

# INTRODUCTION TO INTERNET

Lt. Ibrahim Salim M

Copyright © Instructors.

# OBJECTIVES

- define Internet
- list different types of Internet connections
- explain various services provided by Internet
- download files
- send and receive e-mail
- be acquainted with the terminologies used in Internet

# Internet

- The Internet is a giant network of networks.
  - A network may include PCs, and other devices like servers or printers.
  - A network is connected through a communication channel.
  - No one owns the internet
    - Every person makes a connection owns a slice of the internet
  - There is no central administration to the Internet.

# History of the Internet

- - Grew out of a research network originally funded by U.S. Department of Defense.
- - Development of this network, known as the ARPAnet after the Advanced Research Projects Agency (ARPA), began in 1969.
- As the network grew, it was used for applications beyond research, such as electronic mail.
- In the early 1980s, the current versions of the core Internet protocols, TCP and IP, were introduced across the network.

# History of the Internet

- The term Internet comes from the word inter-network - an interconnected set of networks.
- In 1992, the Center for European Nuclear Research (CERN) released the first versions of World Wide Web software.
- Subsequently, the number of Web servers has grown quickly.

# Applications Of Internet

- Exchange messages using e-mail (Electronic mail).
- Transfer files as well as software.
- Browse through information on any topic on web.
- Communicate in real time (chat) with others connected to the Internet.
- Search databases of government, individuals and organizations.
- Read news available from leading news groups.
- Send or receive animation and picture files from distant places.
- Set up a site with information about your company's products and services.

# Key Milestones in Evolution

- 1950's
  - ARPA (Advanced Research Projects Agency)
- 1970 –
  - ARPANET creates precursor to Transmission Control Protocol (TCP)
- 1971
  - Universities added to net
  - Telnet and FTP are available
- 1972
  - First electronic mail message sent

# Key Milestones in Evolution

- 1973-
  - ARPANET connected to England and Norway
- 1974-
  - TCP starts being used for communicating across a system of networks
- 1982-
  - US DoD starts building defense data networks based on ARPANET technology
- 1983-
  - ARPANET splits into ARPANET and MILNET
  - Internet now in place
  - TCP/IP standardized

# Key Milestones in Evolution

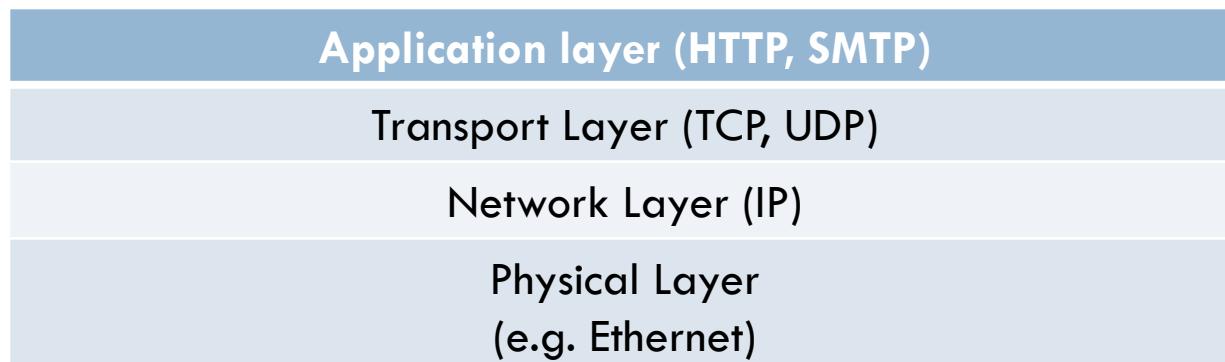
- 1986-
  - National Science Foundation (NSF) implements NFSNET; a system of regional network of routers connected over a backbone network
- 1991-
  - Archie and Gopher released 1992-
  - Internet links more than 17,000 networks
  - in 33 countries; 3 million hosts
    - 1993-
      - World Wide Web is launched
- 1995-
  - Interconnected network providers start offering service
  - About 30 million users

# The Internet Properties

- Key properties of the Internet:
  - The Internet is interoperable.
  - The Internet is global.
  - The Web makes it easy.
  - The costs of the network are shared across multiple applications and borne by the end users.
- The striking characteristic of the Internet --> heterogeneity.

# The Internet Layered Architecture

- The Internet, as a network of connecting many small networks, consists of four layers:
  - Application Layer (HTTP, SMTP..)
  - Transport Layer (TCP, UDP)
  - Network Layer (IP)
  - Physical Layer



# The Internet: Design Principles

- The Internet has been successful because of some fundamental decisions about its design made early in its history.
- **Interoperability:** Independent implementations of Internet protocols actually work together.
- Interoperability means that systems can be assembled using client and server computers and software from different vendors.
- In the context of Internet commerce, interoperability means that buyers and sellers do not have to buy and upgrade software simultaneously from the same vendors to conduct commerce.

# The Internet: Design Principles

- **Layering:**
- Internet protocols are designed to work in layers, with each higher layer building on the facilities provided by lower layers.
- **Simplicity:**
- One way to look at the layering of the Internet is that it grows both up and down from IP. IP is very simple, providing only addressing and formatting of packets.
- Below the level of IP, there is the complexity of many different kinds of network hardware, topologies, and routers

# The Internet: Design Principles

- IP hides that complexity from applications and insulates application developers from:
  - the complexities of different network devices
  - the complexities of implementing low-level network protocols.
- Above IP, higher-level protocols such as TCP offer service abstractions that are easy for application programmers to understand and use.

# The Internet: Design Principles

- Uniform naming and addressing:
- The IP layer offers a uniform addressing structure that assigns a 32-bit address to each computer connected to the network.
- Domain name system (DNS) offers a uniform way to translate human-readable names for computers, such as [www.openmarket.com](http://www.openmarket.com) to the numeric address for that computer.

