

Web Design



HNDIT1022 – Web Design

Week 1: Introduction

M.A.C Jiffriya B.Sc, M.Sc, M.Phil(pera)

- Course Code HNDIT 1022
- No of Credits: 4
- Lecture : 2 hours
- Tutorial / Practical : 4 hours
- Student Activity : 7 hours

Course Aims

- The course focuses on the fundamentals of web internet and develop web site using HTML, CSS, Java Scripts, and the current developments in Web Design theory and the practice.

Learning Outcomes (LO)

- LO1: Design, develop and deploy a simple website using HTML, CSS and Java Scripts
- LO2: Design and develop interactive, visually appealing client-side programs.
- LO3: Describe HTML 5 and web-design tools.

The Internet, WWW, Web Browser & Search Engine



5

- The **Internet**, sometimes called simply "*the Net*," is a worldwide system of computer networks



6

Internet

- The **Internet** is a group of computer networks interconnected around the world.
- It is the largest communication network ever conceived.
- The connected networks are comprised of educational, commercial and government sites.
- The interconnected networks may be made up of any number of computers from two to infinity.
- The collection of sites residing on the Internet form one of the largest repositories of information in history.

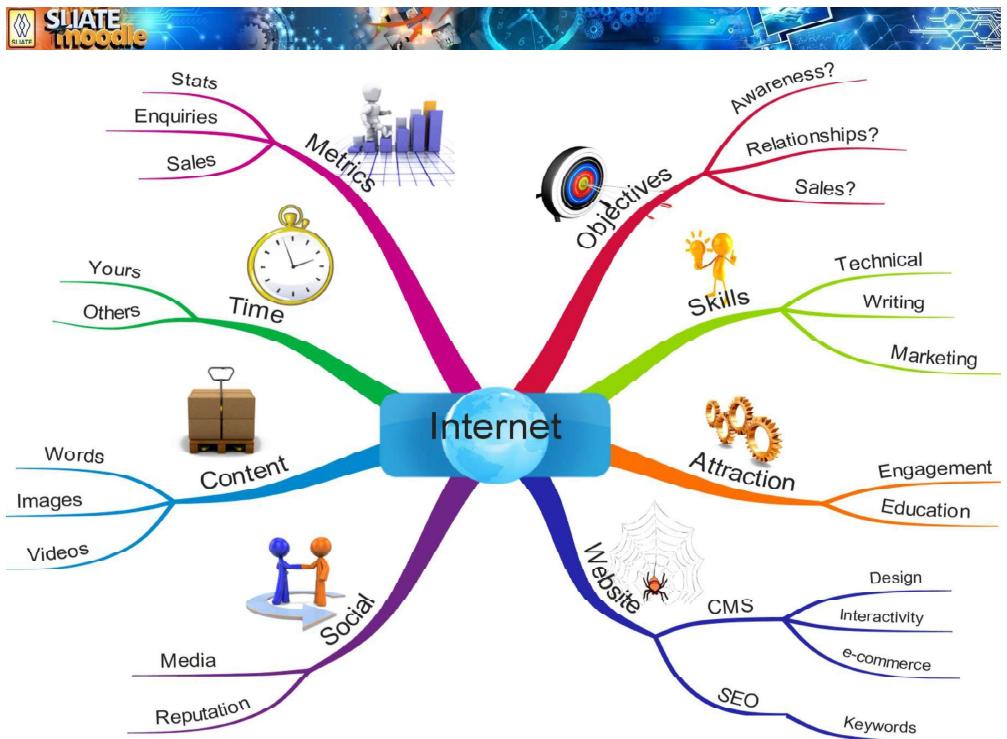
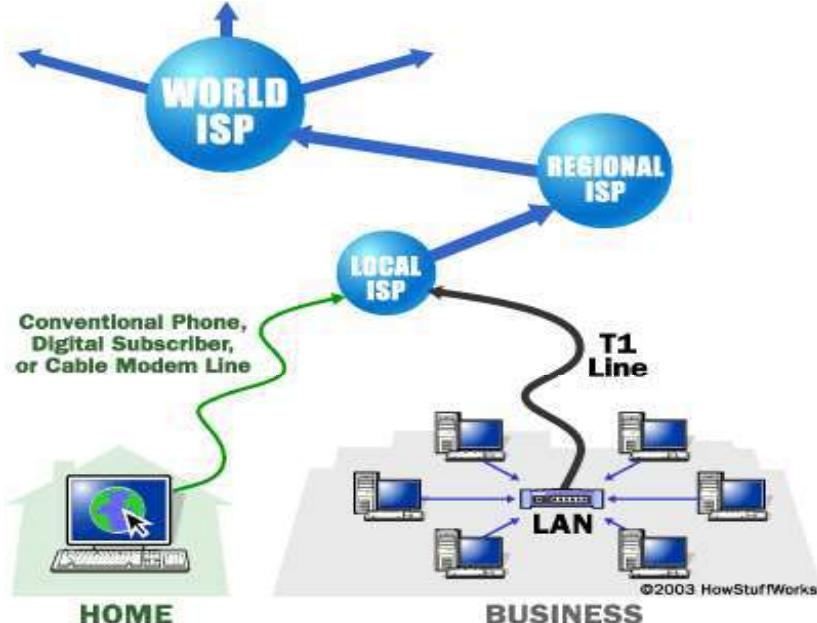
The Internet:

The Internet is a worldwide collection of networks that links millions of businesses, government agencies, educational institutions, and individuals.



8

How Internet structured

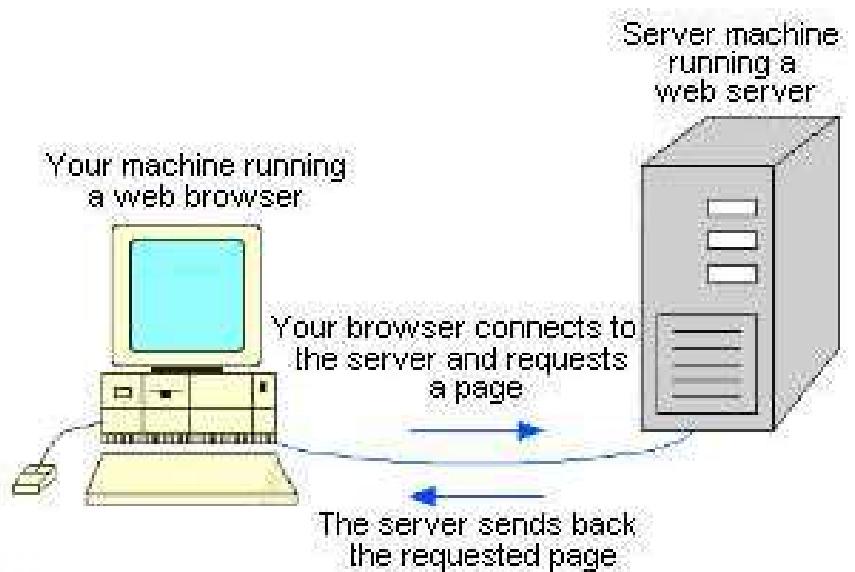


Computer Network

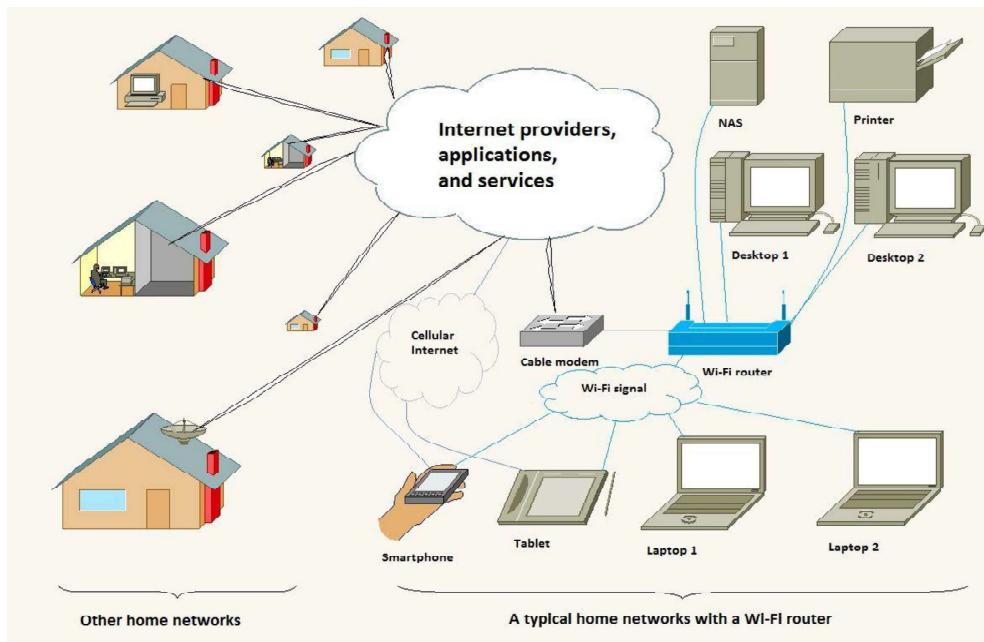
- A **computer network** is a collection of computers and devices interconnected by communications channels that facilitate **communications** among users and allows users to share resources.



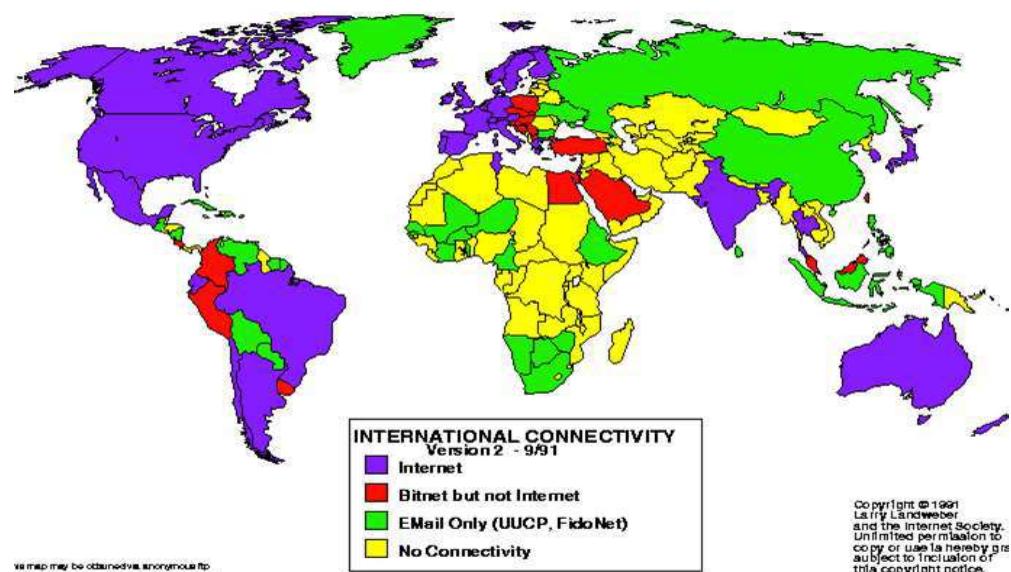
How it works



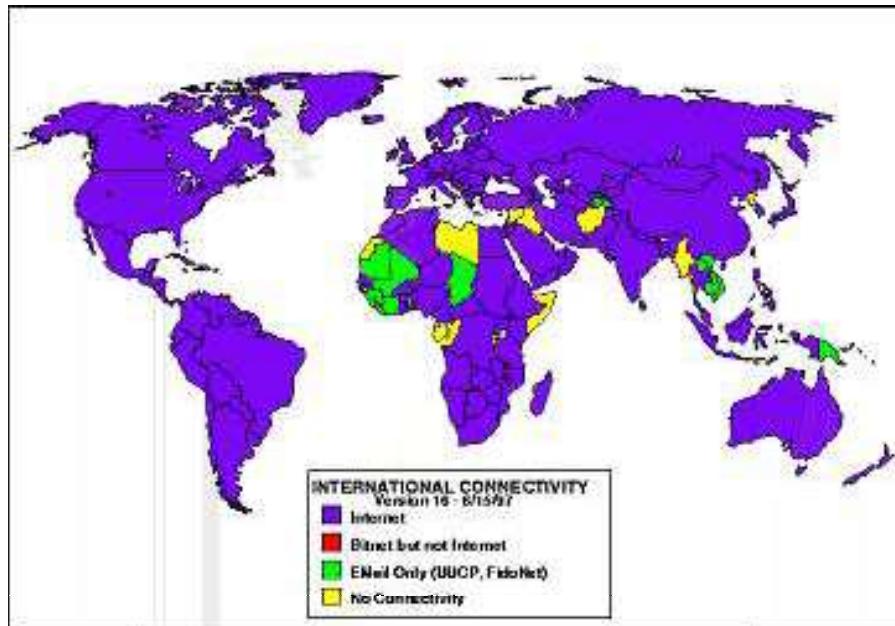
How internet structured ?



World wide Internet presence - 1991



World Internet Presence, 1997



Global Presence

March 2015

Updated



Global cyberspace connections are made by satellites...



Or undersea cables...



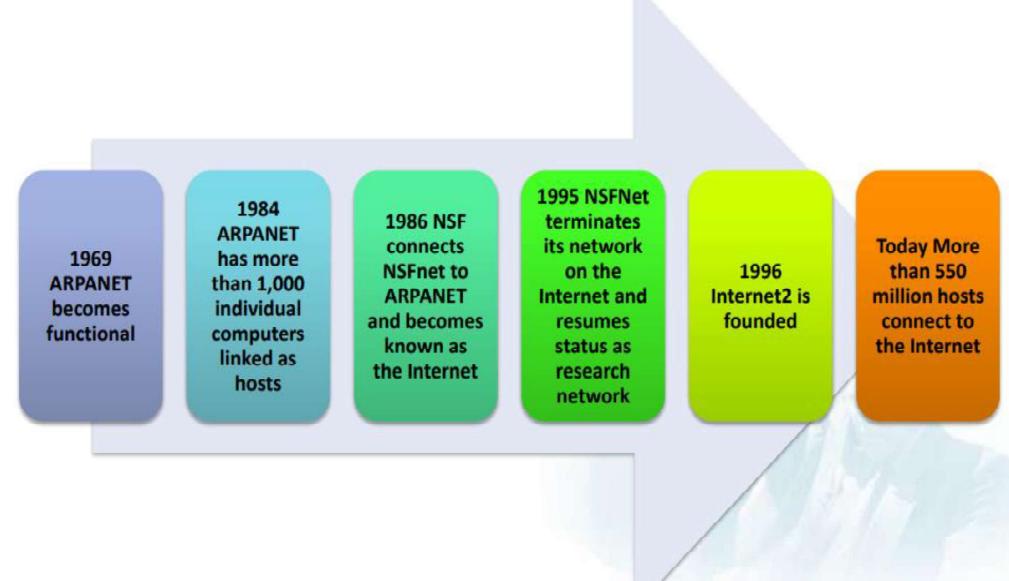
Origins of the Internet

- The Internet originated as ARPANET in September 1969 and had two main goals:

Allow scientists at different physical locations to share information and work together

Function even if part of the network were disabled or destroyed by a disaster

Evolution of the internet



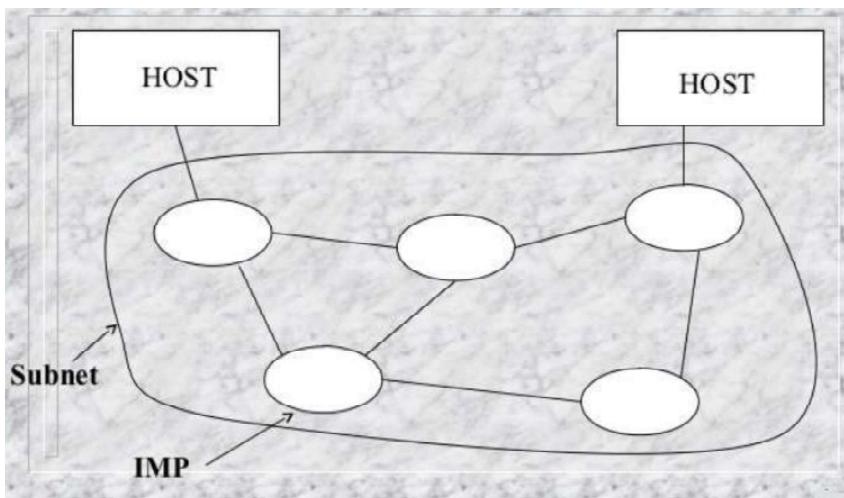
What is ARPANET?

- ARPA (Advanced Research Projects Agency), is an agency of the U.S. Department of Defense.
- The network which called ARPANET, became functional in September 1969, *linking scientific and academics researchers across U.S.*
- ARPANET consisted of 4 main computers located at:
 - University of California, Los Angeles.
 - University of California, Santa Barbara.
 - The Stanford Research Institute.
 - University of Utah.
- Each of these computers served as a **host on the network**, commonly known today as **server**

21

22

ARPANET



23

What is NSF?

- NSF is the National Science Foundation, that connected its huge network of five supercomputer centers, called NSFnet to ARPANET in 1986.
- The configuration of complex networks and hosts became known as the Internet.

30

Owner

- Each organization is responsible only for maintaining its own network.
- **No single person, company, institution, or government agency controls or owns the Internet.**
- The World Wide Web Consortium (W3C) oversees research and sets guidelines and standards.
- Internet2 is a **non-profit research & development project**, that connects more than **200 universities and 115 companies** via a high-speed private network to develop and test advanced network technologies (e.g.: telemedicine, digital libraries & faster Internet services).

31

Some ways to use the Internet

- E-mail
- Research
- Shopping
- News
- Games
- Ed-line
- Entertainment

Major Features of the Internet

- The World Wide Web
- E-Mail
- News
- Telnet
- File Transfer Protocol (FTP)
- VoIP
- Internet Relay Chat (IRC)

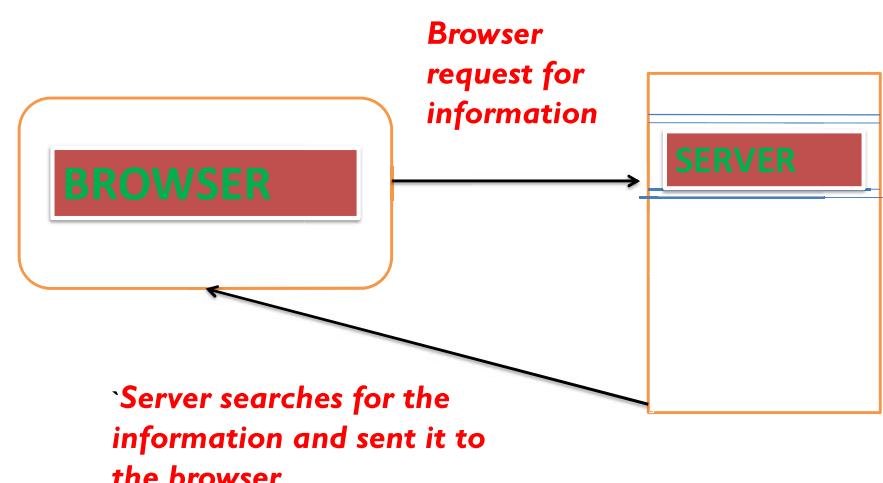
The World Wide Web (WWW)

- ▶ The **World Wide Web**, abbreviated as **WWW** and commonly known as **the Web**, is a system of interlinked hypertext documents accessed via the Internet.
- ▶ WWW is a part of the Internet, which supports hypertext documents, allowing users to view and navigate different types of data.
- ▶ With a web browser, one can view web pages that may contain text, images, videos, and other multimedia and navigate between them via hyperlinks.
- ▶ Every Web page has an address, a Uniform Resource Locator (URL).

Web Browser

- A **web browser** or **Internet browser** is a *software application* for retrieving, presenting, and traversing information resources on the World Wide Web.

Eg: Internet Explore(IE)
Firefox
Opera
Chrome
Safari
Netscape Navigator



Search Engines

- ▶ A search engine is a piece of application software that sits on a powerful computer (a server) on the Internet.
- ▶ Search engines keep track of information available on the Internet.
- ▶ E.g.

- Google (www.google.com)
- Excite (www.excite.com)
- Lycos (www.lycos.com)
- Info seek Guide (www.infoseek.com)
- WebCrawler (www.webcrawler.com)
- World wide web worm (www.goto.com)
- AltaVista (www.altavista.com)



How Internet Works

- TCP/IP
- Routing Traffic Across the Internet
- Addressing Schemes
- Domains and Sub domains

How Internet Works...

TCP/IP

- Every computer and network on the Internet uses the same protocols (set of guidelines or rules and procedures) to control timing and data format.
- The protocol used by the Internet is the Transmission Control Protocol/Internet Protocol, or TCP/IP.
- No matter what type of computer system you connect to the Internet, if it uses TCP/IP, it can exchange data with any other type of computer.

TCP/IP (Cont.)

- Single, common, worldwide standard
- US Department of Defense
 - TCP, transmission control protocol
 - » Handle the movement of data
 - » Breaks the original message into small packets & sequentially labels them.
 - IP, Internet Protocol
 - » Delivery of packets
 - » Disassembling and reassembling of packets

How TCP/IP Works

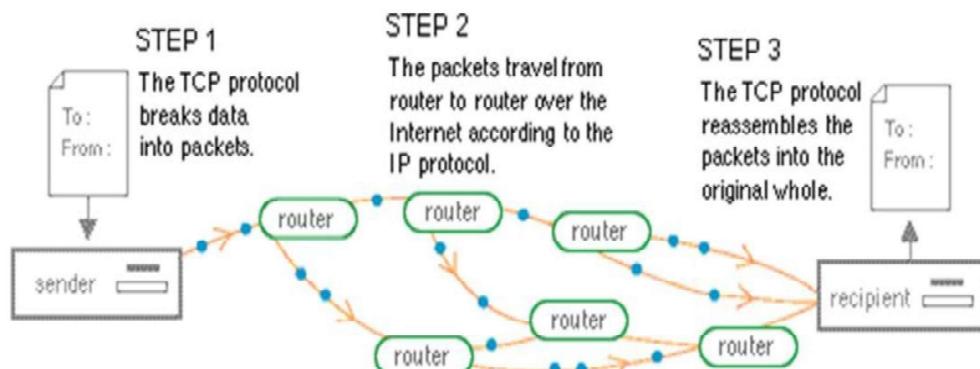
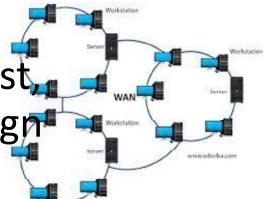


Figure 2. How data travels over the Net.

What about Internet Congestion?

- the packet-switched design solves this
- packets can take multiple paths to destination and get re-assembled
- if one router gets overloaded, buffer overflow messages tell neighbors to route around it
- also TCP/IP “back-off” algorithm
 - monitors throughput of connections and adjusts transmission frequency adaptively
- thus the Internet is amazingly robust, adaptive, and fault tolerant by design



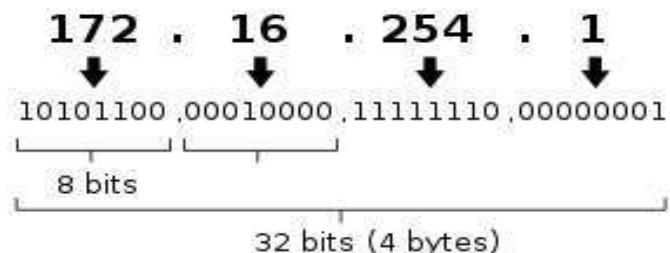
Addressing Schemes

- In order to communicate across the Internet, a computer must have a unique address.
- Every computer on the Internet has a unique numeric identifier, called an Internet Protocol (IP) address.
- Each IP address has four parts – each part a number between 0 and 255. An IP address might look like this:
e.g. 205.46.117.104.

IP addresses

- [0-255].[0-255].[0-255].[0-255]
- 128.194.139.1 (associated with a specific MAC)
- <domain>.<domain>.<subnet>.<host>
- IPv4 (current standard, 4 billion IP addresses)

IPv4 address in dotted-decimal notation



Domains and Sub domains

- In addition to an IP address, most Internet hosts or servers have a Domain Name System (DNS) address, which uses words.
- A domain name identifies the type of institution that owns the computer. An Internet server owned by IBM might have the domain name ibm.com.
- The last part of the domain name is called the *Top-Level Domain (TLD)* or *Zone* and is either two or three letters long.
- The second-to-last part of the domain name is called the *Second-Level Domain*, and it is chosen by the organization that owns the computer.
e.g. whitehouse.gov
ibm.com
slt.net

Three-letter zones (mainly used in the U.S.) indicate the type of organization that owns the domain.

Internet Domains		
Domain	Type of Organization	Example
.com	Business (commercial)	ibm.com (International Business Machines Corp.)
.edu	Educational	center.edu (Centre College, Danville, KY)
.gov	Government	Whitehouse.gov (The White House)
.mil	Military	Navy.mil (The United States Navy)
.net	Gateway or host (or business/commercial)	Mindspring.net (A regional Internet service provider)
.org	Other organization (typically nonprofit)	isoc.org (The Internet Society)

Domains and Sub domains (cont.)

- The two-letter zones indicate the country in which the organization that owns the computer is located.
 e.g. U.S. - .us
 Canadian - .ca
 Sri Lankan - .lk
- Some enterprises have multiple servers, and identify them with sub domains
 e.g. products.ibm.com.
- Currently, domains in the *com*, *edu*, *net* and *org* zones are assigned by Network Solutions' InterNIC Registration Services, at
<http://www.internic.net>

What is internet governance?

The development and application by governments, private sector and civil society, in their respective roles, of shared principles, norms, rules, decision-making procedures, and programmes that shape the evolution and use of the internet

The world summit on the information society(WSIS)(Tunis Agenda 2005)

51

INTERNET GOVERNANCE =

COORDINATION OF
THE MANY ASPECTS

INCLUDING

TECHNICAL STANDARDS,
POLICIES,
INFRASTRUCTURE

THAT MAKE THE INTERNET WORK
& DETERMINE HOW IT IS USED

INVOLVING
GOVERNMENTS
PRIVATE SECTOR
CIVIL SOCIETY

Why is Internet Governance Important?

- Determines how the Internet is managed and used now and in the future
- Outcomes may affect all stakeholders

53

INTERNET STAKEHOLDERS



54

Internet Governance Matters

- The Internet is seen as
 - A global force
 - An ecosystem
- Many implications
 - Economic
 - Social
 - Political



Worldwide Resource => Many Competing Interests

55

Question...?

56



HNDIT1022 – Web Design

Week 1: Introduction

Module Code	HNDIT1022	Module Title	Web Design	
Credit	4	Hours/ Week		Per week(hours)
GPA/NGPA	GPA(Compulsory)		Lectures	02
Semester	1 (15 Weeks)		Lab/Tutorial	04
			Student Activities	07
			Notional hours	13

Module Aims

- The module focuses on the fundamentals of web internet and develop web site using HTML, CSS, JavaScript and the current developments in Web Design theory and the practice.

