if a program is collecting last names to be entered in a database, the program validates that only letters are entered and not numbers.

JavaScript Form Validation

- ✓ HTML form validation can be done by JavaScript.
- ✓ If a form field (fname) is empty, this function alerts a message, and returns false, to prevent the form from being submitted:
- ✓ Example :

```
<form onsubmit="validation()">
    <table>
        <tr>
            <td>Name :</td>
            <td>
                <input type="text" name="" id="name">
            </td>
        </tr>
        <tr>
            <td>
```

```
                    <input type="submit" name="">
                </td>
            </tr>
        </table>
</form>
<script >
        function validation(){
            var name = document.getElementById('name').value;
            if(name == "")
            {
                alert('Plz Enter Your Name');
            }
        }
</script>
```

JavaScript Email Validation

- ✓ Validating email is a very important point while validating an HTML form.
- ✓ An email is a string (a subset of ASCII characters) separated into two parts by @ symbol. a "personal_info" and a domain, that is personal_info@domain.
- ✓ The length of the personal_info part may be up to 64 characters long and domain name may be up to 253 characters.
- ✓ **Valid Email ID :** info@cdmi.in, example@gmail.com
- ✓ **Invalid Email ID :** info.cdmi.in [@ is not present]
- ✓ Example :

```
<form onsubmit="validation()">
        <table>
            <tr>
                <td>Name :</td>
                <td>
                    <input type="text" name="" id="email">
                </td>
            </tr>
            <tr>
                <td>
                    <input type="submit" name="">
                </td>
            </tr>
        </table>
</form>
<script >
        function validation(){
            var email = document.getElementById('name').value;
            var emailRegex = /^\w+[A-Z0-9._%-]+@[A-Z0-9.-]+\.[A-Z]{2,4}\b$/i;

            if(emailRegex.test(email) == false)
            {
                alert('Plz Enter Valid Email ID.');
            }
        }
</script>
```

CREATIVE

Chapter 5
Bootstrap Framework

CREATIVE

❖ Introduction to Bootstrap :

Quick start

Looking to quickly add Bootstrap to your project? Use jsDelivr, provided for free by the folks at jsDelivr. Using a package manager or need to download the source files?

CSS

Copy-paste the stylesheet <link> into your <head> before all other stylesheets to loadour CSS.

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@ 4.0.0/dist/css/bootstrap.min.css">
```

JS

Many of our components require the use of JavaScript to function. Specifically, theyrequire jQuery, Popper.js, and our own JavaScript plugins. Place the following <script>s near the end of your pages, right before the closing </body> tag,to enable them. jQuery must come first, then Popper.js, and then our JavaScript plugins.

We use jQuery's slim build, but the full version is also supported.

```
<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js" ></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js" ></script>
```

Curious which components explicitly require jQuery, our JS, and Popper.js? Click theshow components link below. If you're at all unsure about the general page structure, keep reading for an example page template.

- **Show components requiring JavaScript**
  - Alerts for dismissing
  - Buttons for toggling states and checkbox/radio functionality
  - Carousel for all slide behaviors, controls, and indicators
  - Collapse for toggling visibility of content
  - Dropdowns for displaying and positioning (also requires Popper.js)
  - Modals for displaying, positioning, and scroll behavior
  - Navbar for extending our Collapse plugin to implement responsive behavior
  - Tooltips and popovers for displaying and positioning (also requires Popper.js)
  - Scrollspy for scroll behavior and navigation updates

CREATIVE

- Starter template

Be sure to have your pages set up with the latest design and development standards. That means using an HTML5 doctype and including a viewport meta tag for proper responsive behaviors. Put itall together and your pages should look like this:

```
<!doctype html>
<html lang="en">
 <head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css">

  <title>Hello, world!</title>
 </head>
 <body>
  <h1>Hello, world!</h1>

  <!-- Optional JavaScript -->
  <!-- jQuery first, then Popper.js, then Bootstrap JS -->
  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/js/bootstrap.min.js"></script>
 </body>
</html>
```

- HTML5 doctype

CREATIVE

Bootstrap requires the use of the HTML5 doctype. Without it, you'll see some funkyincomplete styling, but including it shouldn't cause any considerable hiccups.

```
<!doctype html>
<html lang="en">
  ...
</html>
```

Containers

Containers are the most basic layout element in Bootstrap and are **required when using our default grid system**. Choose from a responsive, fixed-width container (meaning its `max-width` changes at each breakpoint) or fluid-width (meaning it's `100%` wide all the time).

While containers *can* be nested, most layouts do not require a nested container.