# The Internet: Design Principles

- **End-to-end:**
- Internet is designed around end-to-end protocols. That is, the interpretation of the data happens on the sending and receiving systems, but nothing in the network needs to look at anything but the destination address for delivering the packet.
- End-to-end protocols have several advantages:
  - hide the internal structure of the network
  - provide simple abstractions to programmers
  - shielding them from such things as the messy details of recovering from lower-level errors.

# The Internet Protocols

- **FTP:(File transfer protocol)**
- One of the most oldest and probably the most popular protocol to be used to move files on the Internet.
- **TCP/IP:(Transmission Control Protocol and Internet Protocol)**
  - ▣ The low-level communications protocol that holds the Internet together.
  - ▣ It provides means to allows two software on difference machines on the Internet find each other, rendezvous, and transfer data.

# The Internet Protocols

- It provides the essential service of making sure that each piece of data is transferred in the correct sequence and without error.
- **SMTP: (the e-mail message protocol)**
  - A protocol to allow two users to communicate through e-mail messages over the Internet.
- **NNTP: (Net News Transfer Protocol)**
  - A protocol, which can be used to access or transfer Usenet news over the Internet.
- **Telnet:**
  - The traditional teletype-style communications protocol for communicating with text-based information services.

# Types of Internet Connections

- **Dial-up**
- This is the most common basic type of connection available from ISPs (Internet Server Providers).
- In Dial-up connection, you use your computer, dial a phone number (provided by ISP) to get connected to server at Providers end through which you access Internet.
- It means you are not directly connected to Internet; you access the Internet through an Internet Service Provider.
- **ISDN (Integrated Services Digital Network)**
- The process of connecting to server to access Internet is almost same as Dial-up, but it offers connectivity through the use of digital phone lines instead of Analog.

# Types of Internet Connections

- It offers Internet connectivity at speeds of up to 128 Kbps, allows the user to receive or make calls simultaneously on the same line.
- ISDN comes through a regular telephone wire from the telephone pole on the street.
- The line combines two 64 Kbps channels to offer 128 Kbps bandwidth broken into three bands:
- One band for the ringing signal of your phone, one band for your telephone conversation, and one band for data transfer.

# Types of Internet Connections

- **Leased Line Connection (Direct Internet Access)**
- A “permanent connection” between a computer system (single CPU or LAN, and the Internet). It is generally used by larger institutions, corporate and government agencies.
- It involves establishing your own Internet gateway (connection) and payment to have a direct full time line with the network.
- Dedicated links are established through an internet service provider who places a computer-controlled router (message director) at your site.
- A router is used to connect your local network to the Internet, allow all the members of network to have complete access to Internet.

# Types of Internet Connections

- DSL (Digital Subscriber Line or Dedicated Service Line)  
**Broadband Connection**
- DSL, an “always-on” data connection is becoming widely available these days.
- It can provide an excellent Internet connection.
- It connects your home or office to the Internet through the same telephone wire that comes from telephone pole on the street. Like ISDN, with DSL, user can make and receive telephone calls while connected to the Internet.
- The difference between DSL and dialup / ISDN is that a DSL Internet connection uses a high-speed dedicated circuit filtering out standard phone calls and Internet signals.

# The World Wide Web

- Latest revolution in the internet scenario.
- WWW Allows multimedia documents to shared between machines
  - Containing text, Image, audio, video, animations.
- Basically a huge collection of inter-linked documents
  - Billions of documents.
  - Inter-linked in any possible way.
  - Resembles cob-web.
- The World Wide Web: History
- March, 1989, Tim Berners-Lee of Geneva's European Particle Physics Laboratory (CERN) circulated a proposal to develop a hypertext system for global information sharing in High Energy Physics community.

# The World Wide Web

- The World Wide Web project began to take shape at the beginning of 1991.
- October 1991, the gateway for WAIS search (a crucial development for the Web's future as search as well as a browsing tool),
- Before the end of 1991, CERN announced the Web to the High Energy Physics community in general.
- Essentially, 1992 was a developmental year. In March of 1993, WWW traffic clocked in at 0.1 percent of total Internet backbone traffic.
- In July of 1994, CERN began to turn over the Web project to a new group called the W3 organization, a joint venture between CERN and MIT to develop the Web further.

# The World Wide Web

- The World Wide Web: HTML
- HTML is a simplified derivative of SGML, or Standard Generalized Markup language.
- Its code can be used to make documents readable across a variety of platforms and software.
- Like SGML, HTML operates through a series of codes placed within an ASCII doc. These codes are translated by a WWW client such as Lynx, Mosaic.
- Cello, Viola, or MacWeb into specific kinds of formats to be displayed on the screen.
  - Items include in a HTML page are:
  - links, lists, headings, titles, images, forms, and maps.
- Due to the limitation of HTML documents, now more advanced technologies are introduced to enrich its functions, such as , JavaScript,

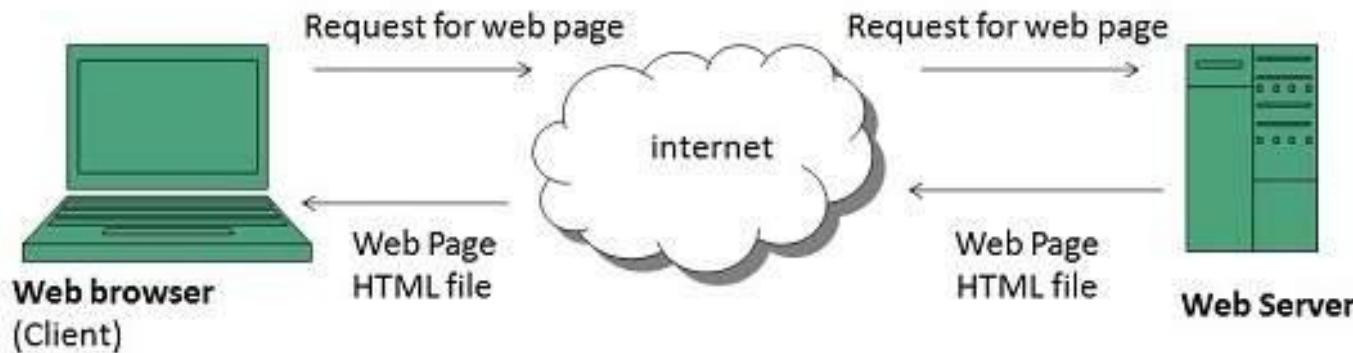
# The World Wide Web

- The World Wide Web: HTTP
- HTTP stands for HyperText Transfer Protocol.
- It is a simple data transfer protocol that binds the Web together.
- It supports the communications between a web client (browser) and its web server.
- It consists of a set of messages and replies for both servers and browsers.
- It treats documents, files, menus, and graphics as objects.
- - It relies on the Universal resource identifier (URI), enclosed in the universal resource locator (URL), to identify files.
- - It uses the Internet's TCP/IP network protocol.

