WEB TECHNOLOGY

MI | AASC

MODULE-I

1.1.1. Introduction to Internet

1.1.1.1. What is the 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 fibres 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.

- It is believed that the internet was developed by "Defence Advanced Projects Agency" (DARPA) department of the United States. And, it was first connected in 1969.

1.1.1.2. Uses of the internet

Generally speaking, the Internet may be used to exchange information with people all over the world, communicate across great distances, and locate information or answers fast on almost any subject.

Here are some examples of specific uses for the Internet:

- Using social media and content sharing.
- Instant messaging, video conferencing, Internet Relay Chat (IRC), Internet telephony, and email are all examples of electronic communication. These all are used through the Internet.
- Access to online degree programs, courses, and workshops for education and self-improvement.
- Searching for jobs: To advertise available positions, submit job applications, and hire candidates identified on social networking sites like LinkedIn, both employers and applicants use the Internet.

1.1.1.3. Other examples include:

- Online dating
- Online gaming
- Research
- Reading electronic newspapers and magazines
- Online shopping or e-commerce.
- Online discussion groups and forums

1.1.1.4. Advantages of the Internet:

- **Instant Messaging:** You can send messages or communicate to anyone using internet, such as email, voice chat, video conferencing, etc.
- **Get directions:** Using GPS technology, you can get directions to almost every place in a city, country, etc. You can find restaurants, malls, or any other service near your location.
- **Online Shopping:** It allows you to shop online such as you can be clothes, shoes, book movie tickets, railway tickets, flight tickets, and more.
- **Pay Bills:** You can pay your bills online, such as electricity bills, gas bills, college fees, etc.
- **Online Banking:** It allows you to use internet banking in which you can check your balance, receive or transfer money, get a statement, request cheque-book, etc.
- **Online Selling:** You can sell your products or services online. It helps you reach more customers and thus increases your sales and profit.

- **Work from Home:** In case you need to work from home, you can do it using a system with internet access. Today, many companies allow their employees to work from home.
- **Entertainment:** You can listen to online music, watch videos or movies, play online games.
- **Cloud computing:** It enables you to connect your computers and internet-enabled devices to cloud services such as cloud storage, cloud computing, etc.
- **Career building:** You can search for jobs online on different job portals and send you CV through email if required.

1.1.1.5. Disadvantages of the Internet

- **Time wastage:** Although, Internet has a lot of advantages, it also contains some limitations. Time wasting is one of among them. It can decrease your productivity if you are spending too much time on the Internet using social media apps while doing nothing. Rather than squandering time, one should use that time to do something useful and even more productive.
- **Bad impacts on health:** You can get health related issues if you spend too much time online; your body needs outside activities, exercise, and many other things. If you look at the screen for a long time, it causes negative effects on the eyes.
- **Cyber Crimes:** These days, crimes including cyber bullying, spam, viruses, hacking, and data theft are increasing day by day. Cybercriminals can quickly break into your system, which store all of your private information.
- **Effects on children:** The constant watching of videos and playing games on the Internet by young children is bad for their social and overall personality development.
- **Bullying and spreading negativity:** Social media applications have provided a free tool to all those people who regularly attempt to spread negativity with really repulsive and humiliating comments and try to bully each other, which are wrong and do bad impact on society.

1.1.2. The World Wide Web

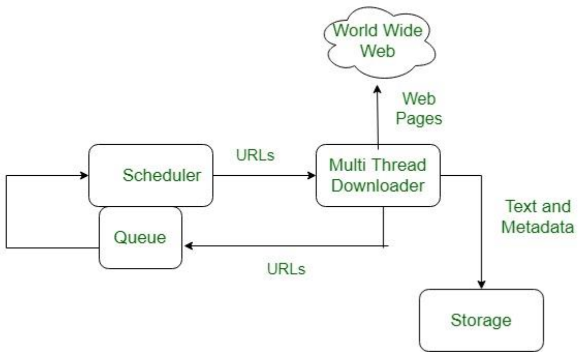
The **World Wide Web** is abbreviated as WWW and is commonly known as the web. The WWW was initiated by CERN (European library for Nuclear Research) in 1989. WWW can be defined as the collection of different websites around the world, containing different information shared via local servers (or computers).

1.1.2.1. History:

It is a project created, by Timothy Berner Lee in 1989, for researchers to work together effectively at CERN. is an organization, named the World Wide Web Consortium (W3C), which was developed for further development of the web. This organization is directed by Tim Berner's Lee, aka the father of the web.

1.1.2.2. System Architecture:

- From the user's point of view, the web consists of a vast, worldwide connection of documents or web pages. Each page may contain links to other pages anywhere in the world.
- The pages can be retrieved and viewed by using browsers of which internet explorer, Netscape Navigator, Google Chrome, etc are the popular ones. The browser fetches the page requested interprets the text and formatting commands on it, and displays the page, properly formatted, on the screen.
- The basic model of how the web works are shown in the figure below. Here the browser is displaying a web page on the client machine. When the user clicks on a line of text that is linked to a page on the abd.com server, the browser follows the hyperlink by sending a message to the abd.com server asking it for the page.



- Here the browser displays a web page on the client machine when the user clicks on a line of text that is linked to a page on abd.com; the browser follows the hyperlink by sending a message to the abd.com server asking for the page.

1.1.2.3. Working of WWW:

- The World Wide Web is based on several different technologies: Web browsers, Hypertext Mark-up Language (HTML) and Hypertext Transfer Protocol (HTTP).
- A Web browser is used to access web pages. Web browsers can be defined as programs which display text, data, pictures, animation and video on the Internet.
- Hyperlinked resources on the World Wide Web can be accessed using software interfaces provided by Web browsers. Initially, Web browsers were used only for surfing the Web but now they have become more universal.
- Web browsers can be used for several tasks including conducting searches, mailing, transferring files, and much more. Some of the commonly used browsers are Internet Explorer, Opera Mini, and Google Chrome.

1.1.2.4. Features of WWW:

- Hypertext Information System
- Cross-Platform
- Distributed
- Open Standards and Open Source
- Uses Web Browsers to provide a single interface for many services
- Dynamic, Interactive and Evolving.
- “Web 2.0”

1.1.2.5. Components of the Web:

There are 3 components of the web:

a) **Uniform Resource Locator (URL):** serves as a system for resources on the web.
b) **Hypertext Transfer Protocol (HTTP):** specifies communication of browser and server.
c) **Hyper Text Mark-up Language (HTML):** defines the structure, organisation and content of a webpage.

1.1.3. WEB BROWSER:

1.1.3.1. What is a Web Browser?

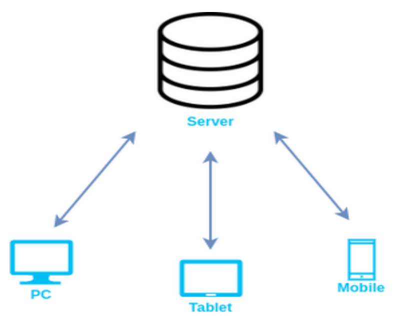
- The web browser is application software to explore www (World Wide Web). It provides an interface between the server and the client and requests to the server for web documents and services.
- It works as a compiler to render HTML which is used to design a webpage. Whenever we search for anything on 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.
- Google Chrome, Microsoft Edge, Mozilla Firefox, and Safari are examples of web browsers.

1.1.3.2. History of the Web Browser

- The first web browser Worldwide Web was invented in the year of 1990 by Tim Berners-Lee. Later, it becomes Nexus.
- In the year of 1993, a new browser Mosaic was invented by Mark Andreessen and their team. It was the first browser to display text and images at a time on the device screen. He also invents another browser Netscape in 1994.
- Next year Microsoft launched a web browser Internet Explorer which was already installed in the Windows operating system. After this many browsers were invented with various features like Mozilla Firefox, Google Chrome, Safari, Opera, etc.

1.1.3.3. How does a Web Browser Work?

- A web browser helps us find information anywhere on the internet. It is installed on the client computer and requests information from the web server such a type of working model is called a client-server model.



Client-server model

- The browser receives information through HTTP protocol. In which transmission of data is defined. When the browser received data from the server, it is rendered in HTML to user-readable form and, information is displayed on the device screen.

1.1.3.4. Website Cookies

- When we visited any website over the internet our web browser stores information about us in small files called cookies. Cookies are designed to remember stateful information about our browsing history.
- Some more cookies are used to remember about us like our interests, our browsing patterns, etc. Websites show us ads based on our interests using cookies.

