RANKA
DATE / /
PAGE

WWW [World Wide Web]

1) WWW stand for World wide web.
2) It is a way of exchanging information b/w computer on the internet.
3) It is a way accessing information all the midden of the internet.
4) It is an information sharing moddle that is on top of the internet.
5) WWW is the collection of all the information [text, images, audio, video]
6) Most of this information is accessed through web -site.
7) WWW uses http protocol to access the information from various server.

Evolution of WWW

1) The invention of WWW can be credited to sir Tim Berners lee. During this work europen organisation for nulecus research in 1989. He had develop basis idea of WWW.

RANKA
DATE / /
PAGE

Web 1.0 [1990 - 2000]

1.) It remains limited to static websites.
2.) Mostly publishing
3.) Limited for reading only
4.) Closed access
5.) Corporation mostly
6.) HTTP, HTML

Web 2.0 [2000 - 2010]

1) Publishing as well as participation
2) Social media
3.) Blogging
4.) Wikis
5.) Keyboward search
6) Ajak, Jquert, XML

Web 3.0 [2010 - onwards]

1.) Mostly drag and drop
2.) Mobile oriental
3) Micro blogging
4) Cloud computing

5) Grid computing
6) Semantic search

Different between www and Internet

	Internet	WWW
1)	Internet is a global network of networks	Its stand for world wide web
2)	Internet is a mean of connecting a computer to any other computer any where in the world.	It is a collection of information which is accessed by via
3)	Nature of internet is hardware	Nature of www is software
4)	Internet consists of computer, routers, cables, modem firewall server etc	www consists of information text like images, audio, video
5)	Internet works on the basis of internet protocol	www uses HTTP

	Internet	WWW
6)	Internet is super set of www.	WWW is sub set of internet
7)	The first version of inter-net uses Arpanet.	In the begning it was none as NSFNET
8)	Computing devices are adantefied by IP address	Information pieces are adantefied by URL
9)	Internet is oregenated in 1960	It was enverted in 1989

- HTTP [Hyper Text Transfer Protocol]

1) HTTP is an application protocol that allow users to transfer data on WWW.

2) It is a set of rules for transfering file such as text file, images, sound, video and other files over the internet.

RANKA
DATE / /
PAGE

3) It gives users a way to interakt with web resources such as html file

4) As soon as user open their browser they are directly using http

- Working of HTTP

1) Using the http protocol resources are exchanged b/w client device and server over the internet.

2) Client and server connected by exchanging messages

3) The message send by client [web browser] are called request and the messages sent by server as an answer are called response

4) Client devices send request to server for the resources needed to load a web page and this server sent response that to the client to full fill the request.

- Features of http

RANKA
DATE / /
PAGE

1) Connection less protocol
2) Media independent
3) Stateless

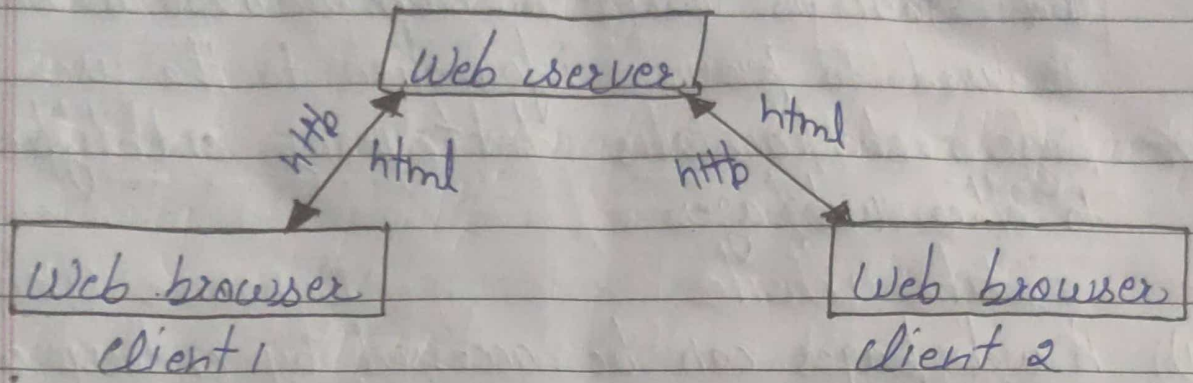
Web Server :-

Web server stores and deliver the contents for a website such as text image, audio or video. A web server is the software and hardware that uses the http, protocols to respond to client request.

1) The main job of a web server is to display website content.
2) Web server hardware is connected to the internet and allows data to be exchange with other connected devices.
3) Web server software controls how a user accesses hosted files.
4) Web server process is an example of client server model.
5) The most commons types of client program is a web browser.

RANKA
DATE / /
PAGE

Working of a web server :-



1) Web browser request the data from the web server
2) The client program request the web servers for some data.
3) Web server accepts the request and process it response the web browser by providing request data. A web server host.

Example of Web Server

1) Apache web server
2) Microsoft Internet information Service (IIS)
3) Sun Jave system web server
4) Nginx Lighttpd

* Web page

RANKA
DATE / /
PAGE

Web page is a document that is return in html and that is viewed in web browser a web pag is used to provide information to the user a w page make contains text, image and hyper lin to other webpages.

1) A web pages can be accessed by endrew a Url address, into a browser address bar.

Web site :-

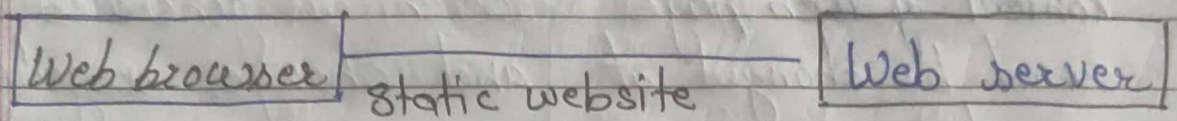
1) It is collection of related web pages that may contain text images, audio, and video.

2) The first page of web page are called home page.

* Static web site

1) Static web site is the basic type of web site that is easy to create.
2) Static website elements are fixed.
3) A static web site content never changes.
4) A static web site does not required web pro-gramming or data base design.

5) The codes are fixed for each page so the informa-tion in the page does not change.
6) In static web site web pages are returned a which are pre built source code file using simple language html, CSS, Javascript.
7) There are no proccessing of the content on the sources.



* Dynamic web site

1) In dynamic web site web pages are return by the webserver which are passed during run time.
2) Dynamic web site is a collection of dynamic web pages how contents changes dynemical.
3) In dynamic web site web page are built during run time with the help of server site scripting languages like PHP, Note.Js
4) Dynamic web site uses client site scripting and server site scripting to generate dynamic contents.

RANKA

Web browser | Web server | Data base ser

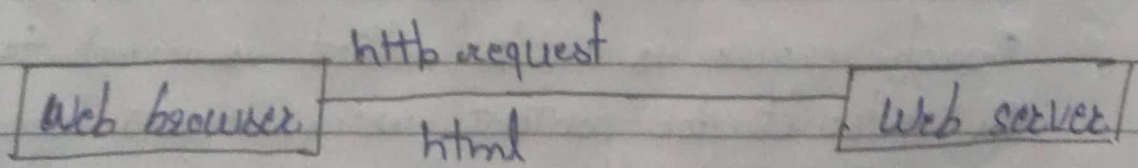
* Difference b/w static and dynamic website.

	Static website	Dynamic website
1)	Content of web page cannot be changed at run time.	Content is generated quick and changes regularly.
2)	No interaction with data base	Content of web page can be changed.
3)	Cheaper development cost	Most development cost.
4)	It sends the same response what every request	It may generate different html for each of request.
5)	HTML, CSS, Javascript is used for development website.	HTML, CSS, Java script and server site scripting language PHP, JSP, ASP. Ne, Node.JS.

	Static Website	Dynamic Website
6)	Same content is delivered every time the page is located.	Content may change every time the page is reloaded.
7)	No user interavity	User interactivity
8)	No features of content managment.	Features of content manag-ment system [CMS]

Web browser :-

A web browser is a type of software that allows you to find and mute web site only internet. A web browser is an application software for accessing WWW. When a user type URL into the browser address bar. The web browser feact the contents from the web server and display on the user devices.



The information is transpate using http protocol which define how, text, image and video are transpeated on the web.

Web browser Example :-

1) Google :- It is the most popular web browser with 70% of global market share.

2) Microsoft edge :- It is microsoft new web browser.

3) Mozilla firefox :- It is the most popular web bro application in US.

4) Safari :- It is the default web browser for all apple devices.

Search engine :-

Search engine is a tool that is used to sea the internet for content using keywords.

It is a software program that helps peop to find the information using keyword.

When a user entre the query into a search engine. A search engine result page is returned.

A search engine is a service that allows internet user to search for content via www.

Example :- Google, Bing, Duck Duck go.

Working of search engine :-

1) Crawling :- The actual discovery of new web page on the internet starts with the process called crawling. Search engine use small program called [crawler, bots, spider bots]. That follows links from known pages to new pages.

2) Indexing :- The search engine will try to understand and categorise the contents on a web page. the web page through keyboard. After crawling it is time for index. The process of validating and storing content from the web page in the search engine database called index.

