Review of Web Technologies

Introduction:

Web technology is related to the interface between client and server.This information include markup language,programming interfaces and languages, and standards for document identification and display. In general web technology incorporates tools and techniques for web development.Web Development is a term for the work involved in developing a web site for World Wide Web(WWW) . This can include web design, web content development, client liaison, client-side/server-side scripting, web server and network security configuration, and e-commerce development.It can range from developing a simple static page to most complex web-based application.

Web design is a term used to encompass the way that content (usually hypertext or hypermedia) is delivered to an end-user through the World Wide Web, using a web browser or other web-enabled software is displayed. The purpose of web design is to create a website—a collection of online content including documents and applications that reside on a web servers. It may include text, images, sounds and other content, and may be interactive.
For the typical web sites, the basic aspects of design are:
The content: the substance, and information on the site should be relevant to the site and should target the area of the public that the website is concerned with.
The usability: the site should be user-friendly, with the interface and navigation simple and reliable.
The appearance: the graphics and text should include a single style that flows throughout, to show consistency. The style should be professional, appealing and relevant.
The structure: of the web site as a whole.

Internet and its Evolution:

The Internet is a global system of interconnected computer networks that use the standard Internet Protocol Suite (TCP/IP) to serve billions of users worldwide.It is a network of networks that consists of millions of private, public, academic, business, and government networks, of local global scope, that are linked by a broad array of electronic and optical networking technologies. The Internet carries a vast range of information resources and services, such as the inter-linked hypertext documents of the World Wide Web (WWW) and the infrastructure to support electronic mail.

The history of internet begins in the 1950s with the development of the electronic computer.

1957:

USSR launches Sputnik into space and, with it, global communications. The United States government creates the Advanced Research Projects Agency (ARPA) in response to Sputnik launch in 1958.

1961: Leonard Kleinrock at MIT wrote a paper called”information flow in large communication Nets” which communicated his theory and concept for packet switching.

1965:

Lawrence Roberts (MIT) and Thomas Marill get an ARPA contract to create the first wide-area network (WAN) connection via long distant dial-up between a TX-2 computer in Massachusetts and a Q-32 computer in California. The system confirms that packet switching offers the most promising model for communication between computers.

1969:

The ARPANET is born .linking four nodes university of California at loss Angeles,Stanford research institute((RSI). University of California at Santa Barbara and university of Utah the network is wired together via 50 kbps circuit. First message sent across the network was supposed to have been “login”.

1970:

Network control protocol (NCP) is designed allowing connection and flow control between process running on different hosts.

1972:
email is introduced . Ray Tomlinson of BBN invents the email program to send messages across a distributed network.The "@" sign is chosen from the punctuation keys on Tomlinson's Model 33 Teletype to separate local from global emails, making "user@host" the email standard.

1973:

Development begins on TCP/IP protocol by a group headed by Vint Cerf (Stanford) and Robert Kahn (DARPA). The new protocol will allow diverse computer networks to interconnect and communicate with each other.

1974:

Vint Cerf and Robert Kahn publish "A Protocol for Packet Network Interconnection" which specifies in detail the design of a Transmission Control Program (TCP) and coins the term “Internet” for the first time.

1978:

The Bulletin board system(BBS) is developed and spam is born.

1983:

Paul Mockapetris expands the Internet beyond its academic origins by inventing the Domain Name System (DNS). John Klensin helps facilitate early procedural and definitional work for DNS administration and top-level domain definitions.

1988:

Internet relay chat(IRC) is developed .

1989:

www proposal was written by Tim burners-lee to convince CERN that a global hypertext system was in their best interest.

1990:

First commercial dial-up internet provider was launched also the code for www was written by Tim Berners-Lee ,based on his proposal from the year before along with the standard for HTML,HTTP,and URLs

1991:
the first web page is created and its purpose was to explain what the www was.it was purely text-based site with lots of highlighted linked embedded into a text which links user to another address and pages .In the same year the first search protocol ,that examined file content instead of just file names was launched called Gopher.

1993:

Mosaic-the first graphical web browser for the general public was introduced .Also the same year marked the beginning of the .gov and .org domain name for government and UN uses.

1995:

The first SSL(source socket layer) encryption was developed by Netscape making it safer to conduct financial transaction which involved credit card payments online this saw the introduction of E-bay and amazon two of the biggest e-commerce business name on the internet. Java and javascript was first introduced in public in 1995 originally called LiveScript it was deployed as a part of Netscape.

1996:

Dr. Henning Schulzrinne co-develops key protocols that enable Voice over the Internet protocol (VoIP). Also in same year Brewster Kahle Founds Internet Archive; Email Surpasses Postal Mail.

1998:
Michael Roberts becomes the first president and CEO of the Internet Corporation for Assigned Numbers (ICANN).

Client server technology:

In a computing model powerful personal computer are connected together with one or more servers.Clients are the powerful personal computer that is a part of a network which requestsfor the service.also called service requestor.Server is a network computer dedicated to common functions that the client computer on the network .it is also called service provider.Web is based on client/server technology. Web servers are included as part of a larger package of internet and intranet related programs for serving e-mail, downloading requests for FTP files and building and publishing web pages.In the client/ server model single machine can be both client and the server The client/ server model utilises a database server in which RDBMS user queries can be answered directly by the server.it reduces the network trafficby providing a

query response to the user rather than transferring total files. The client/ server model improves multi-user updating through a graphical user interface (GUI) front end to the shared database. In client/ server architectures client and server typically communicate through statements made in structured query language (SQL).

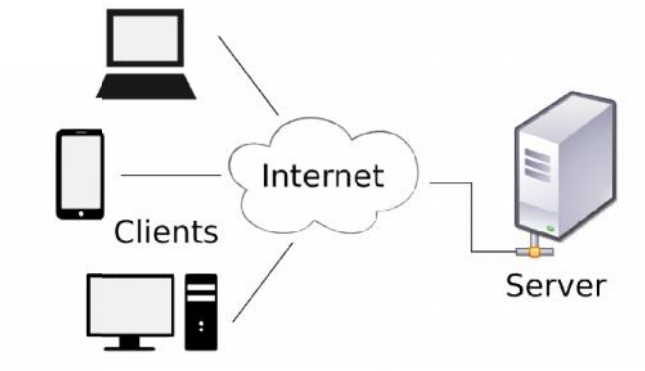


fig:Client/Server model

WWW:

WWW stand for World Wide Web. It is a system of interrelated hypertext document that can be accessed via internet.world wide web ,or simply web is an information sharing model that is built on top of the Internet.It is a collection of image, text sounds and other resources inked by hyperlinks and URLs, which can be viewed using browser software.uses the HTTP protocol to transmit data. Web services, which use HTTP to allow applications to communicate in order to exchange business logic, use the Web to share information. The Web also utilizes browsers, such as Internet Explorer or Firefox, to access Web documents called Web pages that are linked to each other via hyperlinks.

Web pages:

A web page is a document or information resource that is suitable for the World Wide Web and can be accessed through a web browser and displayed on a monitor or mobile device. This information is usually in HTML or XHTML format, and may provide navigation to other web pages via hypertext links. Web pages frequently subsume other resources such as style sheets, scripts and images into their final presentation. they are requested and served from web servers using Hypertext Transfer Protocol (HTTP).It may consist of files of static text and other content stored within the web server's file system (static web pages), or may be constructed by server-side software when they are requested (dynamic web pages). Client-side scripting can make web pages more responsive to user input once on the client browser.

Web sites:

web site is a collection of related web pages containing images, videos or other digital assets.A website is hosted on at least one web server, accessible via a network such as the Internet or a private local area network through an Internet address known as a Uniform Resource Locator. All publicly accessible websites collectively constitute the World Wide Web. web site can be categorized into two types, Static web site and dynamic web site.

Static website:

static websites contain fixed number of web pages and format of web page is fixed which delivers information to the client.This kind of web site is created using HTML and CSS coding on simple text editor like notepad. Static sites never use database connectivity.Static site use for provide some information to the

clients like an organization or institute website.Static sites are easy to develop and a bit experienced people can develop it.

Dynamic website:

dynamic websites can change the web page contents dynamically while the page is running on client's browser. This kind of websites use server- side programming like PHP, Asp.NET. and JSP etc. to modify page contents on run time.Dynamic websites use client side scripting for prepare dynamic design and server-side code to handle event, manage session and cookies, and storing and retrieving data from database.Dynamic websites are not easy to develop because require qualify developers to create it, manage it, test it and maintain security of application and database.

URI:

URI = URL + URN

URI stands for Uniform Resource Identifier is a string of characters used to identify a resources. A URI identifies a resource either by location, or a name, or both. A URI has two specializations known as URL and URN.

URL defines the network location of a specific resource and states how that resource can be obtained.

URL stands for Uniform Resource Locator. It is a subset of the Uniform Resource Identifier (URI which is used by a web browser to identify the location of content on the web. It can be http://, or ftp://.