1.1.3.5. Some Popular Web Browsers

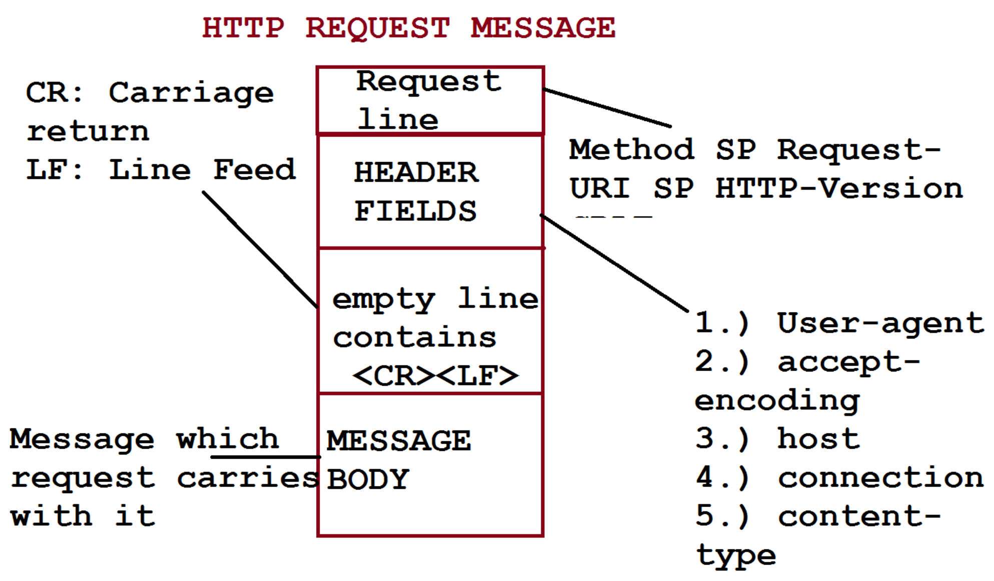
Here is a list of 7 popular web browsers:

- **Google Chrome:** Developed by Google, Chrome is one of the most widely-used web browsers in the world, known for its speed and simplicity.
- **Mozilla Firefox:** Developed by the Mozilla Foundation, Firefox is an open-source browser that is known for its privacy features and customization options.
- **Apple Safari:** Developed by Apple, Safari is the default browser on Mac and iOS devices and is known for its speed and integration with other Apple products.
- **Microsoft Edge:** Developed by Microsoft, Edge is the default browser on Windows 10 and is known for its integration with other Microsoft products and services.
- **Opera:** Developed by Opera Software, Opera is a web browser that is known for its speed and built-in VPN feature.
- **Brave:** Developed by Brave Software, Brave is a web browser that is focused on privacy and security and blocks third-party ads and trackers by default.
- **Tor Browser:** Developed by The Tor Project, Tor Browser is a web browser that is designed for anonymous web browsing and is based on Mozilla Firefox.

1.1.1. HTTP Request Message

- An HTTP client sends an HTTP request to a server in the form of a request message which includes following format:

1. **A Request-line**
2. **Zero or more header (General | Request |Entity) fields followed by CRLF ("Carriage Return" and "Line Feed.")**
3. **An empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields**
4. **Optionally a message-body**



(HTTP Request Message)

- The following sections explain each of the entities used in an HTTP request message.

1.1.1.1. Request-Line

- The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by space SP characters.

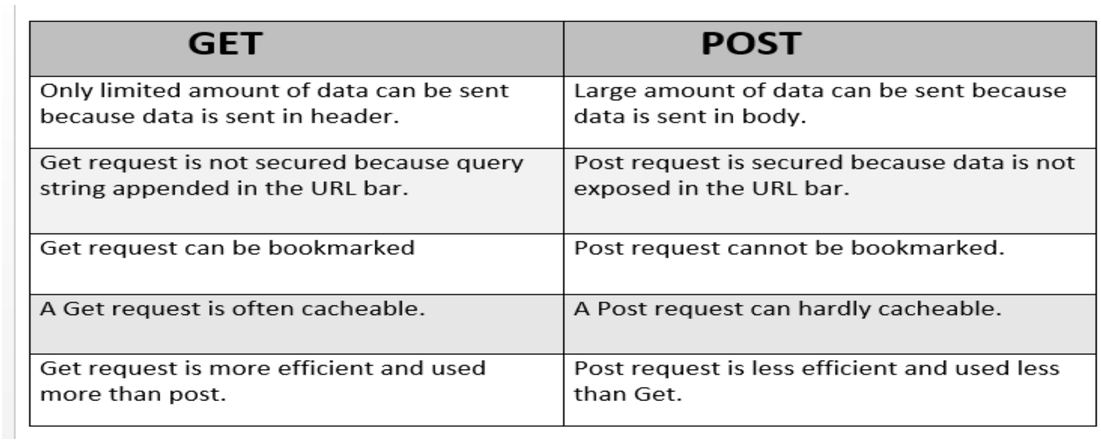
Request-Line = Method SP Request-URI SP HTTP-Version CRLF

Let's discuss each of the parts mentioned in the Request-Line.

1.1.1.1.1. Request Method:

The request **method** indicates the method to be performed on the resource identified by the given **Request-URI**. The method is case-sensitive and should always be mentioned in uppercase. The following table lists all the supported methods in HTTP/1.1.

S.N.	Method and Description
1	**GET** The GET method is used to retrieve information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data.
2	**HEAD** Same as GET, but it transfers the status line and the header section only.
3	**POST** A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.
4	**PUT** Replaces all the current representations of the target resource with the uploaded content.
5	**DELETE** Removes all the current representations of the target resource given by URI.
6	**CONNECT** Establishes a tunnel to the server identified by a given URI.
7	**OPTIONS** Describe the communication options for the target resource.
8	**TRACE** Performs a message loop back test along with the path to the target resource.



GET	POST
Only limited amount of data can be sent because data is sent in header.	Large amount of data can be sent because data is sent in body.
Get request is not secured because query string appended in the URL bar.	Post request is secured because data is not exposed in the URL bar.
Get request can be bookmarked	Post request cannot be bookmarked.
A Get request is often cacheable.	A Post request can hardly cacheable.
Get request is more efficient and used more than post.	Post request is less efficient and used less than Get.

1.1.1.1.2. Request-URI

The Request-URI is a ***Uniform Resource Identifier*** and identifies the resource upon which to apply the request. Following are the most commonly used forms to specify an URI:

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

S.N.	Method and Description
1	The asterisk * is used when an HTTP request does not apply to a particular resource, but to the server itself, and is only allowed when the method used does not necessarily apply to a resource. For example: **OPTIONS * HTTP/1.1**
2	The **absoluteURI** is used when an HTTP request is being made to a proxy. The proxy is requested to forward the request or service from a valid cache, and return the response. For example: **GET http://www.w3.org/pub/WWW/TheProject.html HTTP/1.1**
3	The most common form of Request-URI is that used to identify a resource on an origin server or gateway. For example, a client wishing to retrieve a resource directly from the origin server would create a TCP connection to port 80 of the host "www.w3.org" and send the following lines: **GET /pub/WWW/TheProject.html HTTP/1.1** **Host: www.w3.org** Note that the absolute path cannot be empty; if none is present in the original URI, it MUST be given as "/" (the server root).

1.1.1.2. Request Header Fields:

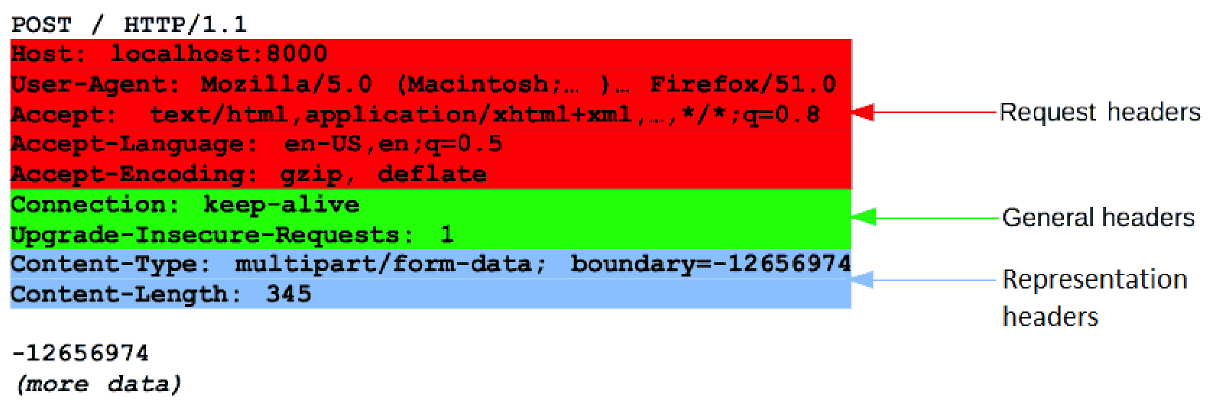
- We will study General-header and Entity-header in a separate chapter when we will learn HTTP header fields. For now, let's check what Request header fields are.
- The request-header fields allow the client to pass additional information about the request, and about the client itself, to the server. These fields act as request modifiers. Here is a list of some important Request-header fields that can be used based on the requirement:
  - ✓ Accept-Charset
  - ✓ Accept-Encoding

- ✓ Accept-Language
- ✓ Authorization
- ✓ Expect
- ✓ From
- ✓ Host
- ✓ If-Match
- ✓ If-Modified-Since
- ✓ If-None-Match
- ✓ If-Range
- ✓ If-Unmodified-Since
- ✓ Max-Forwards
- ✓ Proxy-Authorization
- ✓ Range
- ✓ Referer
- ✓ TE
- ✓ User-Agent

- HTTP headers from a request follow the same basic structure of an HTTP header: a case-insensitive string followed by a colon (':') and a value whose structure depends upon the header. The whole header, including the value, consists of one single line, which can be quite long.

Many different headers can appear in requests. They can be divided in several groups:

1. **General headers**, like Via, apply to the message as a whole.
2. **Request headers**, like User-Agent or Accept, modify the request by specifying it further (like Accept-Language), by giving context (like Referer), or by conditionally restricting it (like If-None).
3. **Representation headers** like Content-Type that describe the original format of the message data and any encoding applied (only present if the message has a body).



```
POST / HTTP/1.1
Host: localhost:8000
User-Agent: Mozilla/5.0 (Macintosh;… )… Firefox/51.0
Accept: text/html,application/xhtml+xml,…,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=-12656974
Content-Length: 345

-12656974
(more data)
```

- You can introduce your custom fields in case you are going to write your own custom Client and Web Server.