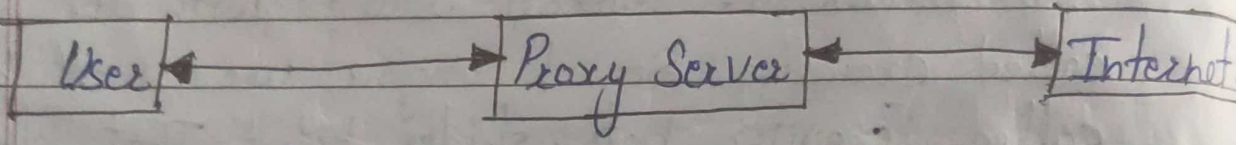
3) Ranking :- In the last step include picking the best result and creating a list of pages that will appear of result page.

RANKA
DATE / /
PAGE

Proxy Server :-

Proxy server works as a gateway b/w user and internet.

The proxy server is a computer on the internet that accepts the incoming request from the client and forwards does request to the server. It seperate the client system and web server from global network.



Working Proxy Server :-

A proxy server is a machine that translates traffic b/w networks it is a server that seperates users from the internet. A proxy server has own address.

1) When a user send a web request to proxy server
2) The proxy server then makes your web request all the behalf.
3) An collect the response from web server and forward the web page data to the user.

RANKA
DATE / /
PAGE

* Proxy server following function :-

1) Firewall
2) Network data filtering
3) Network connection Sharing
4) Data caching
5) Proxy server hide your IP address
6) Using of Proxy server
7) Maintaining and filtering network traffic
8) Content filter
9) Security
10) Logging

RANKA
DATE / /
PAGE

Unit-2

HTML Tag

<HTML> ÷ <HTML> it represent the root of ht
document it is the container for all "htr
elements.

Example :- <Html>
<head>
<title> Web pages </title
<head>
<body>
 esome text here
</body>
</html>

<Head> elements :- It is a container for meta data and is b/w <html> and <bod
Meta data define document title, character set, style and other information.

1) <title> ÷ Title elements defined the title of the documents. Title must be text only and it is esome in the browser, title bar on in the pages tab.

2) <style> :- The <style> elements is used to defined style information for a html page.

3) <meta> :- <Meta> defines meta data abut about html document.

<Meta> always go inside <head> elements and are used to define character set, keywoards, authorised and view port setting

Example :- <meta character set = "Utf. 8">
<meta name = "author" content = "A-Shubhan">

1) Meta data is used by browser and search engine.
2) Meta data will not be displaced on the page.

4) <Link> :- The link element defines the relationship between the current document and external document. This tag is used to link to external style sheet.

Example :- <link rel = "stylesheet" href = "file.css">

rel atribute :- It specified the relationship b/w the current document and external documents.

Href :- It specified the location of external file.

RANKA
DATE / /
PAGE

The link elements is a empty elements.

<body> :- The <body> defines the documents body.
The body element contains all the contents of html document. such as heading, paragraph, image tables, list etc.

Body elements

<Heading>
HTML <h1> to <h6> are used to define html heading

<h1> :- <h1> define it is most popular heading and <h6> define the least importing heading

example :- <body>
<h1> This is a heading </h1>
<h2> heading 2 </h2>
<h3> Heading 3 </h3>
<h4> heading 4 </h4>
<h5> heading 5 </h5>

<hr> :- <hr> stands for horizontal rule. It is used to insert a horizontal rule in a page

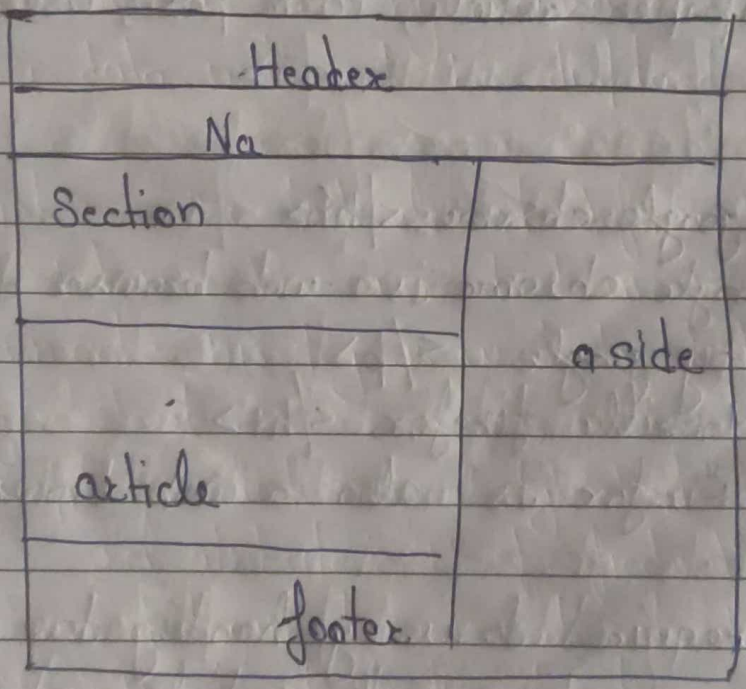
This tag is used defined thematic change in the content. It is also an emity tag.

Semantic Elements in HTML

A semantic elements clearly describe its meaning to bother browser and the developer.

Example of non semantic elements are :- and

Example of semantic elements are :- <table>, <form>, <article>, <header>, <footer>, <section>, <main>



RANKA
DATE / /
PAGE

① <Section> elements :- This section element define in a document.
A section is a thematic grouping of content typically with a heading. A home page can be script in a section for introduction, content, and contact information.

② <Article> elements :- It specify independent, self contain content. An article have it own meaning and it should be possible to read it independently from the rest of web site.
Example of where article elements used Blog post, newspaper article, Form post, User comment

```
<article>
    <h1> google chrome </h1>
    <p> google chrome is a web browser developed
        by google </p>
    <h1> Mozilla firefox. </h1>
    <p> Firefox is also a web browser </p>
</article>
```

It is required to using heading tag ~~with~~ <h1> —— <h6> with article and <section> tag

RANKA
DATE / /
PAGE

③ <Header> element :- Header element specify a header for a document or a section.
Header element should be use has a container for introductive content.
It contain logo, navigation link.
You can have many header elements in one document

```
<header>
<h1> google chrome </h1>
</header>
```

④ <Footer> element :- Footer elements defines a footer for a documents or a web page
A Footer tipikly charge author of the contact coby write information, contact information etc.

```
<footer>
<p> @ copyright </p>
</footer>
```

⑤ <aside> :- IT is use to identife the contant that is related to the primary contant of a web page.
This elements defines a section with additional information or highilighting parts of the information that can be intresting to user.

```
<body>
  <aside>
    <h1> Java </h1>
  <p> Java is a programming language </p>
</aside>
 </body>
```

The content in this or elements should be related to the surrounding content

<nav> :- nav element define a set of navigation li

```
<nav>
<a href = "/html"> HTML </a>
<a href = "/CSS"> CSS </a>
<a href = " Span.html"> Span </a>
</nav>
```

<Figure> and <fig caption> :- <Figure elements is used to handle the group of diagram, photo etc.
This tag is used to markup a photo in a document

<fig caption> :- This is used to define a caption

for a photo this is used to fig caption for a photo as image.

Image element define the image can figcaption define the caption.

<details> ÷ This tag detail specified additional detail that user can open and close on dimeand. This tage is used to create a visit wiget that the user can open and close by default wiget is closed when open it expends and display the content within.

<details>

<summary>
Click to see details
</summary>
 detail content
</details>

<Summary> ÷ This tag define a visible heading for the detail element. The heading can be click to view or hiad a detail.

<Main> ÷ This tag specify the main contain document

RANKA
DATE / /
PAGE

The content inside the main element should be unique to the document. It should not contain any content that is rebeted across document

```
<main>
 <h1> most input web browsers </h1>
 <p> Google chrome, fixefox. </p>
</article>
```

There not must be more than one main elements in a document

<u>Mark</u> :- The <mark> define the text that should be highlighted

 do not forget to bring <mark> practile files </mark> today

<u>Time</u> :- <Time> define a specific time.
The date time atribute of this element is used to translate the time into a machine readable format

 College open at <time> 9:00 AM </time>

RANKA
DATE / /
PAGE

HTML Attributes :-

HTML Attributes provide additional information about HTML elements. Or HTML elements had attributes. Attributes come in name/value/pair.

<href> :- A tag deffine a hyper link dieass attribute specific the URL of the page the link goes to

<Access Key> :- This attribute specific a shortcut key to active a or focus an element.
Its value must be a single character.

<Style> :- Style attributes is used to add style to an element such as colour, size, font etc.

Class :- A class attribute specify a one or more class name for an element

ID :- It is used to specify a unique id for an element. The value of id attributes must be unique b/w html document.

tagtab index :- The tab index specify the tag ord
of an elements.

<a href = "https:// www.google.com" tab index = "2"
google

title :- A title attribute specify extra information
about an element.

Data-* :- This attribute is used to m bag custom
data.
This is used to store custom data private to
the page.

HTML Coding & Conventions :-

1) Always declare document type as first line in
your document.
<!Doctype html>

2) Use lower case element name :-
3) Close all html element.
4) Use lower case attributes name
5) Always quote attribute value.
6) Always specify alt, width, height for images

7) Do not add blank line, spaces or indentation without a reason.

8) Never skip the title element.