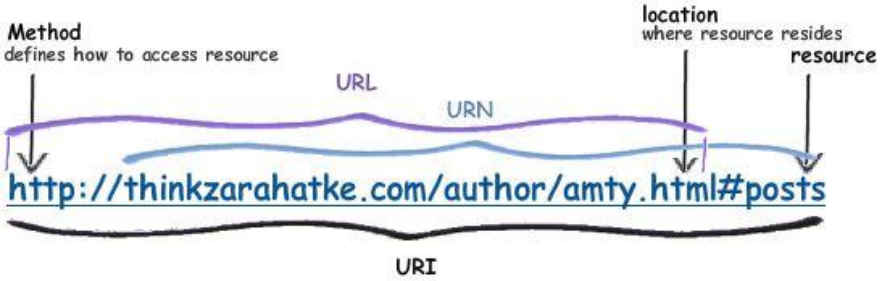
For example:
`http://se.github.io/` is a URL with location `se.github.io` of resource and `http` is a protocol to access that resource.

`ftp://10.10.7.14/ug2011/n2/sht.pdf` is a URL with location `10.10.7.14/ug2011/n2/sht.pdf` of resource and `ftp` is a protocol to access that resource.

Note: All URLs are URIs. But all URIs are not URLs.

URN Stands for Uniform Resource Name. It is a Uniform Resource Identifier (URI) that uses the URN scheme, and does not imply availability of the identified resource**.** Both URNs and URLs are URIs, and a particular URI may be both a name and a locator at the same time.

URI(Uniform Resource Name), a type of URI, basically states what something is, but do not have information on how to access it. For example:
```
author: James F. Kurose
title: Computer Networking: A Top-Down Approach
ISBN-10: 8131790541
```



Web server

Web server is a computer system which receives the HTTP requests via TCP, which are used to distribute information on World Wide Web. Commonly a web browser or web crawler, initiates communication by

making HTTP request for a specific resource and the server responds with the content of that resource or an error message if unable to do so. The main function of a web server is to store, process and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP). Pages delivered are most frequently HTML documents, which may include images, style sheets and scripts in addition to text content.

Web servers are not only used for serving the World Wide Web. They can also be found embedded in devices such as printers, routers, webcams and serving only a local network. The web server may then be used as a part of a system for monitoring and/or administering the device in question. This usually means that no additional software has to be installed on the client computer, since only a web browser is required (which now is included with most operating systems). Apache, Internet information Server(IIS),tomcat are some example of web server.

Web Client

It is a software application capable of communicating with Web servers, requesting and receiving information from them, and processing it for display or other uses. It is used to create a HTTP request message and for processing the HTTP response message. Any web client is designed to directly support user access to web servers is known as user agent. web browser is one kind of web client. Web browsers can run on desktop or laptop computers. Some of the browsers are: Internet Explorer, Mozilla, FireFox, Chrome, Safari, Opera, Netscape Navigator.

Web Browser

Web browser is a software application used to locate, retrieve and display content on the World Wide Web, including Web pages, images, video and other files. As a client/server model, the browser is the client run on a computer that contacts the Web server and requests information. The Web server sends the information back to the Web browser which displays the results on the computer or other Internet-enabled device that supports a browser.

SMTP

SMTP stands for Simple Mail Transfer Protocol..It is an Internet standard for electronic mail (e-mail) transmission across Internet Protocol (IP) networks. SMTP by default uses TCP port25.. SMTP is a connection-oriented, text-based protocol in which a mail sender communicates with a mail receiver by issuing command strings and supplying necessary data over a reliable ordered data stream channel, typically a Transmission Control Protocol(TCP) connection. SMTP is a delivery protocol only. It cannot pull messages from a remote server on demand. Other protocols, such as the Post Office Protocol (POP) and the Internet Message Access Protocol (IMAP) are specifically designed for retrieving messages and managing mail boxes. However, SMTP has a feature to initiate mail queue processing on a remote server so that the requesting system may receive any messages destined for it.Many mail servers now support Extended Simple Mail Transfer Protocol (ESMTP), which allows multimedia files to be delivered as e-mail.

POP

POP stands for post office protocol. It is an application-layer Internet standard protocol used by local e-mail clients to retrieve e-mail from a remote server over a TCP/IP connection. Virtually all modern e-mail clients and servers support POP. The POP protocol has been developed through several versions, with version 3 (POP3).POP3 allows an email client to download an email from an email server. The POP3 protocol is simple and does not offer many features except for download. Its design assumes that the email client downloads all available email from the server, deletes them from the server and then disconnects. POP3 normally uses port 110.

Things to remember

- Web technology is related to the interface between client and server. This information include markup language, programming interfaces and languages, and standards for document identification and display
- The Internet is a global system of interconnected computer networks that use the standard Internet Protocol Suite (TCP/IP) to serve billions of users worldwide
- The history of internet begins in the 1950s with the development of the electronic computer.
- Clients are the powerful personal computer that is a part of a network which requestsfor the service.also called service requestor.
- Server is a network computer dedicated to common functions that the client computer on the network .it is also called service provider.
- WWW is a system of interrelated hypertext document that can be accessed via internet.
- A web page is a document or information resource that is suitable for the World Wide Web and can be accessed through a web browser and displayed on a monitor or mobile device. This information is usually in HTML or XHTML format.
- Web site is a collection of related web pages containing images, videos or other digital assets. It can be categorized into two types, Static web site and dynamic web site.
- URI is a string of characters used to identify a resources. A URI identifies a resource either by location, or a name, or both .A URI has two specializations known as URL and URN.
- Web server is a computer system which receives the HTTP requests via TCP, which are used to distribute information on world wide web.
- Web client is a software application capable of communicating with Web servers, requesting and receiving information from them, and processing it for display or other uses.
- Web browser is a software application used to locate, retrieve and display content on the World Wide Web, including Web pages, images, video and other files.
- Simple Mail Transfer Protocol is an Internet standard for electronic mail (e-mail) transmission across Internet Protocol (IP) networks.SMTP by default uses TCP port25.
- post office protocol is an application-layer Internet standard protocol used by local e-mail clients to retrieve e-mail from a remote server over a TCP/IP connection.POP3 normally uses port 110.

Review of HTML

Html stands for hypertext markup language. It is invented by the scientist Tim burners lee in 1990.the purpose was to make it easier for scientist at different universities to gain access to each others research documents. The project become bigger success than Tim had ever imagined by inventing a HTML he laid the foundation for the web as we known it today.HTML is not a programming language, it is a markup language which use markup tags to describe the web page. HTML tags are keywords surrounded by angle brackets. generally speaking there are two kinds of tags, opening tags: eg<html> and closing tags eg:</html>.the only different between opening and closing tags is the forward slash"/". A web browser read HTML documents and displays them as Web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page. A simple HTML document is given below:

<html>

<head>

<title>This is title</title>

</head>

<body>

<h1>This is first heading</h1>

This is paragraphs

</body>

</html>

Save this page with **.html** or **.htm** extension.

in the head section ,always write a title<title ></title> tag .the title is espically important because it is used by search engines to index your website and shown in the search results.

in the body section ,you write the actual content of the page.

HTML Element:

HTML elements consist of a Start tag ,some content and ,end tag.It gives the structure toa HTML document andtell the browser how you want your website to be presented. for example <h1>This is H1</h1>.Most HTML elements can have attributes.HTML elements with no content are called empty elements. for example

 is an empty element without a closing tag

HTML Attributes

Attributes provide additional information about HTML elements. Attributes are always specified in the start tag. Attributes come in name/value pair like name = "value". For example, HTML links are defined with <a> tag and the link address is provided as an attribute href like Admin page.
Note: Always quote attribute values and use lowercase attributes.

Different sections of HTML Document

<!DOCTYPE html> **identify the version**

<html> **start the browser**

<head><title>Title</title></head> **head section consist title ,metafile etc**

<body>**each and every content of a page is define in body section.**

</body>

</html>

HTML comments

comments are ignored by browser.but they can help document your HTML. With comments you can place notifications and reminders in your HTML.HTML comments are written between <!--and--> eg <!--write comments here-->.