# The World Wide Web

## WWW Operation

4. Then web server receives request using HTTP protocol and checks its search for the requested web page. If found it returns it back to the web browser and close the HTTP connection.
5. Now the web browser receives the web page, It interprets it and display the contents of web page in web browser's window.



# ELECTRONIC MAIL

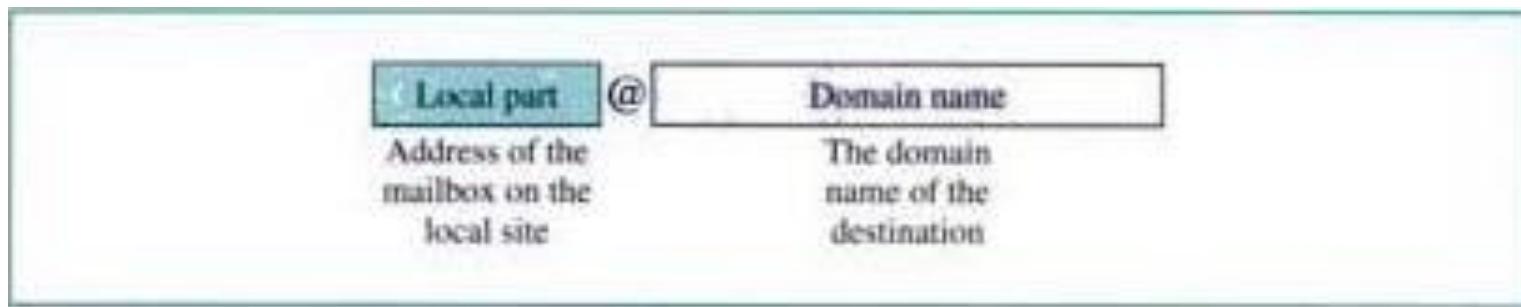
- One of the most popular network services is electronic mail (email).
- Electronic mail is used for sending a single message that includes text, voice, or graphics to one or more recipients.
- Simple Mail Transfer Protocol (SMTP) is the standard mechanism for electronic mail in the Internet.
- **Sending email**
- To send mail, the user creates mail that looks very similar to postal mail. It has an envelope and a message.
- **Envelope:** The envelope usually contains the address, the receiver address, and other information.

# ELECTRONIC MAIL

- **Message:** The message contains the headers and the body. The headers of the message define the sender, the receiver, the subject of the message, and other information.
- The body of the message contains the actual information to be read by the recipient.
- **Receiving Mail :** The email system periodically checks the mailboxes. If a user has mail, it informs the user with a notice.
- If the user is ready to read the mail, a list is displayed in which each line contains a summary of the information about a particular message in the mailbox.
- The summary usually includes the sender mail address, the subject, and the time the mail was sent or received. The user can select any of the messages and display its contents on the screen.

# ELECTRONIC MAIL

- Addresses
- To deliver mail, a mail handling system must use an addressing system with unique address.
- The addressing system used by SMTP consists of two parts: a local part and a domain name, separated by an @ sign



- Local Part: The local part defines the name of a special file, called the user mailbox, where all the mail received for a user is stored to be used by the user agent.

# ELECTRONIC MAIL

- **Domain Name:** The second part of the address is the domain name.
- An organization usually selects one or more hosts to receive and send; they are called mail exchangers.
- The domain name assigned to each mail exchanger either comes from the DNS database or is a logical name (ex.: the name of the organization).

# ELECTRONIC MAIL

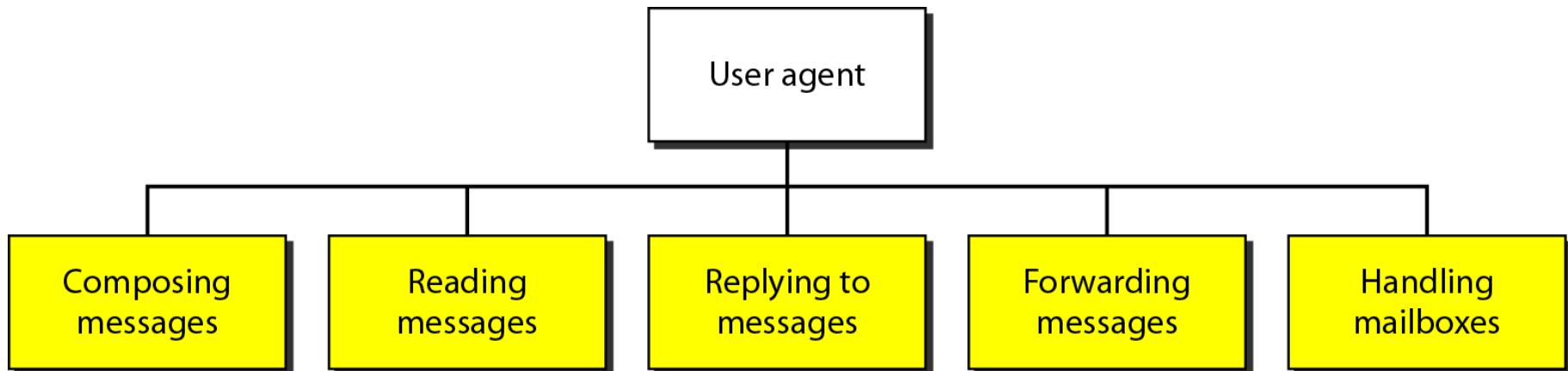
- **User Agent (UA)** The first component of an electronic mail system is the user agent (UA). A user agent sometimes is called a mail reader
- **User Agent Types:** There are two types of user agents: command-driven and GUI-based.
  1. **Command Driven:** Command –driven user agents belong to the early days of electronic mail.
- They are still present as the underlying user agents in servers.
- A command-driven user agent normally accepts a one-character command from the keyboard to perform its tasks.