Comments in HTML :- HTML comments are not display in the browser. You can add comment <!, write comment here --> like this :-

Comparison chart of Internet and WWW

	Internet	**WWW**
1.	Internet originated sometimes in late 1960s.	English scientist Tim Berners-Lee invented the World Wide Web in 1989
2.	Nature of Internet is hardware.	Nature of www is software.
3.	Internet consists of computers, routers, cables, bridges, servers, cellular towers, satellites etc.	www consists of information like text, images, audio, video
4.	The first version of the Internet was known as ARPANET	In the beginning WWW was known as NSFNET
5.	Internet works on the basis of Internet Protocol (IP)	WWW works on the basis of Hyper Text Transfer Protocol (HTTP)
6.	Internet is independent of WWW	WWW requires the Internet to exist
7.	Internet is superset of WWW	WWW is a subset of the Internet. Apart from supporting www, the Internet's hardware infrastructure is used for other things as well (e.g. FTP, SMTP)
8.	Computing devices are identified by IP Addresses	Information pieces are identified by Uniform Resource Locator (URL)

Difference between Internet and WWW :

S.No.	INTERNET	WWW
1	Internet is a global network of networks.	WWW stands for World wide Web.
2	Internet is a means of connecting a computer to any other computer anywhere in the world.	World Wide Web which is a collection of information which is accessed via the Internet.
3	Internet is infrastructure.	WWW is service on top of that infrastructure.
4	Internet can be viewed as a big book-store.	Web can be viewed as collection of books on that store.
5	At some advanced level, to understand we can think of the Internet as hardware.	At some advanced level, to understand we can think of the WWW as software.
6	Internet is primarily hardware-based.	WWW is more software-oriented as compared to the Internet.
7	It is originated sometimes in late 1960s.	English scientist Tim Berners-Lee invented the World Wide Web in 1989.
8	Internet is superset of WWW.	WWW is a subset of the Internet.
9	The first version of the Internet was known as ARPANET.	In the beginning WWW was known as NSFNET.
10	Internet uses IP address.	WWW uses HTTP.

What Is a Proxy Server?

A proxy server provides a gateway between users and the internet. It is a server, referred to as an "intermediary" because it goes between end-users and the web pages they visit online.

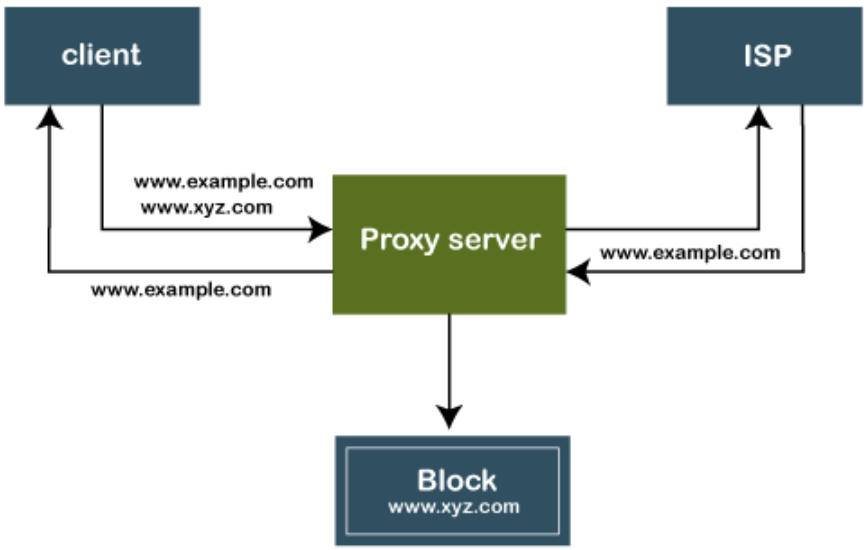
When a computer connects to the internet, it uses an IP address. This is similar to your home's street address, telling incoming data where to go and marking outgoing data with a return address for other devices to authenticate. A proxy server is essentially a computer on the internet that has an IP address of its own.

Proxy Servers and Network Security

Proxies provide a valuable layer of security for your computer. They can be set up as web filters or firewalls, protecting your computer from internet threats like malware.

The **proxy server** is a computer on the internet that accepts the incoming requests from the client and forwards those requests to the destination server. It works as a gateway between the end-user and the internet. It has its own IP address. It separates the client system and web server from the global network.

In other words, we can say that the proxy server allows us to access any websites with a different IP address. It plays an intermediary role between users and targeted websites or servers. It collects and provides information related to user requests.



Mechanism of Proxy Server

What does a proxy server do, exactly?

:

- **Firewalls**: A firewall is a type of network security system that acts as a barrier between a network and the wider internet. Security professionals configure firewalls to block unwanted access to the networks they are trying to protect, often as an anti-malware or anti-hacking countermeasure. A proxy server between a trusted network and the internet is the perfect place to host a firewall designed to intercept and either approve or block incoming traffic before it reaches the network.
- **Content filters**: Just as online proxies can regulate incoming connection requests with a firewall, they can also act as content filters by blocking undesired outgoing traffic. Companies may configure proxy servers as content filters to prevent employees from accessing the blocked websites while at work.
- **Bypassing content filters**: That's right — you can outsmart a web proxy with another proxy. If your company's proxy has blocked your favorite website, but it hasn't blocked access to your personal proxy server or favorite web proxy, you can access your proxy and use it to reach the websites you want.
- **Caching**: Caching refers to the temporary storage of frequently accessed data, which makes it easier and faster to access it again in the future. Internet proxies can cache websites so that they'll load faster than if you were to send your traffic all the way through the internet to the website's server. This reduces *latency* — the time it takes for data to travel through the internet.
- **Security**: In addition to hosting firewalls, proxy servers can also enhance security by serving as the singular public face of the network. From an outside point of view, all the network's users are anonymous, hidden behind the internet proxy's IP address. If a hacker wants to access a specific device on a network, it'll be a lot harder for them to find it.
- **Sharing internet connections**: Businesses or even homes with a single internet connection can use a proxy server to funnel all their devices through that one connection. Using a Wi-Fi router and wireless-capable devices is another solution to this issue.

WEB BROWSER:

A web browser is a type of software that allows you to find and view websites on the Internet.

Each website has a unique address, called a URL (short for Uniform Resource Locator).

When you type a URL into the browser's address bar and press Enter on your keyboard, the browser will load the page associated with that URL.

A web browser (commonly referred to as a browser) is application software for accessing the World Wide Web. When a user follows the URL of a web page from a particular website, the web browser retrieves the necessary content from the website's web server and then displays the page on the user's device.

A web browser takes you anywhere on the internet.

It retrieves information from other parts of the web and displays it on your desktop or mobile device. The information is transferred using the Hypertext Transfer Protocol, which defines how text, images and video are transmitted on the web. This information needs to be shared and displayed in a consistent format so that people using any browser, anywhere in the world can see the information.

Websites save information about you in files called cookies. They are saved on your computer for the next time you visit that site.

Web browser examples

Google Chrome
With 70% of global market share, Google Chrome is the most popular web browser.
Safari
Safari is the default web browser for all Apple devices: Macs, iPads, and iPhones.

Microsoft Edge (formerly Internet Explorer)

Replacing the old and outdated Internet Explorer, Microsoft Edge is Microsoft's new flagship browser. This web browser comes standard with any device using Microsoft's Windows operating system

Mozilla Firefox

Once one of the most popular web browser applications in the US — and the successor of Netscape Navigator, one of the earliest commercially successful web browsers — Firefox has recently lost market share to Chrome and Safari.

Opera

While never the most popular browser, Opera has built a steady user base over the years. This is due in part to the unique features the browser offers, such as a built-in proxy and ad blocker.

What is a search engine?

Search engines allow users to search the internet for content using keywords

When a user enters a query into a search engine, a search engine results page (SERP) is returned, ranking the found pages in order of their relevance.

A search engine is a service that allows Internet users to search for content via the World Wide Web (WWW).

means giving priority to the **highest quality** and most **relevant** pages.