```
<div class="container">
  <!-- Content here -->
</div>
```

Copy

Use `.container-fluid` for a full width container, spanning the entire width of the viewport.

```
<div class="container-fluid">
  ...
</div>
```

Copy

Responsive breakpoints

Since Bootstrap is developed to be mobile first, we use a handful of media queries to create sensible breakpoints for our layouts and interfaces. These breakpoints are mostly based on minimum viewport widths and allow us to scale up elements as the viewport changes.

Bootstrap primarily uses the following media query ranges—or breakpoints—in our source Sass files for our layout, grid system, and components.

Copy

```
// Extra small devices (portrait phones, less than 576px)
// No media query since this is the default in Bootstrap

// Small devices (landscape phones, 576px and up)
@media (min-width: 576px) { ... }

// Medium devices (tablets, 768px and up)
@media (min-width: 768px) { ... }

// Large devices (desktops, 992px and up)
@media (min-width: 992px) { ... }

// Extra large devices (large desktops, 1200px and up)
@media (min-width: 1200px) { ... }
```

Since we write our source CSS in Sass, all our media queries are available via Sass mixins:

Copy

```
@include media-breakpoint-up(xs) { ... }
@include media-breakpoint-up(sm) { ... }
@include media-breakpoint-up(md) { ... }
@include media-breakpoint-up(lg) { ... }
@include media-breakpoint-up(xl) { ... }

// Example usage:
@include media-breakpoint-up(sm) {
  .some-class {
    display: block;
  }
}
```

We occasionally use media queries that go in the other direction (the given screen size *or smaller*):

Copy

```
// Extra small devices (portrait phones, less than 576px)
@media (max-width: 575.98px) { ... }

// Small devices (landscape phones, less than 768px)
@media (max-width: 767.98px) { ... }

// Medium devices (tablets, less than 992px)
@media (max-width: 991.98px) { ... }

// Large devices (desktops, less than 1200px)
@media (max-width: 1199.98px) { ... }

// Extra large devices (large desktops)
// No media query since the extra-large breakpoint has no upper bound on its width
```

Grid options

While Bootstrap uses ems or rems for defining most sizes, pxs are used for grid breakpoints and container widths. This is because the viewport width is in pixels and does not change with the font size.

See how aspects of the Bootstrap grid system work across multiple devices with a handy table.

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
Max container width	None (auto)	540px	720px	960px	1140px
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
# of columns	12				
Gutter width	30px (15px on each side of a column)				
Nestable	Yes				
Column ordering	Yes				

Equal-width

For example, here are two grid layouts that apply to every device and viewport, from xs to xl. Add any number of unit-less classes for each breakpoint you need and every column will be the same width.

1 of 2	2 of 2

1 of 3	2 of 3	3 of 3

Copy

```
<div class="container">
  <div class="row">
    <div class="col">
      1 of 2
    </div>
    <div class="col">
      2 of 2
    </div>
  </div>
  <div class="row">
    <div class="col">
      1 of 3
    </div>
    <div class="col">
      2 of 3
    </div>
    <div class="col">
      3 of 3
    </div>
  </div>
</div>
```

Responsive classes

Bootstrap's grid includes five tiers of predefined classes for building complex responsive layouts. Customize the size of your columns on extra small, small, medium, large, or extra large devices however you see fit.

All breakpoints

For grids that are the same from the smallest of devices to the largest, use the `.col` and `.col-*` classes. Specify a numbered class when you need a particularly sized column; otherwise, feel free to stick to `.col`.

col	col	col	col

col-8	col-4

Copy

```
<div class="row">
  <div class="col">col</div>
  <div class="col">col</div>
  <div class="col">col</div>
  <div class="col">col</div>
</div>
<div class="row">
  <div class="col-8">col-8</div>
  <div class="col-4">col-4</div>
</div>
```

Mix and match

Don't want your columns to simply stack in some grid tiers? Use a combination of different classes for each tier as needed. See the example below for a better idea of how it all works.

.col-12 .col-md-8	.col-6 .col-md-4

.col-6 .col-md-4	.col-6 .col-md-4	.col-6 .col-md-4

.col-6	.col-6

Copy

```
<!-- Stack the columns on mobile by making one full-width and the other half-width -->
<div class="row">
  <div class="col-12 col-md-8">.col-12 .col-md-8</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
</div>

<!-- Columns start at 50% wide on mobile and bump up to 33.3% wide on desktop -->
<div class="row">
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
</div>