Common tags for Heading

HTML headingsare defined with the <h1> to <h6> tags.<h1> defines the most important heading and <h6> defines the least important heading.example:

```
<html>

<body>

<h1>Heading 1 </h1>

<h2>Heading 2 </h2>

<h3>Heading 3</h3>

<h4>Heading 4</h4>

<h5>Heading 5</h5>

<h6>Heading 6</h6>

</body>

</html>
```

Paragraphs

HTML paragraphs are defined with the <P> tags. for example:

```
<p> This is my first paragraph</p>

<p>This is another paragraph</p>
```

Horizontal Lines

HTML <hr> tag create a horizontal line in an Html page.The hr element used to separate the content. eg:

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

<hr>

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

<hr>

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

<hr>
```

Line Breaks

If you want to go to the new line without starting other paragraph,use line break tag.The <br> element is an empty HTML element. It has no end tag. eg:

This is <br> a paragraph <br> with line break.

Formatting

HTML defines specialelementsfor defining text with a special menaing.HTML uses elements like and for formatting output, like**bold**or*italic*text.Formatting elements were designed to display special typees of text:

- Bold text
- Important text
- Italic text
- Emphasized text
- Marked text
- Small text
- Deleted text
- Inserted text
- Subscripts
- Superscripts

Bold and strong formatting

HTML **** element defines **bold** text.And**** element defines **strong** text.eg;

This is normal text

This text is bold

This text is strong

HTML italic and Emphasized forrmatting:

HTML **** element defines *italic* text.And **** element defines *emphasized* text.eg:

This is normal text

This text is italic

This text is emphasized.

HTML small formatting:

HTML <small> element defines small text eg:

<h2>This is HTML <small>Small</small> Formatting</h2>

HTML marked formatting

HTML <mark> element defines marked or highlighted text.eg:

<h2>This is HTML <mark>Marked</mark> Formatting</h2>

HTML delete formatting:

HTML **<del>** element defines **deleted** (removed) text. eg:

My favorite subject is ~~science~~Maths

HTML insert formatting

HTML <ins> element defines inserted (added) text.eg:

My favorite <ins>subject</ins> is Maths.

HTML subscrip formatting:

HTML <sub> element defines subscripted text.eg:

This is _{subscripted} text. .

HTML superscript formatting:

HTML <sup> element defines superscripted text eg:

This is ^{superscripted} text.

HTML links

A link is the address to resource on the web .It is defined under the anchor tag <a>. use href attribute to defiine the link address.eg:

Kullabs.

we can use target attribute to define where the linked document will be opened. for example

Kullabswill open the document in new window.

We can use name attribute to define a named anchor inside a HTML document. Named anchor are invisible to the reader. For example, abc defines a named anchor and we use the syntax abc to link to the named anchor.

We can also use named anchor to link to some content within another document. For example,

Jump to the Useful Tips section

HTML Images

Images are defined with <img> tag. <img> tagis empty, it contains attributes only, and does not have a closing tag. to display the image ,we use src attribute .we can also add height and width attributes with img tag. eg:

we can usealt attribute. That specifies an alternate text for an imageif the image cannot be displayed.The alt attribute provides alternative information for an image if a user for some reason cannot view it (because of slow connection, an error in the src attribute, or if the user uses a screen reader).If a browser cannot find an image, it will display the alt text. eg:

HTML tables

we use <table > tag to use the table on document.A table is divided into rows using <tr> tag and each row is divided into data cells using <td>tag.A data cell can contain text ,image, lists ,paragraphs ,forms etc.eg

```
<table>

<tr>

<td>A</td>

<td>B</td>

<td>C</td>

</tr>

</table>
```

we use border attribute to display table with border ,if we need heading in table datacell ,we use <th>tag .we can use <caption> tag inside a table to display caption for a table.To make a cell span more than one column, use the **colspan** attribute.And to make a cell span more than one row, use the **rowspan** attribute.use cellpadding and cellspacing attributes to create white space between the cell content and its border and to increase the distance between cells respectively.align attribute in <td> tag is used to align the content of the cell .eg:

```
<table border="1" cellspacing="0">
<tr>
<th>Name:</th>
<td align ="center">saisir </td>
</tr>
<tr>
<th rowspan="2">Telephone:</th>
<td>98033</td>
</tr>
<tr>
<td>98426</td>
</tr>
</table>
```

cellpadding and cellspacing attributes are used in <table> tag.

HTML Lists

HTML supports three types of lists:

1. Ordered list
2. Unordered list
3. Defination list

In ordered list ,items are marked with the numbers, letters etc. tag is used for ordered list and list items are start with tag .eg:

```
<ol type="A">

<li>Apple</li>

<li>Ball</li>

<li>Cat</li>

</ol>
```

If we do not use type attribute, items are marked with numbers. We use type = “a” for lowercase letters list, type = “I” for roman numbers list, and type = “i” for lowercase numbers list.

In Unordered list items are marked with bullets. we use tag for unordered list and each item start with <li> tag.

```
<ul type="circle">

<li>Apple</li>

<li>Ball</li>

<li>Cat</li>

</ul>
```

If we do not use type attribute, items are marked with discs. We use type = “circle” for circle bullets list, and type = “square” for square bullets list.

defination list is the list of item with description of each item. we use <dl> tag for defination list .<dt> for defination term and <dd> for defination description.eg:

```
<dl>
<dt>Ram</dt>
<dd>Ram is a boy</dd>
<dt>Sita</dt>
<dd>Sita is a girl</dd>
</dl>
```

HTML Forms

HTML forms are used to collect user input.The **<form>** element defines an HTML form.eg

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

Form elements are different types of input elements, checkboxes, radio buttons, submit buttons, and more.

Input elements

The **<input>** element is the most important **form element**. The <input> element has many variations, depending on the **type** attribute. some commonly used types are:

text:Defines the normal text input.

password: Defines the password

radioDefines the radio buttons

submit:defines the submit button for submitting the form.

<input type="text"> defines a one-line input field for **text input.<input type="radio">** defines a **radio button**.Radio buttons let a user select ONE of a limited number of choices**<input type="submit">** defines a button for **submitting** a form to a **form-handler**.The form-handler is typically a server page with a script for processing input data.The form-handler is specified in the form's **action** attribute.

The **action attribute** defines the action to be performed when the form is submitted.The common way to submit a form to a server, is by using a submit button.Normally, the form is submitted to a web page on a web server.

you can use**method attribute** that specifies the HTTP method (**GET** or **POST**) to be used when submitting the forms

You can use GET (the default method):If the form submission is passive (like a search engine query) and without sensitive information.When you use GET, the form data will be visible in the page addressYou should use POST.If the form is updating data, or includes sensitive information (password).POST offers better security because the submitted data is not visible in the page address.To be submitted correctly, each input field must have a name attribute.The **<fieldset>** element groups related data in a form.The **<legend>** element defines a caption for the <fieldset> element. example:

```
<form action="action_page.php" method="post">
<fieldset>
<legend>Personal information:</legend>
First name:<br>
<input type="text" name="firstname" ><br>
Last name:<br>
<input type="text" name="lastname"><br>