1.1.1.3. Examples of Request Message:

- Now let's put it all together to form an HTTP request to fetch **hello.htm** page from the web server running on tutorialspoint.com

```
GET /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

- Here we are not sending any request data to the server because we are fetching a plain HTML page from the server. Connection is a general-header, and the rest of the headers are request headers. The following example shows how to send form data to the server using request message body:

```
POST /cgi-bin/process.cgi HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive

licenseID=string&content=string&/paramsXML=string
```

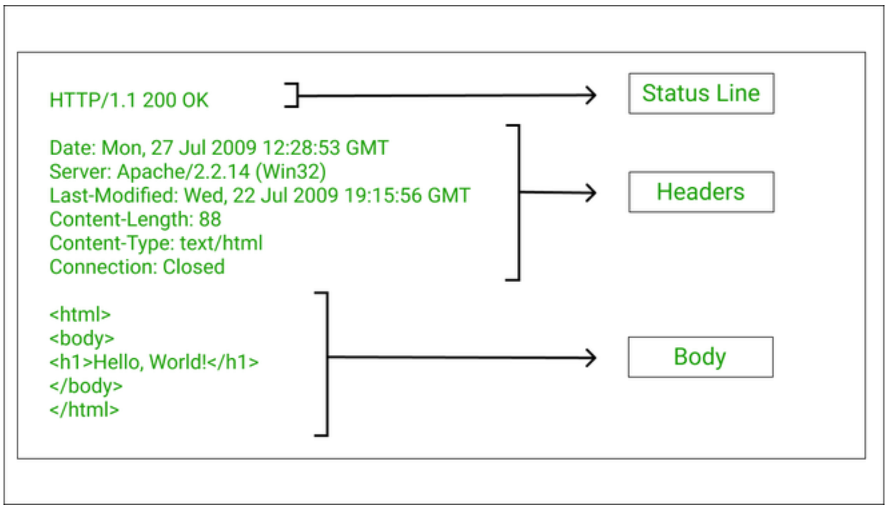
- Here the given URL */cgi-bin/process.cgi* will be used to process the passed data and accordingly, a response will be returned. Here **content-type** tells the server that the passed data is a simple web form data and **length** will be the actual length of the data put in the message body. The following example shows how you can pass plain XML to your web server:

```
POST /cgi-bin/process.cgi HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Content-Type: text/xml; charset=utf-8
Content-Length: length
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://clearforest.com/">string</string>
```

1.1.2. HTTP Response Message.

- After receiving and interpreting a request message, a server responds with an HTTP response message:

1. A Status-line
2. Zero or more header (General|Response|Entity) fields followed by CRLF
3. An empty line (i.e., a line with nothing preceding the CRLF)
4. indicating the end of the header fields
5. Optionally a message-body



```
HTTP/1.1 200 OK

Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT
Content-Length: 88
Content-Type: text/html
Connection: Closed

<html>
<body>
<h1>Hello, World!</h1>
</body>
</html>
```

(HTTP Response Message)

- The following sections explain each of the entities used in an HTTP response message.

1.1.2.1. Message Status-Line:

- A Status-Line consists of the protocol version followed by a numeric status code and its associated textual phrase. The elements are separated by space SP characters.

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

1.1.2.1.1. HTTP Version

- A server supporting HTTP version 1.1 will return the following version information:

HTTP-Version = HTTP/1.1

1.1.2.1.2. Status Code

- The Status-Code element is a 3-digit integer where first digit of the Status-Code defines the class of response and the last two digits do not have any categorization role. There are 5 values for the first digit:

S.N.	Code and Description
1	**1xx: Informational** It means the request was received and the process is continuing.
2	**2xx: Success** It means the action was successfully received, understood, and accepted.
3	**3xx: Redirection** It means further action must be taken in order to complete the request.
4	**4xx: Client Error** It means the request contains incorrect syntax or cannot be fulfilled.
5	**5xx: Server Error** It means the server failed to fulfill an apparently valid request.

- HTTP status codes are extensible and HTTP applications are not required to understand the meaning of all registered status codes. A list of all the status codes has been given in a separate chapter for your reference.

1.1.2.2. Response Header Fields

- We will study General-header and Entity-header in a separate chapter when we will learn HTTP header fields. For now, let's check what Response header fields are.
- The response-header fields allow the server to pass additional information about the response which cannot be placed in the Status- Line. These header fields give information about the server and about further access to the resource identified by the Request-URI.

  - ✓ Accept-Ranges
  - ✓ Age
  - ✓ ETag
  - ✓ Location
  - ✓ Proxy-Authenticate

- ✓ Retry-After
- ✓ Server
- ✓ Vary
- ✓ WWW-Authenticate

- You can introduce your custom fields in case you are going to write your own custom Web Client and Server.

1.1.2.3. Examples of Response Message

- Now let's put it all together to form an HTTP response for a request to fetch the hello.htm page from the web server running on tutorialspoint.com

```
HTTP/1.1 200 OK
Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT
Content-Length: 88
Content-Type: text/html
Connection: Closed

<html>
<body>
<h1>Hello, World!</h1>
</body>
</html>
```

- The following example shows an HTTP response message displaying error condition when the web server could not find the requested page:

```
HTTP/1.1 404 Not Found
Date: Sun, 18 Oct 2012 10:36:20 GMT
Server: Apache/2.2.14 (Win32)
Content-Length: 230
Connection: Closed
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html>
<head>
  <title>404 Not Found</title>
</head>
<body>
  <h1>Not Found</h1>
  <p>The requested URL /t.html was not found on this server.</p>
</body>
</html>
```

- Following is an example of HTTP response message showing error condition when the web server encountered a wrong HTTP version in the given HTTP request:

```
HTTP/1.1 400 Bad Request
Date: Sun, 18 Oct 2012 10:36:20 GMT
```

```
Server: Apache/2.2.14 (Win32)
Content-Length: 230
Content-Type: text/html; charset=iso-8859-1
Connection: Closed

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html>
<head>
  <title>400 Bad Request</title>
</head>
<body>
  <h1>Bad Request</h1>
  <p>Your browser sent a request that this server could not understand.</p>
  <p>The request line contained invalid characters following the protocol string.</p>
</body>
</html>
```

2. *MODULE - II*

MODULE-II

2.1.1. INTODUCTION TO HTML:

HTML is the standard mark-up language for creating Web pages.

2.1.1.1. What is HTML?

- HTML stands for Hyper Text Mark-up Language
- HTML is the standard mark-up language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

2.1.1.2. A Simple HTML Document

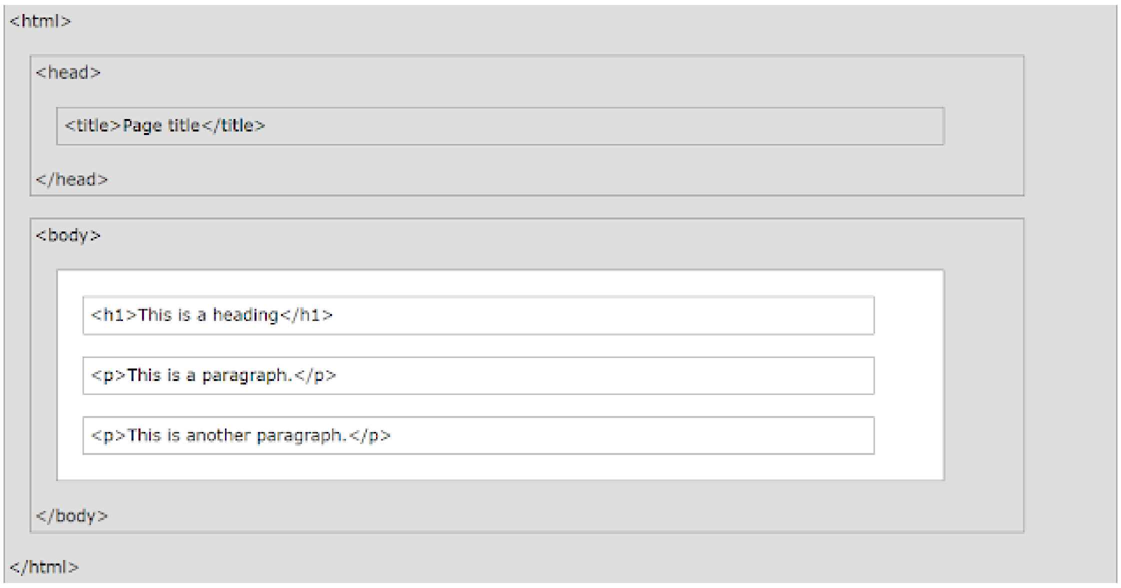
```
<! DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