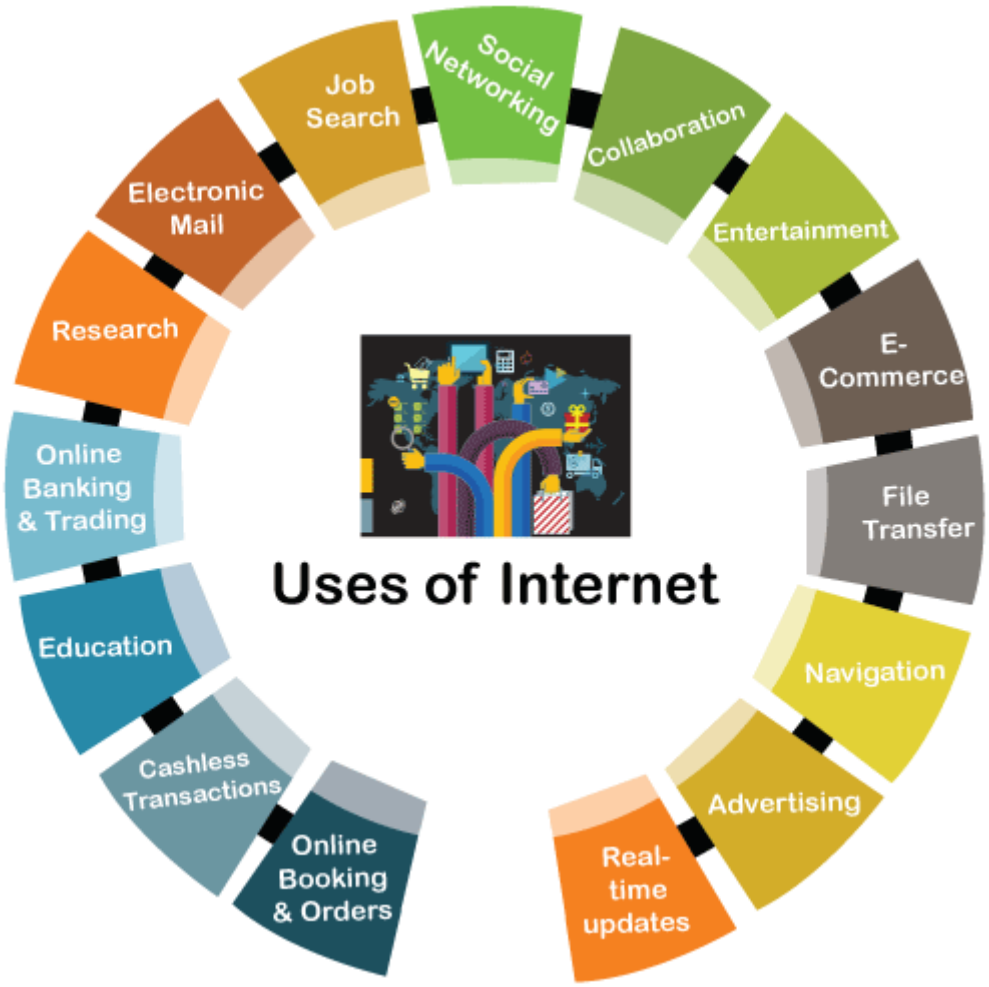
How do search engines work?

There are three key steps to how most search engines work:

- **Crawling** - search engines use programs, called spiders, bots or crawlers, to search the internet.
- **Indexing** - the search engine will try to understand and categorise the content on a web page through 'keywords'.
- **Ranking** - search results are ranked based on a number of factors. These may include keyword density, speed and links. The search engine's aim is to provide the user with the most **relevant** result.

Internet

Internet is a global network that connects billions of computers across the world with each other and to the World Wide Web. It uses standard internet protocol suite (TCP/IP) to connect billions of computer users worldwide. It is set up by using cables such as optical fibers and other wireless and networking technologies. At present, internet is the fastest mean of sending or exchanging information and data between computers across the world.



The Hypertext Transfer Protocol is an application protocol that allows users to communicate data on the World Wide Web.
HTTP gives users a way to interact with web resources such as HTML files by transmitting hypertext messages between clients and servers.

What is HTTP?

HTTP (Hypertext Transfer Protocol) is the set of rules for transferring files -- such as text, images, sound, video and other multimedia files -- over the web.

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,

How HTTP works

Using 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.

Client devices use HTTP to communicate with servers online and access web pages.

HTTP requests and responses

Each interaction between the client and server is called a message.

Client devices submit HTTP requests to servers, which reply by sending HTTP responses back to the clients.

HTTP requests. This is when a client device, such as an internet browser, asks the server for the information needed to load the website.

HTTP responses. The HTTP response message is the data received by a client device from the web server

HTTP

- HTTP stands for **HyperText Transfer Protocol**.
- It is a protocol used to access the data on the World Wide Web (www).
- The HTTP protocol can be used to transfer the data in the form of plain text, hypertext, audio, video, and so on.

HTML Lists

HTML lists allow web developers to group a set of related items in lists.

HTML Lists are used to specify lists of information. All lists may contain one or more list elements. There are three different types of HTML lists:

1. Ordered List or Numbered List (ol)
2. Unordered List or Bulleted List (ul)
3. Description List or Definition List (dl)

HTML offers web authors three ways for specifying lists of information. All lists must contain one or more list elements. Lists may contain −

- **<ul>** − An unordered list. This will list items using plain bullets.
- **<ol>** − An ordered list. This will use different schemes of numbers to list your items.
- **<dl>** − A definition list. This arranges your items in the same way as they are arranged in a dictionary.

Unordered HTML List

An unordered list starts with the `<ul>` tag. Each list item starts with the `<li>` tag.

HTML Unordered List or Bulleted List

In HTML Unordered list, all the list items are marked with bullets. It is also known as bulleted list also. The Unordered list starts with <ul> tag and list items start with the <li> tag.

HTML Unordered Lists

An unordered list is a collection of related items that have no special order or sequence. This list is created by using HTML **<ul>** tag. Each item in the list is marked with a bullet.

The type Attribute

You can use **type** attribute for <ul> tag to specify the type of bullet you like. By default, it is a disc. Following are the possible options −

```
<ul type = "square">
<ul type = "disc">
<ul type = "circle">
```

The list items will be marked with bullets (small black circles) by default:

Example

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

Ordered HTML List

An ordered list starts with the <ol> tag. Each list item starts with the <li> tag.

HTML Ordered List or Numbered List

In the ordered HTML lists, all the list items are marked with numbers by default. It is known as numbered list also. The ordered list starts with <ol> tag and the list items start with <li> tag.

. HTML Ordered Lists

If you are required to put your items in a numbered list instead of bulleted, then HTML ordered list will be used. This list is created by using **<ol>** tag. The numbering starts at one and is incremented by one for each successive ordered list element tagged with <li>.

The type Attribute

You can use **type** attribute for <ol> tag to specify the type of numbering you like. By default, it is a number. Following are the possible options −

```
<ol type = "1"> - Default-Case Numerals.
<ol type = "I"> - Upper-Case Numerals.
<ol type = "i"> - Lower-Case Numerals.
<ol type = "A"> - Upper-Case Letters.
<ol type = "a"> - Lower-Case Letters.
```

The start Attribute

You can use **start** attribute for <ol> tag to specify the starting point of numbering you need. Following are the possible options −

```
<ol type = "1" start = "4">    - Numerals starts with 4.
<ol type = "I" start = "4">    - Numerals starts with IV.
<ol type = "i" start = "4">    - Numerals starts with iv.
<ol type = "a" start = "4">    - Letters starts with d.
<ol type = "A" start = "4">    - Letters starts with D.
```

The list items will be marked with numbers by default:

Example

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

HTML Description Lists

HTML also supports description lists.

A description list is a list of terms, with a description of each term.

The <dl> tag defines the description list, the <dt> tag defines the term (name), and the <dd> tag describes each term:

HTML Description List or Definition List

HTML Description list is also a list style which is supported by HTML and XHTML. It is also known as definition list where entries are listed like a dictionary or encyclopedia.

The definition list is very appropriate when you want to present glossary, list of terms or other name-value list.

HTML Description List: A description list is a list of terms, with a description of each term. The <dl> tag defines the description list, the <dt> tag defines the term name, and the <dd> tag describes each term

HTML Definition Lists

HTML and XHTML supports a list style which is called **definition lists** where entries are listed like in a dictionary or encyclopedia. The definition list is the ideal way to present a glossary, list of terms, or other name/value list.

Definition List makes use of following three tags.

- <dl> − Defines the start of the list
- <dt> − A term
- <dd> − Term definition
- </dl> − Defines the end of the list

The HTML definition list contains following three tags:

1. **<dl> tag** defines the start of the list.
2. **<dt> tag** defines a term.
3. **<dd> tag** defines the term definition (description).

```
<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
```

```
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

HTML <ul> Tag

An unordered HTML list:

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
```

The `<ul>` tag defines an unordered (bulleted) list.

Use the `<ul>` tag together with the <li> tag to create unordered lists.

HTML <ol> Tag

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.

HTML <li> Tag

The `<li>` tag defines a list item.

The `<li>` tag is used inside ordered lists(<ol>), unordered lists (<ul>), and in menu lists (<menu>).

In <ul> and <menu>, the list items will usually be displayed with bullet points.

In <ol>, the list items will usually be displayed with numbers or letters.

HTML <dl> Tag

A description list, with terms and descriptions:

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

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).

HTML <dt> Tag

The <dt> tag defines a term/name in a description list.

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

A description list, with terms and descriptions:

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

HTML <dd> Tag

The <dd> tag is used to describe a term/name in a description list.

The <dd> tag is used in conjunction with <dl> (defines a description list) and <dt> (defines terms/names).

A description list, with terms and descriptions:

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

HTML Nested List

A list within another list is termed as nested list. If you want a bullet list inside a numbered list then such type of list will called as nested list.

```
<!DOCTYPE html>

<html>    <body>

<h2>A Nested List</h2>

<p>Lists can be nested (list inside list):</p>

 <ul>

  <li>Coffee</li>

  <li>Tea

    <ul>

      <li>Black tea</li>         <li>Green tea</li>

    </ul>

  </li>

  <li>Milk</li>

</ul>   </body>

</html>
```

- CSS stands for Cascading Style Sheets
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- CSS saves a lot of work. It can control the layout of multiple web pages all at once
- External stylesheets are stored in CSS files

Cascading Style Sheet(CSS) is used to set the style in web pages that contain HTML elements. It sets the background color, font-size, font-family, color, … etc property of elements on a web page.

Why Use CSS?