Learning Outcomes

- Design , development and deploy a simple website using HTML,CSS and Java Scripts
- Design and develop interactive visually appealing client-side programs
- Describe HTML5 and web design tools

Outline of Details Syllabus

1. Introduction to web Design
2. Understanding HTML and XHTML
3. Define and plan the information hierarchy
4. Understanding Cascading Style sheet
5. Understanding CSS Box Model and Positioning
6. Understanding dynamic web sites and introduction to photoshop
7. Understanding Java Script programming
8. Working with web-based forms
9. Introduction to Dream viewer
10. Introduction to web servers and hosting

5

Assessment Weight

Type	Activity	Weighting
Continuous Assessment	Online quizzes, Labs, Individual Assignment , Group Assignment , Tutorial	40%
End of semester examination	Structured question paper	60%

6

Prescribed Textbook and Resources

- Sam's Teach Yourself HTML, CSS, and JavaScript All in One, Sam's Publishing; 1 edition
- Internet & World Wide Web, How to Programme (5th edition) Dietel&Dietel
- Online materials
- <https://www.w3schools.com/>
- <https://www.tutorialspoint.com/index.htm>



Week 1 - Introduction to web design

Subtopics

- History of HTML and WWW.
- Describe the compatibility of web browsers
- How to select web hosting provider
- How to transfer files to your web server using FTP
- How to distribute web content and web publishing

9

Internet vs. WWW

- Most people use the two terms interchangeably but they are in fact different.
- The Internet is a vast, international network, made up of computers and the physical connections (wires, routers, etc.) allowing them to communicate.
- The World Wide Web (WWW or just the Web) is a collection of software that spans the Internet and enables the interlinking of hypertext documents and resources.
- Hyperlink (or link)- is a reference to a document that the reader can directly follow, or that is followed automatically

History of World Wide Web



- Tim Berners-Lee invented the World Wide Web while working at **CERN** in 1989, applying the concept of hyperlinking that had by then existed for some decades.
- He developed the first **web server**, the first **web browser**, and a document formatting protocol, called **Hypertext Markup Language (HTML)**.
- After publishing the markup language in 1991, and releasing the browser source code for public use in 1993, many other web browsers were soon developed

When was the first web page created ?



- The first webpage was discovered by **Tim Berners-Lee** who was researcher at **CERN** (the European Organization for Nuclear Research).
- The first webpage went live on **August 6, 1991**.
- The first webpage is used for sharing the documents through the internet by using limited simple text and **FTP** (File Transfer Protocol). It is convenient for the workspace and acceptable in browsers.

Cont..

- HTML is derived from SGML (**Standard Generalized Markup Language**).
- The term **SGML** denotes that it is a markup language document and set of tags, it is based on DTD (document type definition).
- SGML is not a document language, but it is used to define the specific part, it is consider as a metadata.
- HTML is the standard markup language for Web pages.
- HTML stands for Hyper Text Markup Language

First webpage

World Wide Web

The WorldWideWeb (W3) is a wide-area [hypermedia](#) information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an [executive summary](#) of the project, [Mailing lists](#), [Policy](#), November's [W3 news](#), [Frequently Asked Questions](#).

[What's out there?](#)

Pointers to the world's online information, [subjects](#), [W3 servers](#), etc.

[Help](#)

on the browser you are using

[Software Products](#)

A list of W3 project components and their current state. (e.g. [Line Mode](#), [X11 Viola](#), [NeXTStep](#), [Servers](#), [Tools](#), [Mail robot](#), [Library](#))

[Technical](#)

Details of protocols, formats, program internals etc

[Bibliography](#)

Paper documentation on W3 and references.

[People](#)

A list of some people involved in the project.

[History](#)

A summary of the history of the project.

[How can I help ?](#)

If you would like to support the web..

[Getting code](#)

Getting the code by [anonymous FTP](#), etc.

What is World Wide Web ?



The **World Wide Web (Web)** is a network of information resources. The Web relies on **three mechanisms** to make these resources readily available to the widest possible audience

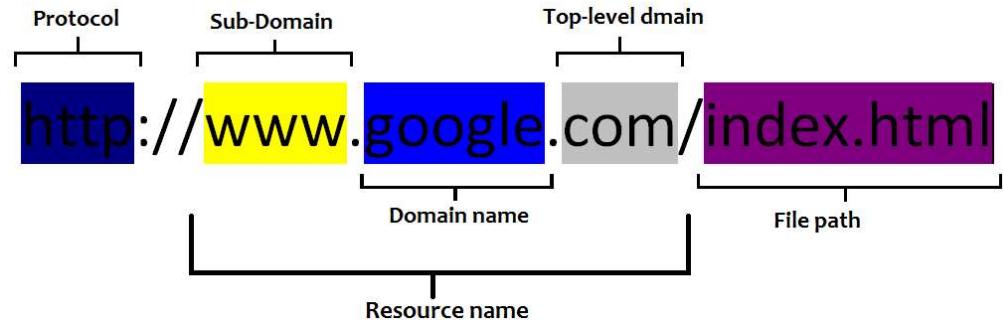
1. A **uniform naming** scheme for locating resources on the Web (e.g., URIs)
2. **Protocols**, for access to named resources over the Web (e.g., [HTTP](#)).
3. **Hypertext**, for easy navigation among resources (e.g., HTML).

URL

Every resource available on the Web -- HTML document, image, video clip, program, etc. .. has an address that may be encoded by a **Uniform Resource Locator, or "URL"**. URLs typically consist of three parts are

- The **naming scheme** of the mechanism used to access the resource(Protocol)
- The **name of the machine** hosting the resource(Domain)
- The **name of the resource** itself, given as a path

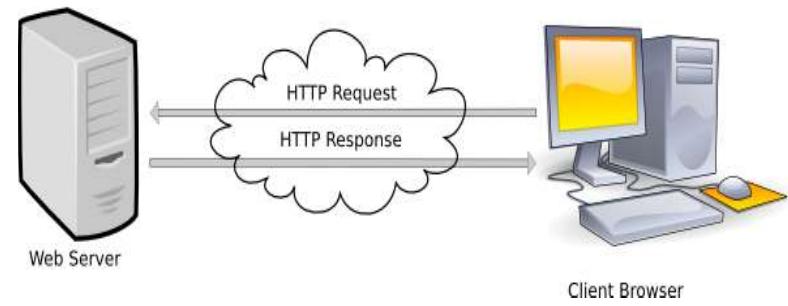
Basic URL structure



- **HTTP**
HTTP (Hypertext Transfer Protocol) is the set of rules for transferring files (text, graphic images, sound, video, and other multimedia files) on the World Wide Web. HTTP is a very simple client-server request/response communication protocol.
Development of HTTP was coordinated by the W3C (World Wide Web Consortium) and the IETF (Internet Engineering Task Force)

- **HTTPS**
URLs of websites that handle private information, such as credit card numbers, often begin with `https://`, the abbreviation for **Hypertext Transfer Protocol Secure (HTTPS)**. HTTPS is the standard for transferring encrypted data on the web.

Web Server-Client Diagram



W3C (World Wide Web Consortium)

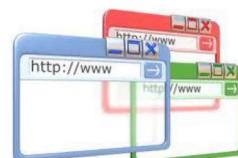
- The Global terms and international standards of HTML was maintained by **W3C** (World Wide Web Consortium) and the **WHATWG** (Web Hypertext Application Technology Working Group).
- To develop the Web standards member of the organizations and the full time staff are working together for W3C.
- **Tim Berners-Lee** who was the web inventor led the W3C and the current CEO of W3C is **Jeffrey Jaffe**.
- The main motto of the W3C is to lead the World Wide Web for developing the guideline and the protocols to certify the long term growth of the Web.

Web page

- A document or information resource that is suitable for the World Wide Web
- Can be accessed through a web browser
- Displayed on a monitor or mobile device
- **This information is usually in HTML or XHTML format.**
- Use other resources such as style sheets, scripts and images into their final presentation
- retrieved from a local computer or from a remote web server

Web Site

- A collection of related web pages containing images, videos or other digital assets
- 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



Web Server

- Web server is a software on a server machine that can be run to answer requests from Web clients using the Hypertext Transfer Protocol (HTTP).
- This can also be defined as a network server that manages access to files, folders and other resources over the Internet or local intranet via HTTP.
- Web servers handle access permission, execute programs, keep track of directories and files and communicate with client computers.

Web browsers

- A software application for retrieving, presenting, and traversing information resources on the World Wide Web.
- Major web browsers are Internet Explorer, Firefox, Google Chrome, Apple Safari, and Opera.



Versions of HTML

YEAR	VERSION
1989	Tim Berners Lee Invented WWW
1991	Time Berners Lee invented HTML
1993	Dave Raggett drafted HTML+
1995	HTML working Group defined HTML2.0
1997	W3C recommendations HTML 3.2
1999	W3C recommendations HTML 4.01
2000	W3C recommendations XHTML 1.0
2008	WHATWG HTML5 First Public Draft
2012	WHATWG HTML5 Living Standard
2014	W3C Recommendations:HTML5
2016	W3C Candidate Recommendation:HTML5.1

Browser Compatibility

- The term browser compatibility refers to the ability of a certain website to appear fully functional on different browsers that are available in the market.
- This means that the website's HTML coding, as well as the scripts on that website, should be compatible to run on the browsers.
- This is of immense importance, especially today when there is a large variety of web browsers available.

Why Is It Important?

- A website's compatibility plays a huge role in the success of an e-commerce business.
- The reason being that incompatibility with certain browsers limits the audience of that website.
- This leads to the limited traffic on that particular website which can bring down its conversion.
- This means that browser compatibility is among the prime aspects to cater to while a website is in the development phase.



How It Works?

- The difference between different browsers is that the mode of translation of the text is different in every web browser.
- This mode of translation is used to translate the HTML coding of a website.
- Although the standard of rules remains, the difference tends to occur with the interpretation of the coding.
- The modern browsers in the market tend to be able to easily decode most modern HTML coding far better than older browsers do.

What is web hosting?

A key element of all websites is that they require hosting services. This generally means they rely on a web hosting provider to give them space on a server.

Types of Web hosting

- Shared Hosting
- Virtual Private Server(VPS)
- Dedicated Hosting
- Cloud Hosting
- WordPress Hosting
- Managed Hosting
- Reseller Hosting



How to select a web hosting provider

- Choosing the hosting company and plan that are right for your site can take some careful consideration.
- Not all hosts are created equal, and you'll want to make sure you're getting all the features you'll need.

Cont..

- Know What Your Website's Needs Are.
- Confirm the Subscription Period for Hosting Contracts.
- Check for Server Reliability and Uptime Guarantees.
- Confirm the Host's Refund Policy and Guarantees.
- Understand What Your Upgrade Options Are.
- Verify the Host's Primary Features and etc..



Hosting Services

The best web hosting services will give you plenty of options when it comes to plans and features. That way, you have flexibility when it comes to getting the resources you need for growing your website.

So which web hosting options are best for you?

If this is your first time building a website, shared hosting is usually a safe , and you can only move upwards from there. On the other hand, if you're a more experienced user and need full control over your configuration, a VPS can be a more fitting option.

	Our Most Popular	Reviews	Rating
1	DreamHost	★★★★★ 61 reviews	Excellent 9.6
		DreamHost - Flexible hosting solutions to ensure your success	
		✓ Free domain name ✓ 100% uptime guaranteed ✓ 24/7 support	
		Visit Site	
2	bluehost	★★★★★ 170 reviews	Excellent 9.3
		Bluehost - Extremely fast & reliable hosting with plenty of advanced security add-ons	
		✓ Powers 2M+ websites worldwide ✓ Automatic WordPress installation ✓ Free domain & website migration	
		Visit Site	
3	HostGator	★★★★★ 108 reviews	Excellent 9.0
		HostGator - Easy and affordable web hosting with intuitive site building tools	
		✓ Easy-to-use control panel ✓ Unmetered disk space & bandwidth	
		Visit Site	

(4) network solutions	★★★★★ 84 reviews	Very Good 8.5	Visit Site
Network Solutions - Cloud-based web hosting with easy-to-upgrade packages			
<ul style="list-style-type: none"> ✓ Domain name included ✓ Professional templates ✓ Support: 24/7 phone & email 			
(5) IONOS by 1&1	★★★★★ 103 reviews	Very Good 8.9	Visit Site
IONOS - Fast and secure hosting to help your business grow			
<ul style="list-style-type: none"> ✓ Free domain & SSL certificate ✓ 99.9% uptime ✓ Scalable performance management 			
(6) HOSTINGER	★★★★★ 77 reviews	Very Good 8.8	Visit Site
HOSTINGER - Scalable hosting solutions for fast websites			
<ul style="list-style-type: none"> ✓ Free templates and builder ✓ High-speed performance 			

What Is FTP(File Transfer Protocol)?

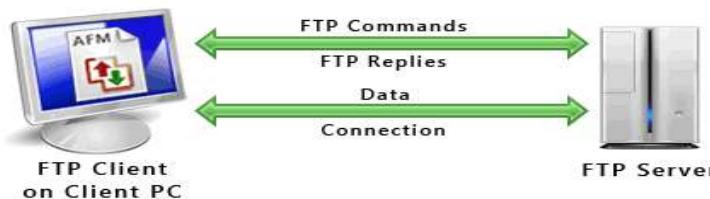
FTP (File Transfer Protocol) is a network protocol for transmitting files between computers over Transmission Control Protocol/Internet Protocol (TCP/IP) connections.

Why is FTP important?

- FTP is a standard network protocol that can enable expansive file transfer capabilities across IP networks.
- Without FTP, file and data transfer can be managed with other mechanisms such as email or an HTTP web service
- But those other options lack the clarity of focus, precision and control that FTP enables.

How does FTP work?

- FTP is a client-server protocol that relies on two communications channels between the client and server
- One computer acts as the server to store information and the other acts as the client to send or request files from the server.
- The FTP protocol typically uses port 21 as its main means of communication.**
- An FTP server will listen for client connections on port 21.**
- Using FTP, a client can upload, download, delete, rename, move and copy files on a server.



How to transfer files to your web server using FTP

Understand the parts of an FTP address :

When you come across FTP addresses on a webpage, they are usually denoted in the same way as a usual webpage address

Determine how you prefer to connect:

There are three main ways to connect to FTP servers: via visual clients, via browser-based clients, or through the command line.

Downloading and installing a visual client is the most widely used and easiest way to connect to an FTP, and also affords you the most power and control over the process.

Cont..

Examples of Visual clients :

- FileZilla
- Cyberduck
- FireFTP
- Classic FTP
- WinSCP
- Free FTP by CoffeeCup

Cont..

Let's use the FileZilla

- FileZilla is a free, downloadable FTP client. Type in the address of the server you wish to access, the port, and the password for accessing the server.
- Once access has been granted, the user's files on their local system as well as the accessed server will be visible.
- The user can download files from the server to the local system, or upload files from the local system to the server.
- They can also make changes to files on the server, as long as they have the proper authorization to do so.

The screenshot shows the FileZilla interface. The left panel displays the 'Local site' with a tree view of files and folders, including 'wp-content', 'wp-includes', 'tweet-trends', 'xampp', 'img', 'install', 'licenses', 'locale', 'mailoutput', 'mailtoidisk', and 'mssql'. The right panel shows the 'Remote site' at '/c11/h03/mr', listing directories like 'home', 'opt', 'proc', 'root', 'run', 'sbin', 'selinux', 'srv', 'sys', 'tmp', 'usr', and 'var'. Below these panels, a detailed list of files is shown with columns for 'Filename', 'Filesize', and 'Filetype'. The 'Filetype' column indicates that most files are PHP Scripts. A status bar at the bottom shows '143 files and 14 directories. Total size: 4,480,467 bytes'.

- Content distribution is the process of promoting your content to your audience in a variety of formats across multiple channels and media platforms.
- Content distribution falls into three categories: owned, earned, and paid distribution.

Why is it Important to Distribute Your Content?

- Content distribution is an essential part of any digital marketing strategy
- Distributing your content properly can lead to more social shares, drive traffic to your website, and create more conversations across your social networks.

Content distribution in website

- Content distribution is the process of promoting your content to your audience in a variety of formats across multiple channels and media platforms.
- Content distribution falls into three categories: owned, earned, and paid distribution.

How to distribute web content?

- Content distribution channels are the online places where you can share and promote the content you've created.
- Nowadays, content can be shared in many different ways, from blogs and social media networks to emails and e-books.
- Also, you can distribute content yourself or through another content publisher, or use a paid content-sharing platform.
- With so many options available, it can be difficult to know which methods best suit your needs and purposes.

Cont..

- However, some content promotion tactics are generally more effective than others.
- Here are a few that can help you reach a larger audience and boost your website traffic.
 - Leverage Influencers
 - Use Social Media
 - Convert Your Content into Different Media
 - Promote Your Content on Content Communities

Cont..

- Some examples of Content Types to Add to Your Website
 - Blog posts
 - Video
 - Infographics
 - Case Studies
 - Ebooks
 - Webinars
 - Podcasts

Web publishing

- Web publishing is the process of publishing original content on the Internet.
- The process includes building and uploading websites, updating the associated webpages, and posting content to these webpages online. Web publishing comprises of personal, business, and community websites in addition to e-books and blogs.
- Publishers must possess a web server, a web publishing software, and an Internet connection to carry out web publishing

What are the options available to publish the website?

- Publishing a website is a complex topic because there are many ways to go about it.
- Such as
 - Getting hosting and a domain name
 - Using an online tool like GitHub or Google App Engine
 - Using a web-based IDE such as CodePen

Questions...?



HNDIT1022 – Web Design

- Create a folder in the documents
- Name:IT2021001_Nisha
- Save the html file : first.html

Week 2: Understanding HTML and XHTML

Subtopics

- Structure of simple web page Including all basic HTML tags
- How to organize your content with headings
- How to validate your web content
- How to differentiate HTML, XML, XHTML and HTML5.

Structure of simple web page Including all basic HTML tags

- **HTML** stands for **HyperText Markup Language**. It is used to design web pages using the **markup language**.
- HTML is the combination of **Hypertext** and **Markup language**.
- Hypertext defines the link between the web pages and markup language defines the text document within the tag that define the structure of web pages.

What is HTML used for ?

- HTML is used to create the structure of web pages that are displayed on the World Wide Web (www).
- It contains Tags and Attributes that are used to design the web pages. Also, we can link multiple pages using Hyperlinks.

Creating HTML Pages

- Web pages can be created and modified by using professional HTML editors.
- However, for learning HTML we recommend a simple text editor like Notepad (PC) or Notepad++ (PC) or TextEdit (Mac)
- An HTML file must have an [.htm](#) or [.html](#) file extension.
- You can use either [.htm](#) or [.html](#) as file extension. There is no difference, it is up to you
- Follow the steps below to create your first web page with Notepad or TextEdit.
 - [Step 1: Open Notepad \(PC\) or Open TextEdit \(Mac\)](#)
 - [Step 2: Write Some HTML](#)
 - [Step 3: Save the HTML Page](#)
 - [Step 4: View the HTML Page in Your Browser](#)

Contd..

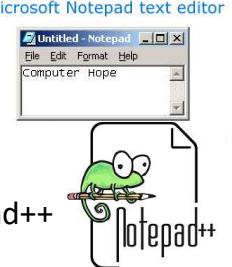
Step 1: Open Notepad (PC) or Notepad++(PC)

Windows 8 or later:

Open the Start Screen (the window symbol at the bottom left on your screen). Type Notepad or Notepad++

Windows 7 or earlier:

Open Start > Programs > Accessories > Notepad or Notepad++



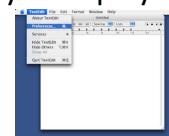
Open TextEdit (Mac)

Open Finder > Applications > TextEdit

Also change some preferences to get the application to save files correctly. In Preferences > Format > choose "Plain Text"

Then under "Open and Save", check the box that says "Display HTML files as HTML code instead of formatted text".

Then open a new document to place the code.



Contd..

Step 2: Write Some HTML

Write or copy the HTML code in the text editor.

Step 3: Save the HTML Page

Save the file on your computer. Select **File > Save as** in the Notepad menu.

or

Save the file in a folder with [.html](#) extension. We can easily save the html file by clicking on the **File** menu and then click on **Save As** option. After that, type the file name with .html extension.

Step 4: View the HTML Page in Your Browser

Open the saved HTML file in your favorite browser (double click on the file, or right-click - and choose "Open with").

Contd..

- You can also use HTML editors (WYSIWYG Editors) to create your web page
- A WYSIWYG editor — short for "What You See Is What You Get" editor is a major asset when building or making changes to a website.
- It enables you to make changes and immediately see how they would be displayed on your live website.

Contd..

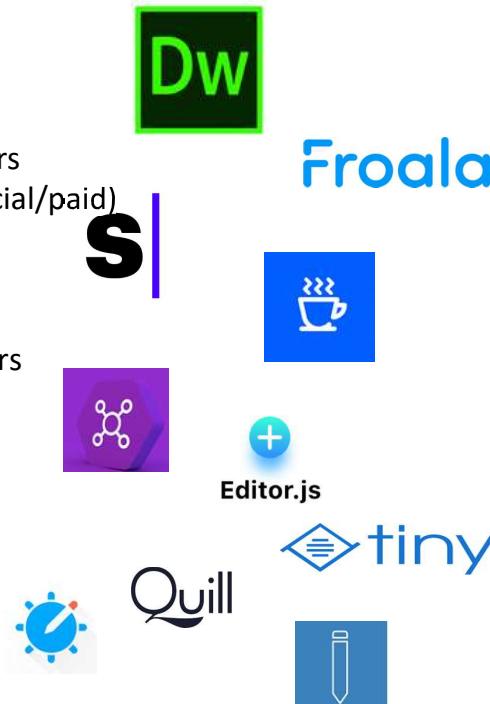
List of best WYSIWYG HTML editors

- Premium editors(commercial/paid)
- Adobe Dreamweaver CC
- Froala
- Setka Editor
- CoffeeCup HTML Editor

List of Free WYSIWYG HTML editors

Free editors(open source)

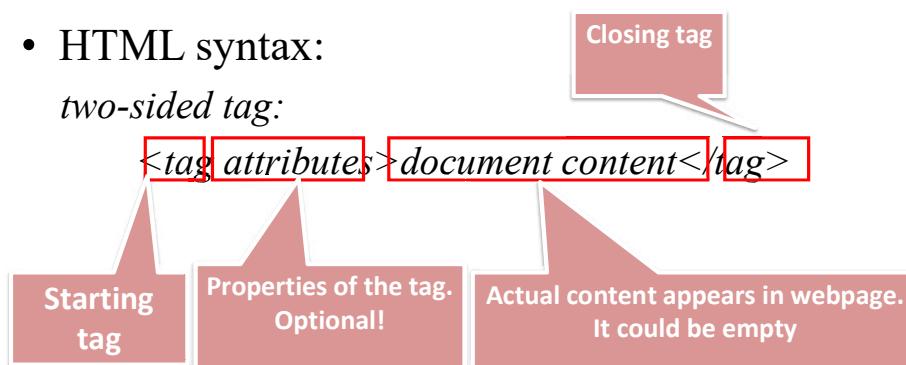
- CKEditor 4
- Editor.js
- TinyMCE
- Quill
- Summernote
- Visual Studio Code
- ContentTools



HTML Syntax

- HTML syntax:

two-sided tag:

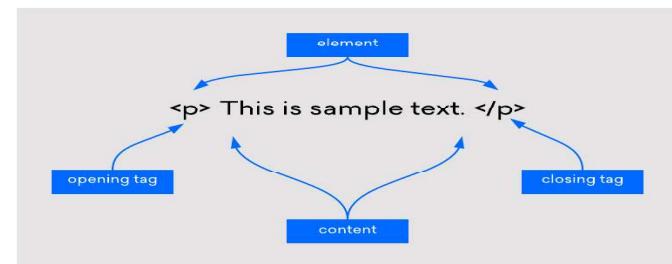


Examples: `<p> Welcome To Web Designing </p>`

```
<body bgcolor = "red">Happiness lies in the joy of achievement and the thrill of creative effort.
</body>
```

HTML tags

- HTML tags are the hidden keywords and used to create web pages in different format.
- Most of the tags contain two parts, opening tags and closing tags. The text of opening and closing tags are the same but the closing tags contains forward slash (/) character.
- There are some tags which do not require closing tags.
- HTML tags are not case sensitive, `` means the same as ``



HTML element

- HTML documents are defined by HTML elements.
- An HTML element is everything from the start tag to the end tag.
 - Example

Start tag

`<p>`

Element content

This is a paragraph

End tag

`</p>`

Syntax of HTML element

- An HTML element starts with a start tag
- An HTML element ends with an end tag
- The element content is everything between the start and the end tag
- Some HTML elements have empty content
- Empty elements are closed in the start tag
- Most HTML elements can have attributes

Example for HTML element

```
<html>
<body>
<p>This is my first web page</p>
</body>
</html>
```

- Contains 3 HTML elements.

Nested HTML elements

- Most HTML elements can be nested
- HTML documents consist of nested HTML elements.

13

14

Empty HTML elements

- HTML elements with no content are called empty elements.
-
 is an empty element without a closing tag (the
 tag defines a line break).
- Empty elements can be "closed" in the opening tag like this:
.
- HTML5 does not require empty elements to be closed. But if you want stricter validation, or you need to make your document readable by XML parsers, you should close all HTML elements.

HTML attributes

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

Structure of simple web page Including all basic HTML tags

HTML Page Structure

```
<!DOCTYPE html> ← Tells version of HTML
<html> ← HTML Root Element

<head> ← Used to contain page HTML metadata
  <title>Page Title</title> ← Title of HTML page
</head>

<body> ← Hold content of HTML
  <h2>Heading Content</h2> ← HTML heading tag
  <p>Paragraph Content</p> ← HTML paragraph tag
</body>

</html>
```

Contd..

•<!DOCTYPE html> – A doctype or document type declaration is an instruction that tells the web browser about the markup language in which the current page is written. It is not an element or tag. The doctype declaration is not case-sensitive.

•<html> – This tag is used to define the root element of HTML document. This tag tells the browser that it is an HTML document. It is the second outer container element that contains all other elements within it.

Contd..

•<head> – This tag is used to define the head portion of the HTML document that contains information related to the document. Elements within the head tag are not visible on the front-end of a webpage.

•<body> – The body tag is used to enclose all the visible content of a webpage. In other words, the body content is what the browser will show on the front end.

Contd..

•<head> – This tag is used to define the head portion of the HTML document that contains information related to the document. Elements within the head tag are not visible on the front-end of a webpage.

•<body> – The body tag is used to enclose all the visible content of a webpage. In other words, the body content is what the browser will show on the front end.