<!-- Columns are always 50% wide, on mobile and desktop -->
<div class="row">
  <div class="col-6">.col-6</div>
  <div class="col-6">.col-6</div>
</div>
```

Reordering

Order classes

Use `.order-` classes for controlling the **visual order** of your content. These classes are responsive, so you can set the `order` by breakpoint (e.g., `.order-1.order-md-2`). Includes support for `1` through `12` across all five grid tiers.

First, but unordered	Third, but first	Second, but last

Copy

```
<div class="container">
  <div class="row">
    <div class="col">
      First, but unordered
    </div>
    <div class="col order-12">
      Second, but last
    </div>
    <div class="col order-1">
      Third, but first
    </div>
  </div>
</div>
```

There are also responsive `.order-first` and `.order-last` classes that change the `order` of an element by applying `order: -1` and `order: 13` (`order: $columns + 1`), respectively. These classes can also be intermixed with the numbered `.order-*` classes as needed.

Third, but first	Second, but unordered	First, but last

Copy

```
<div class="container">
  <div class="row">
    <div class="col order-last">
      First, but last
    </div>
    <div class="col">
      Second, but unordered
    </div>
    <div class="col order-first">
      Third, but first
    </div>
  </div>
</div>
```

Offsetting columns

You can offset grid columns in two ways: our responsive `.offset-` grid classes and our margin utilities. Grid classes are sized to match columns while margins are more useful for quick layouts where the width of the offset is variable.

Offset classes

Move columns to the right using `.offset-md-*` classes. These classes increase the left margin of a column by `*` columns. For example, `.offset-md-4` moves `.col-md-4` over four columns.

Copy

```
<div class="row">
  <div class="col-md-4">.col-md-4</div>
  <div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>
</div>
<div class="row">
  <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
  <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
</div>
<div class="row">
  <div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>
</div>
```

In addition to column clearing at responsive breakpoints, you may need to reset offsets. See this in action in the grid example.

.col-sm-5 .col-md-6	.col-sm-5 .offset-sm-2 .col-md-6 .offset-md-0
.col-sm-6 .col-md-5 .col-lg-6	.col-sm-6 .col-md-5 .offset-md-2 .col-lg-6 .offset-lg-0

Copy

```
<div class="row">
  <div class="col-sm-5 col-md-6">.col-sm-5 .col-md-6</div>
  <div class="col-sm-5 offset-sm-2 col-md-6 offset-md-0">.col-sm-5 .offset-sm-2 .col-md-6 .offset
</div>

<div class="row">
  <div class="col-sm-6 col-md-5 col-lg-6">.col-sm-6 .col-md-5 .col-lg-6</div>
  <div class="col-sm-6 col-md-5 offset-md-2 col-lg-6 offset-lg-0">.col-sm-6 .col-md-5 .offset-md
</div>
```

Typography

Documentation and examples for Bootstrap typography, including global settings, headings, body text, lists, and more.

Headings

All HTML headings, `<h1>` through `<h6>`, are available.

Heading	Example
`<h1></h1>`	h1. Bootstrap heading
`<h2></h2>`	h2. Bootstrap heading
`<h3></h3>`	h3. Bootstrap heading
`<h4></h4>`	h4. Bootstrap heading
`<h5></h5>`	h5. Bootstrap heading
`<h6></h6>`	h6. Bootstrap heading

Copy

```
<h1>h1. Bootstrap heading</h1>
<h2>h2. Bootstrap heading</h2>
<h3>h3. Bootstrap heading</h3>
<h4>h4. Bootstrap heading</h4>
<h5>h5. Bootstrap heading</h5>
<h6>h6. Bootstrap heading</h6>
```

`.h1` through `.h6` classes are also available, for when you want to match the font styling of a heading but cannot use the associated HTML element.

Blockquotes

For quoting blocks of content from another source within your document. Wrap `<blockquote class="blockquote">` around any HTML as the quote.

> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a ante.

Copy

```
<blockquote class="blockquote">
  <p class="mb-0">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a
</blockquote>
```

Images

Documentation and examples for opting images into responsive behavior (so they never become larger than their parent elements) and add lightweight styles to them—all via classes.

Responsive images

Images in Bootstrap are made responsive with `.img-fluid`. `max-width: 100%;` and `height: auto;` are applied to the image so that it scales with the parent element.


```
<img src="..." class="img-fluid" alt="Responsive image">
```

Copy

SVG images and IE 10

In Internet Explorer 10, SVG images with `.img-fluid` are disproportionately sized. To fix this, add `width: 100% \9;` where necessary. This fix improperly sizes other image formats, so Bootstrap doesn't apply it automatically.

Image thumbnails

In addition to our border-radius utilities, you can use `.img-thumbnail` to give an image a rounded 1px border appearance.