# ELECTRONIC MAIL

- **GUI – Based:** Modern user agents are GUI-based.
- They contain Graphical User Interface (GUI) components that allow the user to interact with the software by using both the keyboard and the mouse.
- They have graphical components such as icons, menu bars, and windows that make the services easy to access.
- Some examples of GUI-based user agents are Eudora, Microsoft's Outlook, and Netscape.

# ELECTRONIC MAIL

- **Services provided by a User Agent:** A user agent is a software package (program) that composes (new mail), reads, replies to, and forwards messages.
- It also handles mailboxes



# ELECTRONIC MAIL

1. **Composing Messages** A user agent helps the user compose the e-mail message to be sent out.
  - Most user agents provide a template on the screen to be filled in by the user.
  - Some even have a built-in editor that can do spell checking, grammar checking, and other tasks expected from a sophisticated word processor.
2. **Reading Messages** The second duty of the user agent is to read the incoming messages.
  - When a user invokes a user agent, it first checks the mail in the incoming mailbox.

# ELECTRONIC MAIL

- Most user agents show a one-line summary of each received mail. Each e-mail contains the following fields.
  1. A number field.
  2. A flag field that shows the status of the mail such as new, already read but not replied to, or read and replied to.
  3. The size of the message.
  4. The sender.
  5. The optional subject field.
- 3. **Replying to Messages** After reading a message, a user can use the user agent to reply to a message. A user agent usually allows the user to reply to the original sender or to reply to all recipients of the message.

# ELECTRONIC MAIL

4. **Forwarding Messages** Replying is defined as sending a message to the sender or recipients of the copy. Forwarding is defined as sending the message to a third party.

- A user agent allows the receiver to forward the message, with or without extra comments, to a third party.

5. **Handling Mailboxes** - A user agent normally creates two mailboxes: an inbox and an outbox.

- Each box is a file with a special format that can be handled by the user agent. The inbox keeps all the received e-mails until they are deleted by the user.
- The outbox keeps all the sent e-mails until the user deletes them. Most user agents today are capable of creating customized mailboxes.

# ELECTRONIC MAIL

- **MIME: Multipurpose Internet Mail Extensions**
- Electronic mail can send messages only in (Network Virtual Terminal) NVT 7-bit ASCII format so it has some limitations
- it cannot be used for languages that are not supported by 7-bit ASCII characters (such as French, German)
- Also, it cannot be used to send binary files or video or audio data.
- Multipurpose Internet Mail Extensions is a supplementary protocol that allows non-ASCII data to be sent through e-mail.
- MIME transforms non-ASCII data at the sender site to NVT ASCII data and delivers them to the client MTA to be sent through the Internet. The message at the receiving side is transformed back to the original data.

# ELECTRONIC MAIL

- MIME defines five headers that can be added to the original SMTP header section to define the transformation parameters:
  1. MIME-Version.
  2. Content - Type.
  3. Content – Transfer – Encoding.
  4. Content – ID.
  5. Content – Description.
- 1. **MIME – Version:** This header defines the version of MIME used. The current version is 1.1.
- 2. **Content – Type:** This header defines the type of data used in the body of the message.
- The content type and the content subtype are separated by a slash.

# ELECTRONIC MAIL

- Depending on the subtype, the header may contain other parameters. MIME allows seven different types of data.
3. Content-Transfer-Encoding: This header defines the method used to encode the messages into 0s and 1s for transport.
  4. Content-Id: This header uniquely identifies the whole message in a multiple- message environment.
  5. Content-Description: This header defines if the body is image, audio, or video.

# ELECTRONIC MAIL

- **Mail (or Message) Transfer Agent (MTA)**
- The actual mail transfer is done through Mail Transfer Agents (MTAs).
- To send mail, a system must have a client MTA: and to receive mail, a system must have a server MTA.
- In the Internet, message transfer is done through a protocol (and software) named Simple Mail Transfer Protocol (SMTP).
- To send a message, we need a client SMTP and a server SMTP, we show Alice sending an email to Bob with the SMTP clients and servers needed. Note that mail transfer occurs between the two mail servers, one at Alice's site and the other at Bob's site. The mail servers can belong to the ISPs to which Alice and Bob are subscribers, or they can belong to the companies

# ELECTRONIC MAIL

- **Mail Access Protocols**
- Post Office Protocol, version 3 (POP3):
  - It's simple, but it's limited in functionality.
  - The client POP3 software is installed on the recipient computer; the server POP3 software is installed on the mail server.
  - Mail access starts with the client when the user needs to download email from the mailbox on the mail server.
  - The client (user agent) opens a connection with the server on TCP port 110. It then sends its user name and password to access the mailbox.
  - The user can then list and retrieve the mail messages, one by one.

# ELECTRONIC MAIL

- **IMAP4: Internet Mail Access Protocol, Version 4 (IMAP4).**  
IMAP4 is similar to POP3, but it has more features
- All mail is stored on the server. Messages will appear the same way every time you set up a new e-mail client, no need to move messages. You can switch between an e-mail client and webmail at any time and still have the same messages.
- If your computer crashes and you lose the data stored on your hard drive your e-mail is still safe, because it is stored on the server.
- Makes it easier to access your e-mail using a smartphone because the messages are not removed from the server.
- A user can create, delete or rename mailboxes on the mail server.