A Basic HTML Document Example

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

<h1>My Heading</h1>
<p>My Paragraph.</p>

</body>
</html>
```

Example Explained

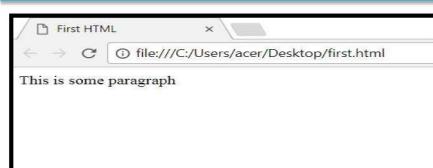
- The `<!DOCTYPE html>` declaration defines that this document is an HTML5 document
- The `<html>` element is the root element of an HTML page
- The `<head>` element contains meta information about the HTML page
- The `<title>` element specifies a title for the HTML page (which is shown in the browser's 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



First HTML Page

first.html

```
<html>          Opening tag
  <head>        Closing tag
    <title>First HTML</title>
  </head>
  <body>
    <p>This is some paragraph</p>
  </body>
</html>
```

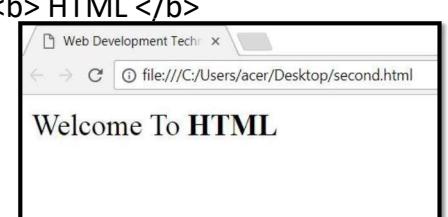


An HTML element consists of an opening tag, a closing tag and the content inside

Example

second.html

```
<html>
  <head>
    <title>Web Development Technology
  </title>
  </head>
  <body>
    Welcome To <b> HTML </b>
  </body>
</html>
```



Example explained

- The first tag in your HTML document is <html>.
 - It tells your browser that this is the start of an HTML document
- The last tag in your document is </html>.
 - It tells your browser that this is the end of the HTML document.
- The text between the <head> tag and the </head> tag is header information.
 - Header information is not displayed in the browser window.

Example explained (contd..)

- The text between the <title> tags is the title of your document.
 - The title is displayed in your browser's caption.
- The text between the <body> tags is the text that will be displayed in your browser.
 - is the visible page content
- The text between the and tags will be displayed in a bold font.

26

The <head> Section

- Contains information that doesn't show directly on the viewable page
- Starts after the <!doctype> declaration
- Begins with <head> and ends with </head>
- Contains mandatory single <title> tag
- Can contain some other tags, e.g.
 - <meta>
 - <script>
 - <style>
 - <!-- comments -->

<head> Section: <title> tag

- Title should be placed between <head> and </head> tags



- Used to specify a title in the window title bar
- Search engines and people rely on titles

Setting Document Properties

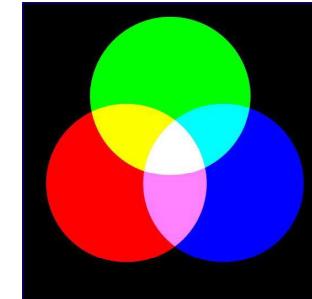
- Document properties are controlled by attributes of the **BODY** element.
- For example,
 - there are color settings for the background color of the page, the document's text and different states of links.

29

Color Codes

- Colors are set using “**RGB**” color codes, which are, represented as hexadecimal values.
- Each 2-digit section of the code represents the amount, in sequence, of **red**, **green** or **blue** that forms the color.
 - E.g.* A **RGB** value with 00 as the first two digits has no red in the color.

Main Colours



30

The Body Element

- The **BODY** element of a web page is an important element in regards to the **page's appearance**.
- This element contains information about the page's background color, the background image, as well as the text and link colors.

E.g.

TEXT="#RRGGBB"

to change the color of **all the text** on the page (**full page text color**)

- Here are the attributes of the **BODY** tag to control all the levels:

➤ **bgcolor**
➤ **Background**
➤ **text**

31

Background Color

- It is very common to see web pages with their background color set to white or some other colors.
- To set your document's background color, you need to edit the **<BODY>** element by adding the **BGCOLOR** attribute.

E.g.

<BODY BGCOLOR="# " ></BODY>

32

TEXT Color

- The TEXT attribute is used to control the color of all the normal text in the document.
- The default color for text is black.

E.g.

The TEXT attribute would be added as follows:

```
<BODY BGCOLOR="#FFFFFF"TEXT="#FF0000"> Document's  
content.  
</BODY>
```

In this example the document's page color is white and the text would be red.

Using Image Background

- The BODY element also gives you ability of setting an image as the document's background.
- An example of a background image's HTML code is as follows:

```
<BODY BACKGROUND="hi.gif" BGCOLOR="#FFFFFF">  
Document's content.  
</BODY>
```

33

34

HTML headings

- HTML headings are defined with the `<h1>` to `<h6>` tags.
 - `<h1>` defines the largest heading.
 - `<h6>` defines the smallest heading.
- Example

```
<h1>This is a heading</h1>  
<h2>This is a heading</h2>  
<h3>This is a heading</h3>
```

- Browsers automatically adds an empty line before and after headings.

HTML paragraphs

- HTML paragraphs are defined with the `<p>` tag
- Example

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

- HTML automatically adds an extra blank line before and after a paragraph

35

Line breaks

- The
 tag is used when you want to end a line, but don't want to start a new paragraph.
- The
 tag forces a line break wherever you place it.
- Example
 - <p>This
 is a para
graph with line breaks</p>
 - The
 tag is an empty tag. It has no closing tag.

Comments in HTML

Comments are written like this:

<!-- This is a comment -->

37

38

HTML Rules (Lines)

- The <hr> or <hr /> tag is used to create an horizontal rule (line).
- Example

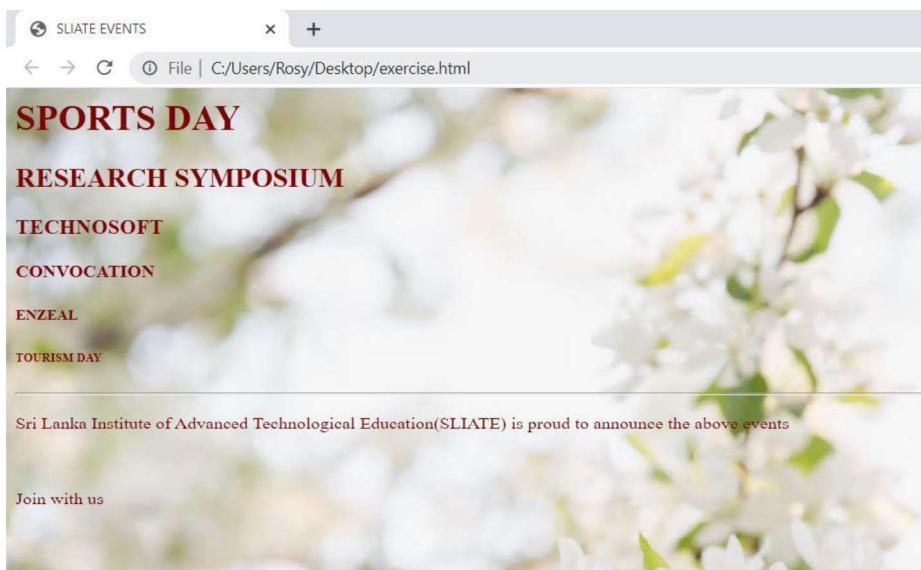
```
<p>This is a paragraph</p>
<hr />
<p>This is a paragraph</p>
<hr />
<p>This is a paragraph</p>
```

Exercise

```
<html>
  <head>
    <title>SLIATE EVENTS</title>
  </head>
  <body background="backgroundpic. avif" text="#800000">
    <h1>SPORTS DAY</h1>
    <h2>RESEARCH SYMPOSIUM</h2>
    <h3>TECHNOSOFT</h3>
    <h4>CONVOCATION</h4>
    <h5>ENZEAL</h5>
    <h6>TOURISM DAY</h6>
    <hr/>
    <p>Sri Lanka Institute of Advanced Technological Education(SLIATE) is proud to announce the above events</p>
    <br>
    <p>Join with us</p>
  </body>
</html>
```



The output of the HTML



- Will create different results.
- With HTML, you cannot change the output by adding extra spaces or extra lines in your HTML code.
- The browser will remove extra spaces and extra lines when the page is displayed.
- Any number of lines count as one space, and any number of spaces count as one space.



Text Formatting

Tag	Description
	bold text
<big>	big text
	emphasized text
<i>	italic text
<u>	Underlined text
<small>	small text
	strong text
<sub>	subscripted text
<sup>	superscripted text

Pre formatted text - Example

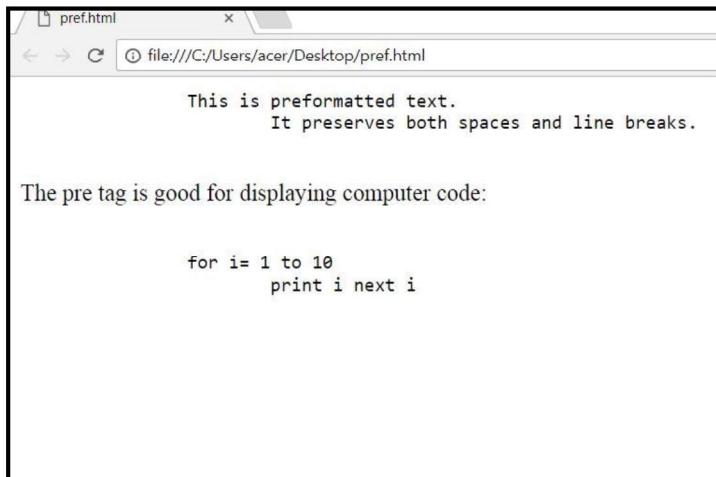
```

<html>
  <body>
    <pre>
      This is preformatted text.
      It preserves both spaces and line breaks.
    </pre>
    <p>The pre tag is good for displaying computer code:</p>
    <pre>
      for i= 1 to 10
          print i next i
    </pre>
  </body>
</html>

```



Pre formatted text – Cont.



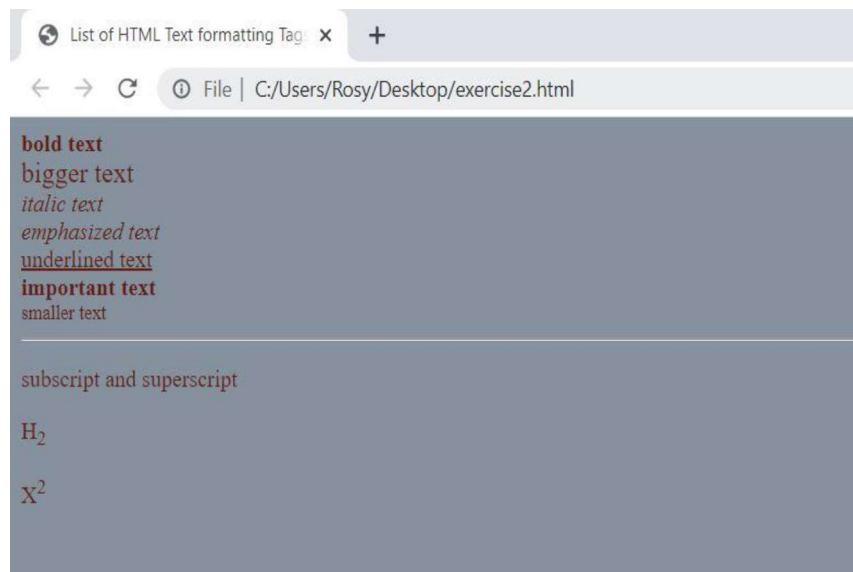
The pre tag is good for displaying computer code:

```
for i = 1 to 10
    print i next i
```

Exercise

```
<html>
    <head>
        <title>List of HTML Text formatting Tags</title>
    </head>
    <body bgcolor="#85929e" text="#641e16">
        <!-- Text formatting in html -->
        <b>bold text</b><br>
        <big>bigger text</big><br>
        <i>italic text</i><br>
        <em>emphasized text</em><br>
        <u>underlined text</u><br>
        <strong>important text</strong><br>
        <small>smaller text</small><br>
        <hr/>
        <p>subscript and superscript</p>
        <p>H<sub>2</sub></p>
        <p>X<sup>2</sup></p>
    </body>
</html>
```

Exercise: Output



How to validate your web content

Why we need to validate the HTML Code?

- As a beginner it is very common that you will make mistake in writing your HTML code. Incorrect or non-standard code may cause unexpected results in how your page displayed or function in browsers.
- To prevent this you can test or validate your HTML code against the formal guidelines and standards set by the Wide Web Consortium (W3C) for HTML/XHTML web pages.
- The World Wide Web Consortium provide a simple online tool (<https://validator.w3.org/>) that automatically check your HTML code and point out any problems/errors your code might have, such as missing closing tags or missing quotes around attributes.

Difference between HTML and XHTML

S.No.	HTML	XHTML
1.	HTML stands for Hypertext Markup Language.	XHTML stands for Extensible Hypertext Markup Language.
2.	It was developed by Tim Berners-Lee.	It was developed by W3C i.e World Wide Web Consortium.
3.	It was developed in 1991.	It was released in 2000.
4.	It is extended from SGML.	It is extended from XML and HTML.
5.	The format is a document file format.	The format is a markup language.
6.	All tags and attributes are not necessarily to be in lower or upper case.	In this, every tag and attribute should be in lower case.
7.	Doctype is not necessary to write at the top.	Doctype is very necessary to write at the top of the file.
8.	It is not necessary to close the tags in the order they are opened.	It is necessary to close the tags in the order they are opened.
9.	While using the attributes it is not necessary to mention quotes. For e.g. <SRC>.	While using the attributes it is mandatory to mention quotes. For e.g. <SRC="Infotec">.
10.	Filename extension used are .html, .htm.	Filename extension are .xhtml, .xht, .xml.

Contd..



Markup Validation Service

Check the markup (HTML, XHTML, ...) of Web documents

Validate by URI Validate by File Upload Validate by Direct Input

Validate by direct input

Enter the Markup to validate:

More Options

Check

This validator checks the [markup validity](#) of Web documents in HTML, XHTML, SMIL, MathML, etc. If you wish to validate specific content such as [RSS/Atom feeds](#) or [CSS stylesheets](#), [MobileOK content](#), or to [find broken links](#), there are other validators and tools available. As an alternative you can also try our [non-DTD-based validator](#).

Difference between XML and XHTML

Parameters of Comparison	XML	XHTML
Full-Form	XML represents Extensive Markup Language.	XHTML represents Extensible Hypertext Markup Language.
Definition	XML refers to a simple text-based format that is used for representing structured information like data, transactions, configuration, documents, invoices, books, and so on.	XHTML refers to a cross between XML and HTML that is used for transitioning from HTML to XML.
Release	XML was first published in 1998.	XHTML was initially released in 2000.
Structure	XML has a hierarchical tree-shaped structure which is called XML tree.	XHTML is developed based on three main components- declaration, head, and body.
Components	XML is composed of Unicode.	XHTML comprises three versions- XHTML Transitional, XHTML 1.0 Frameset, and XHTML 1.0 Strict.

Difference between HTML and HTML5

HTML	HTML5
It didn't support audio and video without the use of flash player support.	It supports audio and video controls with the use of <audio> and <video> tags.
It uses cookies to store temporary data.	It uses SQL databases and application cache to store offline data.
Does not allow JavaScript to run in browser.	Allows JavaScript to run in background. This is possible due to JS Web worker API in HTML5.
Vector graphics is possible in HTML with the help of various technologies such as VML, Silver-light, Flash, etc.	Vector graphics is additionally an integral part of HTML5 like SVG and canvas.
It does not allow drag and drop effects.	It allows drag and drop effects.
Not possible to draw shapes like circle, rectangle, triangle etc.	HTML5 allows to draw shapes like circle, rectangle, triangle etc.
It works with all old browsers.	It supported by all new browser like Firefox, Mozilla, Chrome, Safari, etc.
<HTML>, <Body> , and <Head> tags are mandatory while writing a HTML code.	These tags can be omitted while writing HTML code.
Older version of HTML are less mobile-friendly.	HTML5 language is more mobile-friendly.
Doctype declaration is too long and complicated.	Doctype declaration is quite simple and easy.
Elements like nav, header were not present.	New element for web structure like nav, header, footer etc.
Character encoding is long and complicated.	Character encoding is simple and easy.
It is almost impossible to get true GeoLocation of user with the help of browser.	One can track the GeoLocation of a user easily by using JS GeoLocation API.
It can not handle inaccurate syntax.	It is capable of handling inaccurate syntax.
Being an older version , it is not fast , flexible , and efficient as compared to HTML5.	It is efficient, flexible and more fast in comparison to HTML.
Attributes like charset, async and ping are absent in HTML.	Attributes of charset, async and ping are a part of HTML 5.



Questions...?



HNDIT1022 – Web Design

Week 3 Part 2: HTML Forms

Forms

Forms add the ability to web pages to not only provide the person viewing the document with dynamic information but also to obtain information from the person viewing it, and process the information.

- Forms work in all browsers.
- Forms are Platform Independent.

Objectives:

Upon completing this section, you should be able to

1. Create a FORM.
2. Add elements to a FORM.
3. Specify an action for the FORM.

2

FORMS

- To insert a form we use the <FORM></FORM> tags. The rest of the form elements must be inserted in between the form tags.

```
<HTML>
  <HEAD>
    <TITLE> Sample Form</TITLE>
  </HEAD>
  <BODY BGCOLOR="FFFFFF">
    <FORM ACTION = http://www.xyz.com/formtest.php>
      <P> First Name: <INPUT TYPE="TEXT" NAME="fname" MAXLENGTH="50"> </P>
      <P> <INPUT TYPE="SUBMIT" NAME="fsubmit" VALUE="Send Info"> </P>
    </FORM>
  </BODY>
</HTML>
```

<FORM> element attributes

- **ACTION:** is the **URL** of the **CGI** (Common Gateway Interface) program that is going to accept the data from the form, process it, and send a response back to the browser.
- **METHOD:** **GET** (default) or **POST** specifies which **HTTP** method will be used to send the form's contents to the web server. The CGI application should be written to accept the data from either method.
- **NAME:** is a form name used by **VBScript** or **JavaScripts**.
- **TARGET:** is the target frame where the response page will show up.

Form Elements

- Form elements have properties: **Text boxes, Password boxes, Checkboxes, Option(Radio) buttons, Submit, Reset, File, Hidden and Image.**
- The properties are specified in the **TYPE Attribute of the HTML element <INPUT></INPUT>**

First Name:

Last Name:

Gender::

Male

Female

Please choose type of residence::

Steak

Pizza

Chicken

Enter your favorite quote!

Select a Level of Education:

Diploma

Select your favorite time of day::

Morning

Day

Night

5

6

Form Elements

<INPUT> Element's Properties

TYPE = Type of INPUT entry field.

NAME = Variable name passed to CGI application

VALUE = The data associated with the variable name to be passed to the CGI application

CHECKED = Button/box checked

SIZE = Number of visible characters in text field

MAXLENGTH = Maximum number of characters accepted

7

Text Box

Text boxes: Used to provide input fields for text, phone numbers, dates, etc.

<INPUT TYPE= " TEXT " >

Browser will display



Textboxes use the following attributes:

- **TYPE:** text.
- **SIZE:** determines the size of the textbox in characters. **Default=20** characters.
- **MAXLENGTH :** determines the maximum number of characters that the field will accept.
- **NAME:** is the name of the variable to be sent to the CGI application.
- **VALUE:** will display its contents as the default value.

8



Example on Text Box

```
<TITLE>Form_Text_Type</TITLE>
</HEAD> <BODY>
<h1> <font color=blue> Please enter the following
    BioData</font></h1>
<FORM name="form1" Method= " get " Action= " URL " >
First Name: <INPUT TYPE="TEXT" NAME="FName"
SIZE="15" MAXLENGTH="25"><BR>
Last Name: <INPUT TYPE="TEXT" NAME="LName"
SIZE="15" MAXLENGTH="25"><BR>
Nationality: <INPUT TYPE="TEXT" NAME="Country"
SIZE="25" MAXLENGTH="25"><BR>
The Phone Number: <INPUT TYPE="TEXT" NAME="Phone"
SIZE="15" MAXLENGTH="12"><BR>
</FORM> </BODY> </HTML>
```

9

Output



Please enter the following BioData

First Name:

Last Name:

Nationality:

The Phone Number:

10



Password

Password: Used to allow entry of passwords.

<INPUT TYPE= " PASSWORD " >

Browser will display



Text typed in a password box is starred out in the browser display.

Password boxes use the following attributes:

- **TYPE:** password.
- **SIZE:** determines the size of the textbox in characters.
- **MAXLENGTH:** determines the maximum size of the password in characters.
- **NAME:** is the name of the variable to be sent to the CGI application.
- **VALUE:** is usually blank.

11

Example on Password Box

```
<HTML><HEAD>
<TITLE>Form_Password_Type</TITLE></HEAD>
<BODY>
<h1> <font color=red>To Access, Please
    enter:</font></h1>
<FORM name="form2" Action="url" method="get">
User Name: <INPUT TYPE="TEXT" Name="FName"
SIZE="15" MAXLENGTH="25"><BR>
Password: <INPUT TYPE="PASSWORD"
NAME="PWord" value="" SIZE="15"
MAXLENGTH="25"><BR>
</FORM></BODY> </HTML>
```

12

Output



To Access, Please enter:

User Name: ATI
Password: *****

13

Hidden

Hidden: Used to send data to the CGI application that you don't want the web surfer to see, change or have to enter but is necessary for the application to process the form correctly.

<INPUT TYPE="HIDDEN">

Nothing is displayed in the browser.

Hidden inputs have the following attributes:

- **TYPE:** hidden.
- **NAME:** is the name of the variable to be sent to the CGI application.
- **VALUE:** is usually set a value expected by the CGI application.

14

Check Box

Check Box: Check boxes allow the users to select more than one option.

<INPUT TYPE="CHECKBOX">

Browser will display



Checkboxes have the following attributes:

- **TYPE:** checkbox.
- **CHECKED:** is blank or CHECKED as the initial status.
- **NAME:** is the name of the variable to be sent to the CGI application.
- **VALUE:** is usually set to a value.

45

Example on Check Box

```

<HTML> <HEAD><TITLE>CheckBoxType</TITLE> </HEAD>
<BODY>
<h1> <font color=green>Please check one of the
following</font></h1>
<FORM name="form3" Action="url" method="get">
<font color=red> Select Country: </font><BR>
Sri Lanka:<INPUT TYPE="CheckBox" Name="country" CHECKED><BR>
England<INPUT TYPE="CheckBox" Name="country"><BR>
France<INPUT TYPE="CheckBox" Name="country"><BR> <BR>
<font color=blue>Select Language:</font><BR>
Sinhala<INPUT TYPE="CheckBox" Name="language" CHECKED><BR>
English<INPUT TYPE="CheckBox" Name="language"><BR>
French<INPUT TYPE="CheckBox" Name="language"> <BR></FORM>
</BODY></HTML>

```

16

Output

CheckBoxType - Mozilla Firefox

File Edit View History Bookmarks Tools Help

CheckBoxType

file:///F:/2014 1st Semester/Internet App/Practicals/checkbox.html

Please check one of the following

Select Country:

Sri Lanka:

England:

France:

Select Language:

Sinhala:

English:

French:

Radio Button

Radio Button: Radio buttons allow the users to select only one option.

<INPUT TYPE="RADIO">

Browser will display



Radio buttons have the following attributes:

- **TYPE:** radio.
- **CHECKED:** is blank or CHECKED as the initial status. Only one radio button can be checked
- **NAME:** is the name of the variable to be sent to the CGI application.
- **VALUE:** usually has a set value.

18

Example on Radio Button

```

<HTML> <HEAD><TITLE>CheckBoxType</TITLE> </HEAD>
<BODY>
<h1> <font color=green>Please check one of the
following</font></h1>
<FORM name="form3" Action="url" method="get">
<font color=red> Select Country: </font><BR>
jordan:<INPUT TYPE="RADIO" Name="country" CHECKED><BR>
Yemen<INPUT TYPE="RADIO" Name="country"><BR>
Qatar:<INPUT TYPE="RADIO" Name="country"><BR> <BR>
<font color=blue>Select Language:</font><BR>
Arabic:<INPUT TYPE="RADIO" Name="language" CHECKED><BR>
English:<INPUT TYPE="RADIO" Name="language"><BR>
French:<INPUT TYPE="RADIO" Name="language"> <BR></FORM>
</BODY></HTML>

```

Output

CheckBoxType

file:///F:/2014 1st Semester/Internet App/Practicals/radio.html

Please check one of the following

Select Country:

Sri Lanka:

England:

France:

Select Language:

Sinhala:

English:

French:

19

20



```

<HTML><HEAD>
<TITLE>RADIOBox</TITLE> </HEAD>
<BODY>
Form #1:
<FORM>
  <INPUT TYPE="radio" NAME="choice" VALUE="one"> Yes.
  <INPUT TYPE="radio" NAME="choice" VALUE="two"> No.
</FORM>
<HR color=red size="10" >
Form #2:
<FORM>
  <INPUT TYPE="radio" NAME="choice" VALUE="three" CHECKED>
Yes.
  <INPUT TYPE="radio" NAME="choice" VALUE="four"> No.
</FORM>
</BODY></HTML>

```

21

Output

Form #1:
 Yes. No.

Form #2:
 Yes. No.

22



Push Button

Push Button: This element would be used with JavaScript to cause an action to take place.

<INPUT TYPE="BUTTON">

Browser will display



Push Button has the following attributes:

- **TYPE:** button.
- **NAME:** is the name of the button to be used in scripting.
- **VALUE:** determines the text label on the button.

23

Example on Push Button

```

<DIV align=center><BR><BR>
<FORM>
<FONT Color=red>
<h1>Press Here to see a baby crying:<BR>
<INPUT TYPE="button" VALUE="Press Me"><BR><BR>
<FONT Color=blue>
Click Here to see a baby shouting:<BR>
<INPUT TYPE="button" VALUE="Click Me" > <BR><BR>
<FONT Color=green>
Hit Here to see a baby eating:<BR>
<INPUT TYPE="button" VALUE="Hit ME" > <BR><BR>
<FONT Color=yellow>
</FORM></DIV>

```

24

Output



Press Here to see a baby crying:

Click Here to see a baby shouting:

Hit Here to see a baby eating:

Submit Button

Submit: Every set of Form tags requires a Submit button.

This is the element causes the browser to send the names and values of the other elements to the CGI Application specified by the ACTION attribute of the FORM element.

<INPUT TYPE="SUBMIT">

The browser will display

Submit has the following attributes:

- **TYPE:** submit.
- **NAME:** value used by the CGI script for processing.
- **VALUE:** determines the text label on the button, usually Submit Query.

25

26

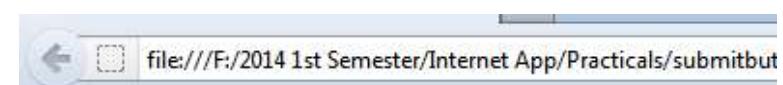
Example on Submit Button

```

<FORM Action="URL"      method="get">
First Name: <INPUT TYPE="TEXT" Size=25
name="firstName"><BR>
Family Name: <INPUT TYPE="TEXT" Size=25
name="LastName"><BR>
<BR>
<FONT Color=red>
Press Here to submit the data:<BR>
<INPUT TYPE="submit" VALUE="Submit Data " >
</FORM>

```

Output



First Name:

Family Name:

Press Here to submit the data:

27

28

Reset Button

Reset: It is a good idea to include one of these for each form where users are entering data. It allows the surfer to clear all the input in the form.

- `<INPUT TYPE="RESET">`

- Browser will display



Reset buttons have the following attributes:

- **TYPE:** reset.
- **VALUE:** determines the text label on the button, usually Reset.

29

Example on Reset Button

```

<FORM Action="URL" method="get">
First Name: <INPUT TYPE="TEXT" Size=25
name="firstName"> <BR>
Family Name: <INPUT TYPE="TEXT" Size=25
name="LastName"><BR>
<BR>
<FONT Color = red>
<STRONG><font size=5>Press Here to submit the
data:</font></STRONG><BR>
<INPUT TYPE="submit" VALUE="SubmitData">
<INPUT TYPE="RESET" VALUE="Reset">
</FORM>

```

30

Output

First Name:

Family Name:

Press Here to submit the data:

Image Submit Button

Image Submit Button: Allows you to substitute an image for the standard submit button.

```
<INPUT TYPE="IMAGE" SRC="image1.gif">
```

Image submit button has the following attributes:

- **TYPE:** Image.
- **NAME:** is the name of the button to be used in scripting.
- **SRC:** URL of the Image file.