```
<img src="..." alt="..." class="img-thumbnail">
```

Copy

Tables

Documentation and examples for opt-in styling of tables (given their prevalent use in JavaScript plugins) with Bootstrap.

Using the most basic table markup, here's how `.table`-based tables look in Bootstrap. **All table styles are inherited in Bootstrap 4**, meaning any nested tables will be styled in the same manner as the parent.

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Copy

```
<table class="table">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First</th>
      <th scope="col">Last</th>
      <th scope="col">Handle</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>
    <tr>
      <th scope="row">2</th>
      <td>Jacob</td>
      <td>Thornton</td>
      <td>@fat</td>
    </tr>
    <tr>
      <th scope="row">3</th>
      <td>Larry</td>
      <td>the Bird</td>
      <td>@twitter</td>
    </tr>
  </tbody>
</table>
```

Forms

Examples and usage guidelines for form control styles, layout options, and custom components for creating a wide variety of forms.

Here's a quick example to demonstrate Bootstrap's form styles. Keep reading for documentation on required classes, form layout, and more.

Copy

```
<form>
  <div class="form-group">
    <label for="exampleInputEmail1">Email address</label>
    <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHel
    <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Password</label>
    <input type="password" class="form-control" id="exampleInputPassword1" placeholder="Passwor
  </div>
  <div class="form-check">
    <input type="checkbox" class="form-check-input" id="exampleCheck1">
    <label class="form-check-label" for="exampleCheck1">Check me out</label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Sizing

Set heights using classes like `.form-control-lg` and `.form-control-sm`.

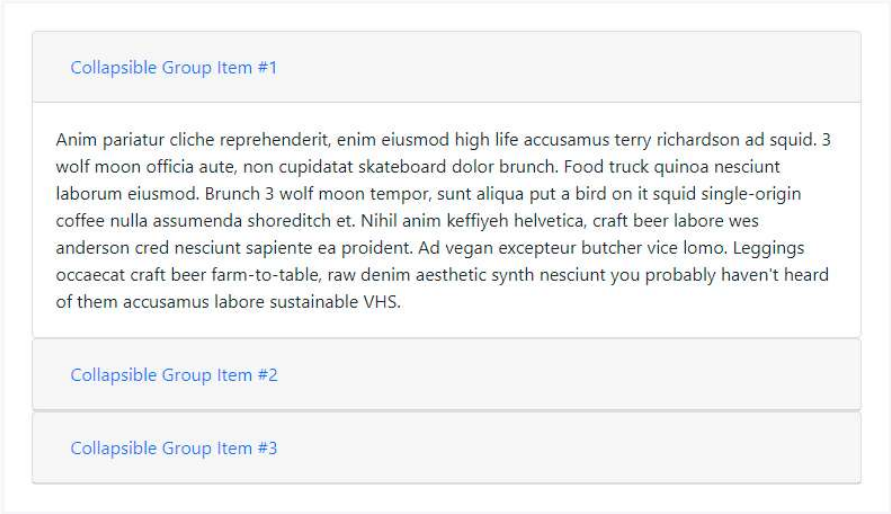
Copy

```
<input class="form-control form-control-lg" type="text" placeholder=".form-control-lg">
<input class="form-control" type="text" placeholder="Default input">
<input class="form-control form-control-sm" type="text" placeholder=".form-control-sm">
```

Bootstrap Components

- **Accordion**

  Using the card component, you can extend the default collapse behavior to create an accordion.



Copy

```
<div id="accordion">
  <div class="card">
    <div class="card-header" id="headingOne">
      <h5 class="mb-0">
        <button class="btn btn-link" data-toggle="collapse" data-target="#collapseOne" aria-expar
          Collapsible Group Item #1
        </button>
      </h5>
    </div>

    <div id="collapseOne" class="collapse show" aria-labelledby="headingOne" data-parent="#accor
      <div class="card-body">
        Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry richardson ad
      </div>
    </div>
  </div>
  <div class="card">
    <div class="card-header" id="headingTwo">
      <h5 class="mb-0">
        <button class="btn btn-link collapsed" data-toggle="collapse" data-target="#collapseTwo"
          Collapsible Group Item #2
        </button>
      </h5>
    </div>
    <div id="collapseTwo" class="collapse" aria-labelledby="headingTwo" data-parent="#accordion">
      <div class="card-body">
        Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry richardson ad
      </div>
    </div>
  </div>
  <div class="card">
    <div class="card-header" id="headingThree">
      <h5 class="mb-0">
        <button class="btn btn-link collapsed" data-toggle="collapse" data-target="#collapseThree
          Collapsible Group Item #3
        </button>
      </h5>
    </div>
    <div id="collapseThree" class="collapse" aria-labelledby="headingThree" data-parent="#accord
      <div class="card-body">
        Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry richardson ad
      </div>
    </div>
  </div>
</div>
```