Password:<br>

<input type="password" name="password"><br><br>

<input type="submit" value="Submit">
</fieldset>
</form>
```

Using colors

With HTML, RGB color values can be specified using this formula: rgb(red, green, blue).Each parameter (red, green, and blue) defines the intensity of the color.The lowest value that can be given to one of the light sources is 0 (hex 00) and the highest values is 255 (hex FF). We can use HEX (e.g. #2000FF) as well as RGB (e.g. rgb(32, 0, 255)) values to define different colors.
The combination of Red, Green and Blue values from 0 to 255 gives a total of more than 16 million different

colors to play with (256 x 256 x 256).
We can also use color names instead of hex and rgb values. The World Wide Web Consortium (W3C) has listed 16 valid color names for HTML and CSS: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow.

for example:

<body style = "background:rgb(12, 100, 255)">
<body style = "background:#0008FF>
<body style = "background:blue">

HTML special characters

HTML Entities and/or ISO Latin-1 codes can be placed in source code like any other alphanumeric characters to produce special characters and symbols that cannot be generated in HTML with normal keyboard commands.commonly used character entities are:

Description	Entity Number	Entity Name	Symbol
Quotation mark	&#34;	&quot;	"
Ampersand	&#38;	&amp;	&
Less than	&#60;	&lt;	<
Greater than	&#62;	&gt;	>
Non-breaking Space			
Inverted exclamation	&#161;	&iexcl;	¡
Cent sign	&#162;	&cent;	¢
Pound sterling	&#163;	&pound;	£
Yen sign	&#165;	&yen;	¥
Copyright	&#169;	&copy;	©
Registered trademark	&#174;	&reg;	®
Fraction one-fourth	&#188;	&frac14;	¼
Fraction one-half	&#189;	&frac12;	½
Fraction three-fourths	&#190;	&frac34;	¾
Inverted question mark	&#191;	&iquest;	¿
en dash	&#8211;	&ndash;	–
em dash	&#8212;	&mdash;	—
dagger	&#8224;	&dagger;	†
horizontal ellipsis	&#8230;	&hellip;	…
euro	&#8364;	&euro;	€
trademark	&#8482;	&trade;	™

fig: character entity

HTML head

The <head> element contains the meta informaton about the document.HTML metadata is data about the HTML document. Metadata is not displayed.According to the HTML standard, only a few tags are legal inside the head section. These are: <base>, <link>, <meta>, <title>, <style>, and <script>.

HTML meta

The purpose of the< meta> element is to provide meta-information about the document.It is used to specify page description, keywords, author, and other metadata.Metadata is used by browsers (how to display content), by search engines (keywords), and other web services.We use three attributes (name, content, and http-equiv) with <meta> tag.We use name = “keywords” to provide information for a search engine.for example

<meta name="keywords" content="HTML, CSS, XML, XHTML, JavaScript"/>

We use name = “description” to define a description of your page. It is sort summary of the content of the page.for example:

<meta name="description" content="Free online educaton "/>

We use name = “author” and name = “copyright” for author and copyright details. For example,
<meta name="author" content="W3schools" />
<meta name="copyright" content="W3schools 2016" />

We use http-equiv = “expires” to refresh itself to the most current version or change to another location (page) entirely after some time. This is useful if you’ve moved a page to a new url and want any visitors to the old address to be quietly sent to the new location. For example,
<meta http-equiv = "refresh" content="50" />Refresh document every 50 seconds:

HTML Div tags

The element defines logical divisions within the document. This means that when you use a element, you are indicating that the enclosed content is a specific section of the page.The tag is used to group block-elements to format them with CSS.

<h3>This is a heading of first div</h3>
This is a paragraph of first div.

<divstyle="background-color:red;text-align:center">

red background of second div

HTML Events

HTML 4 added the ability to let events trigger actions in a browser, like starting a JavaScript when a user clicks on an element.Below is a list of attributes that can be inserted into HTML tags to define event actions.

Window Events

Only valid in body and frameset elements

Attribute	Value	Description
Onload	script	Script to be run when a document loads
Onunload	script	Script to be run when a document unloads

Form Element Events

Only valid in form elements.

Attribute	Value	Description

Onchange	script	Script to be run when the element changes
Onsubmit	script	Script to be run when the form is submitted
Onreset	script	Script to be run when the form is reset
Onselect	script	Script to be run when the element is selected
Onblur	script	Script to be run when the element loses focus
Onfocus	script	Script to be run when the element gets focus

Keyboard Events

Not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, and title elements.

Attribute	Value	Description
onkeydown	script	What to do when key is pressed
onkeypress	script	What to do when key is pressed and released
Onkeyup	script	What to do when key is released

Mouse Events

Not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, and title elements.

Attribute	Value	Description
onclick	script	What to do on a mouse click
ondblclick	script	What to do on a mouse doubleclick
onmousedown	script	What to do when mouse button is pressed
onmousemove	script	What to do when mouse pointer moves
onmouseover	script	What to do when mouse pointer moves over an element
onmouseout	script	What to do when mouse pointer moves out of an element
onmouseup	script	What to do when mouse button is released

Things to remember

- Html stands for hypertext markup language.it is invented by the scientist Tim burners lee in 1990
- .HTML is not a programming language, it is a markup language which use markup tags to describe the web page. HTML tags are keywords surrounded by angle brackets
- HTML document have **.html** or **.htm** extension.
- HTML elements consist of a Start tag ,some content and ,end tag.
- Attributes provide additional information about HTML elements. Attributes are always specified in the start tag. Attributes come in name/value pair
- comments are ignored by browser..HTML comments are written between <!--and-->

Review of CSS

Introduction

CSS stands for cascading style sheets. It defines the look and feel of the Web pages. It was intended to allow developers to separate content from design and layout so that HTML could perform more of the function

that it was originally based on - the markup of content, without worry about the design and layout.(About.com) CSS is used to separate style from content. It can control the layout of multiple web pages all at once.External stylesheets are stored in CSS files.

Syntax

A CSS rule set consists of two blocks a selector and a declaration block:The selector normally 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. A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.(w3school)

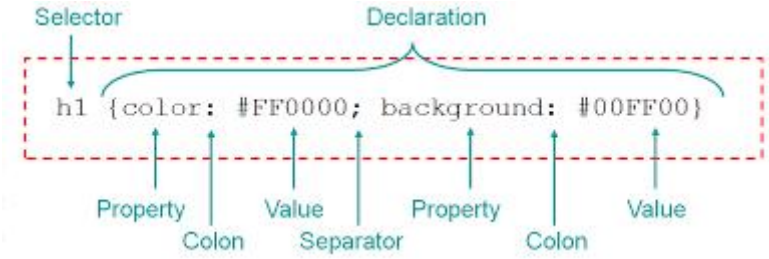


fig:CSS syntax

Inserting CSS

There are three different ways to insert stylesheet in the HTML document they are :(w3school)

1. External style sheet
2. Internal style sheet
3. Inline style

External style sheet

If you want to change thelook of an entire website by changing just one file,use External stylesheet.butEach page must include a reference to the external style sheet file inside the <link> element. The <link> element goes inside the <head> section.for example:

<head>
<link rel="stylesheet" type="text/css" href="style.css">
</head>The style sheet file must be saved with a .css extension.

you can writestyle sheet in any text editor but the file must be saved with a .css extension. an example of an external style sheet

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

h1 {
color: blue;
margin-left: 10px;
}
```

Internal style sheet

If you wanta unique style toone single page,use internal style sheet.Internal styles are defined within the <style> element, inside the <head> section of an HTML page for example:

```
<head>
<style>
body {
background-color: blue;
}

h1 {
color:red;
margin-left: 50px;
}
</style>
</head>
```

Inline style

It is 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.for example:

```
<p style="color:blue;margin-left:20px">This is a paragraph.</p>
```

inline style has the highest priority.i.e,it will override a style defined inside the <head> tag, or in an external style sheet, or in a browser (a default value).

CSS Comment

CSS comment starts with /* and ends with */. Comments can also span multiple lines.they areused to explain the code, and may help when you edit the source code at a later date.CSS comments are ignored by browser.for example

```
<style>
h2 {
color: green;
/* This is a single-line comment */
text-align: left;
}

/* This is
a multi-line
comment */
</style>
```

CSS Selectors

Selectors are used to select the HTML element based on their element name ,class ,id, attributes etc.

Element selector

It selects elements based on the element name.eg :

You can select <h2> elements on a page like this (in this case, all <h2> elements will be center-aligned, with a green text color)

```
<html>
```

```
<head>
<style>
h2 {
text-align: center;
color: green;
}
</style>
</head>
<body>

<h2>Every h2 head will be affected by the style.</h2>
<h2>And me also!</h2>

</body>
</html>
```

ID Selector

It uses the id attribute of an HTML element to select a specific element.The id must be unique within a page, so that id selector is used to select one unique element.To select an element with a specific id, use hash (#) character, followed by the id of the element.for example:

```
<html>
<head>
<style>
#id1 {
text-align: left;
color: blue;
}
</style>
</head>
<body>

<p id="id1">Hello World!</p>
<p>This paragraph is not affected by the style.</p>

</body>
</html>
```

Class selector

It selects elements with a specific class attribute.To select elements with a specific class, use period (.) character, followed by the name of the class.for example:

```
<html>
<head>
<style>
.center {
text-align: center;
color: blue;
}
</style>
```

```
</head>
<body>

<h1 class="center">Hello world .thi is blue and center align heading </h1>
<p class="center">This is paragraph.</p>

</body>
</html>
```

You can also specify only specific HTML elements should be affected by a class for example:

```
<html>
<head>
<style>
h1.center {
text-align: center;
color: blue;
}
</style>
</head>
<body>

<h1 class="center">Hello world .This is blue and center align heading </h1>
<p class="center">This is paragraph.and will not affected</p>

</body>
</html>
```

Grouping Selector

If you have elements with the same style definitions thanIt will be better to group the selectors, to minimize the code you can use comma(,) to separate each selector.for example:

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

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

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

we group selectors from above code as follow:

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

```
color: blue;
}
```

CSS Background

The CSS background properties are used to define the background effects of elements.

CSS background properties are:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position

background color

The background-color property specifies the background color of an element.It is set like this:

```
h2 {
background-color: lightgreen;
}
```

With CSS, a color is most often specified by:

- a valid color name - like "green"
- a HEX value - like "#00ff00"
- an RGB value - like "rgb(0,255,0)"

background image

The background-image property specifies an image to use as the background of an element. By default, the image is repeated so it covers the entire element.The background image for a page can be set like this:

```
body {

background-image:url('photo.gif');

}
```

use an image that does not disturb the text.

Repeat Horizontally or Vertically

By default, the background-image property repeats an image both horizontally and vertically.Some images should be repeated only horizontally or vertically, or they will look strange To repeat an image only horizontally set background-repeat: repeat-x;and verticallyset background-repeat: repeat-y;

Background Image - Set position and no-repeat

Showing the background image only once is specified by the background-repeat property for example:

```
<style>
body {
```

```
background-image: url("myphoto.png");
background-repeat: no-repeat;
}
</style>
```

you can also change the position of the image. The position of the image is specified by the background-position property for example:

```
<head>
<style>
body {
background-image: url("myphoto.png");
background-repeat: no-repeat;
background-position: right top;
}
</style>
</head>
```

Background Image - Fixed position

To specify that the background image should be fixed , use the background-attachment property:for example:

```
<head>
<style>
body {
background-image: url("myphoto.png");
background-repeat: no-repeat;
background-position: right top;
background-attachment: fixed;
}
</style>
</head>
```

Background - Shorthand property

It is also possible to specify all the background properties in one single property. This is called a shorthand property.The shorthand property for background is background for example:

```
<head>
<style>
body {
background: #00ff00 url("myphoto.png") no-repeat right top;
}
</style>
</head>
```

When using the shorthand property the order of the property values is:

- background-color
- background-image
- background-repeat
- background-attachment

- background-position

It does not matter if one of the property values is missing, as long as the other ones are in this order.(w3school)

The CSS Box Model

All HTML elements can be considered as boxes. 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.