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

</body>
</html>
```

Example Explained

- The <!DOCTYPE html> declaration defines that this document is an HTML5 document
- The <html> element is the root element of an HTML page
- The <head> element contains meta information about the HTML page
- The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The <h1> element defines a large heading
- The <p> element defines a paragraph

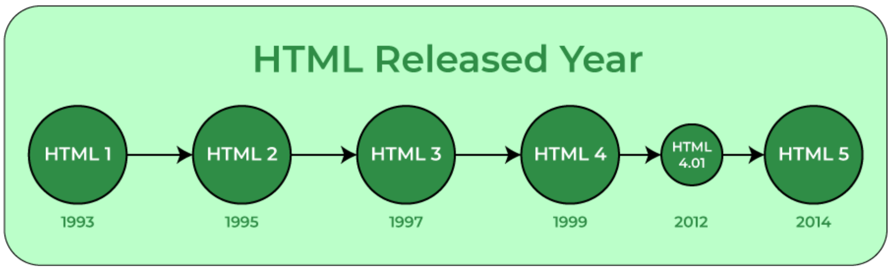
2.1.1.3. HTML Page Structure

Below is a visualization of an HTML page structure:



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

2.1.1.4. HTML Versions



2.1.1.5. Features of HTML

- It is a very **easy and simple language**. It can be easily understood and modified.
- It is very easy to make an **effective presentation** with HTML because it has a lot of formatting tags.
- It is a **mark-up language**, so it provides a flexible way to design web pages along with the text.
- It facilitates programmers to add a **link** on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.
- It is **platform-independent** because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.
- It facilitates the programmer to add **Graphics, Videos, and Sound** to the web pages which makes it more attractive and interactive.
- HTML is a case-insensitive language, which means we can use tags either in lower-case or upper-case.

2.1.1.6. Why learn HTML?

- It is a simple mark-up language. Its implementation is easy.
- It is used to create a website.
- Helps in developing fundamentals about web programming.
- Boost professional career.

2.1.1.7. Advantages:

- HTML is used to build websites.
- It is supported by all browsers.
- It can be integrated with other languages like CSS, JavaScript, etc.

2.1.1.8. Disadvantages:

- HTML can only create static web pages. For dynamic web pages, other languages have to be used.
- A large amount of code has to be written to create a simple web page.
- The security feature is not good.

2.1.2. Elementary tags in HTML:

- <! DOCTYPE> Declaration
- Html Tag
- Title Tag
- Body Tag
- Heading Tags
- Line Break Tag
- Paragraph Tag
- Centring Tag

2.1.2.1. <! DOCTYPE> Declaration:

Definition and Usage:

- All HTML documents must start with a <!DOCTYPE> declaration.
- The declaration is not an HTML tag. It is an "information" to the browser about what document type to expect.
- In HTML 5, the declaration is simple:

```
<!DOCTYPE html>
```

Tips and Notes

Tip: The <!DOCTYPE> declaration is NOT case sensitive.

Examples

```
<!DOCTYPE html>

<!DocType html>
```

```
<!Doctype html>

<!doctype html>
```

2.1.2.2. Html Tag

Definition and Usage

- The <html> tag represents the root of an HTML document.
- The <html> tag is the container for all other HTML elements (except for the <!DOCTYPE> tag).

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

2.1.2.3. Title Tag

Definition and Usage

- The <title> tag defines the title of the document. The title must be text-only, and it is shown in the browser's title bar or in the page's tab.
- The <title> tag is required in HTML documents!
- The contents of a page title are very important for search engine optimization (SEO)! The page title is used by search engine algorithms to decide the order when listing pages in search results.

The <title> element:

- defines a title in the browser toolbar
- provides a title for the page when it is added to favourites
- displays a title for the page in search-engine results

Here are some tips for creating good titles:

- Go for a longer, descriptive title (avoid one- or two-word titles)
- Search engines will display about 50-60 characters of the title, so try not to have titles longer than that
- Do not use just a list of words as the title (this may reduce the page's position in search results)

So, try to make the title as accurate and meaningful as possible!

Note: You can NOT have more than one <title> element in an HTML document.

2.1.2.4. Body Tag

Definition and Usage

- The <body> tag defines the document's body.
- The <body> element contains all the contents of an HTML document, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.

Note: There can only be one <body> element in an HTML document.

2.1.2.5. Heading Tag

Definition and Usage

- The <head> element is a container for metadata (data about data) and is placed between the <html> tag and the <body> tag.
- Metadata is data about the HTML document. Metadata is not displayed.
- Metadata typically define the document title, character set, styles, scripts, and other Meta information.
- The following elements can go inside the <head> element:

  - ✓ <title> (required in every HTML document)
  - ✓ <style>
  - ✓ <base>
  - ✓ <link>
  - ✓ <meta>
  - ✓ <script>
  - ✓ <noscript>

2.1.2.6. Line break Tag

Definition and Usage

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

Tips and Notes

Note: Use the <br> tag to enter line breaks, not to add space between paragraphs.

2.1.2.7. Paragraph Tag

- The HTML <p> element defines a paragraph.
- A paragraph always starts on a new line, and browsers automatically add some white space (a margin) before and after a paragraph.

Example

```
<p> this is a paragraph. </p>
<p> this is another paragraph. </p>
```

2.1.2.8. Centring Tag

- The <center> tag in HTML is used to set the alignment of text into the center. This tag is not supported in HTML5. CSS's property is used to set the alignment of the element instead of the center tag in HTML5.

  Syntax:

  ```
  <center> Contents... </center>
  ```

- **Attributes:** This tag does not accept any attribute.

2.1.3. List in HTML:

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)

Note: We can create a list inside another list, which will be termed as nested List.

2.1.4. Using Ordered List

2.1.4.1. HTML Ordered List | HTML Numbered List

HTML Ordered List or Numbered List displays elements in numbered format. The HTML ol tag is used for ordered list. We can use ordered list to represent items either in numerical order format or alphabetical order format, or any format where an order is emphasized. There can be different types of numbered list:

- Numeric Number (1, 2, 3)
- Capital Roman Number (I II III)
- Small Roman Number (i ii iii)
- Capital Alphabet (A B C)
- Small Alphabet (a b c)

To represent different ordered lists, there are 5 types of attributes in <ol> tag.

Type	Description
Type "1"	This is the default type. In this type, the list items are numbered with numbers.
Type "I"	In this type, the list items are numbered with upper case roman numbers.
Type "i"	In this type, the list items are numbered with lower case roman numbers.
Type "A"	In this type, the list items are numbered with upper case letters.
Type "a"	In this type, the list items are numbered with lower case letters.

HTML Ordered List Example

- Let's see the example of HTML ordered list that displays 4 topics in numbered list. Here we are not defining type="1" because it is the default type.

```
<ol>
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
</ol>
```

Output:

```
1.  HTML
2.  Java
3.  JavaScript
4.  SQL
```

a) ol type="I"

- Let's see the example to display list in roman number uppercase.

```
<ol type="I">
```

```
<li>HTML</li>
<li>Java</li>
<li>JavaScript</li>
<li>SQL</li>
</ol>
```

Output:

```
  I.  HTML
 II.  Java
III.  JavaScript
 IV.  SQL
```

b) ol type="i"

- Let's see the example to display list in roman number lowercase.

```
<ol type="i">
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
</ol>
```

Output:

```
  i.  HTML
 ii.  Java
iii.  JavaScript
 iv.  SQL
```

c) ol type="A"

- Let's see the example to display list in alphabet uppercase.

```
<ol type="A">
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
</ol>
```

Output:

A. HTML

B. Java

C. JavaScript

D. SQL

d) **ol type="a"**

- Let's see the example to display list in alphabet lowercase.

```
<ol type="a">
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
</ol>
```

Output:

a. HTML

b. Java

c. JavaScript

d. SQL

Start attribute

The start attribute is used with ol tag to specify from where to start the list items.

<ol type="1" start="5"> : It will show numeric values starting with "5".

<ol type="A" start="5"> : It will show capital alphabets starting with "E".

<ol type="a" start="5"> : It will show lower case alphabets starting with "e".

<ol type="I" start="5"> : It will show Roman upper case value starting with "V".

<ol type="i" start="5"> : It will show Roman lower case value starting with "v".

```
<ol type="i" start="5">
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
</ol>
```

Output:

```
   v.   HTML
  vi.   Java
 vii.   JavaScript
viii.   SQL
```

Reversed Attribute:

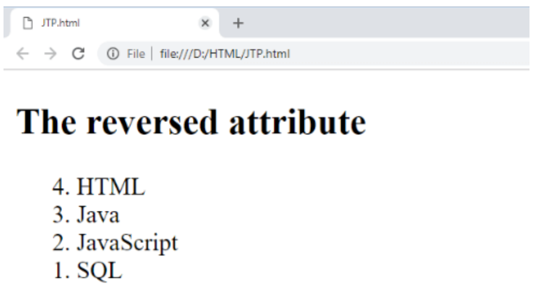
This is a Boolean attribute of HTML <ol> tag, and it is new in HTML5 version. If you use the reversed attribute with

tag then it will numbered the list in descending order (7, 6, 5, 4......1).

Example:

```
<ol reversed>
<li>HTML</li>
<li>Java</li>
<li>JavaScript</li>
<li>SQL</li>
 </ol>
```

Output:


JTP.html

file:///D:/HTML/JTP.html

The reversed attribute

```
4. HTML
3. Java
2. JavaScript
1. SQL
```

2.1.5. Using Unordered Lists

HTML Unordered List or Bulleted List displays elements in bulleted format. We can use unordered list where we do not need to display items in any particular order. The HTML ul tag is used for the unordered list. There can be 4 types of bulleted list:

- disc
- circle
- square
- none

To represent different ordered lists, there are 4 types of attributes in <ul> tag.

Type	Description
Type "disc"	This is the default style. In this style, the list items are marked with bullets.
Type "circle"	In this style, the list items are marked with circles.
Type "square"	In this style, the list items are marked with squares.
Type "none"	In this style, the list items are not marked .

HTML Unordered List Example