- **Alerts**

  Provide contextual feedback messages for typical user actions with the handful of available and flexible alert messages.


```
<div class="alert alert-primary" role="alert">
  This is a primary alert—check it out!
</div>
```

Dismissing

Using the alert JavaScript plugin, it's possible to dismiss any alert inline. Here's how:

- Be sure you've loaded the alert plugin, or the compiled Bootstrap JavaScript.
- If you're building our JavaScript from source, it requires `util.js`. The compiled version includes this.
- Add a dismiss button and the `.alert-dismissible` class, which adds extra padding to the right of the alert and positions the `.close` button.
- On the dismiss button, add the `data-dismiss="alert"` attribute, which triggers the JavaScript functionality. Be sure to use the `<button>` element with it for proper behavior across all devices.
- To animate alerts when dismissing them, be sure to add the `.fade` and `.show` classes.

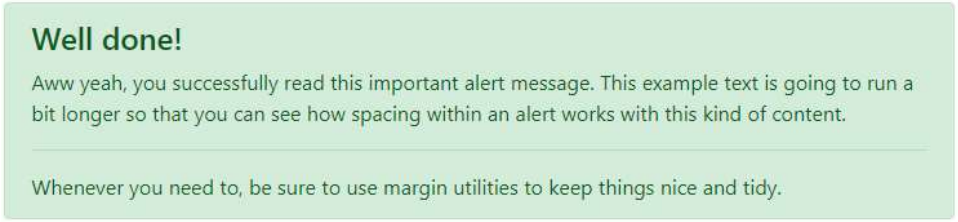
You can see this in action with a live demo:

Copy

```
<div class="alert alert-warning alert-dismissible fade show" role="alert">
  <strong>Holy guacamole!</strong> You should check in on some of those fields below.
  <button type="button" class="close" data-dismiss="alert" aria-label="Close">
    <span aria-hidden="true">&times;</span>
  </button>
</div>
```

Additional content

Alerts can also contain additional HTML elements like headings, paragraphs and dividers.

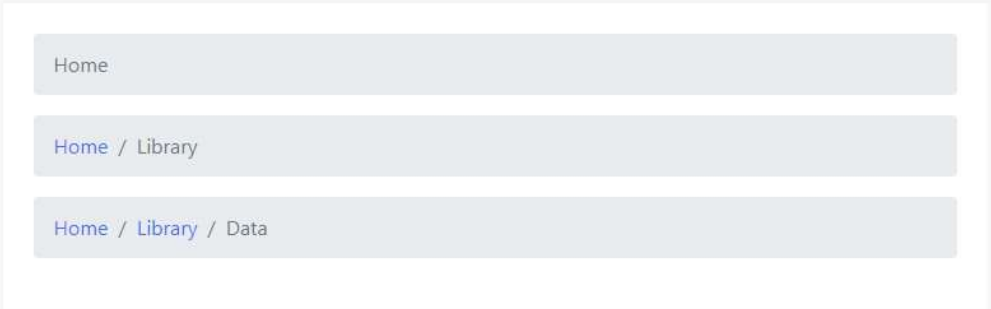


Copy

```
<div class="alert alert-success" role="alert">
  <h4 class="alert-heading">Well done!</h4>
  <p>Aww yeah, you successfully read this important alert message. This example text is going to
  <hr>
  <p class="mb-0">Whenever you need to, be sure to use margin utilities to keep things nice and t
</div>
```

- **Breadcrumb**

  Indicate the current page's location within a navigational hierarchy that automatically adds separators via CSS.

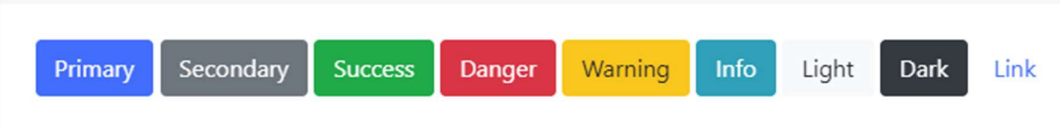


Copy

```
<nav aria-label="breadcrumb">
  <ol class="breadcrumb">
    <li class="breadcrumb-item active" aria-current="page">Home</li>
  </ol>
</nav>

<nav aria-label="breadcrumb">
  <ol class="breadcrumb">
    <li class="breadcrumb-item"><a href="#">Home</a></li>
    <li class="breadcrumb-item active" aria-current="page">Library</li>
  </ol>
</nav>

<nav aria-label="breadcrumb">
  <ol class="breadcrumb">
    <li class="breadcrumb-item"><a href="#">Home</a></li>
    <li class="breadcrumb-item"><a href="#">Library</a></li>
    <li class="breadcrumb-item active" aria-current="page">Data</li>
  </ol>
</nav>
```