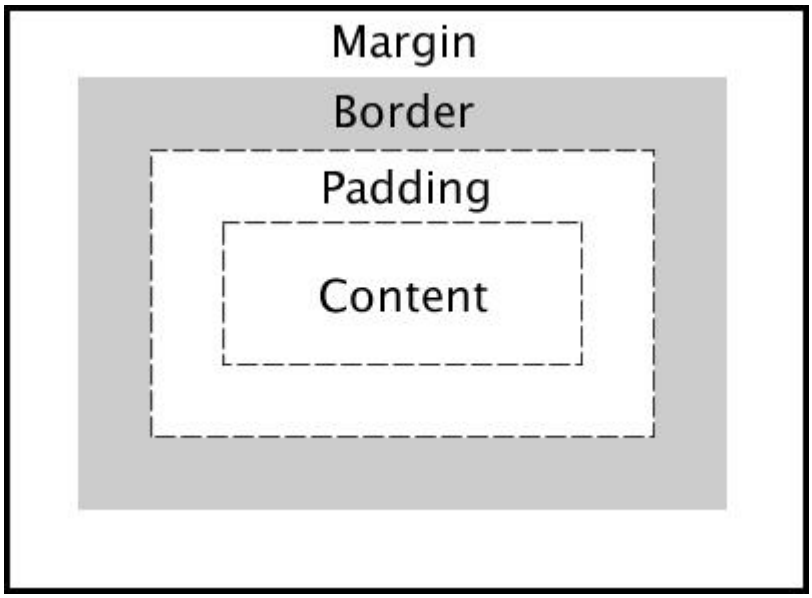


fig: CSS box model

- **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.for example

```
<head>
<style>
div {
background-color: lightgrey;
width: 300px;
padding: 25px;
border: 25px solid navy;
margin: 25px;
}
</style>
</head>
```

CSS Padding Property

The CSS padding properties define the white space between the element content and the element border.The padding clears an area around the content (inside the border) of an element.There are CSS properties for setting the padding for each side of an element (top, right, bottom, and left).

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

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

for example

```
p {
padding-top: 30px;
padding-right: 20px;
padding-bottom: 30px;
padding-left: 600px;
}
```

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

```
p {
padding: 30px 20px 30px 60px;
}
```

this property have one of the four value

If thepaddingproperty has four values:

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

If thepaddingproperty has three values:

- **padding: 25px 50px 75px;**
  - top padding is 25px
  - right and left paddings are 50px
  - bottom padding is 75px

If thepaddingproperty has two values:

- **padding: 25px 50px;**
  - top and bottom paddings are 25px
  - right and left paddings are 50px

If thepaddingproperty has one value:

- **padding: 25px;**
  - all four paddings are 25px

Pseudo classes

A pseudo-class is used to define a special effect of an element.For example,

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

The syntax of pseudo-classes

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

The most commonly used pseudo-classes are as follows:

:link:Use this class to add special style to an unvisited link.

:visited:Use this class to add special style to a visited link.

:hover:Use this class to add special style to an element when you mouse over it.

:active:Use this class to add special style to an active element.

:focus:Use this class to add special style to an element while the element has focus.

:first-child:Use this class to add special style to an element that is the first child of some other element.

:lang:Use this class to specify a language to use in a specified element.

(Tutorialspoint)

Links can be displayed in different ways

```
/* unvisited link */
a:link {
color: #FF0000;
}

/* visited link */
a:visited {
color: #00FF00;
}

/* mouse over link */
a:hover {
color: #FF00FF;
}

/* selected link */
a:active {
color: #0000FF;
}
```

a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective!a:active MUST come after a:hover in the CSS definition in order to be effective! Pseudo-class names are not case-sensitive.

CSS pseudo-elements

A CSS pseudo-element is used to style specified parts of an element.

For example:

- Style the first letter, or line, of an element
- Insert content before, or after, the content of an element

syntax of pseudo-elements

```
selector::pseudo-element {
property:value;
}
```

The ::first-line,::first-letter,::after,::before,::selectionPseudo-elements

::first-line: The ::first-line pseudo-element is used to add a special style to the first line of a text.

::first-letter: The ::first-letter pseudo-element is used to add a special style to the first letter of a text.

::after: The ::after pseudo-element can be used to insert some content after the content of an element.

::before: The ::before pseudo-element can be used to insert some content before the content of an element.

::selection:The ::selection pseudo-element matches the portion of an element that is selected by a user.The following CSS properties can be applied to ::selection: color, background, cursor, and outline

example of ::before:

<html> <head> <style type="text/css"> p:before { content: url("bullet.gif") } </style> </head> <body> This line will be preceded by a bullet. This line will be preceded by a bullet. This line will be preceded by a bullet. </body> </html>

Things to remember

- CSS stands for cascading style sheets. It defines the look and feel of the Web pages.
- A CSS rule set consists of two blocks a selector and a declaration block:The selector normally points to the HTML element you want to style. The declaration block contains one or more declarations separated by semicolons.
- There are three different ways to insert stylesheet in the HTML document they are :(w3school)
    1. External style sheet
    2. Internal style sheet
    3. Inline style
- If you want to change thelook of an entire website by changing just one file,use External stylesheet.butEach page must include a reference to the external style sheet file inside the <link> element. The <link> element goes inside the <head> section
- If you wanta unique style toone single page,use internal style sheet.Internal styles are defined within the <style> element, inside the <head>

- It is used to apply a unique style for a single element.To use inline styles, add the style attribute to the relevant element.
- CSS comment starts with /* and ends with */.
- Element selector selects elements based on the element name .
- ID selector uses the id attribute of an HTML element to select a specific element.To select an element with a specific id, use hash (#) character
- class selector selects elements with a specific class attribute.To select elements with a specific class, use period (.) character, followed by the name of the class
- if you have elements with the same style definitions thanIt will be better to group the selectors, to minimize the code you can use comma(,) to separate each selector
- The CSS background properties are used to define the background effects of elements.The CSS background properties are used to define the background effects of elements.

  CSS background properties are:

  - background-color
  - background-image
  - background-repeat
  - background-attachment
  - background-position

All HTML elements can be considered as boxes. 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.

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

A CSS pseudo-element is used to style specified parts of an element.

Client Side Programming (review of JavaScript)

Introduction

JavaScript is a dynamic client-side computer programming language. JavaScript code is embedded within an HTML page using the JavaScript tag. The <script> tag is used to embed JavaScript code.It is an interpreted programming language with object-oriented capabilities.JavaScript was invented by Brendan Eich in 1995, and became an ECMA standard in 1997. The official name of the javascript is ECMA-262. ECMAScript 6 (released in June 2015) is the latest official version of JavaScript.

JavaScript code can be embedded in:

- An external file
- The header of the page
- The body of the page

Writing comments

To make JavaScript code more readable we use comments.It is also used to prevent execution when testing alternative code.Javascript comments are of two types:single line comment and multi-line comment.

Single line comment: Any text between // and the end of the line is treated as a single line comment.which is ignored by JavaScript.

Multiline comment: Multi-line comments start with /* and end with */.Any text between /* and */ will be ignored by JavaScript.(w3school)

For example:

```
<script language="javascript" type="text/javascript">

// This is is single line comment .

/*

* This is a multiline comment in JavaScript

* and will ignored byJavaScript

*/

</script>
```

JavaScript variables

JavaScript variables are containers for storing data values.You can place data into these containers and then refer to the data simply by naming the container.A variable can have short name like x,y,z, or more descriptive name like First_name, person,carname etc.All JavaScript variables must be identified with unique names.

The general rules for constructing names for variables are:

- Names can contain letters, digits, underscores, and dollar signs.
- Names must begin with a letter
- Names can also begin with $ and _ .
- Names are case-sensitive (a and A are different variables)
- Reserved words (like JavaScript keywords) cannot be used as names.

Declaring a JavaScript variables

var keyword is used to declare the JavaScript variable.for eg:

```
var name;

var age;

var address;
```

After the declaration, the variable has no value.Equal sign("=") is used to assign the value to variable:for eg:

```
name="kullabs";

adderss="Kathmandu";
```

You can also assign a value to the variable when you declare it.for eg:

```
var name="kullabs";
```

You can declare many variables in one statement.Start the statement with **var** and separate the variables by **comma.**for eg:

var name="kullabs",address="Kathmandu";

If you re-declare a JavaScript variable, it will not lose its value.for example:

var name="kullabs";

var name;

The variable name will still have the value "kullabs" after the execution of these statements.

As with algebra, you can do arithmetic operations with JavaScript variables: for eg:

```
var x=10;
var y=x-5;

var z=y+5;
```

but in case of strings will be concatenated.for example;

var companyName = "kul" + "" + "labs";

[removed](companyName);

output:kullabs

JavaScript Operators

JavaScript supports the following types of operators.(tutorialspoint)

- Arithmetic Operators
- Comparison Operators
- Logical (or Relational) Operators
- Assignment Operators
- Conditional (or ternary) Operators

Arithmetic operators

Arithmetic operators are used to perform arithmetic operations on numbers.