```
<ul>
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
</ul>
```

Output:

- HTML
- Java
- JavaScript
- SQL

a) ul type="circle"

```
<ul type="circle">
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
</ul>
```

Output:

- HTML
- Java
- JavaScript
- SQL

b) ul type="square"

```
<ul type="square">
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
</ul>
```

Output:

- HTML
- Java
- JavaScript
- SQL

c) ul type="none"

```
<ul type="none">
 <li>HTML</li>
 <li>Java</li>
 <li>JavaScript</li>
 <li>SQL</li>
```

```
</ul>
```

Output:

- HTML
- Java
- JavaScript
- SQL

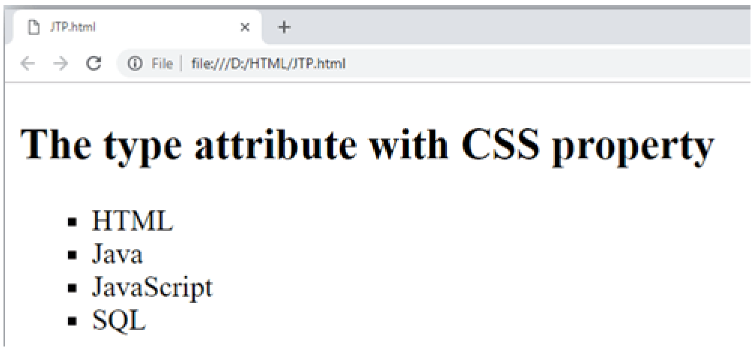
Note: The type attribute is not supported in HTML5; instead of type you can use CSS property of list-style-type. Following is the example to show the CSS property for ul tag.

```
<ul style="list-style-type: square;">
   <li>HTML</li>
   <li>Java</li>
   <li>JavaScript</li>
   <li>SQL</li>
 </ul>
```

Eg Code:

```
<!DOCTYPE html>
<html>
 <head>
  </head>
 <body>
 <h2>The type attribute with CSS property</h2>
   <ul style="list-style-type: square;">
    <li>HTML</li>
     <li>Java</li>
          <li>JavaScript</li>
          <li>SQL</li>
    </ul>
 </body>
</html>
```

Output:



The type attribute with CSS property

- HTML
- Java
- JavaScript
- SQL

2.1.6. HTML Description Lists

- **HTML Description List** or Definition List displays elements in definition form like in dictionary. The <dl>, <dt> and <dd> tags are used to define description list.
- The 3 HTML description list tags are given below:

<dl> tag defines the description list.

<dt> tag defines data term.

<dd> tag defines data definition (description).

Example Program:

```
<!DOCTYPE html>
<html>
<body>
<dl>
  <dt>HTML</dt>
  <dd>is a markup language</dd>
  <dt>Java</dt>
  <dd>is a programming language and platform</dd>
 <dt>JavaScript</dt>
 <dd>is a scripting language</dd>
  <dt>SQL</dt>
  <dd>is a query language</dd>
</dl>
</body>
</html>
```

Output:

```
HTML
        is a markup language
Java
        is a programming language and platform
JavaScript
        is a scripting language
SQL
        is a query language
```

2.1.7. Nested HTML Lists

HTML Nested List

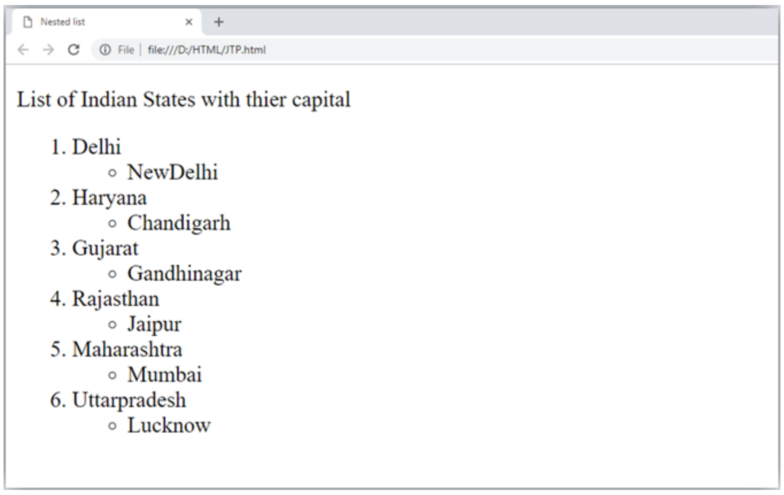
- A list within another list is termed as nested list. If you want a bullet list inside a numbered list then such type of list will called as nested list.

Code:

```
<!DOCTYPE html>
<html>
<head>
  <title>Nested list</title>
</head>
<body>
  <p>List of Indian States with thier capital</p>
<ol>
  <li>Delhi
    <ul>
      <li>NewDelhi</li>
    </ul>
  </li>
  <li>Haryana
    <ul>
      <li>Chandigarh</li>
    </ul>
  </li>
  <li>Gujarat
    <ul>
      <li>Gandhinagar</li>
    </ul>
  </li>
  <li>Rajasthan
    <ul>
```

```
            <li>Jaipur</li>
        </ul>
    </li>
    <li>Maharashtra
        <ul>
            <li>Mumbai</li>
        </ul>
    </li>
    <li>Uttarpradesh
        <ul>
            <li>Lucknow</li></ul>
    </li>
</ol>
</body>
</html>
```

Output:



```
List of Indian States with thier capital

   1. Delhi
         ◦ NewDelhi
   2. Haryana
         ◦ Chandigarh
   3. Gujarat
         ◦ Gandhinagar
   4. Rajasthan
         ◦ Jaipur
   5. Maharashtra
         ◦ Mumbai
   6. Uttarpradesh
         ◦ Lucknow
```

2.1.8. Graphics and Image Format:

HTML Image

- **HTML img tag** is used to display image on the web page. HTML img tag is an empty tag that contains attributes only, closing tags are not used in HTML image element.
- Let's see an example of HTML image.

```
<h2>HTML Image Example</h2>
<img src="good_morning.jpg" alt="Good Morning Friends"/>
```

Output:

Attributes of HTML img tag:

- The src and alt are important attributes of HTML img tag. All attributes of HTML image tag are given below.

1) Src

- It is a necessary attribute that describes the source or path of the image. It instructs the browser where to look for the image on the server.
- The location of image may be on the same directory or another server.

2) Alt

- The alt attribute defines an alternate text for the image, if it can't be displayed. The value of the alt attribute describes the image in words. The alt attribute is considered good for SEO prospective.

3) Width

- It is an optional attribute which is used to specify the width to display the image. It is not recommended now. You should apply CSS in place of width attribute.

4) Height

- It h3 the height of the image. The HTML height attribute also supports iframe, image and object elements. It is not recommended now. You should apply CSS in place of height attribute.

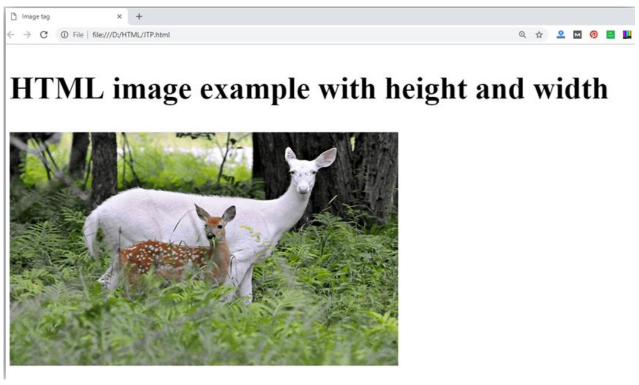
Use of height and width attribute with img tag

You have learnt about how to insert an image in your web page, now if we want to give some height and width to display image according to our requirement, then we can set it with height and width attributes of image.

Example:

```
<img src="animal.jpg" height="180" width="300" alt="animal image">
```

Output:


HTML image example with height and width

Note: Always try to insert the image with height and width, else it may flicker while displaying on webpage.

Use of alt attribute

- We can use alt attribute with tag. It will display an alternative text in case if image cannot be displayed on browser. Following is the example for alt attribute:

```
<img src="animal.png" height="180" width="300" alt="animal image">
```

Output:

How to get image from another directory/folder?

- To insert an image in your web, that image must be present in your same folder where you have put the HTML file. But if in some case image is available in some other directory then you can access the image like this:

```
<img src="E:/images/animal.png" height="180" width="300" alt="animal image">
```

- In above statement we have put image in local disk E------>images folder------>animal.png.

Note: If src URL will be incorrect or misspell then it will not display your image on web page, so try to put correct URL.

Use <img> tag as a link

We can also link an image with other page or we can use an image as a link. To do this, put <img> tag inside the <a> tag.

Example:

```
<a href="https://www.javatpoint.com/what-is-
robotics"><img src="robot.jpg" height="100" width="100"></a>
```

Output:

2.1.9. Frames

- HTML <frame> tag define the particular area within an HTML file where another HTML web page can be displayed.
- A <frame> tag is used with <frameset>, and it divides a webpage into multiple sections or frames, and each frame can contain different web pages.

Note: Do not use HTML <frame> tag as it is not supported in HTML5, instead you can use <iframe> or <div> with CSS to achieve similar effects in HTML.

Syntax < frame src = "URL" >

Following are some specifications about the HTML <frame> tag

Display	Block
Start tag/End tag	Start tag(required), End tag(forbidden)
Usage	Frames

Example 1
Create Vertical frames:

```
<!DOCTYPE html>
<html>
<head>
  <title>Frame tag</title>
</head>
 <frameset cols="25%,50%,25%">
  <frame src="frame1.html" >
  <frame src="frame2.html">
  <frame src="frame3.html">
 </frameset>
</html>
```

Output:

Frame1.html

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

```
    <style>
      div{
                    background-color: #7fffd4;
        height: 500px;
       }
    </style>
</head>
<body>
    <div>
        <h2>This is first frame</h2>
    </div>
 </body>
</html>
```

Frame2.html

```
<!DOCTYPE html>
<html>
<head>
    <style>
      div{
        background-color: #2f4f4f;
        height: 500px;
    }
    </style>
</head>
<body>
    <div>
        <h2>This is Second frame</h2>
    </div>
 </body>
</html>
```

Frame3.html

```
<!DOCTYPE html>
<html>
<head>
    <style>
      div{
        background-color: #c1ffc1;
        height: 500px;   }
    </style>
</head>
<body>
   <div>
         <h2>This is Third frame</h2>
    </div>
 </body>
</html>
```

Example 2:Create Horizontal frames:

```
<!DOCTYPE html>
<html>
<head>
    <title>Frame tag</title>
```

```
</head>
  <frameset rows="30%, 40%, 30%">
    <frame name="top" src="frame1.html" >
    <frame name="main" src="frame2.html">
    <frame name="bottom" src="frame3.html">
  </frameset>
</html>
```

Output:

2.1.10. HTML Form

- An HTML form is a section of a document which contains controls such as text fields, password fields, checkboxes, radio buttons, submit button, menus etc.
- An HTML form facilitates the user to enter data that is to be sent to the server for processing such as name, email address, password, phone number, etc. .

Why use HTML Form

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

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

HTML Form Syntax

```
<form action="server url" method="get|post">
 //input controls e.g. textfield, textarea, radiobutton, button  </form>
```

HTML Form Tags

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

Tag	**Description**
<form>	It defines an HTML form to enter inputs by the used side.
<input>	It defines an input control.
<textarea>	It defines a multi-line input control.
<label>	It defines a label for an input element.
<fieldset>	It groups the related element in a form.

<legend>	It defines a caption for a <fieldset> element.
<select>	It defines a drop-down list.
<optgroup>	It defines a group of related options in a drop-down list.
<option>	It defines an option in a drop-down list.
<button>	It defines a clickable button.

HTML 5 Form Tags

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

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

2.1.10.1. HTML <form> element

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

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

Syntax:

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

2.1.10.2. HTML <input> element

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

Example:

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

Output:

2.1.10.3. HTML TextField Control

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

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

Output:

Note: If you will omit 'name' attribute then the text filed input will not be submitted to server.

2.1.10.4. HTML <textarea> tag in form

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

Example:

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

Output:

2.1.10.5. Label Tag in Form

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

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

NOTE: It is good to use <label> tag with form, although it is optional but if you will use it, then it will provide a focus when you tap or click on label tag. It is more worthy with touchscreens.

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

Output:

2.1.10.6. HTML Password Field Control

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

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

Output:

2.1.10.7. HTML 5 Email Field Control

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

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

It will display in browser like below:

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

2.1.10.8. Radio Button Control

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

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

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

```
<form>
  <label for="gender">Gender: </label>
       <input type="radio" id="gender" name="gender" value="male"/>Male
       <input type="radio" id="gender" name="gender" value="female"/>Female <br/>
</form>
```

Checkbox Control

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

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

```
</form>
```

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

Output:

2.1.10.9. Submit button control

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

Syntax:

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

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

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

The name attribute can be omit here.

Example:

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

Output:

2.1.10.10. HTML <fieldset> element:

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

Example:

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

Output:

HTML Form Example

Following is the example for a simple form of registration.

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

Output:

HTML Form Example

Let's see a simple example of creating HTML form.

```
<form action="#">
<table>
<tr>
   <td class="tdLabel"><label for="register_name" class="label">Enter name:</label></td>
   <td><input type="text" name="name" value="" id="register_name" style="width:160px"/></td>
</tr>
<tr>
   <td class="tdLabel"><label for="register_password" class="label">Enter password:</label></td>
   <td><input type="password" name="password" id="register_password" style="width:160px"/></td>
</tr>
<tr>
   <td class="tdLabel"><label for="register_email" class="label">Enter Email:</label></td>
   <td
><input type="email" name="email" value="" id="register_email" style="width:160px"/></td>
</tr>
<tr>
   <td class="tdLabel"><label for="register_gender" class="label">Enter Gender:</label></td>
   <td>
<input type="radio" name="gender" id="register_gendermale" value="male"/>
<label for="register_gendermale">male</label>
<input type="radio" name="gender" id="register_genderfemale" value="female"/>
<label for="register_genderfemale">female</label>
   </td>
```

```
</tr>
<tr>
   <td class="tdLabel"><label for="register_country" class="label">Select Country:</label></td>
   <td><select name="country" id="register_country" style="width:160px">
   <option value="india">india</option>
   <option value="pakistan">pakistan</option>
   <option value="africa">africa</option>
   <option value="china">china</option>
   <option value="other">other</option>
</select>
</td>
</tr>
<tr>
   <td colspan="2"><div align="right"><input type="submit" id="register_0" value="register"/>
</div></td>
</tr>
</table>
</form>
```

3. *MODULE - III*

MODULE-III

3.1.1. INTODUCTION TO DHTML:

- DHTML stands for Dynamic Hypertext Mark-up language i.e., Dynamic HTML.
- Dynamic HTML is not a mark-up or programming language but it is a term that combines the features of various web development technologies for creating the web pages dynamic and interactive.

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

Components of Dynamic HTML

DHTML consists of the following four components or languages:

- HTML 4.0
- CSS
- JavaScript
- DOM.

HTML 4.0

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

CSS

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

JavaScript

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

DOM

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

Uses of DHTML

Following are the uses of DHTML (Dynamic HTML):

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

- It is also used to add the ticker on various websites, which needs to refresh their content automatically.

Features of DHTML

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

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

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

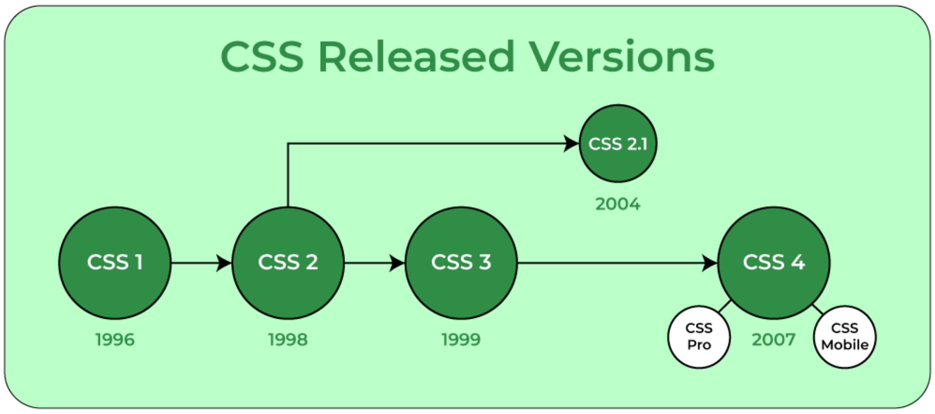
3.1.2. Introduction to style sheets:

- 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

Why CSS?

- **CSS saves time:** You can write CSS once and reuse the same sheet in multiple HTML pages.
- **Easy Maintenance:** To make a global change simply change the style, and all elements in all the webpages will be updated automatically.
- **Search Engines:** CSS is considered a clean coding technique, which means search engines won't have to struggle to "read" its content.
- **Superior styles to HTML:** CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- **Offline Browsing:** CSS can store web applications locally with the help of an offline cache. Using this we can view offline websites.

CSS versions release



years:

CSS Syntax:

CSS comprises style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule set consists of a selector and declaration block.

1. **Selector:** A selector in CSS is used to target and select specific HTML elements to apply styles to.
2. **Declaration:** A declaration in CSS is a combination of a property and its corresponding value.

Selector -- h1
Declaration -- {color:blue;font size:12px;}

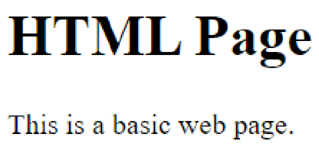
- The selector points to the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons.
- Each declaration includes a CSS property name and a value, separated by a colon.

Let's see how our page looks with & without CSS :
Before CSS: In this example, we have not added any CSS.
<u>Html</u>

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

```
<head>
   <title>Example</title>
</head>
<body>
   <main>
      <h1>HTML Page</h1>
      <p>This is a basic web page.</p>
   </main>
</body>
</html>
```

Output:



HTML Page

This is a basic web page.

[Without CSS]

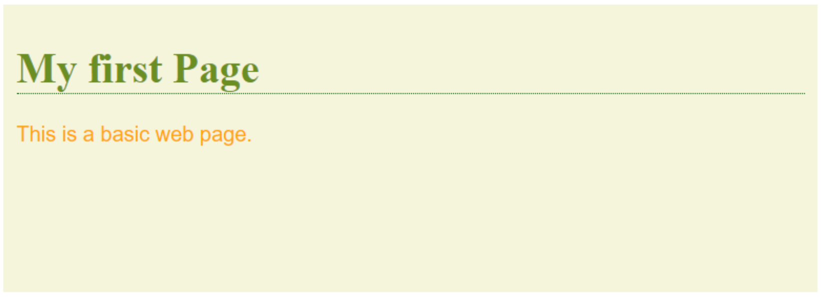
After CSS: In this example, we added some CSS styling inside the HTML code only to show how CSS makes a dull HTML page look beautiful.

<u>Html</u>