- **Button**

  Bootstrap includes several predefined button styles, each serving its own semantic purpose, with a few extras thrown in for more control.




```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>

<button type="button" class="btn btn-link">Link</button>
```

- **Button Group**

  Group a series of buttons together on a single line with the button group, and super-power them with JavaScript.


```
<div class="btn-group" role="group" aria-label="Basic example">
  <button type="button" class="btn btn-secondary">Left</button>
  <button type="button" class="btn btn-secondary">Middle</button>
  <button type="button" class="btn btn-secondary">Right</button>
</div>
```

Copy

- **Card**

  Bootstrap's cards provide a flexible and extensible content container with multiple variants and options.

Card title

Some quick example text to build on the card title and make up the bulk of the card's content.

Go somewhere

```
<div class="card" style="width: 18rem;">
  <img class="card-img-top" src="..." alt="Card image cap">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make up the bul
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

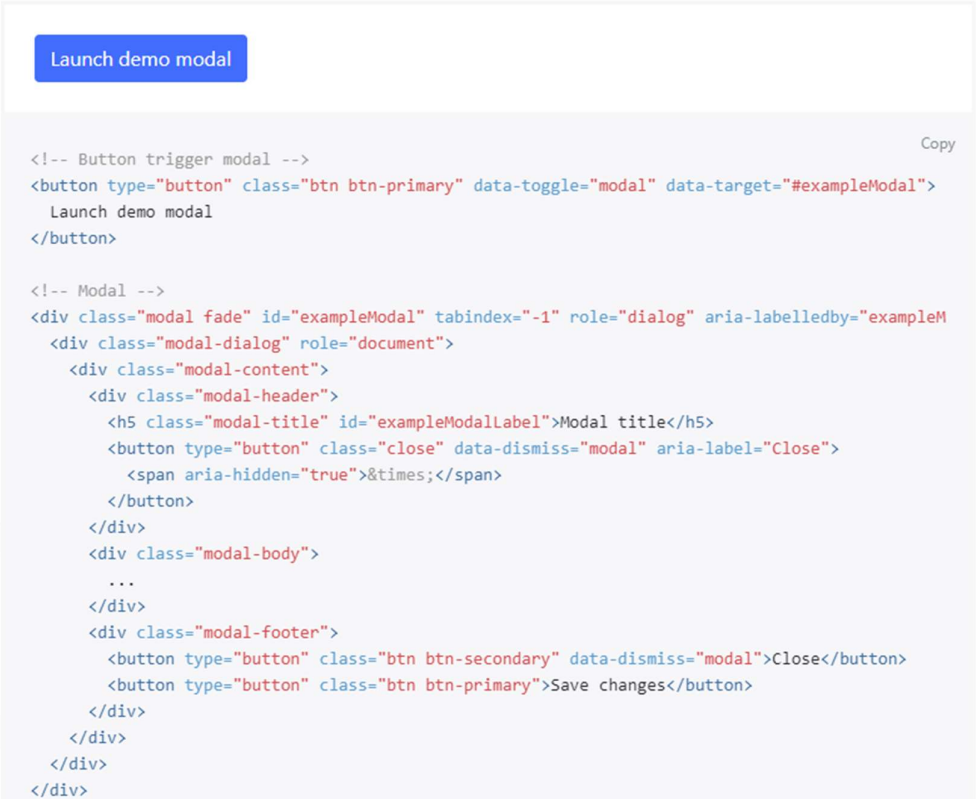
Copy

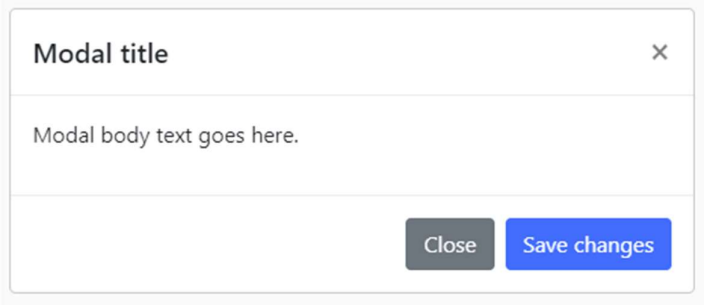
- **Dropdown**

  Toggle contextual overlays for displaying lists of links and more with the Bootstrap dropdown plugin.

- **Modal**

  Use Bootstrap's JavaScript modal plugin to add dialogs to your site for lightboxes, user notifications, or completely custom content.





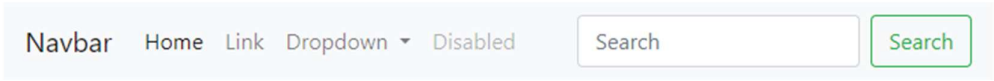
- **Navs**

  Documentation and examples for how to use Bootstrap's included navigation components.