# Archie

- The Internet community has been amassing text, image, software, and database resources for over twenty years.
- Historically, these resources have been stored in public repositories known as anonymous FTP servers.
- FTP is the Internet-standard high-speed file transfer protocol, used for exchange of private information by trusted parties with passwords as well as for publishing information without passwords, i.e., anonymously.
- Hundreds of archives now exist but, up until a year ago, no one tracked them.
- Archie (ARCHIvE server) was developed at McGill University to index the contents of all FTP servers and provide keyword searching of the index.

# Archie

- Its approach is simple but powerful: Every night it re-indexes roughly one thirtieth of the servers; the result is a database that is completely refreshed each month.
- Although Archie enables you to locate information, it does not allow you to view or retrieve the information.
- To do that, you need FTP software on an IP-connected workstation or host .

# WAIS

- Wide Area Information System (a joint project of Apple Computer, Dow Jones, KPMG Peat Marwick, and Thinking Machines Corporation) provides a uniform interface to many full-text databases, together with a sophisticated "relevance search" capability.
- You can search any WAIS database using any word or phrase and the system will return a menu of documents, ordered from more to less relevant.
- WAIS databases are commonly collections of related data (The Bryn Mawr Classical Review), primary source documents (Clinton speeches), or reference works (CIA World Fact Book, Roget's Thesaurus).
- There are currently almost 400 WAIS databases, and new ones appear frequently.

# WAIS

- Since it can be difficult to determine the focus of a WAIS database from its name, a Directory of Servers, itself a WAIS database, was developed.
- You can search this directory for topics that interest you, and it will suggest WAIS databases for you to explore.
- For example, you could search the directory using the keyword "religion," and you would be referred to three WAIS databases: the Book of Mormon, the Qur'an, and the Bible.

# Gopher

- Gopher began as the University of Minnesota's version of PennInfo, a menu-driven campus-wide information system (CWIS).
- Gopher's simplicity as a distributed, client/server CWIS led to its rapid adoption by other institutions, some of which developed new client or server software for desktop or host computers and contributed them to the Gopher software archive (accessible via anonymous FTP, naturally).
- Soon thereafter, Minnesota offered to provide a menu of all Gopher servers that any other Gopher could access.
- The result was what networkers have been talking about for years: an interoperating set of information systems linking several hundred organizations around the world, all with a common user interface

# Gopher

- The next step in Gopher's evolution was addition of gateways to FTP, Telnet (the Internet standard remote terminal protocol), Archie, WAIS, and WWW.
- Gopher was thus transformed from an integrated set of CWIS programs into the most successful Internet navigation tool.
- But success became problematic: As the worldwide menu structure grew, locating information became increasingly tedious.
- Something like Archie was needed to help researchers locate information quickly in this new, ever expanding "Gopherspace."

# Veronica

- In November, 1992 a search tool, Veronica, was contributed to Gopher by a team from University of Nevada at Reno.
- The original Veronica ("Very Easy, Rodent-Oriented, Net-wide Index to Computerized Archives," a comic acronym if ever there was one) provides a search through all menus using a single keyword.
- The result is a dynamically created menu of all Gopher resources that contain the keyword in their menus.
- Now, a second Veronica search tool has appeared--an indexed WAIS database extended to allow Boolean searches of menu documents.

# Veronica

- Although both searches are limited to words in menus (as opposed to the full text of documents), the combination of Veronica and Gopher results in a powerful capability to search for and retrieve information from all over the Internet, with the location of the information effectively irrelevant.

# Web Server

- A **Web server** is a program that generates and transmits responses to client requests for Web resources.
- Handling a client request consists of several key steps:
  - Parsing the request message
  - Checking that the request is authorized
  - Associating the URL in the request with a file name
  - Constructing the response message
  - Transmitting the response message to the requesting client

# Web Server

- The server can generate the response message in a variety of ways:
  - The server simply retrieves the file associated with the URL and returns the contents to the client.
  - The server may invoke a script that communicates with other servers or a back-end database to construct the response message.
- Web site and Web server are different:
  - A Web site consists of a collection of Web pages associated with a particular hostname.
  - A Web server is a program to satisfy client requests for Web resources.

# Web Server

- A Web server proceeds through the following steps in handling an HTTP request:
  - Read and parse the HTTP request message for example GET the resource /foo.htm
  - Translate the URL to a file name for example the resource be located in the base directory such as /www, where the URL `http://www.bar.com/foo/index.html` corresponds to the file of `www/foo/index.html`
  - Determine whether the request is authorized
  - Generate and transmit the response that includes header to show the status information

# Web Server

- A Web server may limit which users can access certain resources. Access control requires a combination of authentication and authorization.
  - **Authentication** identifies the user who originated the request.
  - **Authorization** determines which users have access to a particular resource.

# Web Server

- AUTHENTICATION
- Most client-server systems authenticate a user by asking for a name and password.
- Web server must perform authentication for every request for a resource that has access restrictions.
- The server returns an HTTP response that indicates that the request requires authorization.
- The response also identifies what kind of authentication is required.
- The response also identifies the realm
  - a string that associates a collection of resources at the server

# Web Server

- **AUTHORIZATION** : To control access to Web resources, the server must employ an authorization policy.
- A policy typically expressed in terms of an access control list that enumerates the users who are granted or denied access to the resources.
- In addition to checking the user name, the server may allow or deny access to the resource based on other information associated with the HTTP request, such as the host name or IP address of the requesting client.
- Authenticating HTTP requests can impose a heavy load on the Web server.

# PROXY SERVER

- Proxy server is an intermediary server between client and the internet.
- Proxy servers offers the following basic functionalities:
  - Firewall and network data filtering.
  - Network connection sharing
  - Data caching
- Proxy servers allow to hide, conceal and make your network id anonymous by hiding your IP address.

# PROXY SERVER

- Proxy server is an intermediary server between client and the internet.
- Proxy servers offers the following basic functionalities:
  - Firewall and network data filtering.
  - Network connection sharing
  - Data caching
- Proxy servers allow to hide, conceal and make your network id anonymous by hiding your IP address.