Addition (+) : Adds two operands

Subtraction(-): Subtracts the second operand from the first

Multiplication(*):Multiply both operands

Division(/): Divide the numerator by the denominator

Modulus(%): Outputs the remainder of an integer division

Increment(++): Increases an integer value by one

Decrement(--): Decreases an integer value by one

Comparison Operators

Comparison operators are used in logical statements to determine equality or difference between variables or values.

equal to(==):Checks if the value of two operands are equal or not, if yes, then the condition becomes true.

not equal(!=) :Checks if the value of two operands are equal or not, if the values are not equal, then the condition becomes true.

greater than(>):Checks if the value of the left operand is greater than the value of the right operand, if yes, then the condition becomes true.

less than(<):Checks if the value of the left operand is less than the value of the right operand, if yes, then the condition becomes true.

greater than or equal to(>=):Checks if the value of the left operand is greater than or equal to the value of the right operand, if yes, then the condition becomes true.

less than or equal to(<=):Checks if the value of the left operand is less than or equal to the value of the right operand, if yes, then the condition becomes true.

Logical (or Relational) Operators:

Logical operators are used to determine the logic between variables or values.

AND(&&):If both the operands are non-zero, then the condition becomes true.

OR(||):If any of the two operands are non-zero, then the condition becomes true.

NOT(!):Reverses the logical state of its operand. If a condition is true, then the Logical NOT operator will make it false.

Assignment Operators:

Assignment operators assign values to JavaScript variables

= (Simple Assignment): Assigns values from the right side operand to the left side operand

+= (Add and Assignment): It adds the right operand to the left operand and assigns the result to the left operand.

−= (Subtract and Assignment): It subtracts the right operand from the left operand and assigns the result to the left operand.

***= (Multiply and Assignment):**It multiplies the right operand with the left operand and assigns the result to the left operand.

/= (Divide and Assignment): It divides the left operand with the right operand and assigns the result to the left operand.

%= (Modules and Assignment): It takes modulus using two operands and assigns the result to the left operand.

Conditional operator(? :)

The conditional operator first evaluates an expression for a true or false value and then executes one of the two given statements depending upon the result of the evaluation.

JavaScript Statements

javaScript Statements are the instructions to be executed by the browser.semicolon separates the JavaScript statements.JavaScript statements often start with a **keyword** to identify the JavaScript action to be performed.Statement identifiers are reserved words and cannot be used as variable names.(w3school)

Here is a list of some of the keywords:

break:Terminates a switch or a loop

continue:Jumps out of a loop and starts at the top

debugger:Stops the execution of JavaScript, and calls (if available) the debugging function

do ... while:Executes a block of statements, and repeats the block, while a condition is true

for:Marks a block of statements to be executed, as long as a condition is true

function:Declares a function

if ... else:Marks a block of statements to be executed, depending on a condition

return: Exits a function

switch: Marks a block of statements to be executed, depending on different cases

try ...: catch: Implements error handling to a block of statements

var:Declares a variable

while: Marks a block of statements to be executed while a condition is true

These distinct groups of statements include:(tizag)

- Conditional Statements
- Loop Statements
- Object Manipulation Statements
- Comment Statements
- Exception Handling Statements

Conditional statements:

Conditional statements are used to perform different actions based on different conditions.

In JavaScript we have the following conditional statements:

- Use **if** to specify a block of code to be executed, if a specified condition is true
- Use **else** to specify a block of code to be executed, if the same condition is false

- Use **else if** to specify a new condition to test, if the first condition is false
- Use **switch** to specify many alternative blocks of code to be executed.

If Statement

we use if statement to specify a block of JavaScript code to be executed if a condition is true.

syntax:

```
if (condition) {
block of code to be executed if the condition is true
}
```

else statement

else statement is used to specify a block of code to be executed if the condition is false.

syntax:

```
if (condition) {
block of code to be executed if the condition is true
} else {
block of code to be executed if the condition is false
}
```

else if Statement

Use the **else if** statement to specify a new condition if the first condition is false;

syntax:

```
if (condition1) {
block of code to be executed if condition1 is true
} else if (condition2) {
block of code to be executed if the condition1 is false and condition2 is true
} else {
block of code to be executed if the condition1 is false and condition2 is false
}
```

Switch statement:

The objective of a switch statement is to give an expression to evaluate and several different statements to execute based on the value of the expression. The interpreter checks each case against the value of the expression until a match is found. If nothing matches, a default condition will be used.(tutorialspoint)

syntax:

```
switch (expression)

{case condition 1: statement(s)

break;
```

case condition 2: statement(s)

break;

...

case condition n: statement(s)

break;

default: statement(s)

}

The **break** statements indicate the end of a particular case. If they were omitted, the interpreter would continue executing each statement in each of the following cases.

Loop statements

Loop statements can execute a block of code a number of times.

JavaScript supports different kinds of loops:(w3school)

- **for** - loops through a block of code a number of times
- **for/in** - loops through the properties of an object
- **while** - loops through a block of code while a specified condition is true
- **do/while** - also loops through a block of code while a specifie

for loop:

The for loop is used when you know in advance how many times the script should run.

syntax:

for (loop initialization; test statement; iteration statement) {
code block to be executed
}

example:

<html>

<body>

<script type="text/javascript">

var i=0;

for (i=0;i<=5;i++) {

[removed]("The number is " + i);

[removed]("<br />");

```
}

</script>

</body>

</html>
```

for..In loop

The for...in loop is used to loop through an object's properties.

syntax:

```
for (variablename in object){

statement or block to execute

}
```

In each iteration, one property from object is assigned to variablename and this loop continues till all the properties of the object are exhausted.

example:

```
<html>
<body>

<script type="text/javascript">

var aProperty;

for (aProperty in navigator) {
[removed](aProperty);
[removed]("<br />");
}

</script>

</body>
</html>
```

While Loop:

The while loop loops through a block of code as long as a specified condition is true.(w3school)

syntax:

```
while (condition) {
code block to be executed
}
```

example:

```
<html>
<body>
<script type="text/javascript">
var i=0;
while (i<=5) {
[removed]("The number is " + i);
[removed]("<br />");
i++;
}
</script>
</body>
</html>
```

Do while loop

This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.(w3school)

syntax:

```
do {
code block to be executed
}
while (condition);
```

example:

```
<html>
<body>
<script type="text/javascript">
var i=0;
do {
[removed]("The number is " + i);
[removed]("<br />");
i++;
}while (i<=5);
</script>
</body>
</html>
```

Things to remember

- JavaScript is a dynamic client-side computer programming language. JavaScript code is embedded within an HTML page using the JavaScript tag. The [removed]// <![CDATA[tag is used to embed JavaScript code.
- Javascript comments are of two types:single line comment(//single line comment) and multi-line comment(/*multi line comment*/).
- JavaScript variables are containers for storing data values..A variable can have short name like x,y,z, or more descriptive name like age,total etc.
- **var** keyword is used to declare the JavaScript variable.After the declaration, the variable has no value.Equal sign("=") is used to assign the value to variable
- You can declare many variables in one statement.Start the statement with **var** and separate the variables by **comma..**

- javaScript Statements are the instructions to be executed by the browser.semicolon **separats** the JavaScript statements.
- Conditional statements are used to perform different actions based on different conditions.In JavaScript we have the following conditional statements: if ,else,else if,switch.
- Loop statements can execute a block of code a number of times.JavaScript supports different kinds of loops.they are: for ,for/in,while,do/while .

Client Side Programming(review of JavaScript) II

Dialog Boxes

JavaScript supportthree types of Dialog Boxes. They are:

1. Alert box
2. Confirm box
3. prompt box

Alert box

Alert box is used to give the warning message to the users.When an alert box pops up, the user will have to click "OK" to proceed.

syntax:

alert("message");

example:

<html>

<head>

<script type="text/javascript">

function alertExample() {

alert("This is an alert box click OK to proceed !");

}

</script>

</head>

<body>

<input type="button" onclick="alertExample()" value="Show alert box" />

</body>

```
</html>
```

Confirm box

A confirm box is often used if you want the user to verify or accept something. When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed. If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

syntax

```
confirm("text")
```

example:

```
<html>

<head>

<script type="text/javascript">

function confirmExample() {

var r=confirm("Press a button");

if (r==true) {

[removed]("You pressed OK!");

}

else {

[removed]("You pressed Cancel!");

}

}

</script>

</head>
<body>

<input type="button" onclick="confirmExample()" value="Show confirm box" />

</body>

</html>
```

Prompt box

when you want to pop-up a text box to get user input.use prompt dialog box.it enables you to interact with the user. The user needs to fill in the field and then click either OK or "Cancel" to proceed after entering an

input value. If the user clicks "OK" the box returns the input value. If the user clicks "Cancel" the box returns null.

syntax:

```
prompt("sometext","defaultvalue");
```

example:

```
<html>
<head>

<script type="text/javascript">
function promptExample(){
var name =prompt("Enter your name : ","enter your name here");
[removed]("You have entered : " + name);
}
</script>

</head>

<body>

<form>
<input type="button" value="show Prompt" onclick="promptExample();" />
</form>

</body>
</html>
```

Functions

A function is a block of reusable code which can be called anywhere in your program to perform particular task.A JavaScript function is defined with the **function** keyword, followed by a **name**, followed by parentheses **().**Function names can contain letters, digits, underscores, and dollar signs.The parentheses may include parameter names separated by commas:(parameter1, parameter2, ...)and code is surrounded by curly brackets {}.(tutorialspoint)

syntax:

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

function functionName(parameter1,parameter2,..)