```
<!DOCTYPE html>
<html>

<head>
   <title>Example</title>
   <style>
      main {
         width: 600px;
         height: 200px;
         padding: 10px;
         background: beige;
      }

      h1 {
         color: olivedrab;
         border-bottom: 1px dotted darkgreen;
      }

      p {
         font-family: sans-serif;
         color: orange;
      }
   </style>
</head>

<body>
   <main>
      <h1>My first Page</h1>
      <p>This is a basic web page.</p>
   </main>
</body>
```

```
</html>
```

Output:



My first Page

This is a basic web page.

[With CSS]

THREE WAYS TO APPLY CSS:

To use CSS with HTML document, there are three ways:

- ✓ Inline CSS**: Define CSS properties using style attribute in the HTML elements.**
- ✓ Internal or Embedded CSS: **Define CSS using <style> tag in <head> section.**
- ✓ External CSS: **Define all CSS property in a separate .css file, and then include the file with HTML file using tag in section.**

3.1.3. Inline style information

- We can apply CSS in a single element by inline CSS technique.
- The inline CSS is also a method to insert style sheets in HTML document. This method mitigates some advantages of style sheets so it is advised to use this method sparingly.
- If you want to use inline CSS, you should use the style attribute to the relevant tag.

Syntax:

```
<htmltag style="cssproperty1:value; cssproperty2:value;"> </htmltag>
```

Example:

```
<html>

<body>

<h1 style="color:red;margin-left:40px;">Inline CSS is applied on this heading.</h1>

<p>This paragraph is not affected.</p>

</body>

</html>
```

Output:

Inline CSS is applied on this heading.

This paragraph is not affected.

Disadvantages of Inline CSS

- You cannot use quotations within inline CSS. If you use quotations the browser will interpret this as an end of your style value.
- These styles cannot be reused anywhere else.
- These styles are tough to be edited because they are not stored at a single place.
- It is not possible to style pseudo-codes and pseudo-classes with inline CSS.
- Inline CSS does not provide browser cache advantages.

3.1.4. 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>
```

3.1.5. 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"

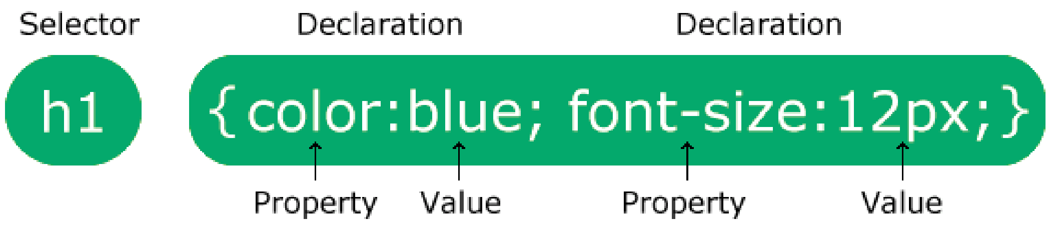
```
body {
  background-color: lightblue;
}

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

Note: Do not add a space between the property value (20) and the unit (px):
Incorrect (space): margin-left: 20 px;
Correct (no space): margin-left: 20px;

3.1.6. Cascading Style sheets:

A CSS rule consists of a selector and a declaration bloc

1) 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.

2) 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:

1. The CSS element Selector
2. The CSS id Selector
3. The CSS class Selector
4. The CSS Universal Selector
5. The CSS Grouping Selector

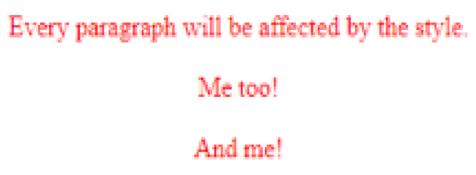
1. The CSS element Selector:

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

Example

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

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  text-align: center;
  color: red;
}
</style>
</head>
<body>

<p>Every paragraph will be affected by the sty
<p id="para1">Me too!</p>
<p>And me!</p>

</body>
</html>
```


Every paragraph will be affected by the style.

Me too!

And me!

2. The CSS id Selector

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

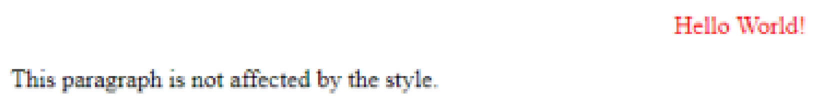
The id of an element is unique within a page, so the id selector is used to select one unique element!

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

Example

The CSS rule below will be applied to the HTML element with id="para1":

```
<!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>
```


Hello World!

This paragraph is not affected by the style.

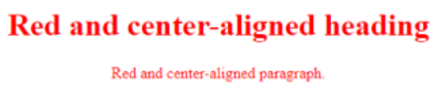
3. The CSS class Selector

- 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.

Example

In this example all HTML elements with class="center" will be red and center-aligned:

```
<!DOCTYPE html>
<html>
<head>
<style>
.center {
  text-align: center;
  color: red;
}
</style>
</head>
<body>

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

</body>
</html>
```



Red and center-aligned heading

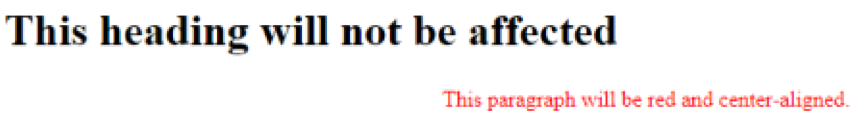
Red and center-aligned paragraph.

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

Example

In this example only <p> elements with class="center" will be red and center-aligned:

```
<!DOCTYPE html>
<html>
<head>
<style>
p.center {
  text-align: center;
  color: red;
}
</style>
</head>
<body>

<h1 class="center">This heading will not be affected</h1>
<p class="center">This paragraph will be red and center-aligned.</p>

</body>
</html>
```



This heading will not be affected

This paragraph will be red and center-aligned.

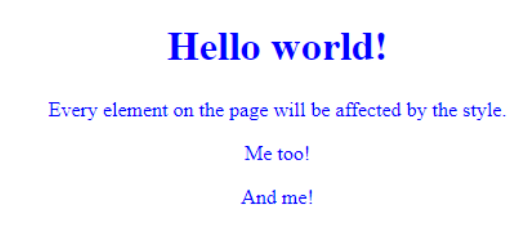
Note: A class name cannot start with a number!

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:



Hello world!

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

Me too!

And me!

```
<!DOCTYPE html>
<html>
<head>
<style>
* {
  text-align: center;
  color: blue;
}
</style>
</head>
<body>

<h1>Hello world!</h1>

<p>Every element on the page will be affected by the style
<p id="para1">Me too!</p>
<p>And me!</p>

</body>
</html>
```

5. The CSS Grouping Selector

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

- Look at the following CSS code (the h1, h2, and p elements have the same style definitions):

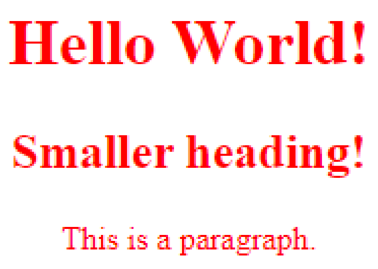
```
h1 {
  text-align: center;
  color: red;
}

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:

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

<h1>Hello World!</h1>
<h2>Smaller heading!</h2>
<p>This is a paragraph.</p>

</body>
</html>
```



Hello World!

Smaller heading!

This is a paragraph.

3) CSS Comments:

- CSS comments are not displayed in the browser, but they can help document your source code.
- 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 */:
- From the HTML tutorial, you learned that you can add comments to your HTML source by using the **<!--...--> syntax**.