```
<form>
<H1><font color=blue>
Click to go Sri Lanka's Map:
<INPUT TYPE="IMAGE" SRC="srilanka.jpg">
</form>
```



Click to go Sri Lanka's Map:



33

File

File Upload: You can use a file upload to allow surfers to upload files to your web server.

<INPUT TYPE="FILE">

Browser will display



File Upload has the following attributes:

- **TYPE:** file.
- **SIZE:** is the size of the text box in characters.
- **NAME:** is the name of the variable to be sent to the CGI application.
- **MAXLENGTH:** is the maximum size of the input in the textbox in characters.

34

Example on File

```
<BODY bgcolor=lightblue>
<form>
<H3><font color=forestgreen>
Please attach your file here to for uploading to
My <font color =red>SERVER...<BR>

<INPUT TYPE="File" name="myFile" size="30">

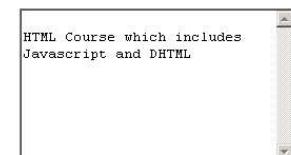
<INPUT TYPE="Submit" value="Submit File">
</form>
</BODY>
```

35

Other Elements used in Forms

<TEXTAREA></TEXTAREA>: is an element that allows for free form text entry.

Browser will display



Textarea has the following attributes:

- **NAME:** is the name of the variable to be sent to the CGI application.
- **ROWS:** the number of rows to the textbox.
- **COLS:** the number of columns to the textbox.

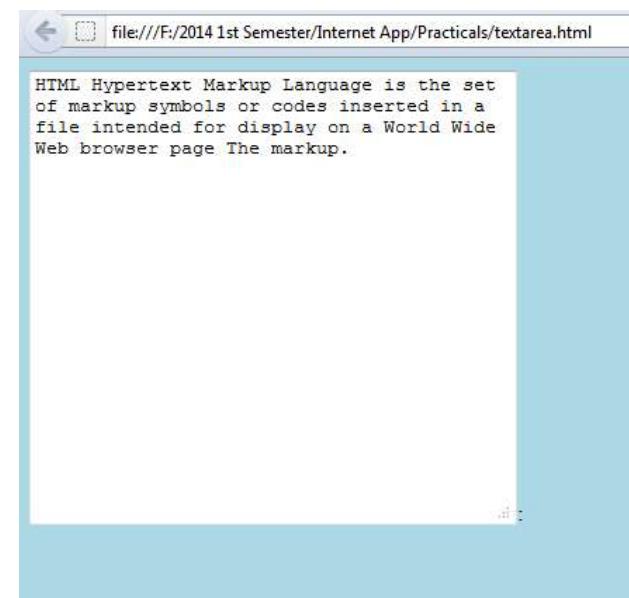
36

Example on <TEXTAREA>

```
<BODY bgcolor=lightblue>
<form>
<TEXTAREA COLS=40 ROWS=20 Name="comments" >
HTML Hypertext Markup Language is the set of markup symbols or
codes inserted in a file intended for display on a World Wide Web
browser page The markup.
</TEXTAREA>;
</form>
</BODY>
```

37

Output



38

Other Elements used in Forms

- The two following examples are <SELECT></SELECT> elements, where the attributes are set differently.
The Select elements attributes are:
 - **NAME:** is the name of the variable to be sent to the CGI application.
 - **SIZE:** this sets the number of **visible** choices.
 - **MULTIPLE:** the presence of this attribute signifies that the user can make multiple selections. By default only one selection is allowed.

39

Example on <SELECT>

```
<BODY bgcolor=lightblue>
<form>
Select the cities you have visited:
<SELECT name="list" size=5>
<option> London</option>
<option> Tokyo</option>
<option> Paris</option>
<option> New York</option>
<option> LA</option>
<option> KL</option>
</SELECT>
</form>
</BODY>
```

40

Output

Select the cities you have visited:

- London
- Tokyo
- Paris
- New York
- LA

Other Elements used in Forms

Drop Down List:



- **Name:** is the name of the variable to be sent to the CGI application.
- **Size: 1**

41

42

Other Elements used in Forms

List Box:



- **Name:** is the name of the variable to be sent to the CGI application.
- **SIZE:** is greater than one.

Other Elements used in Forms

Option

The list items are added to the **<SELECT>** element by inserting **<OPTION></OPTION>** elements.

The Option Element's attributes are:

- **SELECTED:** When this attribute is present, the option is selected when the document is initially loaded. **It is an error for more than one option to be selected.**
- **VALUE:** Specifies the value the variable named in the select element.

43

44

Example

```
<HTML>
<HEAD></HEAD>
<BODY>
<h2><font color=blue>What type of Computer do you
have?</font></h2>
<FORM>
<SELECT NAME="ComputerType" size=4>
<OPTION value="IBM" SELECTED> IBM</OPTION>
<OPTION value="INTEL"> INTEL</OPTION>
<OPTION value=" Apple"> Apple</OPTION>
<OPTION value="Compaq"> Compaq</OPTION>
</SELECT>
</FORM></BODY></HTML>
```

45

Output

What type of Computer do you have?



46

There are eleven different types of form elements:

Button	<input type="button" value="Button"/>
Checkbox	<input type="checkbox"/>
FileUpload	<input type="file"/>
Hidden	<input type="hidden"/>
Password	<input type="password"/>
Radio	<input type="radio"/>
Reset object	<input type="reset" value="Reset"/>
Select object	<input type="select"/>
Submit object	<input type="submit" value="Submit Query"/>
Text	<input type="text"/>
Textarea	<input type="textarea"/>

47

Subtopics



HNDIT1022 – Web Design

- Insert HTML link
- Insert images into web page
- HTML List
- Insert tables into web page

Week 3 Part 1: Understanding more HTML tags

HTML – links and images

Linking

- Most important feature is the hyperlink
 - Which references other resources, such as HTML documents and images.
- Both text and images can act as hyperlinks
- Web browsers underline text hyperlinks and colour their text blue by default.
 - User can distinguish hyperlinks from plain text
- Placing the mouse over this region and clicking the mouse button causes the client to access the indicated document or other Internet resource.

Anchors

- The element marking a hypertext link is called an anchor element, and the marked text is referred to as a hypertext anchor.
- The area between beginning `<a>` and ending `` tags becomes a hot part of text.



HTML Link

- Link syntax:
`Link text`
- The start tag contains attributes about the link.
- The element content (Link text) defines the part to be displayed.
- The element content doesn't have to be text. You can link from an image or any other HTML element.

The href attribute

- The href attribute defines the link "address".
- This `<a>` element defines a link to google:
`Visit Google!`
- The code above will display like this in a browser:
[Visit Google!](http://www.google.com/)



The target attribute

- The target attribute specifies where to open the linked document.
- Syntax:

```
<a target="value">
```

Attribute values

Value	Description
_blank	Open the linked document in a new window
_self	Open the linked document in the same frame as it was clicked (this is default)
_parent	Open the linked document in the parent frameset
_top	Open the linked document in the full body of the window

The target attribute-example

- The code below will open the document in a new browser window with no name:

```
<a href="http://www.google.com/"  
target="_blank">Visit Google!</a>
```

The name Attribute

- The name attribute is used to mark a place as a specific destination of a hypertext link.
- syntax:
`Any content`
 — The value “label” identifies this destination
- When using named anchors we can create links that can jump directly into a specific section on a page, instead of letting the user scroll around to find what he is looking for.
- The link syntax to a named anchor:
`Any content`
- The # in the href attribute defines a link to a named anchor.

Exercise

```
<html>  
<head>  
<title> Showing hyperlink </title>  
</head>  
<body>  
<a href="https://www.youtube.com/watch?v=Gdwst4kSfy4" target="_blank">  
17th Diploma Awarding Ceremony – SLIATE Join With Our Celebration!  
</a>  
</body>  
</html>
```

Output: linked page will be open in new window



Images

- Web pages contain both text and images
- Images are an equal part of web page design
- The three most popular image formats used by Web developers
 - Graphics Interchange Format (gif)
 - Joint Photographic Experts Group (jpeg)
 - Portable Network Graphics (png)
- Users can
 - Create images using software (Adobe Photoshop)
 - Acquire from web sites (Yahoo picture gallery)

img Element

- Use an img element to insert an image in the document
- The tag is empty, which means that it contains attributes only and it has no closing tag.
- The image file's location is specified with the src attribute.
 - src stands for "source"
- The value of the src attribute is the URL of the image you want to display on your page.
- The syntax of defining an image:
``

img Element (contd..)

- The browser puts the image where the image tag occurs in the document.
- If you put an image tag between two paragraphs, the browser shows the first paragraph, then the image, and then the second paragraph.

The alt attribute

- The alt attribute is used to define an "alternate text" for an image.
- The value of the alt attribute is an author defined text:
``
- The "alt" attribute tells the reader what he or she is missing on a page if the browser can't load images.
- The browser will then display the alternate text instead of the image.
- It is a good practice to include the "alt" attribute for each image on a page, to improve the display and usefulness of your document for people who have text-only browsers.

Size of the image

- The optional attributes width and height specify the width and height of the image respectively.
- Can scale an image by increasing or decreasing the values of the width and height
- If these are omitted, the browser uses the actual width and height attributes.
- Images are measured in pixels
- **Example**
 - An image with a height and a width of 100 pixels:
``

Size of the image (contd..)

Syntax:

```
<img height="value" >
```

Attribute Values

Value	Description
pixels	The height in pixels
percent	(like "100px" or just "100") The height in percent of the containing element (like "20%")

Example- image

```
<html>
<head>
<title> Showing IMG </title>
</head>
<body>
<h1> Image inclusion and hypertext links </h1>
<p> Greetings!</p>
<img src = "ATI.jpg" >
</body>
</html>
```

Images as link anchors

- Can create graphical web pages that link to other resources.
- Can turn an image into a hyperlink by putting it inside an anchor element

```
<a href = "https://www.gov.lk/welcome.html">
```

```
 
```

```
 </a>
```

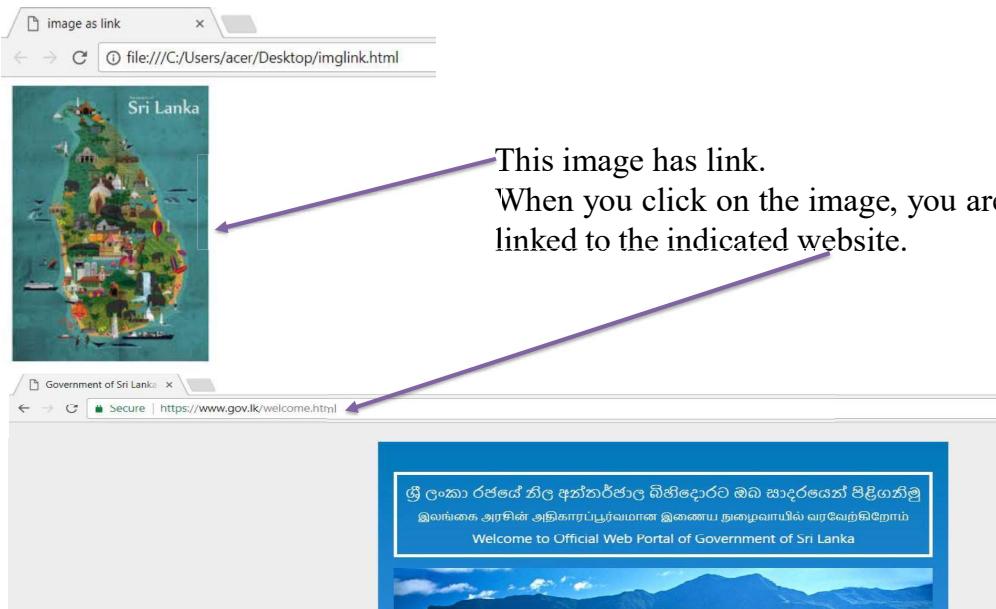
— When you click on the image, you are linked to the indicated document.

Exercise

```
<html>
<head>
<title>image as link</title>
</head>
<body>
<a href = "https://www.gov.lk/welcome.html">

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

Exercise: Output



The align attribute

- The align attribute specifies the horizontal and vertical alignment of an image according to the surrounding element.
- There are five alignment options.
- Syntax:
``

The align attribute(contd..)

- Values:
 - top – aligns the top of the image with the text
 - middle – aligns the middle of the image with the text
 - bottom – aligns the bottom of the image with the text
 - left – aligns the image to the left of text
 - right – aligns the image to the right of the text

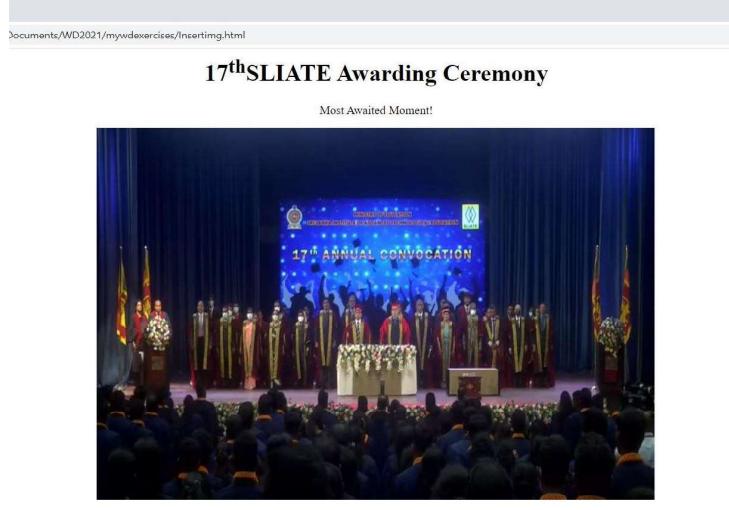
```

```

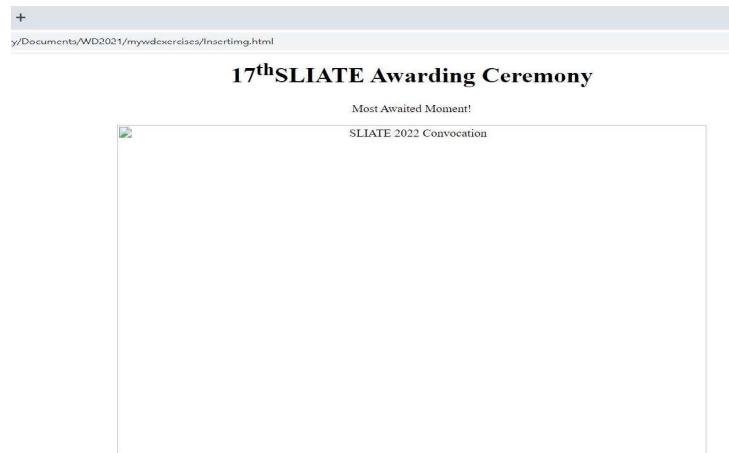
Exercise

```
<html>
  <head>
    <title> Showing Image </title>
  </head>
  <body>
    <h1 align="center"> 17th SLIATE Awarding Ceremony</h1>
    <p><center>Most Awaited Moment!</center></p>
    <center><img src = "convocation.jpg" alt="SLIATE 2022 Convocation" height="500" width="750"></center>
  </body>
</html>
```

Exercise: Output



Exercise: Output With alt attribute



HTML- character entities, styles, lists and tables

Character entities

- Certain character or symbols may be difficult to embed directly into a HTML document.
 - Some keyboards do not provide these symbols
 - Some characters like the < character, have a special meaning in HTML.
- Presence of these symbols may cause syntax errors
- Example:
`<p> if x<10 then increment x by 1</p>`

Character entities (contd..)

- HTML provides character entities for representing special characters.
- A character entity has three parts:
 - an ampersand (&)
 - an entity name or # and an entity number
 - a semicolon (;)
- The corrected line:
`<p> if x <10 then
</p>`
 - Uses the character entity <

increment x by
for the less than symbol

Character entities (contd..)

- Can write it as

```
<p> if x &#60; 10 then increment x by 1
</p>
```

- advantage of using a name instead of a number is that a name is easier to remember.
- The disadvantage is that not all browsers support the newest entity names, while the support for entity numbers is very good in almost all browsers.
- Note that the entities are case sensitive.

Non breaking space

- The most common character entity in HTML is the non-breaking space.
- Normally HTML will truncate spaces in your text.
 - If you write 10 spaces in your text HTML will remove 9 of them.
- To add spaces to your text, use the character entity.

The most common character entities

Result	Description	Entity	Entity
		Name	Number
	non-breaking space	 	
<	less than	<	<
>	greater than	>	>
&	ampersand	&	&
"	quotation mark	"	"
'	apostrophe		'

Some other commonly used Character entities

Result	Description	Entity	Entity
		Name	Number
	cent	¢	¢
£	pound	£	£
¥	yen	¥	¥
§	section	§	§
©	copyright	©	©
®	registered	®	®
	trademark		
×	multiplication	×	×
÷	division	÷	÷



Example

```
<html>
<body>
<p>&copy; reserved</p>
<p>We&hearts;Our Country</p>
<p>you&nbsp;&amp;&nbsp;me forever friends</p>
<p>I&nbsp;&nbsp;&nbsp; will &nbsp;&nbsp;go</p>
<p>number 5 is &#60; number 12</p>
<p>number 5 is &lt; number 12</p>
<p>number 15 is &#62; number 9</p>
<p>number 15 is &gt; number 9</p>
</body>
</html>
```

© reserved
We♥Our Country
you & me forever friends
I will go
number 5 is < number 12
number 5 is < number 12
number 15 is > number 9
number 15 is > number 9

Tabs in HTML

- White space characters
 - The space
 - The tab
 - The carriage return
- In print they act differently
- In HTML they act exactly the same
 - If you place 1 space or 100 or mix it up with tabs and carriage returns, they will all be condensed down to 1 space when the page is rendered by the browser.



Using the Tab Character

- Tab markers along the horizontal axis
 - You can't
- HTML can handle
 - Tabs for layout
 - to get the text to move over a certain amount

Moving text more than one space

- To move more than one space away from the preceding item, you can use the non-breaking space.
- To use the non-breaking space, you simply type as many times as you need it.



Styles attribute

- The style attribute is a new HTML attribute.
- It introduces CSS to HTML.
- The purpose of the style attribute is:
- To provide a common way to style all HTML elements.
- The style attribute specifies an inline style for an element.
- The style attribute will override any style set globally, e.g. styles specified in the <style> tag or in an external style sheet.
- Syntax
`<element style="value">`
- Attribute Values
 - One or more CSS properties and values separated by semicolons

HTML Style Examples

- `style="background-color:yellow"`
- `style="font-size:10px"`
- `style="font-family:Times"`
- `style="text-align:center"`



Example

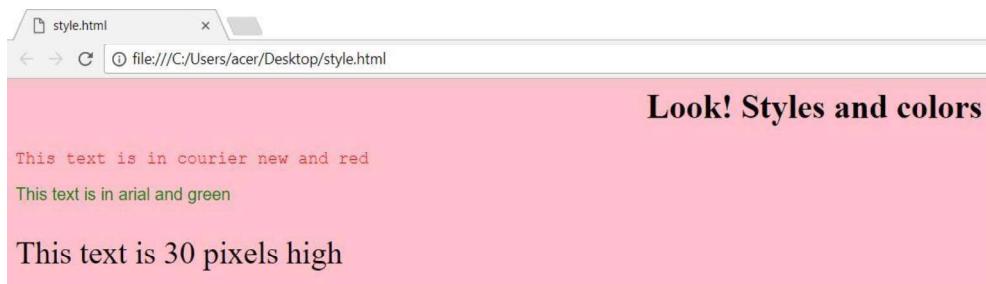
- Use of the style attribute in an HTML document:
`<h1 style="color: blue; text-align: center">This is a header</h1>`
`<p style="color:green">This is a paragraph.</p>`

Style Examples:

- Background Color
 - `<body style="background-color:yellow">`
 - The style attribute defines a style for the <body> element.
- Font Family, Color and Size
 - `<p style="font-family:courier new; color:red; font-size:20px">`
 - The style attribute defines a style for the <p> element.
- Text Alignment
 - `<h1 style="text-align:center">`
 - The style attribute defines a style for the <h1> element.

Style Examples (contd.)

```
<html>
<body style="background-color:pink;">
    <h1 style="text-align:center">Look! Styles and colors</h1>
    <p style="font-family:courier new;color:red">This text is in courier new and red</p>
    <p style="font-family:arial;color:green"> This text is in arial and green</p>
    <p style="font-size:30px">This text is 30 pixels high</p>
</body>
</html>
```



List Elements

- HTML has several elements for defining different types of lists.
- Can be divided into two types
 - Regular lists (the elements OL, and UL)
 - Definition lists (the element DL)
- List of the same or different types can be nested.

Unordered lists(UL)

- The ul element defines an unordered list.
 - A list that does not order its elements by letter or number
- Each list item is indicated by a special symbol
 - Bullet or asterisk
- Each item in an un ordered list is contained within an li(list item) element
 - List element is the only thing that can appear inside an un ordered list .
- Rendered with a line break and a bullet symbol indented from the beginning of the new line

Value	Description
disc	Sets the list item marker to a bullet (default)
circle	Sets the list item marker to a circle
square	Sets the list item marker to a square
none	The list items will not be marked

Example 1

```
<html>
<body>
<h4>Types of computers:</h4>
<ul>
    <li>Mainframe computer</li>
    <li>Mini computer</li>
    <li>Micro computer</li>
</ul>
</body>
</html>
```

Types of computers:

- Mainframe computer
- Mini computer
- Micro computer



Example 2

```

<html>
<body>
<ul type="none">
<li>Banana</li>
<li>Orange</li>
<li>Pineapple</li>
</ul>
<ul type="disc">
<li>Vanila</li>
<li>Chocolate</li>
<li>Mixed</li>
</ul>
<ul type="square">
<li>Red</li>
<li>Yellow</li>
<li>Green</li>
</ul>

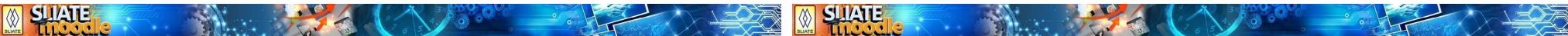
```

Banana
 Orange
 Pineapple
 • Vanila
 • Chocolate
 • Mixed
 ▪ Red
 ▪ Yellow
 ▪ Green
 ○ Rose
 ○ Jasmine
 ○ Sun Flower

Ordered lists(OL)

- The ol defines an ordered list
- A browser indicates ordering by numbering the items
 - Assigning them ascending numbers, letters, etc.
- Each item in an ordered list is contained within an li(list item) element
 - List element is the only thing that can appear inside an ordered list.
- Items are enumerated 1, 2, 3 and so on

Type	Description
type="A"	The list items will be numbered with uppercase letters
type="a"	The list items will be numbered with lowercase letters
type="I"	The list items will be numbered with uppercase roman numbers
type="i"	The list items will be numbered with lowercase roman numbers
type="1"	The list items will be numbered with numbers (default)



Example 1

```

<html>
<body>
<ol>
<li>HNDA</li>
<li>HNDE</li>
<li>HNDIT</li>
</ol>
<ol start="50">
<li>Maths</li>
<li>Science</li>
<li>English</li>
</ol>
</body>
</html>

```

1. HNDA
 2. HNDE
 3. HNDIT
 50. Maths
 51. Science
 52. English

Example 2

```

<html>
<body>
<ol type="1">
<li>Banana</li>
<li>Orange</li>
<li>Pineapple</li>
</ol>
<ol type="i">
<li>Vanila</li>
<li>Chocolate</li>
<li>Mixed</li>
</ol>
<ol type="I">
<li>Red</li>
<li>Yellow</li>
<li>Green</li>
</ol>

```

1. Banana
 2. Orange
 3. Pineapple
 i. Vanila
 ii. Chocolate
 iii. Mixed
 I. Red
 II. Yellow
 III. Green
 A. Rose
 B. Jasmine
 C. Sun Flower
 a. Lecturer
 b. Doctor
 c. Engineer

Definition lists

- Presents a list of items, each with a descriptive paragraph of each item.
 - Can be used for traditional glossaries
- A definition list starts with a `<dl>` tag (**definition list**).
- Each term starts with a `<dt>` tag (**definition term**).
- Each description starts with a `<dd>` tag (**definition description**).
- DT and DD elements should appear in pairs.

DT element

- The DT element contains the term part of the description list
- The contents should be short
 - A few words
 - Shorter than a line

DD element

- Gives the description corresponding to the previous DT element.
- Can be a long description
 - Broken into paragraphs
 - Lists
 - Forms
 - Etc
- A DD element must always follow a DT element.
 - Cannot occur alone.

Example 1: definition list

```
<html>
<body>
<dl>
  <dt>HNDA</dt>
    <dd>Higher National Diploma in Accountancy</dd>
  <dt>HNDIT</dt>
    <dd>Higher National Diploma in Information Technology</dd>
</dl>
</body>
</html>
```

HNDA
Higher National Diploma in Accountancy
HNDIT
Higher National Diploma in Information Technology

Example 1: Nested List

```
<html>
<head>
Nested List
</head>
<body>
<ol>
<li>Believe in yourself</li>
<li>You want
    <!-- Start nested list -->
    <ol>
        <li>to manage time</li>
        <li>to do hard work</li>
        <li>to be happy</li>
        <li>to follow best Attitude</li>
    </ol>
    <!-- end nested list -->
</li><!-- close list item containing nested list -->
<li>Success is yours!!!</li>
</ol>
</body>
</html>
```

Nested List

1. Believe in yourself
2. You want
 1. to manage time
 2. to do hard work
 3. to be happy
 4. to follow best Attitude
3. Success is yours!!!

Tables

- Used to organize data into rows and columns.
- Defined with the `table` element.
- Has three sections
 - Head
 - Contains header information – column names
 - Body
 - Contains primary data
 - Foot
 - Commonly includes calculation results and footnotes

Table tag

- Tables are defined with the `<table>` tag.
- A table is divided into rows (with the `<tr>` tag),
- and each row is divided into data cells (with the `<td>` tag).
 - The letters `td` stands for "table data," which is the content of a data cell.
- A data cell can contain text, images, lists, paragraphs, forms, horizontal rules, tables, etc.

Border Attribute

- Specifies the width (in pixels) of the borders around a table.
- The border attribute applies a border to each cell, and around the table.
- If you do not specify a border attribute the table will be displayed without any borders.
 - Sometimes this can be useful, but most of the time, you want the borders to show.
- Syntax


```
<table border= "value">
```

Width attribute

- The width attribute specifies the width of a table.
 - If the width attribute is not set, a table takes up the space it needs to display the table data.
 - Syntax**
- ```
<table width="value">
```
- Attribute Values**
    - Pixels:** Sets the width in pixels (example: width="50")
    - % :** Sets the width in percent of the surrounding element (example: width="50%")

## Table description elements

The structure and contents of the table are defined by additional elements that can appear inside the table.