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

Applications of CSS

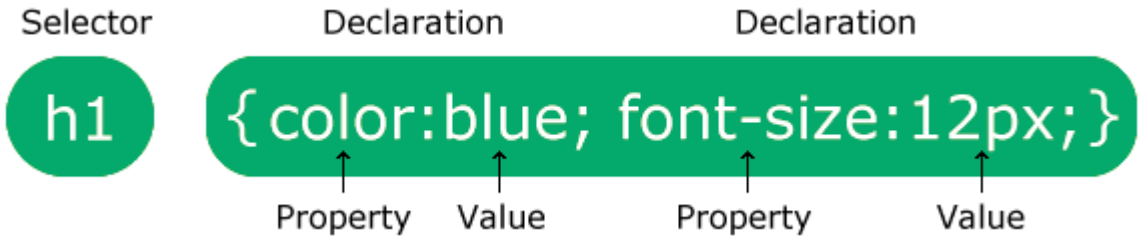
- **CSS saves time** - You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- **Pages load faster** - If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- **Easy maintenance** - To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- **Superior styles to HTML** - CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- **Multiple Device Compatibility** - Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- **Global web standards** - Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

CSS Syntax

A CSS rule consists of a selector and a declaration block.

```
selector { property: value }
```

CSS Syntax



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

```
p {
  color: red;
  text-align: center;
}
```

- `p` is a selector in CSS (it points to the HTML element you want to style: <p>).
- `color` is a property, and `red` is the property value
- `text-align` is a property, and `center` is the property value

CSS Selectors

A CSS selector selects the HTML element(s) you want to style.

CSS Selectors

CSS selectors are used to "find" (or select) the HTML elements you want to style.

We can divide CSS selectors into five categories:

- Simple selectors (select elements based on name, id, class)
- Combinator selectors (select elements based on a specific relationship between them)
- Pseudo-class selectors (select elements based on a certain state)
- Pseudo-elements selectors (select and style a part of an element)
- Attribute selectors (select elements based on an attribute or attribute value)

CSS selectors are used *to select the content you want to style*. Selectors are the part of CSS rule set. CSS selectors select HTML elements according to its id, class, type, attribute etc.

There are several different types of selectors in CSS.

1. CSS Element Selector
2. CSS Id Selector
3. CSS Class Selector
4. CSS Universal Selector
5. CSS Group Selector

1. The CSS element Selector

The element selector selects HTML elements based on the element name.

Here, all <p> elements on the page will be center-aligned, with a red text color:

```
p {
  text-align: center; color: red;
}
```

```
<!DOCTYPE html>
<html>  <head>
<style>
p {
 text-align: center;
 color: red;
}
</style>  </head>  <body>
<p>Every paragraph will be affected by the style.</p>
<p id="para1">Me too!</p>
<p>And me!</p>  </body>  </html>
```

2.The CSS id Selector

The id selector uses the id attribute of an HTML element to select a specific element.

The id of an element is unique within a page, so the id selector is used to select one unique element!

To select an element with a specific id, write a hash (#) character, followed by the id of the element.

```
#para1 {
  text-align: center;
  color: red;
}
```

Note: An id name cannot start with a number!

```
<!DOCTYPE html>
<html>  <head>
<style>
#para1 {
 text-align: center;   color: red;
}
</style> </head><body>
<p id="para1">Hello World!</p>
<p>This paragraph is not affected by the style.</p>    </body>  </html>
```

3.The CSS class Selector

The class selector selects HTML elements with a specific class attribute.

To select elements with a specific class, write a period (.) character, followed by the class name.

```
.center {
  text-align: center;
  color: red;
}
```

You can also specify that only specific HTML elements should be affected by a class.

```
p.center {
  text-align: center;
  color: red;
}
```

Note: A class name cannot start with a number!

<!DOCTYPE html>

<html>

<head>

<style>

.center {

  text-align: center;

  color: red;

}

</style>

</head>

<body>

<h1 class="center">Red and center-aligned heading</h1>

Red and center-aligned paragraph.

</body>

</html>

4.The CSS Universal Selector

The universal selector (*) selects all HTML elements on the page.

Example

The CSS rule below will affect every HTML element on the page:

```
* {
  text-align: center;
  color: blue;
}
```

<!DOCTYPE html> <html><head>

<style> * { text-align: center; color: blue; }

</style> </head> <body>

<h1>Hello world!</h1>

Every element on the page will be affected by the style.

Me too! And me!

</body></html>

The CSS Grouping Selector

The grouping selector selects all the HTML elements with the same style definitions.

```
h1 {
  text-align: center;
```

```
  color: red;
}

h2 {
  text-align: center;
  color: red;
}

p {
  text-align: center;
  color: red;
}
```

It will be better to group the selectors, to minimize the code.

To group selectors, separate each selector with a comma.

Example

In this example we have grouped the selectors from the code above:

```
h1, h2, p {
  text-align: center;
  color: red;
}
```

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

- External CSS
- Internal CSS
- Inline CSS

External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

Example

External styles are defined within the <link> element, inside the <head> section of an HTML page:

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

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

an external style sheet can be written in any text editor, and must be saved with a .css extension.

The external .css file should not contain any HTML tags.

Here is how the "mystyle.css" file looks:

"mystyle.css"

```
body {
  background-color: lightblue;
}

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

Note: Do not add a space between the property value and the unit (such as `margin-left: 20 px;`). The correct way is: `margin-left: 20px;`

Internal CSS

An internal style sheet may be used if one single HTML page has a unique style.

The internal style is defined inside the <style> element, inside the head section.

Example

Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-color: linen;
}

h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
</head>
<body>

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

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

Inline CSS

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

Example

Inline styles are defined within the "style" attribute of the relevant element:

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

<h1 style="color:blue;text-align:center;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>

</body>
</html>
```

Cascading Order

What style will be used when there is more than one style specified for an HTML element?

All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

1. Inline style (inside an HTML element)
2. External and internal style sheets (in the head section)
3. Browser default

So, an inline style has the highest priority, and will override external and internal styles and browser defaults.

Embedded CSS - The <style> Element

You can put your CSS rules into an HTML document using the <style> element. This tag is placed inside the <head>...</head> tags. Rules defined using this syntax will be applied to all the elements available in the document. Here is the generic syntax –

```
<!DOCTYPE html>
<html>
   <head>
      <style type = "text/css" media = "all">
         body {
            background-color: linen;
         }
         h1 {
            color: maroon;
            margin-left: 40px;
         }
      </style>
   </head>
   <body>
      <h1>This is a heading</h1>
      <p>This is a paragraph.</p>
   </body>
</html>
```

Inline CSS ***Advantages:***
Inline CSS can be used for many purposes, some of which include:

1. **Testing:** Many web designers use Inline CSS when they begin working on new projects, this is because its easier to scroll up in the source, rather than change the source file. Some also using it to debug their pages, if they encounter a problem which is not so easily fixed. This can be done in combination with the Important rule of CSS.

2. **Quick-fixes:**There are times where you would just apply a direct fix in your HTML source, using the style attribute, but you would usually move the fix to the relevant files when you are either able, or got the time.

3. **Smaller Websites:** The website such as Blogs where there are only limited number of pages, using of Inline CSS helps users and service provider.

4. **Lower the HTTP Requests**: The major benefit of using Inline CSS is lower HTTP Requests which means the website loads faster than External CSS.

Disadvantages
Inline CSS some of the disadvantages of which includes:

1. **Overriding**: Because they are the most specific in the cascade, they can over-ride things you didn't intend them to.

2. **Every Element**: Inline styles must be applied to every element you want them on. So if you want all your paragraphs to have the font family "Arial" you have to add an inline style to each <p> tag in your document. This adds both maintenance work for the designer and download time for the reader.

3. **Pseudo-elements**: It's impossible to style pseudo-elements and classes with inline styles. For example, with external and internal style sheets, you can style the visited, hover, active, and link color of an anchor tag. But with an inline style all you can style is the link itself, because that's what the style attribute is attached to.

Internal CSS
Advantages

Since the Internal CSS have more preference over Inline CSS. There are numerous advantages of which some of important are an under:

1. **Cache Problem**: Internal styles will be read by all browsers unless they are hacked to hide from certain ones. This removes the ability to use media=all or @import to hide styles from old, crotchety browsers like IE4 and NN4.

2. **Pseudo-elements**: It's impossible to style pseudo-elements and classes with inline styles. With Internal style sheets, you can style the visited, hover, active, and link color of an anchor tag.

3. **One style of same element**: Internal styles need not be applied to every element. So if you want all your paragraphs to have the font family "Arial" you have to add an Inline style <p> tag in Internal Style document.

4. **No additional downloads**: No additional downloads necessary to receive style information or we have less HTTP Request

Disadvantages

1. **Multiple Documents**: This method can't be used, if you want to use it on multiple web pages.
2. **Slow Page Loading**: As there are less HTTP Request but by using the Internal CSS the page load slow as compared to Inline and External CSS.
3. **Large File Size:** While using the Internal CSS the page size increases but it helps only to Designers while working offline but when the website goes online it consumers much time as compared to offline.

External CSS
Advantages

There are many advantages for using external CSS and some of are:

A. **Full Control of page structure**:

B. **Reduced file-size**:

C. **Less load time**:

D. **Higher page ranking** :

CSS Box Model

ll HTML elements can be considered as boxes.