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  color: red; /* Set text color to red */
}
</style>
</head>
<body>

<h2>My Heading</h2>

<!-- These paragraphs will be red -->
<p>Hello World!</p>
<p>This paragraph is styled with CSS.</p>
<p>CSS comments are not shown in the output.</p>

</body>
</html>
```

OUTPUT:

My Heading

Hello World!

This paragraph is styled with CSS.

HTML and CSS comments are not shown in the output.

4) CSS Colors

Colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.

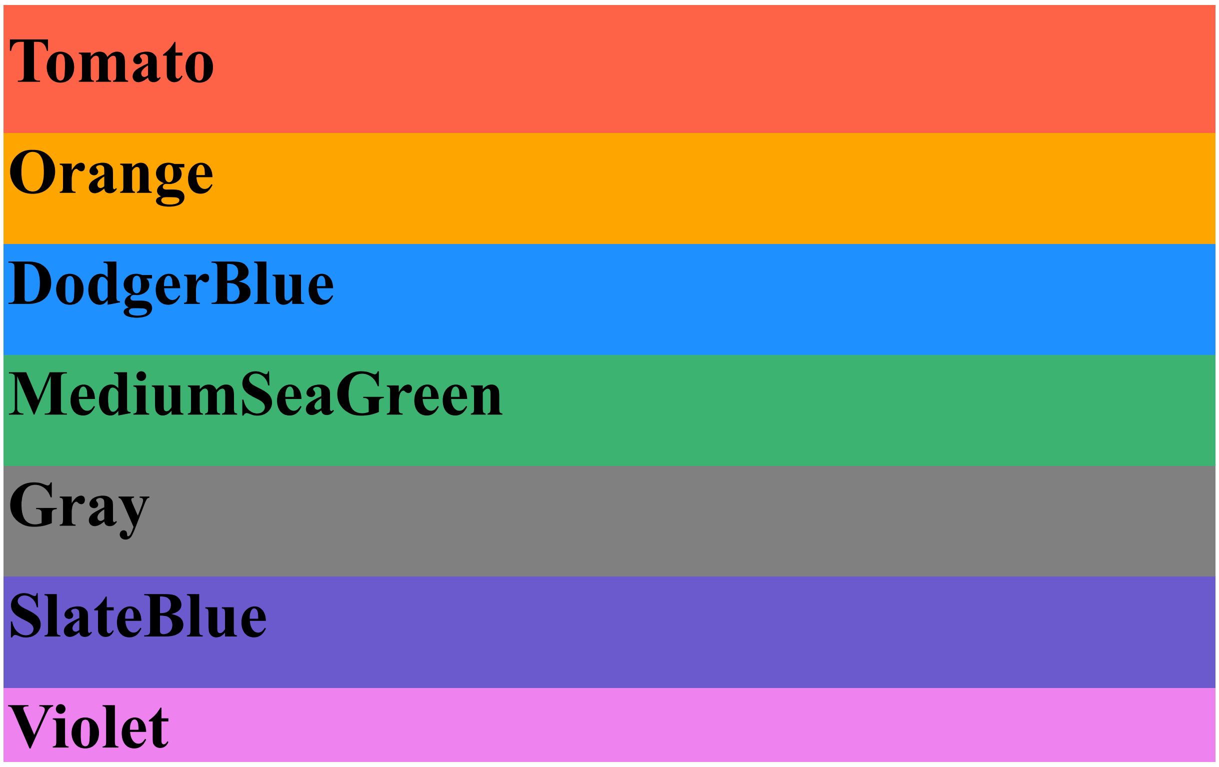
CSS Color Names

In CSS, a color can be specified by using a predefined color name: CSS/HTML support 140 standard color names.

```
<html>

<body>

<h1 style="background-color:Tomato;">Tomato</h1>

<h1 style="background-color:Orange;">Orange</h1>
```

```
<h1 style="background-color:DodgerBlue;">DodgerBlue</h1>

<h1 style="background-color:MediumSeaGreen;">MediumSeaGreen</h1>

<h1 style="background-color:Gray;">Gray</h1>

<h1 style="background-color:SlateBlue;">SlateBlue</h1>

<h1 style="background-color:Violet;">Violet</h1>

</body>

</html>
```

OUTPUT:



```
Tomato
Orange
DodgerBlue
MediumSeaGreen
Gray
SlateBlue
Violet
```

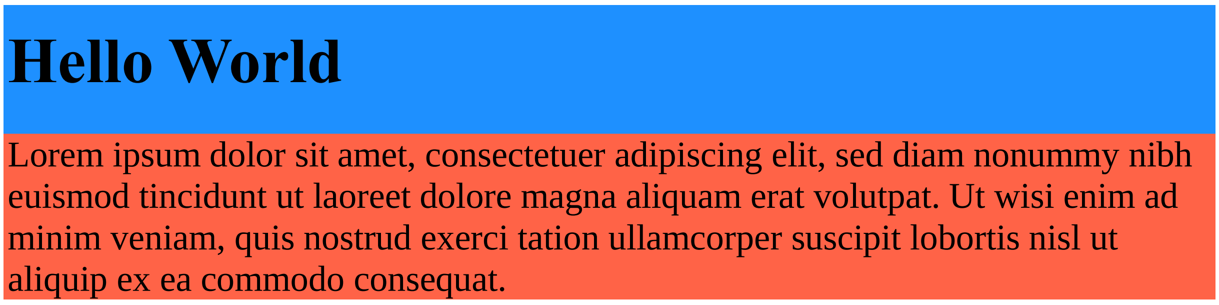
5) CSS Background Color

You can set the background color for HTML elements:

```
<html>
<body>

<h1 style="background-color:DodgerBlue;">Hello World</h1>

<p style="background-color:Tomato;">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.
Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut
aliquip ex ea commodo consequat.
</p>
```

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

OUTPUT:



Hello World

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

6) CSS Text Color

You can set the color of text:

```
<html>
<body>

<h3 style="color:Tomato;">Hello World</h3>
<p style="color:DodgerBlue;">Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed
diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
<p style="color:MediumSeaGreen;">Ut wisi enim ad minim veniam, quis nostrud exerci
tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>
</body>
</html>
```

OUTPUT:

Hello World

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

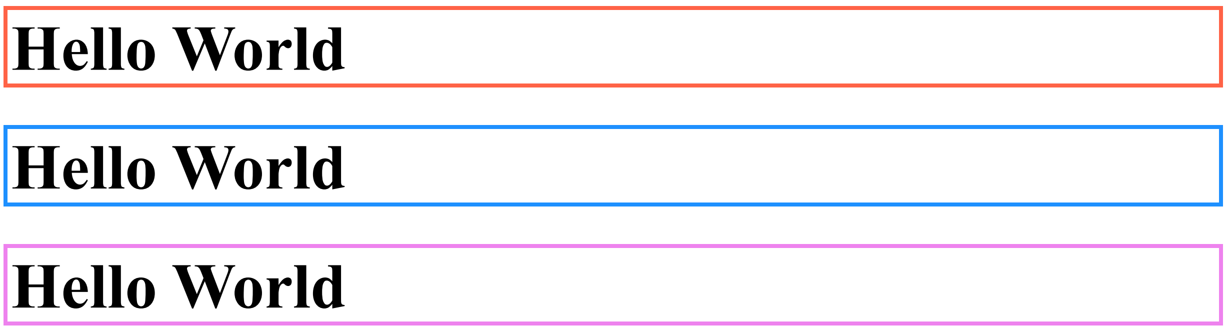
7) CSS Border Color

- You can set the color of borders:

```
<html>
<body>

<h1 style="border: 2px solid Tomato;">Hello World</h1>

<h1 style="border: 2px solid DodgerBlue;">Hello World</h1>

<h1 style="border: 2px solid Violet;">Hello World</h1>

</body>
```

```
        </html>
```

OUTPUT:



Hello World

Hello World

Hello World

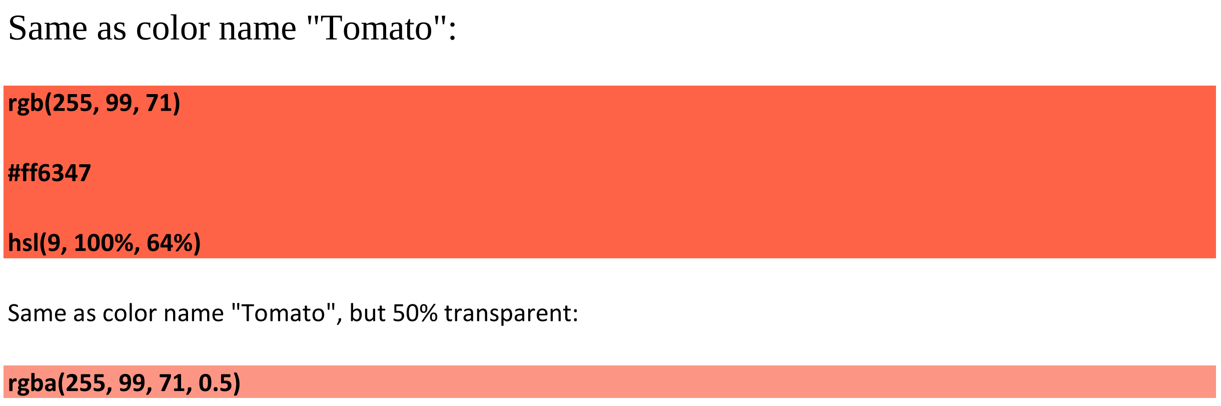
8) CSS Color Values

In CSS, colors can also be specified using RGB values, HEX values, HSL values, RGBA values, and HSLA values:

Same as color name "Tomato":

```
<html>
<body>

<p>Same as color name "Tomato":</p>

<h1 style="background-color:rgb(255, 99, 71);">rgb(255, 99, 71)</h1>
<h1 style="background-color:#ff6347;">#ff6347</h1>
<h1 style="background-color:hsl(9, 100%, 64%);">hsl(9, 100%, 64%)</h1>

<p>Same as color name "Tomato", but 50% transparent:</p>
<h1 style="background-color:rgba(255, 99, 71, 0.5);">rgba(255, 99, 71, 0.5)</h1>
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">hsla(9, 100%, 64%, 0.5)</h1>

<p>In addition to the predefined color names, colors can be specified using RGB, HEX, HSL, or even
transparent colors using RGBA or HSLA color values.</p>

</body>
</html>
```

OUTPUT:



Same as color name "Tomato":

rgb(255, 99, 71)

#ff6347

hsl(9, 100%, 64%)

Same as color name "Tomato", but 50% transparent:

rgba(255, 99, 71, 0.5)

hsla(9, 100%, 64%, 0.5)

In addition to the predefined color names, colors can be specified using RGB, HEX, HSL, or even transparent colors using RGBA or HSLA color values.