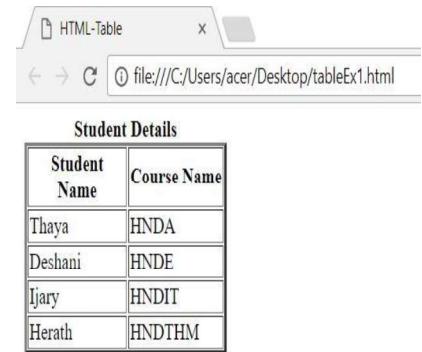
Tag	Description
<table>	Defines a table
<th>	Defines a table header
<tr>	Defines a table row
<td>	Defines a table cell
<caption>	Defines a table caption
<rowspan>	Defines the number of rows a cell should span/merge.
<colspan>	Defines the number of columns a cell should span.
<colgroup>	Defines groups of table columns
<col>	Defines the attribute values for one or more columns in a table
<thead>	Defines a table head
<tbody>	Defines a table body
<tfoot>	Defines a table footer

## Example 1 - Table

```
<html>
 <head>
 <title>HTML-Table</title>
 </head>
 <body>
 <table border="2">
 <caption>
 Student Details
 </caption>
 <tr>
 <th>Student Name</th>
 <th>Course Name</th>
 </tr>
 <tr>
 <td>Thaya</td>
 <td>HNDA</td>
 </tr>
 </table>
 </body>
</html>
```

```
<tr>
 <td>Deshani</td>
 <td>HNDE</td>
</tr>
<tr>
 <td>Ijary</td>
 <td>HNDIT</td>
</tr>
<tr>
 <td>Herath</td>
 <td>HNDTHM</td>
</tr>
</table>
</body>
</html>
```

## Example 1 – Table: Output



Student Name	Course Name
Thaya	HNDA
Deshani	HNDE
Ijary	HNDIT
Herath	HNDTHM



## Example 2 - Table

```
<html>
 <head>
 <title>Table with colspan</title>
 </head>
 <body>
 <table border="1">
 <tr>
 <td colspan="3" align="center">Subject Details</td>
 </tr>
 <tr>
 <th>Code No</th>
 <th>Subject</th>
 <th>No Of Credits</th>
 </tr>
 <tr>
 <td>HNDIT 1012</td>
 <td>Visual Application Programming</td>
 <td>04</td>
 </tr>
 </table>
 </body>
</html>
```

```
<tr>
 <td>HNDIT 1022</td>
 <td>Web Design</td>
 <td>04</td>
</tr>
<tr>
 <td>HNDIT 1032</td>
 <td>Computer and Network Systems</td>
 <td>03</td>
</tr>
<tr>
 <td colspan ="2" align="center">Total Credits</td>
 <td>07</td>
</tr>
</table>
</body>
</html>
```



## Example 2 - Table: Output

Subject Details		
Code No	Subject	No Of Credits
HNDIT 1012	Visual Application Programming	04
HNDIT 1022	Web Design	04
HNDIT 1032	Computer and Network Systems	03
Total Credits		07

## Example 3 - Table

```
<html>
 <head>
 <title>Table with rowspan</title>
 </head>
 <body>
 <table border="1" width="40%">
 <tr>
 <th colspan="2" align="center">Personal Details</th>
 </tr>
 <tr>
 <th>Name</th>
 <td>Rozan</td>
 </tr>
 <tr>
 <th rowspan="2">Contact Number</th>
 <td>021 226 3245</td>
 </tr>
 </table>
 </body>
</html>
```



<tr>

```
<td>077 8675443</td>
```

</tr>

<tr>

```
<th>Address</th>
```

```
<td>Colombo</td>
```

</tr>

</table>

</body>

</html>

## Example 3 - Table: Output

Personal Details	
Name	Rozan
Contact Number	021 226 3245 077 8675443
Address	Colombo



## Example 4 - Table

<html>

<head>

<title>HTML -Tables</title>

</head>

<body>

```
<table border = "1" width = "40%">
```

```
<caption>Price of Fruit</caption>
```

<thead>

<tr>

```
<th>Fruit</th>
```

```
<th>Price</th>
```

</tr>

</thead>

<tfoot>

<tr>

```
<th>Total</th>
```

```
<th>Rs.180.00</th>
```

</tr>

</tfoot>

## Example 4(contd..)

<tbody>

```
<tr>
```

```
<td>Mango</td>
```

```
<td>Rs.60.00</td>
```

</tr>

<tr>

```
<td>Banana</td>
```

```
<td>Rs.45.00</td>
```

</tr>

<tr>

```
<td>Pineapple</td>
```

```
<td>Rs.75.00</td>
```

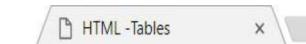
</tr>

</tbody>

</table>

</body>

</html>



file:///C:/Users/acer/Desktop/tableex.html

Price of Fruit

Fruit	Price
Mango	Rs.60.00
Banana	Rs.45.00
Pineapple	Rs.75.00
Total	Rs.180.00

## Questions...?

## Subtopics

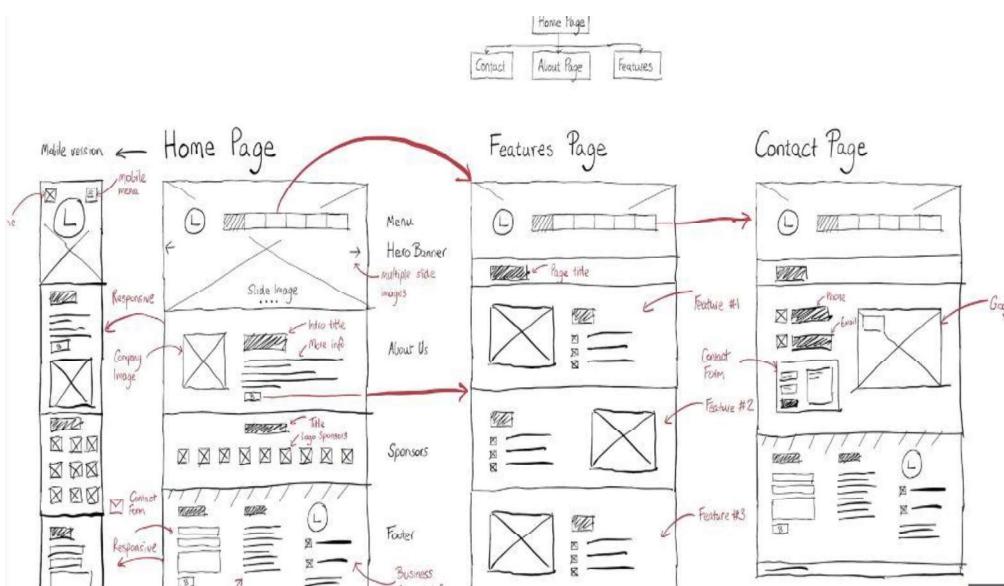


# HNDIT1022 – Web Design

- Why designer use wireframes
- Design wireframe

## Week 4: Define and plan the information hierarchy

## Wireframe?



## Wireframe?



# What is Wireframe?

Basically wireframe is a sketch of your website, before any kind of design elements or development even takes place.

It is a visual representation of elements on a website.

It usually doesn't include any styling, color, or graphics.

A wireframe is constructed using basic boxes, lines, and other shapes to create an outline of the functional parts of a web page without wasting time creating an intricate, polished design.

It's often the first step in the website design and web development process.

## Contd..

Definition in detail : A visual representation of an interface , used to communicate the structure, content, information hierarchy, functionality and behavior of an interface

### Structure

How will the pieces of this site be put together?

### Content

What will be displayed on this site?

### Information Hierarchy

How is this information organized and displayed?

### Functionality

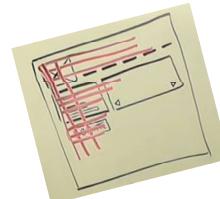
How will this interface work?

### Behavior

How does it interact with the user? How does it behave?



## Why designer use wireframe? / Why is a wireframe important?



- Establish hierarchy of information**

You want to follow that F-Shape pattern that users are looking for the information in. And it lets you understand what are the most important parts of your web site. What do you need people to click on first?

- Simplifies communication**

It simplifies the communication between developer and your client, which is always good for everyone involved.

- Blueprint for the design**  
it means an outline.

## Type of wireframes

- Sketches
  - Quick/experiment
  - Good for feedback
- Low fidelity
  - Block diagrams
  - Good for flows
- High fidelity
  - Detailed wireframes
  - Including comments
  - Describe content & behavior
  - Should be understood without explanation

## How to design wireframes

Simply there are two types of designing

- **Physical hand-drawn Sketches** can be made with paper, pens and markers
- **Digital wireframe** can be made using a variety of tools

## Digital Wireframe Tools :

### Free and Paid Wireframe Tools

You can find a range of great tools for wireframing with a quick Google search, many of which are free to use and often browser-based!



- Figma
- Diagrams.net
- Mydrawt.cc
- Pencil Project
- Justinmind
- Wirefy



Contd..



- Wireframe.cc
- Balsamiq Mockups
- Sketch
- MockFlow
- Cacoo
- InVision
- Miro
- Adobe XD
- Marvel



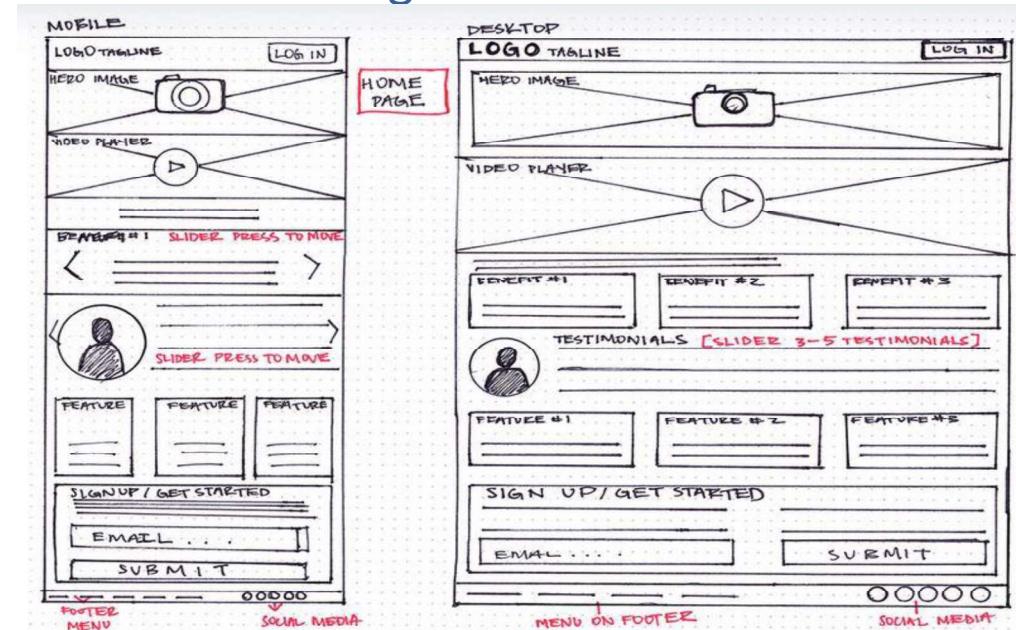
## How to design wireframes

- Whether the hand-drawn sketch wireframe or digital wireframe they're both made up of the same things basic fundamental shapes to represent the key elements that would you see on a screen in their simplest form and they're usually in black white and sometimes some gray
- The important thing about wireframes is to keep it simple more detail can be distracting and it's important to keep things to their most basic fundamental forms.
- Once you've got some initial ideas down on paper you might notice that not everything fits quite as you envisaged.

## How to design wireframes Contd..

- Take the time to do a quick second sketch and refine key elements if they don't fit quite right.
- Digital wireframes are a great step after hand-drawn sketches because they're easy to share with colleagues and the rest of the design team who can continue to add layers of complexity and polish.

## How to design wireframes Contd..



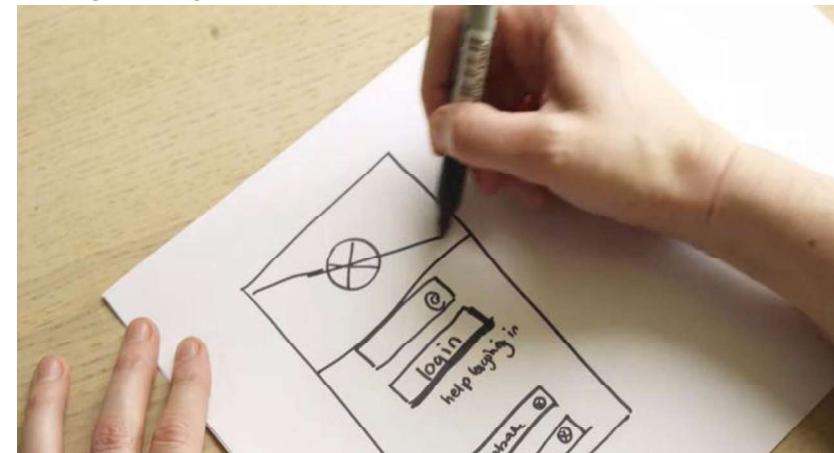
## Standard Elements on Wireframe

A wireframe usually includes the following standard elements.

1. Logo
2. Search field
3. Breadcrumb
4. Headers, including page title as the H1 and subheads H2-Hx
5. Navigation systems, including global navigation and local navigation
6. Body content
7. Share buttons
8. Contact information
9. Footer

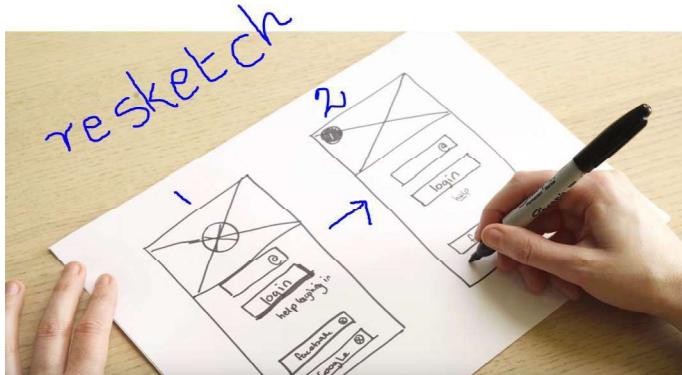
## Let's build your first wireframe

- Let's get started with the login frame
- Login frame elements : image, logo, text field to enter user id or email id, login button, forgot password link, alternative login options such as Facebook , Google
- Step 1: Sketching the login frame elements



# Let's build your first wireframe

- Step 2 : re-sketch the wireframe and place the elements in right place

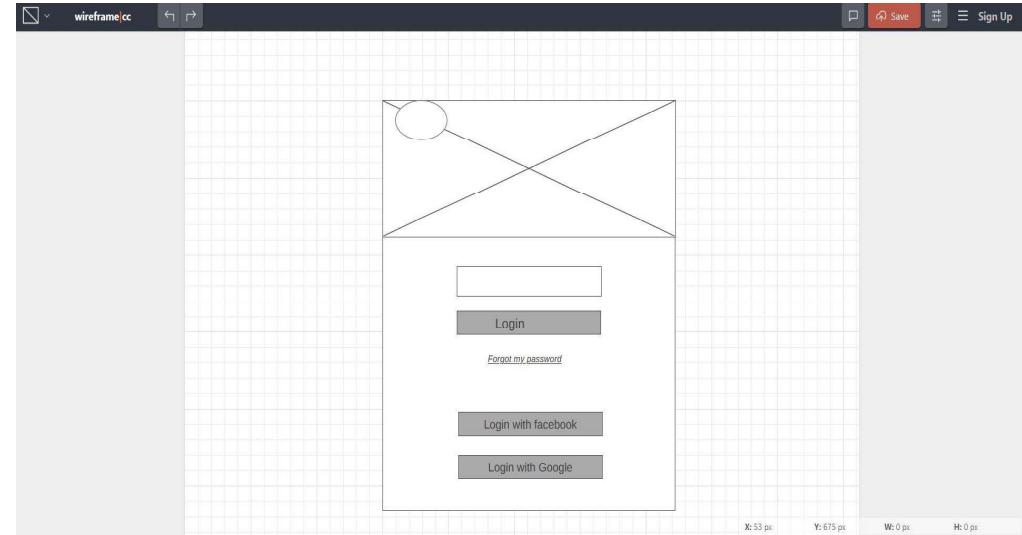


- Step 3 : After that you need to digitizing the wireframe using any tool

## Contd..

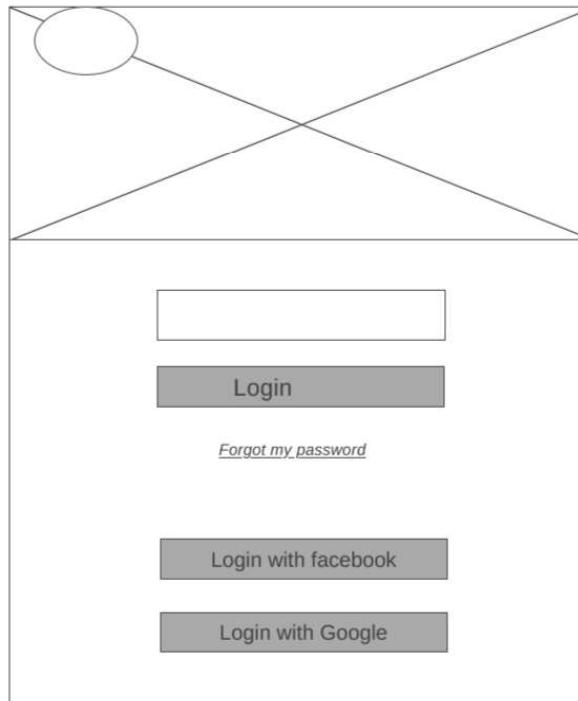
- let's use 7 days free trial online tool wireframe.cc to draw the digital format.

Note: you can use any wireframing tool



## Contd..

Digital wireframe



## Bad wireframe vs Good design wireframe

This is a bad wireframe!

Forget visual design

Avoid using color / gradients

Adding unnecessary elements makes the wireframe less powerful, remember:

Wireframes focus only on the content and interaction of the interface!





## Contd..

Looks good, right?



## Let's Start.....

Start big! Start with the largest parts of the interface, the frame, header, footer, etc. Then work you way through the smaller elements of the interface. The following slides will show a quick wireframe build-up.

### Final tip!

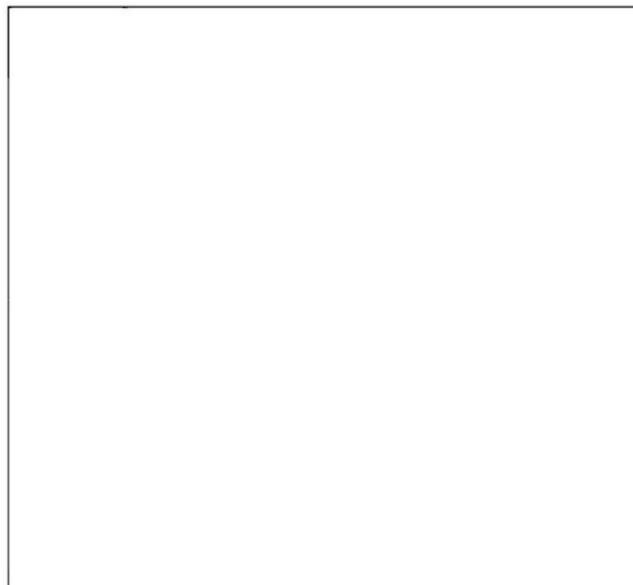
Remember, your wireframe will be used by lots of different people for different purposes:

Designers  
Developers  
Project leaders  
Usability testers  
Clients

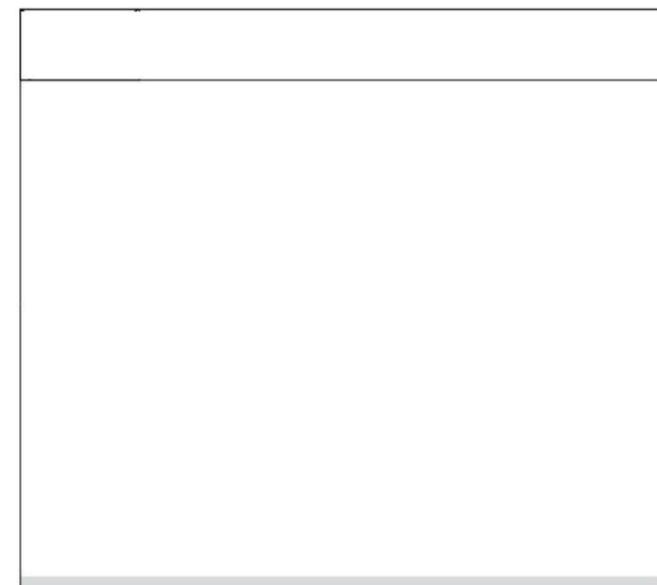
So be as clear as you can!



## Contd..



## Contd..





Contd..

A screenshot of a community page from the Slate Moodle theme. The header includes a logo, the word "Slate", and "Moodle". Below the header is a navigation bar with links: Features (unchecked), Demo (unchecked), Downloads (unchecked), Support (unchecked), Community (unchecked), Styles (unchecked), and Mods (unchecked). The main content area is completely blank.

Contd..

A screenshot of a community page from the Slate Moodle theme. The header includes a logo, the word "Slate", and "Moodle". Below the header is a navigation bar with links: Features (unchecked), Demo (unchecked), Downloads (unchecked), Support (unchecked), Community (unchecked), Styles (unchecked), and Mods (unchecked). The main content area is completely blank. At the bottom right of the header, there is a small menu with links: FAQ (unchecked), Search (unchecked), Members (unchecked), and Logout (unchecked).



Contd..

A screenshot of a community page from the Slate Moodle theme. The header includes a logo, the word "Slate", and "Moodle". Below the header is a navigation bar with links: Features (unchecked), Demo (unchecked), Downloads (unchecked), Support (unchecked), Community (unchecked), Styles (unchecked), and Mods (unchecked). The main content area contains a "Community title" section with a "Community" link and a "Logout" link. Below this is a forum table titled "General". The table has columns: Forum, Topics, Posts, and Date. It lists several forum categories with their respective counts and last post details. For example, the "Community" category has 145 topics, 234 posts, and was last updated on Thu May 25, 2008 by John Smith.

Contd..

A screenshot of a community page from the Slate Moodle theme. The header includes a logo, the word "Slate", and "Moodle". Below the header is a navigation bar with links: Features (unchecked), Demo (unchecked), Downloads (unchecked), Support (unchecked), Community (unchecked), Styles (unchecked), and Mods (unchecked). The main content area contains a "Community title" section with a "Community" link and a "Logout" link. Below this is a forum table titled "General". The table has columns: Forum, Topics, Posts, and Date. It lists several forum categories with their respective counts and last post details. For example, the "Community" category has 145 topics, 234 posts, and was last updated on Thu May 25, 2008 by John Smith. The table also includes sections for "School", "Music", and "Books".



Contd..

1

Community title

FAQ Search Members Logout

Forum Topics Posts Date

Category	Topic	Posts	Last Post
General	Community	145	Thu May 25, 2006 posted by John Smith
School	VID	215	Thu May 25, 2006 posted by John Smith
	IAD	123	Thu May 25, 2006 posted by John Smith
	MMS	143	Thu May 25, 2006 posted by John Smith
Music	Artist	435	Thu May 25, 2006 posted by John Smith
	Tracks	235	Thu May 25, 2006 posted by John Smith
	Radio	64	Thu May 25, 2006 posted by John Smith
	Albums	12	Thu May 25, 2006 posted by John Smith

2

Contd..

1

Community title

FAQ Search Members Logout

Forum Topics Posts Date

Category	Topic	Posts	Last Post
General	Community	145	Thu May 25, 2006 posted by John Smith
School	VID	215	Thu May 25, 2006 posted by John Smith
	IAD	123	Thu May 25, 2006 posted by John Smith
	MMS	143	Thu May 25, 2006 posted by John Smith
Music	Artist	435	Thu May 25, 2006 posted by John Smith
	Tracks	235	Thu May 25, 2006 posted by John Smith
	Radio	64	Thu May 25, 2006 posted by John Smith
	Albums	12	Thu May 25, 2006 posted by John Smith

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

9999



## Week 5 - Cascading Style Sheet (CSS)



## Learning Outcome

- Use style sheets to enhance the user interface of a web site.

## Cascading Style Sheet

- Cascading Style Sheet (CSS) is a style sheet language used to describe the presentation of a document written in a markup language (e.g. HTML and XHTML).
- CSS is a list of statements (rules) that define colors, fonts, layout, and other aspects to HTML elements.

## Why Use Style Sheets?

- It allows much greater degree of layout and display control.
- The amount of format coding necessary to control display characteristics can be greatly reduced.
- It allows multiple styles to be attached to a document at once.
- It also allows for all the style formatting in a document to be changed at once, thus a document can be easily formatted for different purposes (online, brochures, printing, etc.).

## Three ways to insert CSS

- **External style sheets**, i.e. a separate CSS-file referenced from the document
- **Embedded style (internal)**, blocks of CSS information inside the HTML document itself
- **Inline style**, attaches a style definition within the HTML element it is modifying

## Inline Style

- To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property.

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

</body>
</html>
```

## Embedded / internal Style Sheet

- An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section of an HTML page, by using the <style> tag

```
<html><head>
<style type="text/css">
.border{
border-bottom-width:thick;
border-bottom-style:double;
border-bottom-color:red;
}
</style></head>
<body>

</body>
</html>
```

# External Style Sheet

- An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing one file. Each page must link to the style sheet using the <link> tag. The <link> tag goes inside the head section

```
<html>
<head>
<link rel="stylesheet" type="text/css"
 href="externalstyle.css"/>
</head>
<body>

</body>
</html>
```

```
.border{
 border-bottom-width:thick;
 border-bottom-style:double;
 border-bottom-color:red;
}
```

*Externalstyle.css*

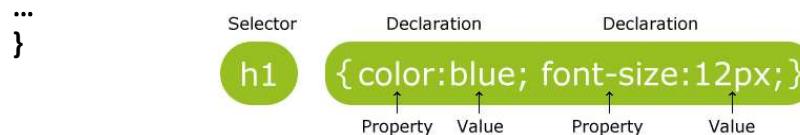
# Cascading order

- What style will be used when there is more than one style specified for an HTML element? (multiple style sheets)
  - Inline style (inside an HTML element)
  - Internal style sheet (in the head section)
  - External style sheet
  - Browser default

# Defining a Style

- CSS style definitions follow the format given below:

```
selector_expression {
 element_property: property_value(s);
 element_property: property_value(s);
```



- The **selector\_expression** is an expression that can be used to match specific elements (HTML elements/tags) in the document; the **element\_property** specifies which properties of the element the definition will affect, and each property can take a value. The property and value are separated by a colon, and surrounded by curly braces.

# Example Code

```
<html>
<head>
<style type="text/css">
 body {
 background-color:rgb(255,0,255);
 }
 h1 {
 background-color:yellow;
 }
</style>
</head>

<body>
 <h1>This is heading 1</h1>
 <p>This is a paragraph.</p>
</body>
</html>
```

## Example -view



## Property Values and Units

CSS supports the following metrics for property values :

- CSS keywords and other properties, such as thin, thick, transparent, ridge, and etc.
- Real-world measures
  - Inches (in)
  - Centimeters (cm)
  - Millimeters (mm)
  - Points (pt) – The points used by CSS2 are equal to 1/72 of an inch
  - Picas (pc) – 1 pica is equal to 12 points
- Screen measures in pixels (px)
- Relational to font size (font size (em) or x-height size (ex))
- Percentages (%)
- Color codes (#rrggbb or rgb(r,g,b) )

## Property Values and Units

- Angles
  - Degrees (deg)
  - Radians (rad)
- Time values (seconds (s) and milliseconds (ms)) – Used with aural style sheets
- Frequencies (hertz (Hz) and kilohertz (kHz))
  - Used with aural style sheets

## Selectors

- Selectors are essentially patterns that enable a user agent to identify what elements get what styles.
  - E.g. “If it is a paragraph tag, give it this style”:  
`p { text-indent: 2em; }`
- The ways of matching:
  - Matching elements by name
  - Matching using the universal selector
  - Matching elements by class
  - Matching elements by identifier
  - Matching elements by specific attributes

# Matching Elements by Name

- Example:
- **h1 { color: green; }**
  - Causes all occurrences of the selector tags (h1) to be formatted with the property/value section of the definition (color: green)
- Multiple selectors
  - Example
  - **h1, h2, h3, h4 { color: green; }**

## Matching Using the Universal Selector

- The universal selector (\*) can be used to match any element in the document.
- E.g.
  - Match every tag and apply the color red:  
**\* { color: red; }**
  - Match any **<ol>** tag that is a descendant of a **<td>** tag, which is a descendant of a **<tr>** tag: **tr td ol { color: red; }**

# Matching Elements by Class

- With the class selector you can define different styles for the same type of HTML element.
- Say that you would like to have two types of paragraphs in your document: one right-aligned paragraph, and one center-aligned paragraph. Here is how you can do it with styles:

- **p.right {text-align: right}**
  - **p.center {text-align: center}**

- You have to use the class attribute in your HTML document:

```
<p class="right">This paragraph will be rightaligned.</p>
```

```
<p class="center">This paragraph will be
centeraligned.</p>
```

# Matching Elements by Class

- To apply more than one class per given element, the syntax is:  
**<p class="center bold"> This is a paragraph. </p>**  
The paragraph above will be styled by the class "center" AND the class "bold".
- You can also omit the tag name in the selector to define a style that will be used by all HTML elements that have a certain class. In the example below, all HTML elements with class="center" will be center-aligned:

```
.center {text-align: center}
```

```
<h1 class="center">This heading will be center-aligned.
</h1>
```

```
<p class="center">This paragraph will be centeraligned.</p>
```



# Example

```

<html>
<head>
<style type="text/css">
p{
color:blue;
text-align:center;
}
#para1
{
text-align:left;
color:green;
}

p.left{text-align:left;}
.center{text-align:center;}
</style>
</head>
<body>

```

Sri Lanka's documented history spans 3,000 years

<p id="para1">

Sri Lanka's documented history spans 3,000 years

<p class="left">

Sri Lanka's documented history spans 3,000 years

</p>

<p class="center">

Sri Lanka's documented history spans 3,000 years

</p>

Sri Lanka's documented history spans 3,000 years

</body> </html>

Sri Lanka's documented history spans 3,000 years



## Matching elements by Specific Attributes

- To match any attribute in elements, not just class and id, the attribute and the value(s) have to be specified at the end of the selector, offset in square brackets. E.g.
  - Match any table with a border attribute set to 3, **table[border="3"]**
  - Match any table with a border attribute, **table[border]**
  - Match any table elements with a class value of *datalist* and *border* value of 3, **table.datalist[border="3"]**

# Example

```

<html>
<head>
<style>
a[target] {
background-color: yellow;
}
</style></head>
<body>

```

The links with a target attribute gets a yellow background:

[w3schools.com](http://www.w3schools.com) [disney.com](http://www.disney.com) [wikipedia.org](http://www.wikipedia.org)

<p>The links with a target attribute gets a yellow background:</p>

<a href="http://www.w3schools.com">w3schools.com</a>

<a href="http://www.disney.com" target="\_blank">disney.com</a>

<a href="http://www.wikipedia.org" target="\_top">wikipedia.org</a>

</body>

</html>

# Pseudo-classes

- Pseudo classes are identifiers that are understood by user agents and apply to elements of certain types without the elements having to be explicitly styled. E.g. Anchor styles

```
<html>
<head>
<style type="text/css">
:link{color:#FF00FF}
:visited{color:#FF00FF}
:active{color:#FF00FF}
</style>
</head>
<body>
 Visit SLIATE Web
</body>
</html>
```

# Fonts

- Fonts are styled collection of letters and symbols. Different fonts can be used to convey different information.
- CSS supports five different font family types:
  - Serif fonts** They are used in body text; the finishing strokes, flared or tapering ends, or serifed endings, make the lines of characters flow and tend to be easier on the eyes.
  - Sans serif fonts** They are used for headings or other large areas of emphasis.
  - Cursive fonts** They are used in extreme cases where emphasis is on ornamentation rather than legibility.
  - Fantasy fonts** They are used for logos and other ornamentation purposes where legibility is secondary.
  - Monospace fonts** They are used in code listings and other approximating terminal output.

## Font Characteristics

- Fonts are mapped according to a system similar to ruled paper.
  - The line that the characters or symbols sit on is called the **baseline**.
  - The distance between the baseline and the top of the highest characters (usually capital letters and lowercase letters such as l, f or t ) is known as the **ascension**.
  - The distance between the baseline and the lowest point of characters that dip below it (such as p, g or q) is known as the **descension**.



## Defining a Font Family

- The font-family property defines the font or fonts should be used for elements in the document.
- More than one font family names is defined with a generic family name for versatility.

### Introduction

This chapter introduces what is expected from a final year project and highlights its objectives. It also describes on how to select a project and a supervisor. A brief summary of the respect of the chapters is also given.

# CSS font

- p{font-family:"Times New Roman", Times, serif;}
- p.normal {font-style:normal;}
- p.italic {font-style:italic;}
- p.oblique {font-style:oblique;}
- h1 {font-size:40px;}
- h2 {font-size:30px;}
- p {font-size:14px;}
- h1 {font-size:2.5em;} /\* 40px/16=2.5em \*/
- h2 {font-size:1.875em;} /\* 30px/16=1.875em \*/
- p {font-size:0.875em;} /\* 14px/16=0.875em \*/
- H1{font-weight:normal;}
- H1{font-weight: bold ;}
- H1{font-weight:lighter;}

## Defining a Font Family

```

<head>
<title>Font Family</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<style type="text/css">
<!--
body (font-family: "Courier New", Courier, mono;)
h3 {
 font-family: Verdana, Arial, Helvetica, sans-serif;
 font-weight: bold;
}
-->
</style>
</head>

<body>
<h3>Introduction</h3>
<p>This chapter introduces what is expected from a final year project and highlights its objectives. It also describes on how to select a project and a supervisor. A brief summary of the respect of the chapters is also given.</p>
</body>

```

# Some Other Font Properties

- Font sizing
  - Two properties can be used to control font sizing: **font-size** and **font-size-adjust**. Both properties can adjust a font absolutely or relative to the current font size.
- Font styling
  - Four properties can be used to affect font styling: **font-style**, **font-variant**, **font-weight**, and **font-stretch**.
- Line spacing
  - The **line-height** property controls the line height of text, i.e. the distance between the baseline of two vertically stacked lines of text.

## Example -Code

```

<head>
<title>Font Properties</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<style type="text/css">
<!--
p.heading { font-size: x-large; font-weight: bold; }
p.italic { font-style: italic; font-variant: small-caps; }
p.spaced { line-height: 200%; }
-->
</style>
</head>

<body>
<p class="heading">Introduction</p>
<p class="italic">This chapter introduces what is expected from a final year project and highlights its objectives.</p>
<p class="spaced">The individual project is by far the most important single piece of work in the BIT degree programme. It provides the opportunity for a candidate to demonstrate independence and originality</p>
</body>

```

## Example - View

### Introduction

*THIS CHAPTER INTRODUCES WHAT IS EXPECTED FROM A FINAL YEAR PROJECT AND HIGHLIGHTS ITS OBJECTIVES.*

The individual project is by far the most important single piece of work in the BIT degree programme. It provides the opportunity for a candidate to demonstrate independence and originality

## Colors

- CSS options for defining colors can entirely replace the color attributes in plain HTML. In addition CSS has the following advantages:
  - In plain HTML, when you wanted to create an area with a specified color, you were forced to create a table. However, with CSS, you can define an area to have a specific color without that area being part of a table.
  - In plain HTML, specially when working with tables, you had to specify font attributes and colors etc. for each and every table cell. However, with CSS you can simply refer to a certain class.

## Specifying a Color

- The color value can be expressed using one of three methods:
  - Color keywords
    - blue, black, green, and so on
  - Color hexadecimal values
    - values specified in the form #rrggbba
      - E.g. #FF0000 for red, #000000 for black
  - Color decimal or percentage values
    - Values specified using the rgb() function. The value can be an integer between 0 and 255 or a percentage.
      - E.g. rgb(100, 0, 100), rgb(50%, 0, 50%)

## Specifying a Color

- Most elements in a HTML document have two color properties: a foreground color and a background color.
  - Setting the foreground color for contents  
**color: <color\_value>;**
  - Setting the background color for an area  
**background-color: <color\_value>;**
  - Setting a background image to fill out an area  
**background-image: url("<url\_to\_image>");**

## Example- Code

```
<head>
<title>Colors</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<style type="text/css">
<!--
.redtext { color: #FF0000; }
.filled { background-color: #0000FF; color: #00FF00; }
-->
</style>
</head>

<body>
<p class="redtext">Although the project is done for a client a candidate should remember that the purpose of the project is to fulfil an examination requirement of the BIT degree programme. </p>
<p class="filled">It is the responsibility of the candidate to identify a suitable project. The project should comprise a substantial amount of individual work to satisfy the PEB that the project objectives have been met as well as the time spent on the project is justified. </p>
</body>
```

## CSS Background image -Example

- Body { background-image:url('paper.gif');}
- Body { background-image:url('paper.gif'); background-repeat:repeat-x; }
- Body { background-image:url('paper.gif'); background-repeat:no-repeat; background-position:right top; }

## CSS Background color-Example

- Body { background-color:#b0c4de;}
- h1 { background-color:red;}
- div { background-color:rgb(255,255,0);}

## Text color- Example

- body {color:blue;}
- h1 {color:#00ff00;}
- h2 {color:rgb(255,0,0);}

## Text alignment- Example

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

```
p.date {text-align:right;}
```

```
p.main {text-align:justify;}
```

Text decoration -Example

- ```
h1 {text-decoration:overline;}
```
- ```
h2 {text-decoration:line-through;}
```
- ```
h3 {text-decoration:underline;}
```
- ```
h4 {text-decoration:blink;}
```
- ```
a{text-decoration:none;}
```

## Text Transformation

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

Text Indentation

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

## CSS Lists -Example

```

• ul.a {list-style-type: circle;}
• ul.b {list-style-type: square;}
• ol.c {list-style-type: upper-roman;}
• ol.d {list-style-type: lower-alpha;}
• ul
{
list-style-image: url('sqpurple.gif');
}

```

## Values for Unordered Lists

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

## Values for Ordered Lists

Value	Description
armenian	The marker is traditional Armenian numbering
decimal	The marker is a number
decimal-leading-zero	The marker is a number padded by initial zeros (01, 02, 03, etc.)
georgian	The marker is traditional Georgian numbering (an, ban, gan, etc.)
lower-alpha	The marker is lower-alpha (a, b, c, d, e, etc.)
lower-greek	The marker is lower-greek (alpha, beta, gamma, etc.)
lower-latin	The marker is lower-latin (a, b, c, d, e, etc.)
lower-roman	The marker is lower-roman (i, ii, iii, iv, v, etc.)
upper-alpha	The marker is upper-alpha (A, B, C, D, E, etc.)
upper-latin	The marker is upper-latin (A, B, C, D, E, etc.)
upper-roman	The marker is upper-roman (I, II, III, IV, V, etc.)

## Tables

### CSS properties available for table attributes:

Purpose	Table Attribute	CSS Property(ies)
Borders	<b>border</b>	<b>border</b> properties
Spacing inside cell	<b>cellpadding</b>	<b>padding</b> properties
Spacing between cells	<b>cellspacing</b>	<b>border-spacing</b> properties
Width of a table	<b>width</b>	<b>width</b> and <b>table-layout</b> properties
Table framing	<b>frame</b>	<b>border</b> properties
Alignment	<b>align, valign</b>	<b>text-align, vertical-alignment</b>

## Table-Examples

- ```
table, th, td
{
border: 1px solid black;
}
```
- ```
td
{
padding:15px;
}
```
- ```
td
{
text-align:right;
}
```
- ```
td
{
height:50px;
vertical-align:bottom;
}
```
- ```
th
{
background-color:green;
color:white;
}
```

CSS Border- Example

- ```
p.one
{
border-style:solid;
border-width:5px;
}
```
- ```
p.two
{
border-style:solid;
border-width:medium;
}
```
- ```
p.one
{
border-style:solid;
border-color:red;
}
```
- ```
p
{
border-top-style:dotted;
border-right-style:solid;
border-bottom-style:dotted;
border-left-style:solid;
}
```

CSS Margin- Examples

- ```
margin-top:100px;
margin-bottom:100px;
margin-right:50px;
margin-left:50px;
```

## CSS Margin- Examples

- The margin property can have from one to four values..
- **margin:25px 50px 75px 100px;**
  - top margin is 25px
  - right margin is 50px
  - bottom margin is 75px
  - left margin is 100px
- **margin:25px 50px 75px;**
  - top margin is 25px
  - right and left margins are 50px
  - bottom margin is 75px
- **margin:25px;**
  - all four margins are 25px

## CSS Padding - Examples

- padding-top:25px;
- padding-bottom:25px;
- padding-right:50px;
- padding-left:50px;

## CSS Padding - Examples

- The padding property can have from one to four values.
- **padding:25px 50px 75px 100px;**
  - top padding is 25px
  - right padding is 50px
  - bottom padding is 75px
  - left padding is 100px
- **padding:25px 50px 75px;**
  - top padding is 25px
- right and left paddings are 50px
- bottom padding is 75px
- **padding:25px 50px;**
  - top and bottom paddings are 25px
  - right and left paddings are 50px
- **padding:25px;**
  - all four paddings are 25px

## Subtopics



### HNDIT1022 – Web Design

## Week 6: Understanding the CSS Box Model and Positioning

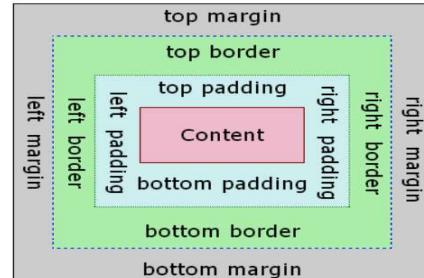
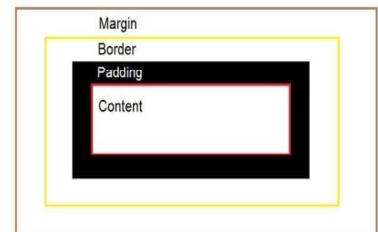
- Conceptualize CSS box model
- How to position elements
- Use CSS to do more with lists, text and navigations

## What is 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.

The image below illustrates the box model:



## Contd..

- The CSS Box Model defines the layout of HTML block elements such as divisions (<div>s) and paragraphs (<p>s). It does not apply to inline elements (<span>s, anchors (<a>s), and a bunch of others).
- Every block element is laid out (by the browser) as a rectangular box with four parts (boxes). The innermost part (box) is the content (or content box) which is surrounded by padding (or padding box). The border (or border box) encloses the padding; the outermost part is the margin (margin box).
- The padding is used to specify the amount of spacing surrounding the content within the border box, while the margin is used to specify the spacing separating the element from its surroundings.
- For screen display, padding and margin are normally specified in pixels.



## Contd..

Explanation of the different parts:

**Content** - The content of the box, where text and images appear

**Padding** - Clears an area around the content. The padding is transparent

**Border** - A border that goes around the padding and content

**Margin** - Clears an area outside the border. The margin is transparent

## Width and Height of an Element

When you set the width and height properties of an element with CSS, you just set the width and height of the **content area**.

To calculate the full size of an element, you must also add padding, borders and margins.

## Contd..

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

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

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

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

*By default (if no width is set), a block element takes up the whole width of the browser window and is resized when the browser window is resized. The element's height (if no height is set) adjusts to accommodate its content (the user may have to scroll).*



## Contd..

Example :

This <div> element will have a total width of 350px:

```
div {
 width: 300px;
 padding: 20px;
 border: 5px solid gray;
 margin: 0;
}
```

Here is the calculation:

300px (width)  
+ 40px (left + right padding)  
+ 10px (left + right border)  
+ 0px (left + right margin)  
**= 350px**

## Exercise 1

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 background-color: grey;
 width: 300px;
 border: 5px solid red;
 padding: 20px;
 margin: 10px;
}
</style>
</head>
```

```
<body>
<h2>Example on CSS Box
Model</h2>
<p>The CSS box model</p>
<div>This text is the content of
the box..</div>
</body>
</html>
```

# Output

## Example on CSS Box Model

The CSS box model

This text is the content of the box..

## Exercise 2

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
 margin: 25px;
 padding: 35px;
 width: 200px;
 background-color:
lightseagreen;
 border: 20px dashed
indianred;
}
</style>
</head>
```

```
<body>
<div>
Content box

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



## CSS Layout - The position Property

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

There are five different position values:

- static
- relative
- fixed
- absolute
- sticky

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

## Position : static

HTML elements are positioned static by default.

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

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



## Exercise 3

```
<!DOCTYPE html>
<html>
<head>
<style>
div.static {
 position: static;
 border: 8px solid #08a9bf;
}
</style>
</head>
```

### Static Position

An element with position.....

This div element has position: static;

```
<body>
<h2>Static Position</h2>
<p>An element with
position.....</p>
<div class="static">
This div element has position:
static;
</div>
</body>
</html>
```

## Position : relative

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

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



## Exercise 4

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
 position: relative;
 top: 30px;
 right: 50px;
 border: 5px solid #eb14ca;
}
</style>
</head>
```

### Relative Position

An element with position.....

This div element has position: relative;

```
<body>
<h2>Relative Position</h2>
<p>An element with
position.....</p>
<div class="relative">
This div element has position:
relative;
</div>
</body>
</html>
```

## Position : fixed

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

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



## Exercise 5

```
<!DOCTYPE html>
<html>
<head>
<style>
div.fixed {
 position: fixed;
 bottom: 0;
 right: 0;
 width: 300px;
 border: 3px solid #73AD21;
}
</style>
</head>
```

```
<body>
<h2>fixed position</h2>
<p>An element with
position...</p>
<div class="fixed">
This div element has position:
fixed;
</div></body>
</html>
```

## Contd...

### fixed position

An element with position...

This div element has position: fixed;



## Position : absolute

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

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

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

## Exercise 6

```
<!DOCTYPE html>
<html>
<head>
<style>
div.relative {
 position: relative;
 width: 500px;
 height: 200px;
 border: 3px solid #0a27b3;
}
div.absolute {
 position: absolute;
 top: 80px;
 right: 0;
 width: 250px;
 height: 100px;
 border: 3px dotted #f83e90;
}
</style>
</head>
```

```
<body>
<h2>absolute position</h2>
<p>An element with ...</p>
<div class="relative">This div element
has position: relative;
<div class="absolute">This div
element has position: absolute;</div>
</div>
```

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

Contd...

## absolute position

An element with ...

This div element has position: relative;

This div element has position: absolute;

## Position : sticky

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

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

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

## Exercise 7

```
<!DOCTYPE html>
<html>
<head>
<style>
div.sticky {
 position: -webkit-sticky;
 position: sticky;
 top: 0;
 padding: 5px;
 background-color: #4bdbc7;
 border: 4px double #f10d7f;
}
</style>
</head>

<body>
<p>Try to scroll inside this frame to understand how sticky positioning works.</p>
<div class="sticky">I am sticky element!</div>
<div style="padding-bottom:2000px">
 <p></p>
 <p>Scroll back up to remove the stickyness.</p>
 <p>Some text to enable scrolling..</p>
</div>
</body>
</html>
```

Contd...

Try to **scroll** inside this frame to understand how sticky positioning works.

I am sticky element!

Scroll back up to remove the stickyness.

Some text to enable scrolling..

## Subtopics



### HNDIT1022 – Web Design

- What are dynamic web sites
- Introduction to the photoshop  
Resizing &cropping image Photo retouching

## Week 7: understanding dynamic web sites and introduction to photoshop

### What is a STATIC WEBSITE ?

- Any Site that has fixed content usually written in html code.
- Every page will have the Code written separately, just as you see it on the web.
- Every page has to be saved separately on the server.
- Changes have to be made manually every time, and you need coding knowledge to make any and all changes.

### Advantages & Disadvantages of STATIC Websites

#### Advantages

Easy to develop

Cheap to develop

Cheap to host

#### Disadvantages (Cons)

Requires web development expertise to update site

Changes and updates are very time consuming

Site not as useful for the user

Content can get stagnant

OUT OF DATE!!

## DYNAMIC WEBSITE

- A site whose construction is controlled by an application server processed by server-side scripts. Pages of the website are not coded and saved separately.
- The design/template ( look and feel ) is saved separately.
- Corresponding content are saved separately.
- The pages are dynamically populated every time.

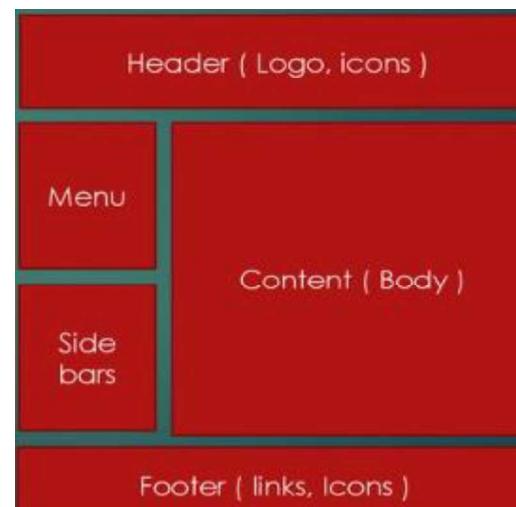
## Website Structure



## A DYNAMIC Website Structure

A request to view a page will Dynamically populate the different sections of the site according to a template file

Header  
Menu  
Content  
Side bars  
Footer



## Advantages & Disadvantages of DYNAMIC websites

### Advantages

- Much more functional website
- Much easier to update
- Much Easier to add new content/pages
- New content brings people back to the site and helps in the search engines
- Can work as a system to allow staff or users to collaborate
- It provides the attractive look.
- It is useful for E-commerce sites.
- It provide a clear and the well-organized look.

### Disadvantages

- More expensive to develop
- Slower to develop
- Hosting costs a little more ( This might not be true now, because of great shared web hosting plans these days)

# Static Website Vs Dynamic Website

## Static Website

- It is used the HTML code to develop a website.
- Flexibility is the important in the static website.
- The same content webpage load in the website .

## Dynamic Website

- It is used the sever side Scripting language .
- CMS is the important in the dynamic website.
- The content is generate quickly.
- **CMS-content management system**



## HNDIT1022 – Web Design

### Week 8, 9 & 10: understanding JavaScript Programming

#### Subtopics

- What is scripting
- How scripting and programming are different
- How to include java script commands into web page Functions and build in functions
- How objects use in java scripts Responding to events
- Conditional statements and looping
- Syntax rules avoid errors
- Adding comments

Contd...

- Name and declare variables
- Assign variable
- Use variables and literal expressions
- Create arrays of numbers and strings
- How event handlers related to objects
- Create new window with java script
- How to build frameset
- Link between frames and windows
- Use JavaScript to work with frames

#### What is scripting?

- As the name suggest, it's all about giving the script to perform some certain task.
- Scripting languages are basically the subcategory of programming languages which is used to give guidance to another program or we can say to control another program, so it also involves instructions.
- It basically connects one language to one another languages and doesn't work standalone.
- Scripting languages need to be interpreted (Scanning the code line by line, not like compiler in one go) instead of compiled.

## Contd...

- When it comes to making a website or application coding involves basically three types of languages i.e the programming language, Scripting Language and Markup Language.



## Programming VS Scripting

A programming language is a <b>computer language</b> that is used to communicate with computers using a set of instructions.	A scripting language is a type of <b>programming language</b> designed for a runtime system to automate the execution of tasks.
It is compiled language or <b>compiler-based language</b> .	It is interpreted language or <b>interpreter-based language</b>
It is used to develop an <b>application or software from scratch</b> .	It is used to <b>combine existing components and automate a specific task</b> .
It runs or executes independently and does not depend on the parent (exterior) program.	It runs or executes inside another program.
It uses a compiler to convert source code into machine code.	It uses an interpreter to convert source code into machine code.
As it uses a compiler, hence the complete program is converted into machine code in one shot.	As it uses an interpreter, hence the program is converted into machine code line by line.
These languages are required to be compiled.	There is no need for compilation.
It is comparatively difficult to write code in a programming language, and it requires numerous lines of code for each task.	It is comparatively easy to write code in the scripting language, and it requires few lines of code for each task.
The development time in programming languages is high as more lines are required.	The development time in a scripting language as a smaller number of lines are required.
There is the high maintenance cost.	There is less maintenance cost.
All programming languages are not scripting languages	All scripting languages are programming languages
It generates a .exe file.	It does not create a .exe file.
Usually, programming languages do not support or provide very little support for user interface designing, data types, and graphic designing.	Scripting languages provide great support to user interface design, data types, and graphic design.
Some popular examples are C, C++, Java, Scala, COBOL, etc.	Some popular examples are Perl, Python, JavaScript, etc.

## Characteristics of Client-side Scripting

Client-side scripts can modify the pages at runtime, and therefore, they also falls under the heading of DHTML (dynamic HTML).

Client-side scripts have greater access to the information and functions available on the user's computer, whereas for server-side scripts its for the server.

Client-side scripts require that the user's web browser understand the scripting language in which they are written.

## Some Uses of Client-side Scripting

### Form verification

Client-side scripts can parse form data prior to the data being submitted to a handler on the server, ensuring that there are no obvious errors (missing or improperly formatted data).

### Document animation and automation

Accessing element data and properties via the DOM, client-side scripts can affect changes in elements' content, appearance, size, position, and so forth.

### Basic document intelligence

Client-side scripts can embed a base level of intelligence to the document by linking elements to scripts via events (mouse clicks, key presses, and so on).

# Client-side Scripting Languages

- JavaScript
- JScript
- VBScript
- ActionScript
- AJAX

## JAVA SCRIPT Introduction

**JavaScript** is a object-based scripting language and it is light weighted.

It is first implemented by Netscape (with help from Sun Microsystems).

JavaScript was created by **Brendan Eich** at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side).

It is not compiled but translated. JavaScript Translator is responsible to translate the JavaScript code which is embedded in browser.

9

## Where it is used?

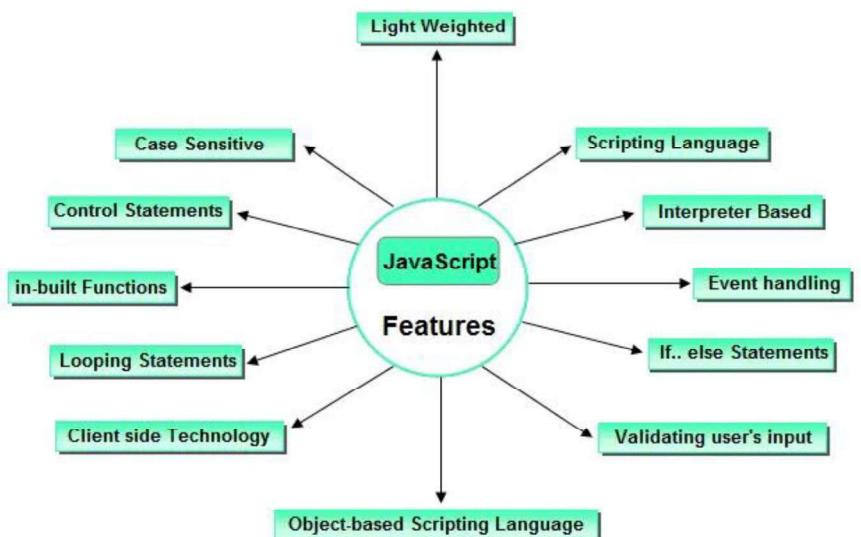
It is used to create interactive websites.

It is mainly used for:

- Client-side validation
- Dynamic drop-down menus
- Displaying date and time
- Build small but complete client side programs .
- Displaying popup windows and dialog boxes (like alert dialog box, confirm dialog box and prompt dialog box)
- Displaying clocks etc.

## Features of JavaScript

JavaScript is a client side technology, it is mainly used for gives client side validation, but it have lot of features which are given below;



# Way of Using JavaScript

There are three places to put the JavaScript code.

Between the <body> </body> tag of html (Inline JavaScript)

Between the <head> </head> tag of html (Internal JavaScript)

In .js file (External JavaScript)

## How to Put a JavaScript Into an HTML Page?