# Purpose of Proxy Servers

## 1. Improve performance

- Can dramatically improve performance for a group of users.
- It saves all the results of requests in a cache.
- Can greatly conserve bandwidth.

## 2. Filter requests

- Prevent users from accessing a specific set of web sites.
- Prevent users for accessing pages containing some specified strings.
- Prevent users from accessing video files

# Major types of websites

## 1. Blog : The term blog comes from the word weblog.

- Until 2009 blogs were usually the work of a single individual, occasionally of a small group, and often covered a single subject.
- More recently "multi-author blogs" (MABs) have developed, with posts written by large numbers of authors and professionally edited.
- This type of site is usually displayed in a reverse chronological order, such as the most recent post or upload appears first.
- The rise of Twitter and other "microblogging" systems helps integrate MABs and single-author blogs into societal news streams.

# Major types of websites

- Until 2009 blogs were usually the work of a single individual, occasionally of a small group, and often covered a single subject.
- More recently "**multi-author blogs**" (MABs) have developed, with posts written by large numbers of authors and professionally edited.
- This type of site is usually displayed in a reverse chronological order, such as the most recent post or upload appears first.
- The rise of Twitter and other "microblogging" systems helps integrate MABs and single-author blogs into societal news streams.

# Major types of websites

2. **Wiki** : These websites are created to serve as a detail way of passing descriptive information to the society, e.g. WIKIPEDIA.
  - Text is usually written using a simplified markup language or a rich-text editor.
  - While a wiki is a type of content management system, it differs from a blog or most other such systems in that the content is created without any defined owner or leader, and wikis have little implicit structure, allowing structure to emerge according to the needs of the users.
  - Trustworthiness and Security - the two biggest attribute for wikis. Critics of publicly editable wiki systems argue that these systems could be easily tampered with,

# Major types of websites

3. **Social** : Social network site is a site that enable user to create a public profile within that website and form relationship with other users of the web, however it is referred to a profile site.
  - Social site on the Internet is describes the community based site where it brings people together to talk, share ideas, share interests, make new friends, etc.
  - However this type of collaboration and sharing of data is often referred to as social media. Below are examples of social site; Facebook, twitter, YouTube, instagram etc.

# Web Browser

- Web browsers are mainly used to access pages of the World Wide Web. Or
- A Web browser acts as an interface between the user and Web server
- A Web browser contains the basic software you need in order to find, retrieve, view, and send information over the Internet. This includes software that lets you:
  - Send and receive electronicmail (or email) messages worldwide nearly instantaneously.
  - Read messages from newsgroups (or forums) about thousands of topics in which users share information and opinions.
  - Browse the World Wide Web (or Web) where you can find a rich variety of text, graphics, and interactive information.

# Web Browser

## □ COMPONENTS OF WEB BROWSER

### 1. User Interface

□ this includes the address bar, back/forward button , bookmarking menu etc

### 2. Rendering Engine

□ Rendering, that is display of the requested contents on the browser screen.

□ By default the rendering engine can display HTML and XML documents and images

# Web Browser

- **HISTROY**
- The history of the Web browser dates back in to the late 1980s, when a variety of technologies laid the foundation for the first Web browser, WorldWideWeb, by Tim Berners-Lee in 1991.
- Microsoft responded with its browser Internet Explorer in 1995 initiating the industry's first browser war
- Opera first appeared in 1996; although it have only 2% browser usage share as of April 2010, it has a substantial share of the fast-growing mobile phone Web browser market, being preinstalled on over 40 million phones.
- In 1998, Netscape launched Mozilla

# Web Browser

- Different types of browser

## 1. MOZILLA FIREFOX

- The Firefox Web Browser is the faster, more secure, and fully customizable way to surf the web
- Mozilla is a global community dedicated to building free, open source products like the award winning Firefox web browser and Thunderbird email software.

Mozilla Firefox

# Web Browser

## 2. MOSIAC

- Mosaic was developed at the National Center for Supercomputing Applications(NCSA) at the University of Illinois Urbana-Champaign beginning in late 1992. NCSA released the browser in 1993, and officially discontinued development and support on January 7, 1997.
- Mosaic was also the first browser to display images inline with text instead of displaying images in a separate window

# Web Browser

## 3. NETSCAPE NAVIGATOR

- Netscape Navigator and Netscape are the names for the proprietary web browser popular in the 1990s
- It was the flagship product of the Netscape Communications Corporation and the dominant web browser in terms of usage share, although by 2002 its usage had almost disappeared

# Web Browser

## 4. WINDOWS INTERNET EXPLORER

- Windows Internet Explorer (formerly Microsoft Internet Explorer), is a series of graphical web browsers developed by Microsoft and included as part of the Microsoft Windows line of operating systems starting in 1995

## 4. OPERA

- Opera is a web browser and Internet suite developed by Opera Software.
- The browser handles common Internet-related tasks such as displaying web sites, sending and receiving e-mail messages, managing contacts, chatting on IRC

# Web Browser

- The browser handles common Internet-related tasks such as displaying web sites, sending and receiving e-mail messages, managing contacts, chatting on IRC, downloading files via BitTorrent, and reading web feeds. Opera is offered free of charge for personal computers and mobile phones.

## 6. SAFARI

- Safari is a graphical web browser developed by Apple and included as part of the Mac OS X operating system.
- On the company's Mac OS X operating system, it became Apple's default browser

# Web Browser

## 7. GOOGLE CHROME

- Google Chrome is a web browser developed by Google that uses the WebKit layout engine and application framework
- It was first released as a beta version for Microsoft Windows on 2 September 2008, and the public stable release was on 11 December 2008.

# Web Browser

## 8. Mobile Browsers

- Mobile Browsers A mobile browser, also called a micro browser, minibrowser or wireless internet browser(WIB), is a web browser designed for use on a mobile device such as a mobile phone or PDA.
- Opera Mini, offered free of charge, is designed primarily for mobile phones, but also for smartphones and personal digital assistants.