{

statements

}

</script
```

Calling a function

To call a function somewhere later in the script, you would simply need to write the name of that function.for example:

```
<html>

<head>

<script type="text/javascript">

function helloFunction()

{

[removed] ("Hello world !");

}

</script>

</head>

<body>

<form>

<p>click the button to call function</p>

<input type="button" onclick="helloFunction()" value="click here">

</form>

</body>

</html>
```

Function parameter

JavaScript gives facility to pass different parameters while calling a function. These passed parameters can be captured inside the function and any manipulation can be done over those parameters. A function can take multiple parameters separated by comma. example:

```
<html>

<head>

<script type="text/javascript">

function printNameAge(name, age)

{

[removed] (name + " is " + age + " years old.");
```

```
}

</script>

</head>

<body>

<p>Click the following button to call the function</p>

<form>

<input type="button" onclick="printNameAge('ram', 8)" value="click here">

</form>

</body>

</html>
```

Return statement:

A JavaScript function can have an optional **return** statement. This is required if you want to return a value from a function. When JavaScript reaches a **return statement**, the function will stop executing.

for example:

```
<html>
<head>

<script type="text/javascript">
function sum(a, b)
{
return a+b;
}
function result()
{
var total;
total = sum(2, 3);
[removed] (total );
}
</script>

</head>

<body>

<form>
<input type="button" onclick="result()" value="click here">
</form>

</body>
</html>
```

JavaScript Events

Events are things that happen to HTML element .When JavaScript is used in HTML page JavaScript can reacton these events.An HTML event can be something the browser does or something a user does.for example:When the page loads, it is called an event. When the user clicks a button, that click too is an event. Other examples include events like pressing any key, closing a window, resizing a window, etc.when events happen, you may want to do something.JavaScript lets you execute code when events are detected. HTML allows event handler attributes, with JavaScript code, to be added to HTML elements.

common HTML events and their description are given :

onchange:An HTML element has been changed

onclick:The user clicks an HTML element

onmouseover:The user moves the mouse over an HTML element

onmouseout:The user moves the mouse away from an HTML element

onkeydown:The user moves the mouse away from an HTML element

onload:The browser has finished loading the page

Error handling

There are three types of errors in programming:

1. **Syntax Errors**:eg:alert("I am missing a closing parenthesis **//syntax error**
2. **Runtime Errors**: Runtime errors, also called exceptions, occur during execution (after compilation/interpretation) eg:undefinedfunction()
3. **Logical Errors**: Logic errors can be the most difficult type of errors to track down. These errors are not the result of a syntax or runtime error. Instead, they occur when you make a mistake in the logic that drives your script and you do not get the result you expected.

There are two types of mechanism to handle error:

1. Using try…catch statement
2. Using onerror event

Try, catch and finally:

The**try**statement allows you to define a block of code to be tested for errors while it is being executed.The**catch**statement allows you to define a block of code to be executed, if an error occurs in the try block.The JavaScript statements**try**and**catch**come in pairs:

syntax:

try {

Block of code to try

}

```
catch(err) {
Block of code to handle errors
}
```

The **finally** statement lets you execute code, after try and catch, regardless of the result

```
try {
Block of code to try
}
catch(err) {
Block of code to handle errors
}
finally {
Block of code to be executed regardless of the try / catch result
}
```

example:

```
<html>
<head>
<script type="text/javascript">
function myFunc()
{
var a = 100;
try {
var result=a/0;
}
catch ( e ) {
alert("Error: " + e.description );
}
finally{
```

```
[removed] (“this block is always execute”);

}

</script>

</head>

<body>

<p>Click the following to see the result:</p>

<form>

<input type="button" value="Click Me" onclick="myFunc();" />

</form>

</body>

</html>
```

Throw keyword

You can use**throw**statement to create your costume exceptions. Later these exceptions can be captured and you can take an appropriate action.

```
<html>

<head>

<script type="text/javascript">

function myFunc()

{

var a = 100;

var b = 0

try{

if ( b == 0 ){

throw( "Divide by zero error." );

}

else

{
```

```
var c = a / b;

}

}

catch ( e ) {

alert("Error: " + e );

}

}

</script>

</head>

<body>

<p>Click the following to see the result:</p>

<form>

<input type="button" value="Click Me" onclick="myFunc();" />

</form>

</body>

</html>
```

Using onerror event

The onerror event handler was the first feature to facilitate error handling in JavaScript. The error event is fired on the window object whenever an exception occurs on the page.The onerror event handler provides three pieces of information to identify the exact nature of the error :

Error message: The same message that the browser would display for the given error

URL: The file in which the error occurred

Line number: The line number in the given URL that caused the error

Here is the example to show how to extract this information.

```
<html>

<head>

<script type="text/javascript">

window.onerror = function (msg, url, line) {
```

```
alert("Message : " + msg );

alert("url : " + url );

alert("Line number : " + line );

}

</script>

</head>

<body>

<p>Click the following to see the result:</p>

<form>

<input type="button" value="Click Me" onclick="myFunc();" />

</form>

</body>

</html>
```

Form validation

Form validation is the process of checking that a form has been filled correctly before it is processed. For example, if your form has a box for the user to type their name you might want your form handler to check that they've filled in their name before you deal with the rest of the form.There are two main approaches for validating forms: server-side validation , and client-side side validation(usually javaScript is used).Server-side validation is more secure but often more tricky to code and it also increases load of server computer, whereas client-side (JavaScript) validation is easier to do and quicker too (the browser doesn't have to connect to the server to validate the form, so the user finds out instantly if they've missed out that required field!) and it also decreases the load .(Elated)(tizag)

Form validation- checking empty field:

It is a most common type of validationYou want to be sure that your visitors enter data into the HTML fields you have "required" for a valid submission.below code checks if a field has been left empty.If s field is blank it alert a alert message .

```
<html>
<body>

<script type='text/javascript'>
function chkEmpty()
{
var v= document.getElementById('elem').value;
if(v.length ==0)
{
alert("Field should not be empty:");
```

```
document.getElementById('elem').value="";
document.getElementById('elem').focus();
}
}
</script>
<form>
Required Field: <input type='text' id='elem'/>
<input type='button' onclick="chkEmpty()" value='Check'/>
</form>
</body>
</html>
```

Form Validation - Checking for All Numbers

If someone is entering a credit card, phone number, zip code, similar information you want to be able to ensure that the input is all numbers. The quickest way to check if an input's string value is all numbers is to use a regular expression /^[0-9]+$/ that will only match if the string is all numbers and is at least one character long.

```
<html>
<body>

<script type='text/javascript'>
function allNumber()
{
var patt=/^[0-9]+$/;
var v= document.getElementById('elem').value;
if(v.match(patt))
{
alert("valid entry");
}
else
{
alert("Invalid entry:");
document.getElementById('elem').value="";
document.getElementById('elem').focus();
}
}
</script>
<form>
Required Field: <input type='text' id='elem'/>
<input type='button' onclick="allNumber()" value='Check'/>
</form>
</body>
</html>
```

Form Validation - Checking for All Letters

If we wanted to see if a string contained only letters we need to specify an expression that allows for both lowercase and uppercase letters: /^[a-zA-Z]+$/ .

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

```
<script type='text/javascript'>
function allLetter()
{
var patt=/^[a-zA-Z]+$/;
var v= document.getElementById('elem').value;
if(v.match(patt))
{
alert("valid entry");
}
else{
alert("Invalid entry:");
document.getElementById('elem').value="";
document.getElementById('elem').focus();
}
}
</script>
<form>
Required Field: <input type='text' id='elem'/>
<input type='button' onclick="allLetter()" value='Check'/>
</form>


</body>
</html>
```

Form Validation - Restricting the Length

Being able to restrict the number of characters a user can enter into a field is one of the best ways to prevent bad data. Below we have created a function that checks for length of input.

```
<html>
<body>
<script type='text/javascript'>
function validate()
{
var minlen=6;
var v= document.getElementById('elem').value;
if(v.length<6)
{
alert("User password must have at least 6 Characters");
document.getElementById('elem').value="";
document.getElementById('elem').focus();
}
else
{alert("Valid entry:");
}
}
</script>
<form>
Password: <input type='password' id='elem'/>
<input type='button' onclick="validate()" value='Check'/>
</form>
```

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

Form Validation - Selection Made

To be sure that someone has actually selected a choice from an HTML select input you can use a simple trick of making the first option as helpful prompt to the user and a red flag to you for your validation code. By making the first option of your select input something like "Please Choose" you can spur the user to both make a selection and allow you to check to see if the default option "Please Choose" is still selected when he/she submit the form.