The CSS Box Model

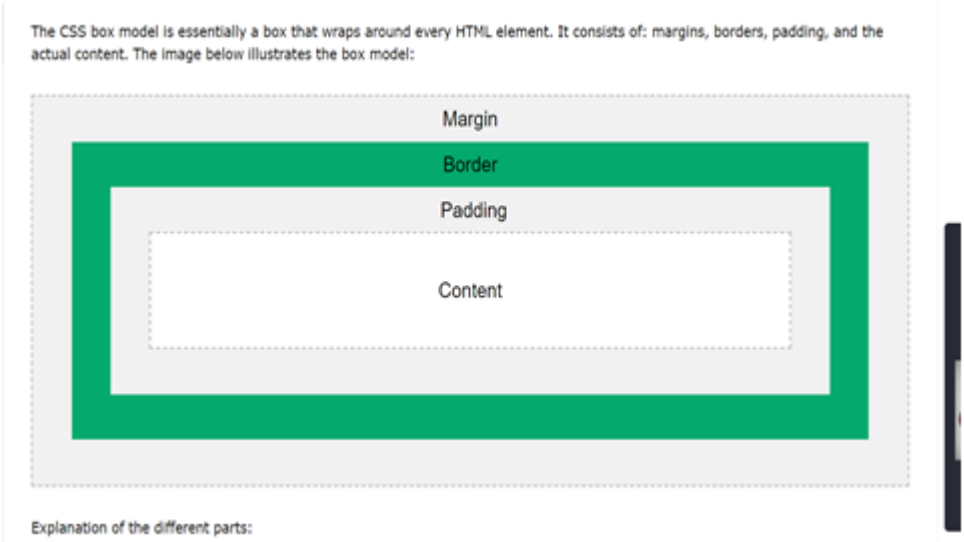
In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:

Explanation of the different parts:

- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

The box model allows us to add a border around elements, and to define space between elements.



Width and Height of an Element

In order to set the width and height of an element correctly in all browsers, you need to know how the box model works.

Important: When you set the width and height properties of an element with CSS, you just set the width and height of the **content area**. To calculate the full size of an element, you must also add padding, borders and margins.

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 320px;
  padding: 10px;
  border: 5px solid gray;
  margin: 0;
}
</style>
</head>
<body>

<h2>Calculate the total width:</h2>

<img src="klematis4_big.jpg" width="350" height="263" alt="Klematis">
<div>The picture above is 350px wide. The total width of this element is
also 350px.</div>

</body>
</html>
```

Here is the calculation:

320px (width)
+ 20px (left + right padding)
+ 10px (left + right border)
+ 0px (left + right margin)
= 350px

The total width of an element should be calculated like this:

Total element width = width + left padding + right padding + left border + right border + left margin + right margin

The total height of an element should be calculated like this:

Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

CSS Borders

The CSS border properties allow you to specify the style, width, and color of an element's border.

CSS Border Style

The `border-style` property specifies what kind of border to display.

The following values are allowed:

- `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

The `border-style` property can have from one to four values (for the top border, right border, bottom border, and the left border).

```
<!DOCTYPE html>
<html>
<head>
<style>
p.dotted {border-style: dotted;}
p.dashed {border-style: dashed;}
p.solid {border-style: solid;}
p.double {border-style: double;}
p.groove {border-style: groove;}
p.ridge {border-style: ridge;}
p.inset {border-style: inset;}
p.outset {border-style: outset;}
p.none {border-style: none;}
p.hidden {border-style: hidden;}
p.mix {border-style: dotted dashed solid double;}
</style>
</head>
<body>
```

```
<h2>The border-style Property</h2>
<p>This property specifies what kind of border to display:</p>

<p class="dotted">A dotted border.</p>
<p class="dashed">A dashed border.</p>
<p class="solid">A solid border.</p>
<p class="double">A double border.</p>
<p class="groove">A groove border.</p>
<p class="ridge">A ridge border.</p>
<p class="inset">An inset border.</p>
<p class="outset">An outset border.</p>
<p class="none">No border.</p>
<p class="hidden">A hidden border.</p>
<p class="mix">A mixed border.</p>

</body>
</html>
```

CSS Border Width

The `border-width` property specifies the width of the four borders.

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:

Example

Demonstration of the different border widths:

```
p.one {
  border-style: solid;
  border-width: 5px;
}

p.two {
  border-style: solid;
  border-width: medium;
}

p.three {
  border-style: dotted;
  border-width: 2px;
}

p.four {
  border-style: dotted;
  border-width: thick;
```

CSS 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)"
- HSL - specify a HSL value, like "hsl(0, 100%, 50%)"
- transparent

Note: If `border-color` is not set, it inherits the color of the element.

Example

Demonstration of the different border colors:

```
p.one {
  border-style: solid;
  border-color: red;
}

p.two {
  border-style: solid;
  border-color: green;
}

p.three {
  border-style: dotted;
  border-color: blue;
```

CSS Margins

Margins are used to create space around elements, outside of any defined borders.

CSS Margins

The CSS `margin` properties are used to create space around elements, outside of any defined borders.

With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

Margin - Individual Sides

CSS has properties for specifying the margin for each side of an element:

- `margin-top`
- `margin-right`
- `margin-bottom`
- `margin-left`

All the margin properties can have the following values:

- auto - the browser calculates the margin
- *length* - specifies a margin in px, pt, cm, etc.
- *%* - specifies a margin in % of the width of the containing element
- inherit - specifies that the margin should be inherited from the parent element

Tip: Negative values are allowed.

Example

Set different margins for all four sides of a <p> element:

```
p {
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 80px;
}
```

Margin - Shorthand Property

To shorten the code, it is possible to specify all the margin properties in one property.

The `margin` property is a shorthand property for the following individual margin properties:

- `margin-top`
- `margin-right`
- `margin-bottom`
- `margin-left`

So, here is how it works:

If the `margin` property has four values:

- **margin: 25px 50px 75px 100px;**
  - top margin is 25px
  - right margin is 50px
  - bottom margin is 75px
  - left margin is 100px

Example

Use the margin shorthand property with four values:

```
p {
  margin: 25px 50px 75px 100px;
}
```

```
<!DOCTYPE html>

<html>

<head>

<style>

div {

  border: 1px solid black;

  margin-top: 100px;

  margin-bottom: 100px;

  margin-right: 150px;

  margin-left: 80px;

  background-color: lightblue;

}

</style>

</head>

<body>


<h2>Using individual margin properties</h2>
```

This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.

</body>

</html>

If the `margin` property has three values:

- **margin: 25px 50px 75px;**
  - top margin is 25px
  - right and left margins are 50px
  - bottom margin is 75px

Example

Use the margin shorthand property with three values:

```
p {
  margin: 25px 50px 75px;
}
```

If the `margin` property has two values:

- **margin: 25px 50px;**
  - top and bottom margins are 25px
  - right and left margins are 50px

Example

Use the margin shorthand property with two values:

```
p {
  margin: 25px 50px;
}
```

If the `margin` property has one value:

- **margin: 25px;**
  - all four margins are 25px

Example

Use the margin shorthand property with one value:

```
p {
  margin: 25px;
}
```

The auto Value

You can set the margin property to `auto` to horizontally center the element within its container.

The element will then take up the specified width, and the remaining space will be split equally between the left and right margins.

Example

Use margin: auto:

```
div {
  width: 300px;
  margin: auto;
  border: 1px solid red;
}
```

CSS Padding

Padding is used to create space around an element's content, inside of any defined borders.

The CSS `padding` properties are used to generate space around an element's content, inside of any defined borders.

With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

Padding - Individual Sides

CSS has properties for specifying the padding for each side of an element:

- `padding-top`

- padding-right
- padding-bottom
- padding-left

All the padding properties can have the following values:

- *length* - specifies a padding in px, pt, cm, etc.
- % - specifies a padding in % of the width of the containing element
- inherit - specifies that the padding should be inherited from the parent element

Note: Negative values are not allowed.

Example

Set different padding for all four sides of a element:

```
div {
  padding-top: 50px;
  padding-right: 30px;
  padding-bottom: 50px;
  padding-left: 80px;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  border: 1px solid black;
  background-color: lightblue;
  padding-top: 50px;
  padding-right: 30px;
  padding-bottom: 50px;
  padding-left: 80px;
}
</style>
</head>
<body>

<h2>Using individual padding properties</h2>

<div>This div element has a top padding of 50px, a right padding of 30px,
a bottom padding of 50px, and a left padding of 80px.</div>

</body>
</html>
```

CSS background Property

Set different background properties in one declaration:

The `background` property is a shorthand property for:

- background-color
- background-image
- background-position
- background-size
- background-repeat
- background-origin
- background-clip
- background-attachment

CSS background-color Property

Set the background color for a page:

The `background-color` property sets the background color of an element.

The background of an element is the total size of the element, including padding and border (but not the margin).

```
body {background-color: coral;}
```

CSS background-image Property

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.

Set two background images for the <body> element:

```
body {
```

```
  background-image: url("img_tree.gif"), url("paper.gif");
  background-color: #cccccc;
}
```

CSS background-position Property

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

Tip: By default, a background-image is placed at the top-left corner of an element, and repeated both vertically and horizontally.