```
<ul class="nav">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

- **Nav & Tabs**

  Navbars come with built-in support for a handful of sub-components




```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSuppo
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
        <a class="nav-link" href="#">Home <span class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Link</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-tog
          Dropdown
        </a>
        <div class="dropdown-menu" aria-labelledby="navbarDropdown">
          <a class="dropdown-item" href="#">Action</a>
          <a class="dropdown-item" href="#">Another action</a>
          <div class="dropdown-divider"></div>
          <a class="dropdown-item" href="#">Something else here</a>
        </div>
      </li>
      <li class="nav-item">
        <a class="nav-link disabled" href="#">Disabled</a>
      </li>
    </ul>
    <form class="form-inline my-2 my-lg-0">
      <input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search
      <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>
    </form>
  </div>
</nav>
```

- **Navigation**

  Documentation and examples for showing pagination to indicate a series of related content exists across multiple pages.

- **Labels & Progressbars**

  Add labels to your progress bars by placing text within the

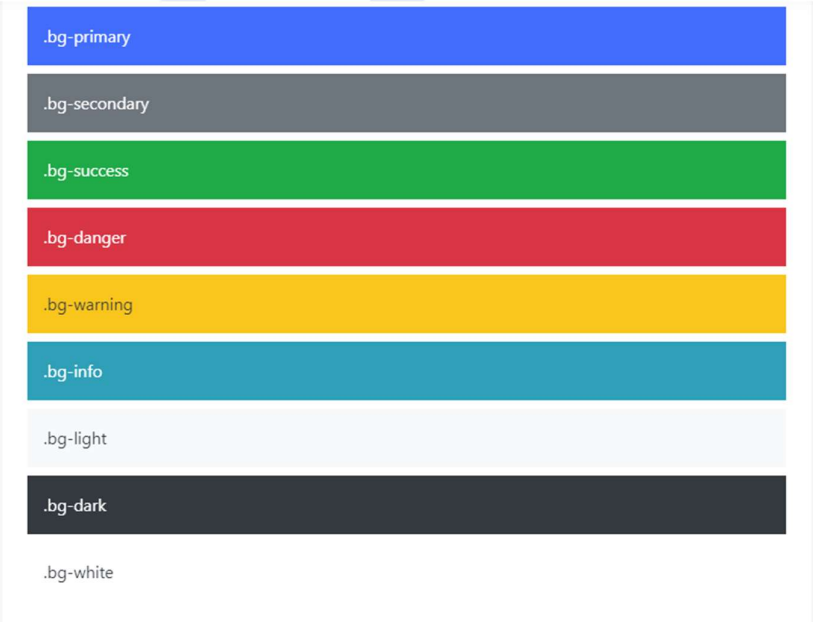
Bootstrap Components

- **Colors**

Copy

```
<p class="text-primary">.text-primary</p>
<p class="text-secondary">.text-secondary</p>
<p class="text-success">.text-success</p>
<p class="text-danger">.text-danger</p>
<p class="text-warning">.text-warning</p>
<p class="text-info">.text-info</p>
<p class="text-light bg-dark">.text-light</p>
<p class="text-dark">.text-dark</p>
<p class="text-muted">.text-muted</p>
<p class="text-white bg-dark">.text-white</p>
```

- **Background**



Copy

```
<div class="p-3 mb-2 bg-primary text-white">.bg-primary</div>
<div class="p-3 mb-2 bg-secondary text-white">.bg-secondary</div>
<div class="p-3 mb-2 bg-success text-white">.bg-success</div>
<div class="p-3 mb-2 bg-danger text-white">.bg-danger</div>
<div class="p-3 mb-2 bg-warning text-dark">.bg-warning</div>
<div class="p-3 mb-2 bg-info text-white">.bg-info</div>
<div class="p-3 mb-2 bg-light text-dark">.bg-light</div>
<div class="p-3 mb-2 bg-dark text-white">.bg-dark</div>
<div class="p-3 mb-2 bg-white text-dark">.bg-white</div>
```

- **Border**

  Use border utilities to add or remove an element's borders. Choose from one at a time.


```
<span class="border"></span>
<span class="border-top"></span>
<span class="border-right"></span>
<span class="border-bottom"></span>
<span class="border-left"></span>
```

- **Display**

  Quickly and responsively toggle the display value of components and more with our display utilities.