```
<html>
<body>
<script type="text/javascript">
document.write("Hello World!")
</script>
</body>
</html>
```

- **Inline JavaScript**

- When java script was written within the html element using attributes related to events of the element then it is called as inline java script.

- **Example of Inline JavaScript**

- **Example How to use JavaScript**

```
<html> <form> <input type="button"
value="Click" onclick="alert('Button Clicked')"/>
</form> </html>
```

- Output:

Click

## Internal JavaScript

- When java script was written within the section using element then it is called as internal java script.

- **Example of Internal JavaScript**

```
<html>
<head>
<script>
function msg()
{ alert("Welcome in JavaScript"); }
</script>
</head>
<form>
<input type="button" value="Click" onclick="msg()"/>
</form>
</html>
```

- Output:

click

# External JavaScript

- Writing java script in a separate file with extension .js is called as external java script. For adding the reference of an external java script file to your html page, use tag with src attribute as follows

- Example**

```
<script type="text/javascript" src="filename.js"/>
```

- Create a file with name functions.js and write the following java script functions in it.

- message.js**

- Example**

```
function msg() { alert("Welcome in JavaScript"); }
```

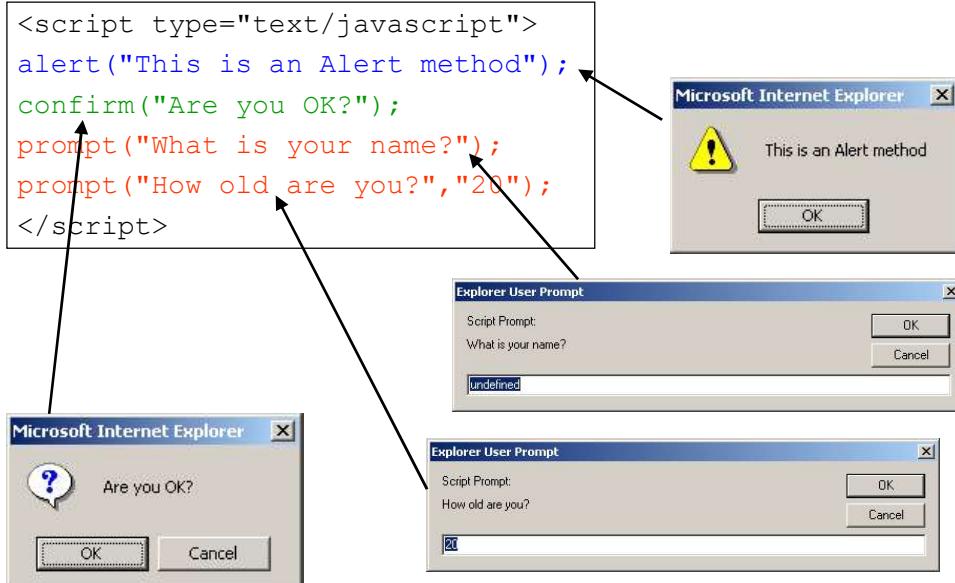
- Example**

```
<html>
<head>
<script type="text/javascript" src="message.js">
</script>
</head>
<body>
<form>
<input type="button" value="click" onclick="msg()">
</form>
</body>
</html>
```

- output:

click

## alert(), confirm(), and prompt()



## alert() and confirm()

```
alert("Text to be displayed");
```

- Display a message in a dialog box.
- The dialog box will block the browser.

```
var answer = confirm("Are you sure?");
```

- Display a message in a dialog box with two buttons: "OK" or "Cancel".
- confirm() returns **true** if the user click "OK". Otherwise it returns **false**.

## **prompt()**

```
prompt("What is your student id number?");
prompt("What is your name?", "No name");
```

- Display a message and allow the user to enter a value
- The second argument is the "default value" to be displayed in the input textfield.
- Without the default value, "undefined" is shown in the input textfield.
- If the user click the "OK" button, **prompt()** returns the value in the input textfield as a string.
- If the user click the "Cancel" button, **prompt()** returns null.

## **Identifier**

- Same as Java/C++ except that it allows an additional character – '\$'.
- Contains only 'A' – 'Z', 'a' – 'z', '0' – '9', '\_', '\$'
- First character cannot be a digit
- Case-sensitive
- Cannot be reserved words or keywords

## **Variable and Variable Declaration**

```
<head><script type="text/javascript">
 // We are in the default scope - the "window" object.
 x = 3; // same as "window.x = 3"
 var y = 4; // same as "y = 4" or "window.y = 4"

 { // Introduce a block to creat a local scope
 x = 0; // Same as "window.x = 0"
 var y = 1; // This is a local variable y
 }

 alert("x=" + x + ", y=" + y); // Print x=0, y=4
</script></head>
```

- Local variable is declared using the keyword 'var'.
- Dynamic binding – a variable can hold any type of value
- If a variable is used without being declared, the variable is created automatically.
  - If you misspell a variable name, program will still run (but works incorrectly)

## **Data Types**

- **Primitive data types**
  - **Number**: integer & floating-point numbers
  - **Boolean**: true or false
  - **String**: a sequence of alphanumeric characters
- **Composite data types (or Complex data types)**
  - **Object**: a named collection of data
  - **Array**: a sequence of values (an array is actually a predefined object)
- **Special data types**
  - **Null**: the only value is "null" – to represent nothing.
  - **Undefined**: the only value is "undefined" – to represent the value of an uninitialized variable

## Strings

- A string variable can store a sequence of alphanumeric characters, spaces and special characters.
- Each character is represented using 16 bit
  - You can store Chinese characters in a string.
- A string can be enclosed by a pair of single quotes (' ) or double quote ( " ).
- Use escaped character sequence to represent special character (e.g.: \" , \n , \t)

## typeof operator

```
var x = "hello", y;
alert("Variable x value is " + typeof x);
alert("Variable y value is " + typeof y);
alert("Variable x value is " + typeof z);
```

- An unary operator that tells the type of its operand.
  - Returns a string which can be "number", "string", "boolean", "object", "function", "undefined", and "null"
  - An array is internally represented as an object.

## Operators

- Arithmetic operators
  - +, -, \*, /, %
- Post/pre increment/decrement
  - ++, --
- Comparison operators
  - ==, !=, >, >=, <, <=
  - ===, !== (Strictly equals and strictly not equals)
    - i.e., Type and value of operand must match / must not match

## == VS ===

```
// Type conversion is performed before comparison
var v1 = ("5" == 5); // true

// No implicit type conversion.
// True if only if both types and values are equal
var v2 = ("5" === 5); // false

var v3 = (5 === 5.0); // true

var v4 = (true == 1); // true (true is converted to 1)

var v5 = (true == 2); // false (true is converted to 1)

var v6 = (true == "1") // true
```

# Logical Operators

- **!** – Logical NOT
- **&&** – Logical AND
  - OP1 && OP2
  - If OP1 is true, expression evaluates to the value of OP2.  
Otherwise the expression evaluates to the value of OP1.
  - Results may not be a boolean value.
- **||** – Logical OR
  - OP1 || OP2
  - If OP1 is true, expression evaluates to the value of OP1.  
Otherwise the expression evaluates to the value of OP2.

```
var tmp1 = null && 1000; // tmp1 is null
var tmp2 = 1000 && 500; // tmp2 is 500
var tmp3 = false || 500; // tmp3 is 500
var tmp4 = "" || null; // tmp4 is null
var tmp5 = 1000 || false; // tmp5 is 1000

// If foo is null, undefined, false, zero, NaN,
// or an empty string, then set foo to 100.
foo = foo || 100;
```

## Operators Contd ...

- String concatenation operator
  - **+**
  - If one of the operand is a string, the other operand is automatically converted to its equivalent string value.
- Assignment operators
  - **=, +=, -=, \*=, /=, %=**
- Bitwise operators
  - **&, |, ^, >>, <<, >>>**

## Control Structures

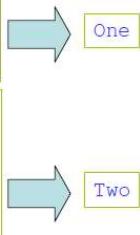
- JavaScript supports the following control structures,
  - If...Else
  - Switch
  - Do...While
  - While
  - For and For...In

## If...Else

- The if and if...else constructs execute a block of code depending on the evaluation (true or false) of an expression.
- E.g.

```
var i = 1;
if (i == 1) {
 document.write("One");
}

var i = 2;
if (i == 1) {
 document.write("One");
} else if (i == 2) {
 document.write("Two");
} else {
 document.write("Other");
}
```



33

## Conditional Operator

- JavaScript also contains a conditional operator that assigns a value to a variable based on some condition.
- E.g.

```
var x = 2;
y = (x==1)? "One": "Other";
document.write(y);
```

Similar to:

```
var x = 1;
if (x==1)
 y = "One";
else
 y = "Other";
document.write(y);
```



34

## Switch

- The switch construct executes specific block(s) of code based on the value of a particular expression.
  - Note: Use break statement to prevent the code from running into the next case automatically.
- E.g.

```
switch (day) {
 case "Mon": {
 document.write("Mon");
 break;
 }
 case "Tue": {
 document.write("Tue");
 break;
 }
 default: {
 document.write("Other");
 }
}
```



## Do...While

- The do...while loop executes one or more lines of code as long as a specified condition remains true.
  - E.g.
- ```
var i = 0;
do {
    document.write(i);
    i++;
} while (i < 10);
```
- A diagram illustrating the execution flow of the do...while loop code. It shows a single execution path that leads to the output '0123456789'.
- 36
- Note: Due to the expression being evaluated at the end of the structure, statement(s) in the do...while loop are executed at least once, even if the condition is false.

While

- The while loop executes one or more lines of code while specified expression remains true.
- E.g.

```
var i = 0;
while (i < 10) {
    document.write(i);
    i++;
}
```



0123456789

- Note: Because the expression is evaluated at the beginning of the loop, the statement(s) will not be executed if the expression is false at the beginning.

37

For and For...In

- The for loop executes statement(s) a specified number of times governed by two expressions and a condition.
- E.g.

```
for (i=1; i<10; i++) {
    document.write(i);
}
```



123456789

- The for...in loop executes statement(s) while assigning a variable to the properties of an object or elements of an array.
- E.g.

```
days = new Array("M", "T", "W");
for (i in days) {
    document.write(i);
}
```

38

Break and Continue

- The break command will break the loop and continue executing the code that follows after the loop (if any).
- E.g.

```
for (i=1; i<10; i++) {
    if (i==3) break;
    document.write(i);
}
```



12

- The continue command will break the current loop and continue with the next value.
- E.g.

```
for (i=1; i<10; i++) {
    if (i==3) continue;
    document.write(i);
}
```



12456789

19

Labels

- Labels are used to mark statements for reference by other statements in other sections of a script.
- Labels can solve the problem of breaking from a loop that is nested inside another loop.
- E.g.

```
top:
for (i=0; i<5; i++) {
    for (j=0; j<5; j++) {
        if (j==3) break top;
        document.write(j);
    }
}
```



012



Without the label

012012012012012012

40

Conditional Statements Examples

```
<script>
x=3
if(x<0)
{
alert ("negative")
}
else
{
alert ("positive")
}
</script>
```

Conditional Statements Examples Contd...

```
<script>
c=confirm("Do you read books?")
if(c)
{
alert ("Good. Keep it up")
}
else
{
alert ("Shame on you")
}
</script>
```

Conditional Statements Examples Contd ...

```
<script>
p=prompt("Enter your lucky number?", " ")
if(p=="08")
{
alert("True")
}
else
{
alert("False")
}
</script>
```

Functions and Procedures

- A function in a JavaScript is a set of statements that performs a specific task (a procedure).
- A function will be executed by an event or by a call to that function.
- Functions can be defined both in the `<head>` and in the `<body>` section of a document. However, to assure that the function is read/loaded by the browser before it is called, it could be wise to put it in the `<head>` section.
- A function can be called from anywhere within the page (or even from other pages if the function is embedded in an external .js file).

Contd...

```
function name() {  
    statement ;  
    statement ;  
    ...  
    statement ;  
}
```

JS

```
function myFunction() {  
    alert("Hello!");  
    alert("How are you?");  
}
```

JS

- the above could be the contents of example.js linked to our HTML page
- statements placed into functions can be evaluated in response to user events

CS380

45

Built-In Functions

- **eval(expr)**

- evaluates an expression or statement

- eval("3 + 4"); // Returns 7 (Number)
 - eval("alert('Hello')"); // Calls the function alert('Hello')

- **isFinite(x)**

- Determines if a number is finite

- **isNaN(x)**

- Determines whether a value is “Not a Number”

Built-In Functions

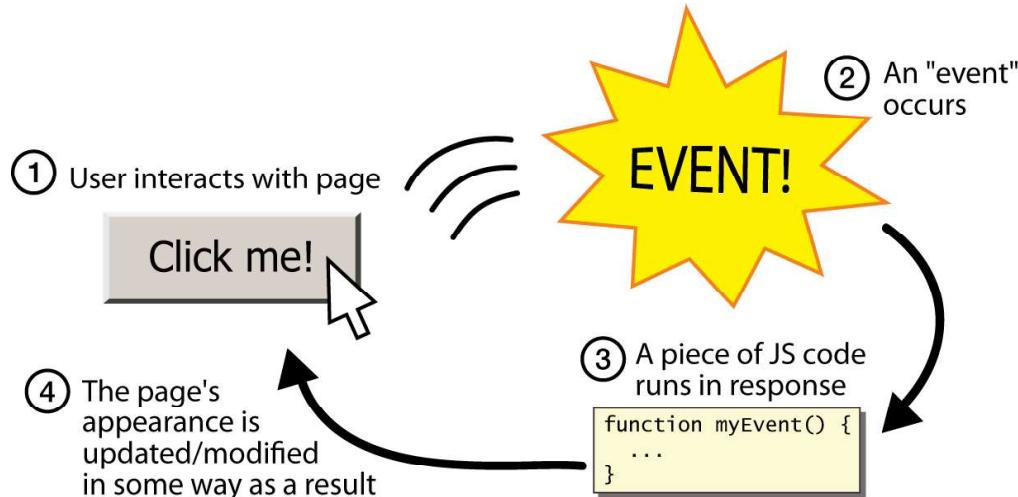
- **parseInt(s)**
- **parseInt(s, radix)**
 - Converts string literals to integers
 - Parses up to any character that is not part of a valid integer
 - parseInt("3 chances") // returns 3
 - parseInt(" 5 alive") // returns 5
 - parseInt("How are you") // returns NaN
 - parseInt("17", 8) // returns 15

- **parseFloat(s)**
 - Finds a floating-point value at the beginning of a string.
 - parseFloat("3e-1 xyz") // returns 0.3
 - parseFloat("13.5 abc") // returns 13.5

Events

- An event occurs as a result of some activity
 - e.g.:
 - A user clicks on a link in a page
 - Page finished loaded
 - Mouse cursor enter an area
 - A preset amount of time elapses
 - A form is being submitted

Event-driven programming



CS380

49

Event Handlers

- Event Handler – a segment of codes (usually a function) to be executed when an event occurs
- We can specify event handlers as attributes in the HTML tags.
- The attribute names typically take the form "**onXXX**" where **XXX** is the event name.
 - e.g.:
`Other Website`

Event Handlers

| Event Handlers | Triggered when |
|----------------|--|
| onChange | The value of the text field, textarea, or a drop down list is modified |
| onClick | A link, an image or a form element is clicked once |
| onDoubleClick | The element is double-clicked |
| onMouseDown | The user presses the mouse button |
| onLoad | A document or an image is loaded |
| onSubmit | A user submits a form |
| onReset | The form is reset |
| onUnLoad | The user closes a document or a frame |
| onResize | A form is resized by the user |

For a complete list, see http://www.w3schools.com/html/dom_obj_event.asp

onClick Event Handler Example

```
<html>  
<head>  
<title>onClick Event Handler Example</title>  
<script type="text/javascript">  
function warnUser() {  
    return confirm("Are you a student?");  
}  
</script>  
</head>  
<body>  
<a href="ref.html" onClick="return warnUser()">  
<!--<br/>If onClick event handler returns false, the link  
is not followed.  
-->  
Students access only</a>  
</body>  
</html>
```

onLoad Event Handler Example

```
<html><head>
<title>onLoad and onUnload Event Handler Example</title>
</head>
<body>
  onLoad="alert('Welcome to this page')"
  onUnload="alert('Thanks for visiting this page')"
>
Load and UnLoad event test.
</body>
</html>
```

onMouseOver & onMouseOut Event Handler

```
<html>
<head>
<title>onMouseOver / onMouseOut Event Handler Demo</title>
</head>
<body>
<a href="http://www.cuhk.edu.hk"
  onMouseOver="window.status='CUHK Home'; return true;"
  onMouseOut="status=''"
>CUHK</a>
</body>
</html>
```

- When the mouse cursor is over the link, the browser displays the text "CUHK Home" instead of the URL.
- The "return true;" of onMouseOver forces browser not to display the URL.
- window.status and window.defaultStatus are disabled in Firefox.

onSubmit Event Handler Example

```
<html><head>
<title>onSubmit Event Handler Example</title>
<script type="text/javascript">
  function validate() {
    // If everything is ok, return true
    // Otherwise return false
  }
</script>
</head>
<body>
<form action="MessageBoard" method="POST"
  onSubmit="return validate();"
>
...
</form></body></html>
```

- If onSubmit event handler returns false, data is not submitted.
- If onReset event handler returns false, form is not reset

Build-In JavaScript Objects

Object	Description
Array	Creates new array objects
Boolean	Creates new Boolean objects
Date	Retrieves and manipulates dates and times
Error	Returns run-time error information
Function	Creates new function objects
Math	Contains methods and properties for performing mathematical calculations
Number	Contains methods and properties for manipulating numbers.
String	Contains methods and properties for manipulating text strings

- See online references for complete list of available methods in these objects:
<http://javascript-reference.info/>

String Object (Some useful methods)

- `length`
 - A string property that tells the number of character in the string
- `charAt(idx)`
 - Returns the character at location "idx"
- `toUpperCase(), toLowerCase()`
 - Returns the same string with all uppercase/lowercase letters
- `substring(beginIdx)`
 - Returns a substring started at location "beginIdx"
- `substring(beginIdx, endIdx)`
 - Returns a substring started at "beginIdx" until "endIdx" (but not including "endIdx")
- `indexOf(str)`
 - Returns the position where "str" first occurs in the string

How to use “onError” event handler?

```
<html>
<head>
<title>onerror event handler example</title>
<script type="text/javascript">
function errorHandler() {
  alert("Error Ourred!");
}
// JavaScript is casesensitive
// Don't write onerror!
window.onerror = errorHandler;
</script>
</head>
<body>
<script type="text/javascript">
  document.write("Hello there;
</script>
</body>
</html>
```



Error and Exception Handling in JavaScript

- Javascript makes no distinction between Error and Exception (Unlike Java)
- **Handling Exceptions**
 - The `onError` event handler
 - A method associated with the window object.
 - It is called whenever an exception occurs
 - The `try ... catch ... finally` block
 - Similar to Java try ... catch ... finally block
 - For handling exceptions in a code segment
 - Use `throw` statement to throw an exception
 - You can throw value of any type
 - The `Error` object
 - Default object for representing an exception
 - Each Error object has a `name` and `message` properties

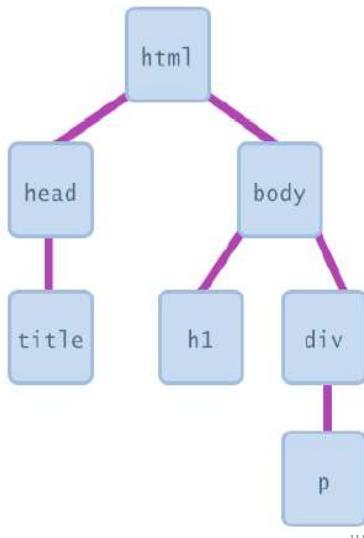
Document Object Model (DOM)

- When a web page is loaded, the browser creates a **Document Object Model** of the page.
- Representation of the current web page as a tree of Javascript objects
- allows you to view/modify page elements in script code after page has loaded
- client side = highly responsive interactions
- browser-independent
- allows progressive enhancement.

What is the DOM?

Contd...

- most JS code manipulates elements on an HTML page
- we can examine elements' state
 - e.g. see whether a box is checked
- we can change state
 - e.g. insert some new text into a div
- we can change styles
 - e.g. make a paragraph red



- The DOM is a W3C (World Wide Web Consortium) standard.
- The DOM defines a standard for accessing documents:
- "*The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document.*"
- The W3C DOM standard is separated into 3 different parts:
 - Core DOM - standard model for all document types
 - XML DOM - standard model for XML documents
 - HTML DOM - standard model for HTML documents

What is the HTML DOM?

- The HTML DOM is a standard **object** model and **programming interface** for HTML. It defines:
- The HTML elements as **objects**
- The **properties** of all HTML elements
- The **methods** to access all HTML elements
- The **events** for all HTML elements

JavaScript - HTML DOM Methods

- HTML DOM methods are **actions** you can perform (on HTML Elements).
- HTML DOM properties are **values** (of HTML Elements) that you can set or change.

The DOM Programming Interface

- The HTML DOM can be accessed with JavaScript (and with other programming languages).
- In the DOM, all HTML elements are defined as **objects**.
- The programming interface is the properties and methods of each object.
- A **property** is a value that you can get or set (like changing the content of an HTML element).
- A **method** is an action you can do (like add or deleting an HTML element).

Accessing elements: document.getElementById

```
var name = document.getElementById("id");
```

JS

```
<button onclick="changeText();">Click me!</button>
<span id="output">replace me</span>
<input id="textbox" type="text" />
```

HTML

```
function changeText() {
    var span = document.getElementById("output");
    var textBox = document.getElementById("textbox");

    textBox.style.color = "red";
}
```

JS

The getElementById Method

- The most common way to access an HTML element is to use the id of the element.
- In the example above the getElementById method used id="demo" to find the element.

The innerHTML Property

- The easiest way to get the content of an element is by using the **innerHTML** property.
- The innerHTML property is useful for getting or replacing the content of HTML elements.
- The innerHTML property can be used to get or change any HTML element, including <html> and <body>.

Contd...

- document.getElementById returns the DOM object for an element with a given id
- can change the text inside most elements by setting the innerHTML property
- can change the text in form controls by setting the value property

Example

- The following example changes the content (the innerHTML) of the <p> element with id="demo":

```
<html>
<body>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML =
"Hello World!";
</script>
</body>
</html>
```

The HTML DOM Document Object

- The document object represents your web page.
- If you want to access any element in an HTML page, you always start with accessing the document object.
- Below are some examples of how you can use the document object to access and manipulate HTML.

JavaScript Objects

- Real Life Objects, Properties, and Methods
 - In real life, a car is an **object**.
 - A car has **properties** like weight and color, and **methods** like start and stop:
-
- ObjectProperties**
1.car.name = Celerio 2.car.model = 2018
3.car.weight = 850kg 4.car.color = silky silver
 - Methods**
1.car.start() 2.car.drive()
3.car.brake() 4.car.stop()

- All cars have the same **properties**, but the property values differ from car to car.
- All cars have the same **methods**, but the methods are performed at different times.
- JavaScript Objects
- You have already learned that JavaScript variables are containers for data values.
- This code assigns a **simple value** (Celerio) to a **variable** named car:

```

<!DOCTYPE html>
<html>
<body>
<p>Creating a JavaScript Variable.</p>
<p id="demo"></p>
<script>
var car = "Celerio";
document.getElementById("demo").innerHTML =
car;
</script>
</body>
</html>

```

- Objects are variables too. But objects can contain many values.
- This code assigns **many values** (Celerio, 2018, silky silver) to a **variable** named car:

```

<!DOCTYPE html>
<html>
<body>
<p>Creating a JavaScript Object.</p>
<p id="demo"></p>
<script>
var car = {type:"Celerio", model:"2018", color:"silky silver"};
document.getElementById("demo").innerHTML = car.type;
</script>
</body>
</html>

```

- The values are written as **name:value** pairs (name and value separated by a colon).

Object Properties

- The name:values pairs (in JavaScript objects) are called **properties**.
- var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
- **Object Methods:**
- Methods are **actions** that can be performed on objects.
- Methods are stored in properties as **function definitions**.

Object Definition

- You define (and create) a JavaScript object with an object literal:

Example

```

var person = {firstName:"John",
lastName:"Doe", age:50, eyeColor:"blue"};

```

```

var person = {
  firstName:"John",
  lastName:"Doe",
  age:50,
  eyeColor:"blue"
};

```

Accessing Object Properties

- You can access object properties in two ways:
 - *objectName.propertyName*
- Or*
- *objectName["propertyName"]*
- Example1
 - *person.lastName;*
 - Example2
 - *person["lastName"];*

```
<!DOCTYPE html>
<html>
<body>
<p>
There are two different ways to access an object property:
</p>
<p>You can use person.property or person["property"].</p>

<p id="demo"></p>

<script>
var person = {
  firstName: "John",
  lastName : "Doe",
  id      : 5566
};
document.getElementById("demo").innerHTML =
person.firstName + " " + person.lastName;
</script>

</body>
</html>
```

Accessing Object Methods

- You access an object method with the following syntax:
 - *objectName.methodName()*
- Example
 - *name = person.fullName();*

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

<p>
There are two different ways to access an object property:
</p>
<p>You can use person.property or person["property"].</p>

<p id="demo"></p>

<script>
var person = {
  firstName: "John",
  lastName : "Doe",
  id      : 5566
};
document.getElementById("demo").innerHTML =
person["firstName"] + " " + person["lastName"];
</script>

</body>
</html>
```

```

<!DOCTYPE html>
<html>
<body>

<p>Creating and using an object method.</p>

<p>An object method is a function definition, stored as a property value.</p>

<p id="demo"></p>

<script>
var person = {
  firstName: "John",
  lastName : "Doe",
  id      : 5566,
  fullName : function() {
    return this.firstName + " " + this.lastName;
  }
};

document.getElementById("demo").innerHTML = person.fullName();
</script>
</body>
</html>

```

- **bgcolor and fgcolor**

These are used to set background and foreground(text) color to webpage. The methods accept either hexadecimal values or common names for colors.

Syntax: `document.bgcolor="#1f9de1"; document.fgcolor="silver";`

- **anchors**

The anchors property is an array of anchor names in the order in which they appear in the HTML Document. Anchors can be accessed like this:

- **Syntax:**

- `document.anchors[0];`

- `document.anchors[n-1];`

- **Links**

Another array holding all links in the order in which they were appeared on the Webpage

- **Forms**

Another array, this one contains all of the HTML forms. By combining this array with the individual form objects each form item can be accessed.

The Document Object

- A document is a web page that is being either displayed or created. The document has a number of properties that can be accessed by JavaScript programs and used to manipulate
 - the content of the page.
 - **write or writeln**
- Html pages can be created on the fly using JavaScript. This is done by using the write or writeln methods of the document object.
- Syntax:`document.write ("String"); document.writeln ("String");`
- In this document is object name and write () or writeln () are methods. Symbol period is used as connector between object name and method name. The difference between these two methods is carriage form feed character that is new line character automatically added into the document.
- Example: `document.write("<body>"); document.write("<h1> Hello </h1>");`

The Window Object

- The window object is used to create a new window and to control the properties of window.
- **Methods:**
 - `open("URL","name")` : This method opens a new window which contains the document specified by URL and the new window is identified by its name.
 - `close()`: this shutdowns the current window.
- **Properties:**
 - `toolbar = [1|0]` `location= [1|0]` `menubar = [1|0]` `scrollbars = [1|0]` `status = [1|0]` `resizable = [1|0]`
 - where as 1 means *on* and 0 means *off*
 - `height=pixels, width=pixels` : These properties can be used to set the window size.
 - The following code shows how to open a new window
 - `newWin = open("first.html","newWin","status=0,toolbar=0,width=100,height=100");`

- Window object supports three types of message boxes.
- Alert box Confirm box Prompt box
- **Alert box** is used to display warning/error messages to user. It displays a text string with *OK* button.
- Syntax: `window.Alert ("Message");`

- **Confirm Box** is useful when submitting form data. This displays a window containing message with two buttons: *OK* and *Cancel*. Selecting *Cancel* will abort the any pending action, while *OK* will let the action proceed.
- Syntax
- `window.confirm("String");`

The Form Object

- **Prompt box** used for accepting data from user through keyboard. This displays simple window that contains a prompt and a text field in which user can enter data. This method has two parameters: a text string to be used as a prompt and a string to use as the default value. If you don't want to display a default then simply use an empty string. Syntax
- `Variable=window.prompt("string","default value");`

- Two aspects of the form can be manipulated through JavaScript. First, most commonly and probably most usefully, the data that is entered onto your form can be checked at submission. Second you can actually build forms through JavaScript.
- Form object supports three events to validate the form
 $\text{onClick} = \text{"method()"}$
- This can be applied to all form elements. This event is triggered when the user clicks on the element.
 $\text{onSubmit} = \text{"method()"}$
- This event can only be triggered by form itself and occurs when a form is submitted.
 $\text{onReset} = \text{"method()"}$
- This event can only be triggered by form itself and occurs when a form is reset.