```
body {
  background-image: url('w3css.gif');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-position: center;
}
```

Values
left top
left center
left bottom
right top
right center
right bottom
center top
center center
center bottom

CSS background-repeat Property

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

Repeat a background-image only vertically:

```
body {
  background-image: url("paper.gif");
  background-repeat: repeat-y;  }
```

CSS background-attachment Property

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

```
body {
  background-image: url("img_tree.gif");
  background-repeat: no-repeat;
  background-attachment: fixed;
}
CSS Syntax:
background-attachment: scroll|fixed|local|initial|inherit;
```

CSS Text

CSS has a lot of properties for formatting text.

Text Color

The `color` property is used to set the color of the text. The color is specified by:

- a color name - like "red"
- a HEX value - like "#ff0000"
- an RGB value - like "rgb(255,0,0)"

```
body {
  color: blue;
}
```

```
h1 {
  color: green;
}
```

Text Color and Background Color

In this example, we define both the `background-color` property and the `color` property:

```
h1 {
  background-color: black;
  color: white;
}
```

CSS text-align Property

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

A text can be left or right aligned, centered, or justified.

```
h1 {
  text-align: center;
}

div.a {
  text-align: center;
}

//<div class="a">
```

CSS Text Decoration

Text Decoration

The `text-decoration` property is used to set or remove decorations from text.

The value `text-decoration: none;` is often used to remove underlines from links:

The other `text-decoration` values are used to decorate text:

```
h2 {
  text-decoration: overline;
}

h3 {
  text-decoration: line-through;
}

h4 {
  text-decoration: underline;
}
h5 {
  text-decoration: underline overline dotted red;
}
```

CSS Text Transformation

The `text-transform` property is used to specify uppercase and lowercase letters in a text.

It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word:

```
p.uppercase {
  text-transform: uppercase;
}
p.lowercase {
  text-transform: lowercase;
}



p.capitalize {
  text-transform: capitalize;
}
```

CSS Text Spacing

The `text-indent` property is used to specify the indentation of the first line of a text:

```
p {
  text-indent: 50px;
}
```

Letter Spacing

The `letter-spacing` property is used to specify the space between the characters in a text.

The following example demonstrates how to increase or decrease the space between characters:

```
h1 {

  letter-spacing: 5px;

}
```

Word Spacing

The `word-spacing` property is used to specify the space between the words in a text.

The following example demonstrates how to increase or decrease the space between words:

```
p.one {

  word-spacing: 10px;

}
```

CSS Fonts

Choosing the right font for your website is important!

Generic Font Families

In CSS there are five generic font families:

1. Serif fonts have a small stroke at the edges of each letter. They create a sense of formality and elegance.
2. Sans-serif fonts have clean lines (no small strokes attached). They create a modern and minimalistic look.
3. Monospace fonts - here all the letters have the same fixed width. They create a mechanical look.
4. Cursive fonts imitate human handwriting.
5. Fantasy fonts are decorative/playful fonts.

All the different font names belong to one of the generic font families.

The CSS font-family Property

In CSS, we use the `font-family` property to specify the font of a text.

Note: If the font name is more than one word, it must be in quotation marks, like: "Times New Roman".

Tip: The `font-family` property should hold several font names as a "fallback" system, to ensure maximum compatibility between browsers/operating systems. Start with the font you want, and end with a generic family (to let the browser pick a similar font in the generic family, if no other fonts are available). The font names should be separated with comma

Font Style

The `font-style` property is mostly used to specify italic text.

This property has three values:

- normal - The text is shown normally
- italic - The text is shown in italics
- oblique - The text is "leaning" (oblique is very similar to italic, but less supported)

```
p.normal {
  font-style: normal;
}

p.italic {
  font-style: italic;
}

p.oblique {
  font-style: oblique;
}
```

Font Weight

The `font-weight` property specifies the weight of a font:

```
p.normal {
  font-weight: normal;
}

p.thick {
  font-weight: bold;
}
```

Font Variant

The `font-variant` property specifies whether or not a text should be displayed in a small-caps font.

In a small-caps font, all lowercase letters are converted to uppercase letters. However, the converted uppercase letters appears in a smaller font size than the original uppercase letters in the text.

```
p.normal {
  font-variant: normal;
}

p.small {
  font-variant: small-caps;
}
```

CSS Font Size

Font Size

The `font-size` property sets the size of the text.

Being able to manage the text size is important in web design. However, you should not use font size adjustments to make paragraphs look like headings, or headings look like paragraphs.

Always use the proper HTML tags, like <h1> - <h6> for headings and <p> for paragraphs.

The font-size value can be an absolute, or relative size.

Absolute size:

- Sets the text to a specified size
- Does not allow a user to change the text size in all browsers (bad for accessibility reasons)
- Absolute size is useful when the physical size of the output is known

Relative size:

- Sets the size relative to surrounding elements
- Allows a user to change the text size in browsers

Note: If you do not specify a font size, the default size for normal text, like paragraphs, is 16px (16px=1em).

```
h1 {
  font-size: 40px;
}

h2 {
  font-size: 30px;
}
```

Responsive Font Size

The text size can be set with a `vw` unit, which means the "viewport width".

That way the text size will follow the size of the browser window:

```
<h1 style="font-size:10vw">Hello World</h1>
```

CSS Lists

HTML Lists and CSS List Properties

In HTML, there are two main types of lists:

- unordered lists (<ul>) - the list items are marked with bullets
- ordered lists (<ol>) - the list items are marked with numbers or letters

The CSS list properties allow you to:

- Set different list item markers for ordered lists
- Set different list item markers for unordered lists
- Set an image as the list item marker
- Add background colors to lists and list items

Different List Item Markers

The `list-style-type` property specifies the type of list item marker.

```
<!DOCTYPE html>
<html>
<head>
<style>
ul.a {
  list-style-type: circle;
}

ul.b {
  list-style-type: square;
}

ol.c {
  list-style-type: upper-roman;
}

ol.d {
  list-style-type: lower-alpha;
}
</style>
</head>
<body>

<h2>The list-style-type Property</h2>

<p>Example of unordered lists:</p>
<ul class="a">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>

<ul class="b">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>

<p>Example of ordered lists:</p>
<ol class="c">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>
```

```
<ol class="d">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>

</body>
</html>
```

The `list-style` property is a shorthand for the following properties:

- list-style-type
- list-style-position
- list-style-image

If one of the values are missing, the default value for that property will be used.

The `list-style-image` property specifies an image as the list item marker:

```
ul {
  list-style-image: url('sqpurple.gif');
}
```

CSS list-style-position Property

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

"list-style-position: outside;" means that the bullet points will be outside the list item. The start of each line of a list item will be aligned vertically. This is default:

• Coffee - A brewed drink prepared from roasted coffee beans...
• Tea
• Coca-cola

"list-style-position: inside;" means that the bullet points will be inside the list item. As it is part of the list item, it will be part of the text and push the text at the start:

- Coffee - A brewed drink prepared from roasted coffee beans...
- Tea
- Coca-cola

Example

```
ul.a {
  list-style-position: outside;
}
ul.b {
  list-style-position: inside;
}
```

The `list-style-type:none` property can also be used to remove the markers/bullets. Note that the list also has default margin and padding. To remove this, add `margin:0` and `padding:0` to <ul> or <ol>:

```
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}
```

CSS list-style-type Property

Set some different list styles:

```
ul.a {list-style-type: circle;}
ul.b {list-style-type: square;}
ol.c {list-style-type: upper-roman;}
ol.d {list-style-type: lower-alpha;}
```

List - Shorthand property

The `list-style` property is a shorthand property. It is used to set all the list properties in one declaration:

```
ul {
  list-style: square inside url("sqpurple.gif");
}
```

CSS Tables

Table Borders

To specify table borders in CSS, use the `border` property.

The example below specifies a black border for <table>, <th>, and <td> elements:

```
table, th, td {
  border: 1px solid black;
}
```

Full-Width Table

The table above might seem small in some cases. If you need a table that should span the entire screen (full-width), add `width: 100%` to the <table> element:

```
table {
  width: 100%;
}
```

Collapse Table Borders

The `border-collapse` property sets whether the table borders should be collapsed into a single border:

Set the collapsing borders model for two tables:

```
table {
  border-collapse: collapse;
}
```

CSS border-spacing Property

The `border-spacing` property sets the distance between the borders of adjacent cells.This property works only when border-collapse is separate.

```
#table1 {
  border-collapse: separate;
  border-spacing: 15px;
}
```

CSS Table Size

Table Width and Height

The width and height of a table are defined by the `width` and `height` properties.

The example below sets the width of the table to 100%, and the height of the <th> elements to 70px:

```
table {
  width: 100%;
}

th {
  height: 70px;
}
```

To create a table that should only span half the page, use `width: 50%`:

Horizontal Alignment

The `text-align` property sets the horizontal alignment (like left, right, or center) of the content in <th> or <td>.

By default, the content of <th> elements are center-aligned and the content of <td> elements are left-aligned.

To center-align the content of <td> elements as well, use `text-align: center`:

```
td {
  text-align: center;
}
```

Vertical Alignment

The `vertical-align` property sets the vertical alignment (like top, bottom, or middle) of the content in <th> or <td>.

By default, the vertical alignment of the content in a table is middle (for both <th> and <td> elements).

The following example sets the vertical text alignment to bottom for <td> elements:

```
td {
  height: 50px;
  vertical-align: bottom;
}
```

Table Padding

To control the space between the border and the content in a table, use the `padding` property on <td> and <th> elements:

```
th, td {
  padding: 15px;
  text-align: left;
}
```

Add the `border-bottom` property to <th> and <td> for horizontal dividers:

```
th, td {
  border-bottom: 1px solid #ddd;
}
```

Hoverable Table

Use the `:hover` selector on <tr> to highlight table rows on mouse over:

```
tr:hover {background-color: yellow;}
```

Table Color

The example below specifies the background color and text color of <th> elements:

```
th {
  background-color: #04AA6D;
  color: white;
```

Responsive Table

}Add a container element (like) with `overflow-x:auto` around the <table> element to make it responsive:

Example

```
<div style="overflow-x:auto;">

<table>

... table content ...