As such, the classes are named using the format:

- `.d-{value}` for `xs`
- `.d-{breakpoint}-{value}` for `sm`, `md`, `lg`, and `xl`.

Where *value* is one of:

- `none`
- `inline`
- `inline-block`
- `block`
- `table`
- `table-cell`
- `table-row`
- `flex`
- `inline-flex`

The media queries effect screen widths with the given breakpoint *or larger*. For example, `.d-lg-none` sets `display: none;` on both `lg` and `xl` screens.

Examples

Copy

```
<div class="d-inline p-2 bg-primary text-white">d-inline</div>
<div class="d-inline p-2 bg-dark text-white">d-inline</div>
```

- **Position**

Use these shorthand utilities for quickly configuring the position of an element.

Common values

Quick positioning classes are available, though they are not responsive.

```
<div class="position-static">...</div>
<div class="position-relative">...</div>
<div class="position-absolute">...</div>
<div class="position-fixed">...</div>
<div class="position-sticky">...</div>
```

Copy

Fixed top

Position an element at the top of the viewport, from edge to edge. Be sure you understand the ramifications of fixed position in your project; you may need to add aditional CSS.

```
<div class="fixed-top">...</div>
```

Copy

Fixed bottom

Position an element at the bottom of the viewport, from edge to edge. Be sure you understand the ramifications of fixed position in your project; you may need to add aditional CSS.

```
<div class="fixed-bottom">...</div>
```

Copy

Sticky top

Position an element at the top of the viewport, from edge to edge, but only after you scroll past it. The `.sticky-top` utility uses CSS's `position: sticky`, which isn't fully supported in all browsers.

IE11 and IE10 will render `position: sticky` as `position: relative`. As such, we wrap the styles in a `@supports` query, limiting the stickiness to only browsers that can render it properly.

```
<div class="sticky-top">...</div>
```

Copy

- **Spacing**

Bootstrap includes a wide range of shorthand responsive margin and padding utility.

The classes are named using the format `{property}{sides}-{size}` for `xs` and `{property}{sides}-{breakpoint}-{size}` for `sm`, `md`, `lg`, and `xl`.

Where *property* is one of:

- `m` - for classes that set `margin`
- `p` - for classes that set `padding`

Where *sides* is one of:

- `t` - for classes that set `margin-top` or `padding-top`
- `b` - for classes that set `margin-bottom` or `padding-bottom`
- `l` - for classes that set `margin-left` or `padding-left`
- `r` - for classes that set `margin-right` or `padding-right`
- `x` - for classes that set both `*-left` and `*-right`
- `y` - for classes that set both `*-top` and `*-bottom`
- blank - for classes that set a `margin` or `padding` on all 4 sides of the element

Where *size* is one of:

- `0` - for classes that eliminate the `margin` or `padding` by setting it to `0`
- `1` - (by default) for classes that set the `margin` or `padding` to `$spacer * .25`
- `2` - (by default) for classes that set the `margin` or `padding` to `$spacer * .5`
- `3` - (by default) for classes that set the `margin` or `padding` to `$spacer`
- `4` - (by default) for classes that set the `margin` or `padding` to `$spacer * 1.5`
- `5` - (by default) for classes that set the `margin` or `padding` to `$spacer * 3`
- `auto` - for classes that set the `margin` to auto

- **Text**

  Documentation and examples for common text utilities to control alignment, wrapping, weight, and more.

- **Vertical Allignment**
  Easily change the vertical alignment of inline, inline-block, inline-table, and table cell elements.

baseline top middle bottom text-top text-bottom

Copy

```
<span class="align-baseline">baseline</span>
<span class="align-top">top</span>
<span class="align-middle">middle</span>
<span class="align-bottom">bottom</span>
<span class="align-text-top">text-top</span>
<span class="align-text-bottom">text-bottom</span>
```

With table cells:

baseline top text-top text-bottom middle bottom

Copy

```
<table style="height: 100px;">
  <tbody>
    <tr>
      <td class="align-baseline">baseline</td>
      <td class="align-top">top</td>
      <td class="align-middle">middle</td>
      <td class="align-bottom">bottom</td>
      <td class="align-text-top">text-top</td>
      <td class="align-text-bottom">text-bottom</td>
    </tr>
  </tbody>
</table>
```