# **Introduction to HTML**



# Definitions

- **WWW** – World Wide Web.
- **HTML** – HyperText Markup Language – The Language of Web Pages on the World Wide Web.
- HTML is a text formatting language.
- **URL** – Uniform Resource Locator.
- **Browser** – A software program which is used to show web pages.
-

# Introduction

- “Normal text” surrounded by bracketed tags that tell browsers how to display web pages
- Pages end with “`.htm`” or “`.html`”
- HTML Editor – A word processor that has been specialized to make the writing of HTML documents more effortless.

# Introduction

- HTML stands for Hypertext Markup Language, and it is the most widely used language to write Web Pages.
- As its name suggests, HTML is a markup language.
- **Hypertext** refers to the way in which Web pages (HTML documents) are linked together. When you click a link in a Web page, you are using hypertext.

# Introduction

- Markup Language describes how HTML works.
- With a markup language, you simply "mark up" a text document with tags that tell a Web browser how to structure it to display.

# Structure of HTML

- The basic structure for all HTML documents is simple and should include the following minimum elements or tags:
- **<html>** - The main container for HTML pages
- **<head>** - The container for page header information
- **<title>** - The title of the page
- **<body>** - The main body of the page

# Basic Structure Document

- <HTML>
- <HEAD>
- <TITLE>MES College Marampally</TITLE>
- </HEAD>
- <BODY>
- This is what is displayed.
- </BODY>
- </HTML>
-

# Tags

- Codes enclosed in brackets
- Usually paired
  - <TITLE>My Web Page</TITLE>
- Not case sensitive
  - <TITLE> = <title> = <TITLE>

# **<!DOCTYPE> Declaration**

- The <!DOCTYPE> declaration tag is used by the web browser to understand the version of the HTML used in the document. Current version of HTML is 5 and it makes use of the following declaration –
- <!DOCTYPE html>There are many other declaration types which can be used in HTML document depending on what version of HTML is being used.

# The <html> Element

- The <html> element is the containing element for the whole HTML document. Each HTML document should have one <html> and each document should end with a closing </html> tag.
- Following two elements appear as direct children of an <html> element:
  - <head>
  - <body>
- As such, start and end HTML tags enclose all the other HTML tags you use to describe the Web page.

# The <head> Element

- The <head> element is just a container for all other header elements. It should be the first thing to appear after the opening <html> tag.
- Each <head> element should contain a <title> element indicating the title of the document.

# The <title> Element

- You should specify a title for every page that you write inside the <title> element. This element is a child of the <head> element). It is used in several ways:
  1. It displays at the very top of a browser window.
  2. It is used as the default name for a bookmark in browsers such as IE and Netscape.

# The <title> Element

3. It is used by search engines that use its content to help index pages.
- Therefore it is important to use a title that really describes the content of your site. The <title> element should contain only the text for the title and it may not contain any other elements.
- Example:
- Here is the example of using title tag.
- `<head><title>HTML Basic tags</title> </head>`

# The <body> Element

- The <body> element appears after the <head> element and contains the part of the Web page that you actually see in the main browser window, which is sometimes referred to as body content.
- A <body> element may contain anything from a couple of paragraphs under a heading to more complicated layouts containing forms and tables.
- Most of what you will be learning in this and the following five chapters will be written between the opening <body> tag and closing </body> tag.

# The <body> Element

- Example:
- Here is the example of using body tag.
- `<body> <p>This is a paragraph tag.</p></body>`

# Attributes

- Attributes are another important part of HTML markup.
- An attribute is used to define the characteristics of an element and is placed inside the element's opening tag.
- All attributes are made up of two parts: a name and a value:
- The name is the property you want to set.
- For example, the <font> element in the example carries an attribute whose name is face, which you can use to indicate which typeface you want the text to appear in.

# Attributes

- The **value** is what you want the value of the property to be.
- The first example was supposed to use the Arial typeface, so the value of the face attribute is Arial.
- you can see that a color for the text has been specified as well as the typeface in this **<font>** element:
- **<font face="arial" color="#CC0000">**

# Generic Attributes

Attribute	Options	Function
align	right, left, center	Horizontally aligns tags
valign	top, middle, bottom	Vertically aligns tags within an HTML element.
bgcolor	numeric, hexidecimal, RGB values	Places a background color behind an element
background	URL	Places an background image behind an element

# Generic Attributes

Attribute	Options	Function
id	User Defined	Names an element for use with Cascading Style Sheets.
class	User Defined	Classifies an element for use with Cascading Style Sheets.
width	Numeric Value	Specifies the width of tables, images, or table cells.
height	Numeric Value	Specifies the height of tables, images, or table cells.
title	User Defined	"Pop-up" title for your elements.

# 16 Basic Colors

Color Name	RGB Triplet	Hexadecimal	Color Name	RGB Triplet	Hexadecimal
Aqua	(0,255,255)	00FFFF	Navy	(0,0,128)	000080
Black	(0,0,0)	000000	Olive	(128,128,0)	808000
Blue	(0,0,255)	0000FF	Purple	(128,0,128)	800080
Fuchsia	(255,0,255)	FF00FF	Red	(255,0,0)	FF0000
Gray	(128,128,128)	808080	Silver	(192,192,192)	C0C0C0
Green	(0,128,0)	008000	Teal	(0,128,128)	008080
Lime	(0,255,0)	00FF00	White	(255,255,255)	FFFFFF
Maroon	(128,0,0)	800000	Yellow	(255,255,0)	FFFF00

# Color Codes

- WHITE • #FFFFFF
- BLACK • #000000
- RED • #FF0000
- GREEN • #00FF00
- BLUE • #0000FF
- MAGENTA • #FF00FF
- CYAN • #00FFFF
- YELLOW • #FFFF00
- AQUAMARINE • #70DB93
- BAKER'S CHOCOLATE • #5C3317
- VIOLET • #9F5F9F
- BRASS • #B5A642
- COPPER • #B87333

# 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.
- The following example will display a document with a white background color:

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

# **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. The TEXT attribute would be added as follows:
- **<BODY BGCOLOR="#FFFFFF" TEXT="#FF0000"></BODY>**
- In this example the document's page color is white and the text would be red.