</table>

</div>
```

CSS Layout - The display Property

he `display` property is the most important CSS property for controlling layout.

The display Property

The `display` property specifies if/how an element is displayed.

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is `block` or `inline`.

Block-level Elements

A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

The element is a block-level element.

Examples of block-level elements:

-
- <h1> - <h6>
- <p>
- <form>
- <header>
- <footer>
- <section>

Inline Elements

An inline element does not start on a new line and only takes up as much width as necessary.

This is an inline element inside a paragraph.

Examples of inline elements:

-
- <a>
- <img>

Display: none;

`display: none;` is commonly used with JavaScript to hide and show elements without deleting and recreating them. Take a look at our last example on this page if you want to know how this can be achieved.

The `<script>` element uses `display: none;` as default.

Override The Default Display Value

As mentioned, every element has a default display value. However, you can override this.

Changing an inline element to a block element, or vice versa, can be useful for making the page look a specific way, and still follow the web standards.

A common example is making inline `<li>` elements for horizontal menus:

```
li {
  display: inline;
}
```

CSS Layout - The position Property

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

The position Property

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

There are five different position values:

- `static`
- `relative`

- `fixed`
- `absolute`
- `sticky`

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

---

position: static;

HTML elements are positioned static by default.

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

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

This element has position: static;

Here is the CSS that is used:

Example

```
div.static {
  position: static;
  border: 3px solid #73AD21;
}
```

position: relative;

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

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

This element has position: relative;

Here is the CSS that is used:

```
div.relative {
  position: relative;
  left: 30px;
  border: 3px solid #73AD21;
}
```

position: fixed;

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

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

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

```
div.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
```

position: absolute;

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

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

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

position: sticky;

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

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

```
div.sticky {
  position: -webkit-sticky; /* Safari */
  position: sticky;
  top: 0;
  background-color: green;
  border: 2px solid #4CAF50;
}
```

CSS Layout - float and clear

The CSS `float` property specifies how an element should float.

The CSS `clear` property specifies what elements can float beside the cleared element and on which side.

The float Property

The `float` property is used for positioning and formatting content e.g. let an image float left to the text in a container.

The `float` property can have one of the following values:

- `left` - The element floats to the left of its container
- `right` - The element floats to the right of its container
- `none` - The element does not float (will be displayed just where it occurs in the text). This is default
- `inherit` - The element inherits the float value of its parent

In its simplest use, the `float` property can be used to wrap text around images.

```
img {
  float: right;
```

```
}

img {
  float: left;
}

img {
  float: none;
}
```

Example - Float Next To Each Other

Normally div elements will be displayed on top of each other. However, if we use `float: left` we can let elements float next to each other:

```
div {
  float: left;
  padding: 15px;
}

.div1 {
  background: red;
}
```

CSS Pseudo-classes

A Pseudo class in CSS is used to define the special state of an element. It can be combined with a CSS selector to add an effect to existing elements based on their states. For Example, changing the style of an element when the user hovers over it, or when a link is visited. All of these can be done using Pseudo Classes in CSS.

A pseudo-class is used to define a special state of an element.

For example, it can be used to:

- Style an element when a user mouses over it
- Style visited and unvisited links differently
- Style an element when it gets focus

Syntax

The syntax of pseudo-classes:

```
selector:pseudo-class {

  property: value;

}
```

pseudo-class	Description
:active	IIt is used to add style to an active element.
:hover	IIt adds special effects to an element when the user moves the mouse pointer over the element.
:link	IIt adds style to the unvisited link.
:visited	IIt adds style to a visited link.
:focus	IIt selects the element which is focused by the user currently.
:first-child	It adds special effects to an element, which is the first child of another element.

:hover pseudo-class

This pseudo-class adds a special style to an element when the user moves the cursor over it. If you want to make it effective, it must be applied after the "**:link**" and "**:visited**" pseudo-classes.

:active pseudo-class

It applies when the elements are clicked or activated. It selects the activated element.

We can understand it with the example given below.

Example

```
<!DOCTYPE html>
<html>
<head>
   <style>
   body{
   text-align:center;
   }
   a:active{
      color: yellow;
   }
   h1, h2, h3{
      color:red; ;
   }
   </style>
</head>

<body>
   <h1>Hello World</h1>
   <h2>The :active pseudo-class</h2>
   <h3>Click the following link to see the effect</h3>
   <a href="#">Click the link</a>
</body>
</html>
```

:visited pseudo-class

It selects the visited links and adds special styles to them. Its possible values can be any color name in a valid format.

Example

```
<!DOCTYPE html>
<html>
<head>
   <style>
   body{
   text-align:center;
   }
   a:visited{
```

```
        color: red;
    }

    </style>
</head>

<body>
    <h1>Hello World</h1>
    <h2>The :visited pseudo-class</h2>
    <h3>Click the following link to see the effect</h3>
    <a href="#">Click the link</a>
</body>
</html>
```

:focus pseudo-class

It selects the elements that are currently focused on by the user. It is generally used in input elements of the forms and triggers when the user clicks on it.

Example

```
<!DOCTYPE HTML>
<html>
 <style>
 form{
 text-align:center;
 }
 input:focus{
        border:5px solid lightblue;
        box-shadow:10px 10px 10px black;
        color: blue;
        width:300px;
        }
  </style>
 <body>
  <form>
    <h1>Name: <input type="text" value="Enter your name"></h1>
  </form>
</body>
</html>
```

:first-child pseudo-class

It matches a particular element, which is the first child of another element and adds a special effect to the corresponding element.

Note: We have to declare a <!DOCTYPE> at the top of the document to make ":first-child" pseudo-class to work in IE8 and its earlier versions.

The following illustration explains it more clearly.

Example

```
<!DOCTYPE HTML>
<html>
  <head>
    <style>
       h1:first-child {
         text-indent: 200px;
   color:blue;
       }
    </style>
  </head>

  <body>

    <div>
      <h1>It is the first heading in div. It will be indented, and its color will be blue.</h1>
      <h1>It is the Second heading in div, which will not be affected.</h1>
    </div>
  </body>
</html>
```

The simple tooltip hover

A tooltip specifies extra information about something when the user moves the cursor over the element. Let's create a tooltip by using the **":hover"** pseudo-class.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
text-align:center;
}
h2{
  display: none;
  background-color: red;
  color:white;
  padding: 20px;
}
div{
font-size:40px;
}
div:hover h2 {
  display: block;
```

```
}
</style>
</head>
<body>
<h1>Move your mouse to the below text to see the effect</h1>
<div>Hello World
  <h2>Welcome to the javaTpoint</h2>
</div>

</body>
</html>
```

CSS classes and pseudo-classes

The classes in CSS can be combined with **pseudo-classes.** We can write it as-

Syntax

```
selector.class: pseudo-class {
  property: value;
}
```

We can understand it with the following example.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body{
text-align:center;
}
div.hello:hover {
  color: red;
  font-size:40px;
}
</style>
</head>
<body>
<h1>CSS Classes and pseudo-classes</h1>
<h2>Move your cursor to the below text</h2>
<div class="hello">Hello World</p>

</body>
</html>
```

Anchor Pseudo-classes

Links can be displayed in different ways:

```
<!DOCTYPE html>
<html>
<head>
<style>
/* unvisited link */
a:link {
  color: red;
}

/* visited link */
a:visited {
  color: green;
}

/* mouse over link */
a:hover {
  color: hotpink;
}

/* selected link */
a:active {
  color: blue;
}
</style>
</head>
<body>

<p><b><a href="default.asp" target="_blank">This is a link</a></b></p>
```

Pseudo-classes and HTML Classes

Pseudo-classes can be combined with HTML classes:

When you hover over the link in the example, it will change color:

```
<style>
a.highlight:hover {
  color: #ff0000;
  font-size: 22px;
}
</style>
</head>
<body>

<h2>Pseudo-classes and HTML Classes</h2>

<p>When you hover over the first link below, it will change color and font size:</p>

<p><a class="highlight" href="css_syntax.asp">CSS Syntax</a></p>

<p><a href="default.asp">CSS Tutorial</a></p>

</body>
</html>
```

Hover on

An example of using the `:hover` pseudo-class on a element:

Example

```
div:hover {
  background-color: blue;
}
```

CSS - The :first-child Pseudo-class

The `:first-child` pseudo-class matches a specified element that is the first child of another element.

Match the first <p> element

In the following example, the selector matches any <p> element that is the first child of any element:

Example

```
p:first-child {
  color: blue;
}
```

Match the first <i> element in all <p> elements

In the following example, the selector matches the first <i> element in all <p> elements:

Example

```
p i:first-child {
  color: blue;
}
```

Match all <i> elements in all first child <p> elements

In the following example, the selector matches all <i> elements in <p> elements that are the first child of another element:

Example

```
p:first-child i {
  color: blue;
}
```