```
<html>
<head>
<script type='text/javascript'>
function check()
{
var si=document.getElementById('con').selectedIndex;
var v= document.getElementById('con').options[si].text;
if(v=="Please Choose")
{
alert("You must choose the number");
}
else
{
alert("You choose :"+v);

}
}
</script>
<form>
Choose number: <select id='con'>
<option>Please Choose</option> <option>1</option>
<option>2</option>
<option>3</option><option>4</option><option>5</option><option>6</option><option>7</option><optio
n>8</option><option>9</option><option>10</option>
</select>
<input type='button' onclick='check()' value='submit'/>
</form>

</body>
</html>
```

Validating radio buttons

Radio buttons are used if we want to choose any one out of many options such as gender. In such case any one of the radio button must be selected. We can validate radio button selection as below:

```
<html>
<head>
<script type='text/javascript'>
function validate()
{
var gender=document.getElementsByName("gen");
if(gender[0].checked==false && gender[1].checked==false)
```

```
{
alert("You must choose Gender");
}
else
{
if(gender[0].checked==true)
alert("Male");
else
alert("Female");
}
}

</script>
<form>
Select Gender:
<input type=radio name='gen'>Male
<input type=radio name='gen'>Female
<input type='button' onclick='validate()' value='Check'/>
</form>

</body>
</html>
```

Form Validation - Email Validation

Every email is made up for 5 parts:
1. A combination of letters, numbers, periods, hyphens, plus signs, and/or underscores
2. The at symbol @
3. A combination of letters, numbers, hyphens, and/or periods
4. A period
5. The top level domain (com, net, org, us, gov, ...)

below code used to validate the email :

```
<html>
<head>
<script type='text/javascript'>
function validEmail()
{
var patt=/^[\w\-\.\+]+\@[a-zA-Z0-9\.\-]+\.[a-zA-z0-9]{2,4}$/;
var v= document.getElementById('elem').value;
if(v.match(patt))

{
alert("valid Email");
}
else
{
alert("Invalid Email"); document.getElementById('elem').value="";
document.getElementById('elem').focus();
}
}
</script>
```

```
<form>
Email ID: <input type='text' id='elem'/>
<input type='button' onclick="validEmail()" value='Check'/>
</form>

</body>
</html>
```

complete form validation example

(docslide)

```
<html>
<head>
<script type='text/javascript'>

function formValidator(){
// Make quick references to our fields
var firstname = document.getElementById('firstname');
var addr = document.getElementById('addr');
var phone = document.getElementById('phone');

var email = document.getElementById('email');
var age = document.getElementById('age');


// Check each input in the order that it appears in the form!
if(isAlphabet(firstname, "Please enter only letters for your name")){
if(isAlphanumeric(addr, "Numbers and Letters Only for Address")){
if(isNumeric(phone, "Please enter a valid phone number ")){

if(emailValidator(email, "Please enter a valid email address")){
if(madeSelection(Age, "Please Choose your age")){
return true;

}
}
}
}
}


return false;

}

function notEmpty(elem, helperMsg){
if(elem.value.length == 0){
elem.focus(); // set the focus to this input
return false;
}
return true;
}
```

```
function isNumeric(elem, helperMsg){
var numericExpression = /^[0-9]+$/;
if(elem.value.match(numericExpression)){
return true;
}else{
alert(helperMsg);
elem.focus();
return false;
}
}

function isAlphabet(elem, helperMsg){
var alphaExp = /^[a-zA-Z]+$/;
if(elem.value.match(alphaExp)){
return true;
}else{
alert(helperMsg);
elem.focus();
return false;
}
}

function isAlphanumeric(elem, helperMsg){
var alphaExp = /^[0-9a-zA-Z]+$/;
if(elem.value.match(alphaExp)){
return true;
}else{
alert(helperMsg);
elem.focus();
return false;
}
}


function madeSelection(elem, helperMsg){
if(elem.value == "Please Choose"){
alert(helperMsg);
elem.focus();
return false;
}else{
return true;
}
}

function emailValidator(elem, helperMsg){
var emailExp = /^[\w\-\.\+]+\@[a-zA-Z0-9\.\-]+\.[a-zA-z0-9]{2,4}$/;
if(elem.value.match(emailExp)){
return true;
}else{
alert(helperMsg);
elem.focus();
return false;
}
```

```
}
</script>

<form onsubmit='return formValidator()' >
First Name: <input type='text' id='firstname' /><br />
Address: <input type='text' id='addr' /><br />
Phone number: <input type='text' id='phone' /><br />
Email: <input type='text' id='email' /><br />
Age: <select id='age'>
<option>Please Choose</option>
<option>15</option>
<option>16</option>
<option>17</option>
<option>16</option>
<option>18</option>


<input type='submit' value='submit' />
</form>

</body>
</html>
```

Cookies

A cookie is a variable that is stored on the visitor's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With JavaScript, you can both create and retrieve cookie values.

Example of cookies:

- Name cookie - The first time a visitor arrives to your web page, he or she must fill in her/his name. The name is then stored in a cookie. Next time the visitor arrives at your page, he or she could get a welcome message like "Welcome samrat" The name is retrieved from the stored cookie.
- Date cookie - The first time a visitor arrives to your web page, the current date is stored in a cookie. Next time the visitor arrives at your page, he or she could get a message like "Your last visit was on saturday april 1, 2017!" The date is retrieved from the stored cookie

Create a Cookie with JavaScript.(w3school)

JavaScript can create, read, and delete cookies with the **[removed]** property.With JavaScript, a cookie can be created like this

[removed] = "username=amir ";

You can also add an expiry date (in UTC time). By default, the cookie is deleted when the browser is closed:

[removed] = "username=amir; expires=Thu, 18 Dec 2017 12:00:00 UTC";

With a path parameter, you can tell the browser what path the cookie belongs to. By default, the cookie belongs to the current page.

[removed] = "username=amir; expires=Thu, 18 Dec 2017 12:00:00 UTC; path=/";

Delete a Cookie with JavaScript

Deleting a cookie is very simple. Just set the expires parameter to a passed date:

[removed] = "username=; expires=Thu, 01 Jan 1990 00:00:00 UTC";

Cookie Example

In the example to follow, we will create a cookie that stores the name of a visitor.The first time a visitor arrives to the web page, he will be asked to fill in his name. The name is then stored in a cookie.The next time the visitor arrives at the same page, he will get a welcome message.

For the example we will create 3 JavaScript functions:

1. A function to set a cookie value
2. A function to get a cookie value
3. A function to check a cookie value

```
<html>
<head>
<script>

function setCookie(cname,cvalue,exdays) {
var d = new Date();
d.setTime(d.getTime() + (exdays*24*60*60*1000));
var expires = "expires=" + d.toGMTString();<br />[removed] = cname+"="+cvalue+"; "+expires;
}

function getCookie(cname) {
var name = cname + "=";
var ca = [removed].split(';');
for(var i=0; i<ca.length; i++) {
var c = ca[i];
while (c.charAt(0)==' ') {
c = c.substring(1);
}
if (c.indexOf(name) == 0) {
return c.substring(name.length, c.length);
}
}
return "";
}

function checkCookie() {
var user=getCookie("username");
if (user != "") {
alert("Welcome again " + user);
} else {
user = prompt("Please enter your name:","");
if (user != "" && user != null) {
setCookie("username", user, 30);
```

```
}
}
}

</script>
</head>
<body onload="checkCookie()">
</body>
</html>
```

The parameters of the first function above are the name of the cookie (cname), the value of the cookie (cvalue), and the number of days until the cookie should expire (exdays). The function sets a cookie by adding together the cookie name, the cookie value, and the expires string

In second function,

Take the cookiename as parameter (cname).Create a variable (name) with the text to search for (cname + "=").Split [removed] on semicolons into an array called ca (ca = [removed].split(';')).Loop through the ca array (i=0;i<ca.length;i++), and read out each value c=ca[i]).If the cookie is found (c.indexOf(name) == 0), return the value of the cookie (c.substring(name.length,c.length).If the cookie is not found, return "".

Last, we create the function that checks if a cookie is set.If the cookie is set it will display a greeting.If the cookie is not set, it will display a prompt box, asking for the name of the user, and stores the username cookie for 365 days, by calling the setCookie function.

Things to remember

- JavaScript supportthree types of Dialog Boxes. They are:
  - Alert box
  - Confirm box
  - prompt box
- Alert box is used to give the warning message to the users.When an alert box pops up, the user will have to click "OK" to proceed.syntax:alert("message");
- A confirm box is often used if you want the user to verify or accept something. When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed. syntax: confirm("text")
- when you want to pop-up a text box to get user input.use prompt dialog box.it enables you to interact with the user. The user needs to fill in the field and then click either OK or "Cancel" to proceed after entering an input value.syntax: prompt("sometext","defaultvalue");
- A function is a block of reusable code which can be called anywhere in your program to perform particular task.A JavaScript function is defined with the **function** keyword, followed by a **name**, followed by parentheses **().**code is surrounded by curly brackets {}.
- Events are things that happen to HTML element .When JavaScript is used in HTML page JavaScript can reacton these events.An HTML event can be something the browser does or something a user does.
- Common HTML events are :onchange,onclick,onmouseover,onmouseout,onkeydown,onload.
- By Using try…catch statement and Using onerror event we handle the error in javaScript.
- Form validation is the process of checking that a form has been filled correctly before it is processed.
- A cookie is a variable that is stored on the visitor's computer. Each time the same computer requests a page with a browser, it will send the cookie too