The browser Object

- The browser is JavaScript object that can be used to know the details of browser. Some of the properties of the browser object is as follows:

Property	Description
navigator.appCodeName	It returns the internal name for the browser. For major browsers it is Mozilla
navigator.appName	It returns the public name of the browser – navigator or Internet Explorer
navigator.appVersion	It returns the version number, platform on which the browser is running.
navigator.userAgent	The strings appCodeName and appVersion concatenated together
navigator.plugins	An array containing details of all installed plug-ins
Navigator.mimeTypes	An array of all supported MIME Types

The Math Object

- The *Math* object holds all mathematical functions and values. All the functions and attributes used in complex mathematics must be accessed via this object.
- Syntax:
- Math.methodname();**
- Math.value;**

Method	Description	Example
Math.abs(x)	Returns the absolute value	Math.abs(-20) is 20
Math.ceil(x)	Returns the ceil value	Math.ceil(5.8) is 6 Math.ceil(2.2) is 3
Math.floor(x)	Returns the floor value	Math.floor(5.8) is 5 Math.floor(2.2) is 2
Math.round(x)	Returns the round value, nearest integer value	Math.round(5.8) is 6 Math.round(2.2) is 2
Math.trunc(x)	Removes the decimal places it returns only integer value	Math.trunc(5.8) is 5 Math.trunc(2.2) is 2
Math.max(x,y)	Returns the maximum value	Math.max(2,3) is 3 Math.max(5,2) is 5
Math.min(x,y)	Returns the minimum value	Math.min(2,3) is 2 Math.min(5,2) is 2
Math.sqrt(x)	Returns the square root of x	Math.sqrt(4) is 2

Math.pow(a,b)	This method will compute the a^b	Math.pow(2,4) is 16
Math.sin(x)	Returns the sine value of x	Math.sin(0.0) is 0.0
Math.cos(x)	Returns cosine value of x	Math.cos(0.0) is 1.0
Math.tan(x)	Returns tangent value of x	Math.tan(0.0) is 0
Math.exp(x)	Returns exponential value i.e e^x	Math.exp(0) is 1
Math.random(x)	Generates a random number in between 0 and 1	Math.random()
Math.log(x)	Display logarithmic value	Math.log(2.7) is 1
Math.PI	Returns a π value	$a = \text{Math.PI}$ $a = 3.141592653589793$

The Date Object

- This object is used for obtaining the date and time. In JavaScript, dates and times represent in milliseconds since 1st January 1970 UTC. JavaScript supports two time zones: UTC and local. UTC is Universal Time, also known as Greenwich Mean Time(GMT), which is standard time throughout the world. Local time is the time on your System. A JavaScript *Date* represents date from -1,000,000,000 to -1,000,000,000 days relative to 01/01/1970.
- Date Object Constructors:
- new Date();* Constructs an empty date object.
- new Date("String");* Creates a Date object based upon the contents of a text string.
- new Date(year, month, day[,hour, minute, second]);* Creates a Date object based upon the numerical values for the year, month and day.

```
var dt=new Date();
document.write(dt); // Fri Nov 11 2022 01:47:11 GMT+0530 (Sri Lanka Standard Time)
```

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML =
Date();
</script>
</body>
</html>
```

output:
Fri Nov 11 2022 01:48:40 GMT+0530 (Sri Lanka Standard Time)

JavaScript Date Methods

- Date methods let you get and set date values (years, months, days, hours, minutes, seconds, milliseconds)

Method	Description
getDate()	Get the day as a number (1-31)
getDay()	Get the weekday as a number (0-6)
getFullYear()	Get the four digit year (yyyy)
getHours()	Get the hour (0-23)
getMilliseconds()	Get the milliseconds (0-999)
getMinutes()	Get the minutes (0-59)
getMonth()	Get the month (0-11)

JavaScript Arrays

JavaScript arrays are used to store multiple values in a single variable.

Displaying Arrays

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var cars = ["Saab", "Volvo", "BMW"];
document.getElementById("demo").innerHTML = cars[0];
</script>
</body>
</html>
```

Creating an Array

Using an array literal is the easiest way to create a JavaScript Array.

Syntax:

```
var array-name = [item1, item2, ...];
```

Using the JavaScript Keyword new

The following example also creates an Array, and assigns values to it:

```
var cars = new Array("Saab", "Volvo", "BMW");
```

Access the Elements of an Array

You refer to an array element by referring to the **index number**.

```
var name = cars[0];
```

This statement modifies the first element in cars:

```
cars[0] = "Opel";
```

Can Have Different Objects in One Array

- JavaScript variables can be objects. Arrays are special kinds of objects.
- Because of this, you can have variables of different types in the same Array.

Example:

```
myArray[0] = Date.now;  
myArray[1] = myFunction;  
myArray[2] = myCars;
```

Note:

- ❖ var points = new Array(); // Bad
- ❖ var points = []; // Good

When to Use Arrays? When to use Objects?

- JavaScript does not support associative arrays.
- You should use objects when you want the element names to be strings.
- You should use arrays when you want the element names to be sequential numbers.

Arrays are Objects

- Arrays are a special type of objects. The **typeof** operator in JavaScript returns "object" for arrays.
- But, JavaScript arrays are best described as arrays.

Arrays use **numbers** to access its "elements". In this example, person[0] returns John:

Array: var person = ["John", "Doe", 46];

Objects use **names** to access its "members". In this example, person.firstName returns John:

Object: var person = {firstName:"John", lastName:"Doe", age:46};

Array Properties and Methods

The real strength of JavaScript arrays are the built-in array properties and methods:

Examples

```
var x = cars.length; // The length property returns the number  
                     of elements in cars  
var y = cars.sort(); // The sort() method sort cars in  
                     alphabetical order
```

97

98

Array Object Methods

Method	Description
concat()	Joins two or more arrays, and returns a copy of the joined arrays
indexOf()	Search the array for an element and returns its position
join()	Joins all elements of an array into a string
lastIndexOf()	Search the array for an element, starting at the end, and returns its position
pop()	Removes the last element of an array, and returns that element
push()	Adds new elements to the end of an array, and returns the new length
reverse()	Reverses the order of the elements in an array
shift()	Removes the first element of an array, and returns that element
slice()	Selects a part of an array, and returns the new array
sort()	Sorts the elements of an array
splice()	Adds/Removes elements from an array
toString()	Converts an array to a string, and returns the result
unshift()	Adds new elements to the beginning of an array, and returns the new length
valueOf()	Returns the primitive value of an array

99

100

Concat(): Joins two or more arrays, and returns a copy of the joined arrays.

Syntax: array.concat(array1,array2.....);

Ex:

```
var hege = ["Cecilie", "Lone"];
var stale = ["Emil", "Tobias", "Linus"];
var kai = ["Robin"];
var children = hege.concat(stale,kai);
```

indexOf(): this method searches the array for the specified item, and returns its position.

- The search will start at the specified position, or at the beginning if no start position is specified, and end the search at the end of the array.

Returns -1 if the item is not found.

If the item is present more than once, the indexOf method returns the position of the first occurrence.

Syntax: array.indexOf(item);

Ex:

```
var fruits=["Banana","Orange","Apple","Mango"];
var x=fruits.indexOf("Apple");
```

Here the value of x is:2 i.e. Apple is at position 2.

Join(): The join() method joins the elements of an array into a string, and returns the string. The elements will be separated by a specified separator. The default separator is comma (,).

Syntax: array.join();

Ex:

```
var fruits=[“Banana” “Orange”,“Apple”,“Mango”];
var x=fruits.join();
```

the result of x will be Banana,Orange,Apple,Mango.

Slice(): The slice() method returns the selected elements in an array, as a new array object. The slice() method selects the elements starting at the given *start* argument, and ends at, *but does not include*, the given *end* argument.

Syntax: array.slice (start, end);

Ex:

```
var fruits=[“Banana”,“Orange”,“Apple”,“Mango”];
var x=fruits.slice(1,3);
```

the result of x will be [Orange,Apple];

Splice(): The splice() method adds/removes items to/from an array, and returns the removed item(s).

Syntax: array.splice(index,howmany,item1,item2....);

Ex:

```
var fruits=[“Banana”,“Orange”,“Apple”,“Mango”];
fruits.splice(2,1,”Lemon”,“Kiwi”);
the result of fruits will be [Banana,Orange,Lemon,Kiwi,Mango]
```

101

Arrays Operations and Properties

- Declaring new empty array:

```
var arr = new Array();
```

- Declaring an array holding few elements:

```
var arr = [1, 2, 3, 4, 5];
```

- Appending an element / getting the last element:

```
arr.push(3);
var element = arr.pop();
```

- Reading the number of elements (array length):

```
arr.length;
```

- Finding element's index in the array:

```
arr.indexOf(1);
```

102

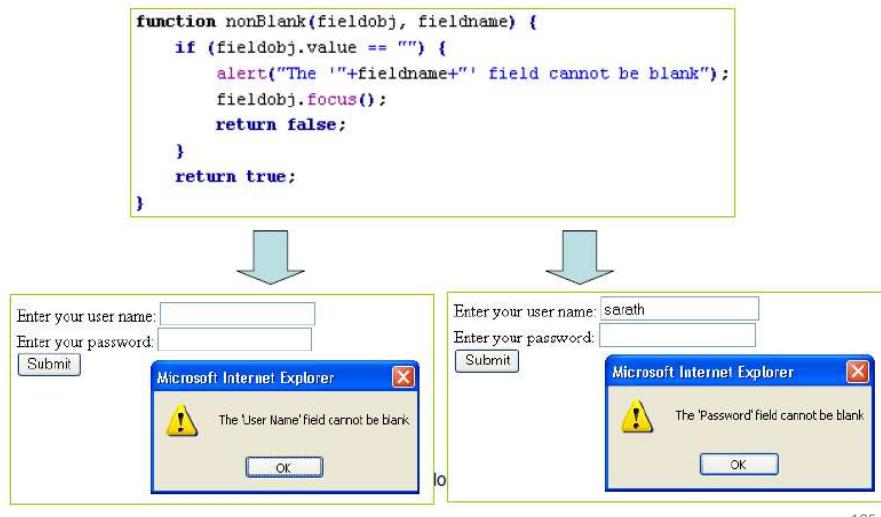
Form Validation Example

- A form to get user name and password.
Validation required to check
 - has the user left required fields empty
 - has the user entered text in correct length

Enter your user name:

Enter your password:

Checking the Field is Not Blank



Checking the Field is in Correct Length

Enter your user name:

Enter your password:

Submit

Microsoft Internet Explorer

The 'Password' field must be atleast 5 characters long

OK

↓

↓

106

Checking the Field is in Correct Length

```
function checkLength(fieldobj, fieldname, minlen, maxlen) {
    if (fieldobj.value.length < minlen) {
        alert("The '"+fieldname+"' field must be atleast "+minlen+
        " characters long");
        fieldobj.focus();
        return false;
    }
    if (fieldobj.value.length > maxlen) {
        alert("The '"+fieldname+"' field must be less than "+maxlen+
        " characters long");
        fieldobj.focus();
        return false;
    }
    return true;
}
```

Main Form Handler

```
<form action="handler.php" method="post" name="myform" id="myform"
onSubmit="return validate();"><p>
Enter your user name:<input name="username" type="text" id="username"><br>
Enter your password:<input name="password" type="password" id="password"><br>
<input type="submit" name="Submit" value="Submit">
</p></form>
```

↓

```
function validate() {
    if (nonBlank(myform.username,"User Name") &&
    nonBlank(myform.password,"Password") &&
    checkLength(myform.password,"Password",5,20))
        return true;
    else
        return false;
}
```

Form validation example contd...

```
<html>
<head>
<script language="javascript">
function validate()
{
    var x=f1.t1.value;
    var y=f1.t2.value;
    var re=/\d|\W|_/.g;

    if(x=="" || y=="")
        window.alert("Enter both user id and password");
    else if(x.length<6 || y.length<6)
        window.alert("both user id and password should greater than or equal to 6");
    else if((re.test(x)))
        window.alert("user name should be alphabets");
    else
        window.alert("welcome"+ " Mr."+x);
}
}
```

```
</script>
</head>
<body bgcolor="green">
<form name="f1">
<center>
User Name<input type="text" name="t1"><br><br>
Password <input type="password" name="t2"><br><br>
<input type="submit" value="submit" onClick=validate()>
<input type="reset" value="reset">
</center>
</form>
</body>

</html>
```

DYNAMIC HTML

- **DHTML** is combination of HTML, CSS and JavaScript. It gives pleasant appearance to web page.
- Difference between HTML and DHTML

HTML	DHTML
HTML is used to create static web pages.	DHTML is used to create dynamic web pages.
HTML is consists of simple html tags.	DHTML is made up of HTML tags+cascading style sheets+javascript.
Creation of html web pages is simplest but less interactive.	Creation of DHTML is complex but more interactive.

Contd...

Dynamic HTML, or DHTML, is for a collection of technologies used together to create interactive and animated by using a combination of a static markup language (such as HTML), a client-side scripting language (such as JavaScript), a presentation definition language (such as CSS), and the Document Object Model.

Uses

- DHTML allows authors to add effects to their pages that are otherwise difficult to achieve.
- For example, DHTML allows the page author to:
- Animate text and images in their document, independently moving each element from any starting point to any ending point, following a predetermined path or one chosen by the user.
- · Embed a ticker that automatically refreshes its content with the latest news, stock quotes, or other data.
- · Use a form to capture user input, and then process and respond to that data without having to send data back to the server.
- · Include rollover buttons or drop-down menus.

INTRODUCTION OF JAVA SCript

- A scripting language is a lightweight programming language.
- JavaScript is programming code that can be inserted into HTML pages.
- JavaScript inserted into HTML pages, can be executed by all modern web browsers.

DEMO

- ```
<script>
 alert("My First JavaScript")
</script>
```

### EVENT

- we want to execute code when an **event** like when the user clicks a button

### FUNCTIONS

- if we put JavaScript code inside a **function** call that function when an event occurs.

## SCRIPT TAG

- Scripts in HTML must be inserted between `<script>` and `</script>` tags.
- Scripts can be put in the `<body>` and in the `<head>` section of an HTML page.

### The `<script>` Tag

- To insert a JavaScript into an HTML page, use the `<script>` tag.
- The `<script>` and `</script>` tells where the JavaScript code starts and ends.

## JAVASCRIPT IN `<HEAD>` OR `<BODY>`

- You can place an unlimited number of scripts in an HTML document.
- Scripts can be in the `<body>` or in the `<head>` section of HTML, and/or in both.
- It is a common practice to put functions in the `<head>` section, or at the bottom of the `<body>` section.

## EXTERNAL JAVASCRIPTS

- Scripts can also be placed in external files. These files often contain code to be used by several different web pages.
- External JavaScript files have the file extension .js
- To use an external script, point to the .js file

## MANIPULATING HTML ELEMENTS

- To access an HTML element from JavaScript, you can use the `document.getElementById(id)` method.

### EXAMPLE

- `<body>`

```
<h1>My First Web Page</h1>
<p id="demo">My First Paragraph</p>
```

- `<script>`

```
document.getElementById("demo")
```

- `.innerHTML="My First JavaScript"`

## WRITING TO THE DOCUMENT OBJECT

- `<h1>My First Web Page</h1>`
- `<script>`
- `document.write("<p>My First JavaScript</p>")`

## JAVASCRIPT STATEMENTS

- JavaScript statements are "commands" to tell the browser. The purpose of the statements is to tell the browser what to do.
- This JavaScript statement tells the browser "Hello Dolly" inside an HTML element with `id="demo"`:

## SEMICOLON ;

- Semicolon separates JavaScript statements.
- Normally you add a semicolon at the end of each executable statement.
- Using semicolons also makes it possible to write multiple statements on one line.

## JAVASCRIPT CODE

- JavaScript code (or just JavaScript) is a set of JavaScript statements.
- Each statement is executed by the browser in sequence they are written.

## JAVASCRIPT IS CASE SENSITIVE

- JavaScript is case sensitive.
- Watch your capitalization closely when you write JavaScript statements:
- A function getElementById is not the same as getElementbyID.

## JAVASCRIPT CODE BLOCKS

- JavaScript statements can be grouped together into blocks.
- Blocks start with a left curly bracket, and end with a right curly bracket.
- The purpose of a block is to make the sequence of statements execute together.
- A good example of statements grouped together into blocks, are JavaScript **functions**.

## WHITE SPACE

- JavaScript ignores extra spaces. You can add space to your script to make it more readable.
- `var name="Hege";`

## BREAK UP A CODE LINE

- `document.write("Hello \n World!")`

## JAVASCRIPT COMMENTS

- // Write to a heading:  
`document.getElementById("myH1").innerHTML="Welcome to my Homepage";`
  
- /\*  
The code below will write  
to a heading and to a paragraph,  
etc.  
\*/  
**COMMENT CONT...**
  
- `var x=5; // declare x and assign 5`  
`var y=x+2; // declare y and assign`

## DATA TYPE EXAMPLE

- `var pi=3.14;`  
`var name="John Doe";`  
`var age=26;`
  
**DECLARING JAVASCRIPT VARIABLE**  
- You declare JavaScript variables with keyword
  
- `var carname;`
  
- `carname = "Volvo";`

## JAVASCRIPT VARIABLES

- Variable names must begin with a letter
  
- Variable names can also begin with \$ and will not use it)

## JAVASCRIPT DATA TYPES

- JavaScript variables can also hold other types like text values (`name="John Doe"`).
- In JavaScript a text like "John Doe" is called a string.
- There are many types of JavaScript variables now, just think of numbers and strings.
- When you assign a text value to a variable, put single quotes around the value.
- When you assign a numeric value to a variable, put commas around the value.

## JAVASCRIPT IS LOOSELY TYPED

- JavaScript is weakly typed. This means that the same variable can be used as different types.

### ○ Example

- `var x;` // Now x is undefined

## JAVASCRIPT OPERATORS

- = is used to assign values.
- The assignment operator = is used to ass to JavaScript variables.
  
- + is used to add values.
- The arithmetic operator + is used to add \

## ARITHMETIC OPERATOR

Operator	example
+	x=y+2
-	x=y-2
*	x=y*2 10
/	x=y/2
%	x=y%2
++	x=++y x=y++

## THE + OPERATOR USED ON STI

```
txt1="What a very";
txt2="nice day";
txt3=txt1+txt2;
```

## ADDING STRINGS AND NUM

```
x=5+5;
y="5"+5;
z="Hello"+5;
```

```
output
10
```

```
<script>
document.getElementById("demo").inner
HTML =
"John' is " + typeof "John" + "
" +
"3.14 is " + typeof 3.14 + "
" +
"NaN is " + typeof NaN + "
" +
"false is " + typeof false + "
" +
"[1, 2, 3, 4] is " + typeof [1, 2, 3, 4] +
"
" +
"{name:'John', age:34} is " + typeof
{name:'John', age:34} + "
" +
"new Date() is " + typeof new Date() +
"
" +
"function () {} is " + typeof function () {} +
"
" +
"myCar is " + typeof myCar + "
" +
>null is " + typeof null;
</script>
```

## COMPRESSION OPERATOR

- == is equal to
- === is exactly equal to (value and type)
- != is not equal
- !== is not equal (neither value nor type)
- > is greater than
- < is less than
- - is greater than or equal to

## LOGICAL OPERATORS

- && and  $(x < 10 \& & y > 0)$
- || or  $(x == 5 \mid\mid y == 5)$
- ! not  $!(x == y)$

## CONDITIONAL STATEMENTS

- Very often when you write code, you want to perform different actions for different decisions. You can use conditional statements in your code.

## CONDITIONAL STATEMENTS CONT..

- In JavaScript we have the following conditional statements:
- **if statement** - use this statement to execute some code only if a specified condition is true
- **if...else statement** - use this statement to execute one code if the condition is true and another code if the condition is false
- **if...else if....else statement** - use this statement to execute one of many blocks of code to be executed

## CONDITIONAL OPERATOR

- JavaScript also contains a conditional operator which assigns a value to a variable based on some condition.

### ○ Syntax

- $\text{Variable name} = (\text{condition}) ? \text{value1} : \text{value2}$

#### - Examples -

```
<script>
 let x=5;
 let y=(x<18)? "A": "B";
 alert(y);
</script>
```

## IF STATEMENT

- **if (condition)**  
    {  
        code to be executed if condition is true  
    }

## IF...ELSE

- **if (condition)**  
    {  
        code to be executed if condition is true  
    }  
    **else**  
    {  
        code to be executed if condition is false  
    }

## IF ... ELSE IF ... ELSE

- o if (*condition1*)  
  {  
    *code to be executed if condition1 is true*  
  }  
else if (*condition2*)  
  {  
    *code to be executed if condition2 is true*  
  }  
else  
  {  
    ...  
    ...  
    ...  
  }

## THE JAVASCRIPT SWITCH STATEMENT

- o switch(*n*)  
  {  
    case 1:  
      *execute code block 1*  
      break;  
    case 2:  
      *execute code block 2*  
      break;  
    default:  
      *code to be executed if n is different from 1 or 2*  
  }

## Next Week LOOP

Loops can execute a block of code a number of times.

### JAVASCRIPT LOOPS

- o Loops are handy, if you want to run the same code over and over again, each time with a different value.

## THE FOR LOOP

- o for (*variable initialization ; Condition Check ; increment or decrement* )  
  {  
    *the code block to be executed*  
  }

- <body>
- <h2>JavaScript For Loop</h2>
- <p id="demo"></p>
- <script>
- let text = "";
- for (let i = 0; i < 5; i++) {
- text += "The number is " + i + "<br>";
- }
- document.getElementById("demo").innerHTML = text;
- </script>
- </body>

JavaScript For Loop  
The number is 0  
The number is 1  
The number is 2  
The number is 3  
The number is 4

- ```
for (var i = 0; i < 5; i++) {  
  for (var j = 0; j < 5; j++) {  
    console.log(i + " " + j); } }  
#####  
var i = 0;  
for (; i < 5; i++)  
{ console.log(i); }
```
- ```
for (var i = 0; ; i++)
{ console.log(i); if(i == 5)
{ break; } }

for(var i = 0; i < 5;) {
 console.log(i); i++; }
```
- ```
var i = 0;  
for (; ; )  
{ console.log(i);  
if(i == 5)  
{ // condition  
break; } // update the  
variable  
i++; }
```
- ```
for (var i = 0; i < 5; i++)
 console.log(i);

var m = 10, n = 20;
for (var i = m; i <= n; i++)
{ if (i % 2 === 0)
{ console.log(i); } }
```

- var n = 20;
- for (var i = 2; i <= n; i++) {
- var isPrime = true;
- for (var j = 2; j <= Math.sqrt(i); j++) { // nested loop
- if (i % j === 0) {
- isPrime = false;
- break; } }
- if (isPrime) {
- console.log(i);
- } }

- OutPut

```

* *
* *
* *
```

```
*


```

- <script>
- let n = 5; // row or column count
- // defining an empty string
- let string = "";
- for (let i = 0; i < n; i++) { // external loop
- for (let j = 0; j < n; j++) { // internal loop
- if (i === 0 || i === n - 1) {
- string += "\*";
- }
- else {
- if (j === 0 || j === n - 1) {
- string += "\*";
- }
- else {
- string += " ";
- }
- }
- }
- // newline after each row
- string += "\n";
- }
- // printing the string
- document.write(`<pre>\${string}</pre>`);
- </script>

## The While Loop

- The while loop loops through a block of code as long as a specified condition is true
- **while (*condition*) {**  
*// code block to be executed*  
**}**
- Example
- **while (i < 10) {**  
*text += "The number is " + i;*  
*i++;*  
**}**

# The Do While Loop

- The do while loop is a variant of the 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.

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

```
<!DOCTYPE html>  
<html>  
<body>  
<h2>JavaScript Iterables</h2>  
<p>Iterate over an Array:</p>  
  
<p id="demo"></p>  
<script>  
let sum = 0;  
  
// take input from the user  
let number = parseInt(prompt('Enter a number: '));  
  
while(number >= 0) {  
  
    // add all positive numbers  
    sum += number;  
  
    // take input again if the number is positive  
    number = parseInt(prompt('Enter a number: '));  
}  
  
// display the sum  
console.log(`The sum is ${sum}.`);  
</script>  
</body>  
</html>
```

• for Vs while Loop

- A for loop is usually used when the number of iterations is known. For example,

- ```
// this loop is iterated 5 times
for (let i = 1; i <=5; ++i) { // body of loop }

And while and do...while loops are usually
used when the number of iterations are
unknown. For example,

• while (condition) { // body of loop }
```

- ```
// infinite while loop  
while(true){  
    // body of loop  
}
```



HNDIT1022 – Web Design

Subtopics

- What is domain name
- DNS
- What is hosting space
- What is email service

Week 13: introduction to web servers and hosting

What is domain name

A domain name is a string of text that maps to a numeric IP address, used to access a website from client software.

In plain English, a domain name is the text that a user types into a browser window to reach a particular website.

The actual address of a website is a complex numerical IP address (e.g. 103.21.244.0), but thanks to Domain Name System (DNS), users are able to enter human-friendly domain names and be routed to the websites they are looking for. This process is known as a DNS lookup.

Who manages domain names?

Domain names are all managed by domain registries, which delegate the reservation of domain names to registrars. Anyone who wants to create a website can register a domain name with a registrar, and there are currently over 300 million registered domain names.

What is DNS

The Domain Name System (DNS) is the phonebook of the Internet. Humans access information online through domain names, like nytimes.com or espn.com. Web browsers interact through Internet Protocol (IP) addresses. DNS translates domain names to IP addresses so browsers can load Internet resources.

Each device connected to the Internet has a unique IP address which other machines use to find the device. DNS servers eliminate the need for humans to memorize IP addresses such as 192.168.1.1 (in IPv4), or more complex newer alphanumeric IP addresses such as 2400:cb00:2048:1::c629:d7a2 (in IPv6).

What is hosting space

The web hosting space, also known as web space, storage space or disk space, generally refers to the amount of space on a web server that is allocated to website owners by the web hosting companies. It is made up of the total quantity of all text files, images, scripts, databases, emails and other files related to your website.

Contd...

Web hosting is the process of storing all the files that make up your website on a server.

Think of it as your digital office. Like you need a physical space for a physical office, you need a digital space for your digital office. A web host rents out that digital space on servers for your website files. It helps you establish your website on the internet.

Features To look When Choosing A Reliable Web Hosting Provider

- Look for reliability
- customer service
- web space and bandwidth
- domain name service
- site management extras
- price

What is email service

An email service is a company that provides businesses with tools to send bulk emails and implement email marketing.

Email services offer user-friendly features to manage mailing lists, email design, and metrics to monitor your success.

Why is email service important?

- Fits any business
- Provides useful tools for creating emails
- Gathers a lot of data
- Results in high Return On Investment(ROI)
- Offers an inexpensive or free trial
- Simplifies sending emails