# Headings, <Hx> </Hx>

- Inside the BODY element, heading elements H1 through H6 are generally used for major divisions of the document.
- Headings are permitted to appear in any order, but you will obtain the best results when your documents are displayed in a browser if you follow these guidelines:
- H1: should be used as the highest level of heading, H2 as the next highest, and so forth.

# Headings, <Hx> </Hx>

- <HTML>
  - <HEAD><TITLE> Example page</TITLE>
  - </HEAD>
  - <BODY>
  - <H1> Heading 1 </H1>
  - <H2> Heading 2 </H2>
  - <H3> Heading 3 </H3>
  - <H4> Heading 4 </H4>
  - <H5> Heading 5 </H5>
  - <H6> Heading 6 </H6>
  - </BODY>
  - </HTML>
- Heading 1
  - Heading 2
  - Heading 3
  - Heading 4
  - Heading 5
  - Heading 6

# Paragraphs, <P> </P>

- Paragraphs allow you to add text to a document in such a way that it will automatically adjust the end of line to suite the window size of the browser in which it is being displayed.
- Each line of text will stretch the entire length of the window.

# Paragraphs, <P> </P>

- <HTML><HEAD>
- <TITLE> Example Page</TITLE>
- </HEAD>
- <BODY><H1> Heading 1 </H1>
- <P> Paragraph 1, ....</P>
- <H2> Heading 2 </H2>
- <P> Paragraph 2, ....</P>
- <H3> Heading 3 </H3>
- <P> Paragraph 3, ....</P>
- <H4> Heading 4 </H4>
- <P> Paragraph 4, ....</P>
- <H5> Heading 5 </H5>
- <P> Paragraph 5, ....</P>
- <H6> Heading 6</H6>
- <P> Paragraph 6, ....</P>
- </BODY></HTML>
- Heading 1
- Paragraph 1,....
- Heading 2
- Paragraph 2,....
- Heading 3
- Paragraph 3,....
- Heading 4
- Paragraph 4,....
- Heading 5
- Paragraph 5,....
- Heading 6
- Paragraph 6,....

# Paragraphs, <P> </P>

- <p align="left">This is left aligned.</p>  
<p align="center">This is center aligned.</p>  
<p align="right">This is right aligned.</p>  
<p align="justify">This is justified. This works  
when you have multiple lines in your paragraph  
and you want to justify all the lines so that they  
can look more nice.</p>

# Break, <BR>

- Line breaks allow you to decide where the text will break on a line or continue to the end of the window.
- A <BR> is an empty Element, meaning that it may contain attributes but it does not contain content.
- The <BR> element does not have a closing tag.

# Break, <BR>

- <HTML>
- <HEAD>
- <TITLE> Example  
Page</TITLE>
- </HEAD>
- <BODY>
- <H1> Heading 1 </H1>
- <P>Paragraph 1, <BR>
- Line 2 <BR> Line 3  
<BR>....
- </P>
- </BODY>
- </HTML>
- Heading 1
- Paragraph 1,....
- Line 2
- Line 3
- ....

# Horizontal Rule, <HR>

- The <HR> element causes the browser to display a horizontal line (rule) in your document.
- <HR> does not use a closing tag, </HR>.
- Horizontal rules are used to visually break up sections of a document.
- The <hr> tag creates a line from the current position in the document to the right margin and breaks the line accordingly.
- For example
- <p>This is paragraph one and should be on top</p> <hr> <p>This is paragraph two and should be at bottom</p>

# Horizontal Rule, <HR>

Attribute	Description	Default Value
SIZE	Height of the rule in pixels	2 pixels
WIDTH	Width of the rule in pixels or percentage of screen width	100%
NOSHADE	Draw the rule with a flat look instead of a 3D look	Not set (3D look)
ALIGN	Aligns the line (Left, Center, Right)	Center
COLOR	Sets a color for the rule (IE 3.0 or later)	Not set

# Horizontal Rule, <HR>

- <HTML>
  - <HEAD>
  - <TITLE> Example  
Page</TITLE>
  - </HEAD>
  - <BODY>
  - <H1> Heading 1 </H1>
  - <P>Paragraph 1, <BR>
  - Line 2 <BR>
  - <HR>Line 3 <BR>
  - </P>
  - </BODY>
  - </HTML>
- Heading 1
  - Paragraph 1,....
  - Line 2
  - \_\_\_\_\_
  - \_\_\_\_\_
  - Line 3

# Character Formatting

- The <center> Element
- You can use <center> tag to put any content in the center of the page or any table cell.
- Example:
- <p>This is not in the center.</p> <center><p>This is in the center.</p> </center>

# Character Formatting

- Nonbreaking Spaces:
- Suppose you were to use the phrase "12 Angry Men." Here you would not want a browser to split the "12" and "Angry" across two lines:
- A good example of this technique appears in the movie "12 Angry Men."
- In cases where you do not want the client browser to break text, you should use a **nonbreaking space entity** (&nbsp;) instead of a normal space. For example, when coding the "12 Angry Men" paragraph, you would use something similar to the following code:
- <p>A good example of this technique appears in the movie "12&nbsp;Angry&nbsp;Men."</p>

# Character Formatting

- Preserve Formatting - The `<pre>` Element:
- Sometimes a text to follow the exact format of how it is written in the HTML document. In those cases, use the preformatted tag (`<pre>`).
- Any text between the opening `<pre>` tag and the closing `</pre>` tag will preserve the formatting of the source document.
- `<pre>` function testFunction( strText )
  - { alert (strText) }
  - `</pre>`

# Presentational Tags

- If you use a word processor, you are familiar with the ability to make text **bold**, **italicized**, or **underlined**; these are just three of the ten options available to indicate how text can appear in HTML.
- Bold Text - **The `<b>` Element:**
- Anything that appears in a `<b>...</b>` element is displayed in bold, like the word bold here:
- `<p>The following word uses a <b>bold</b> typeface.</p>`
- This will produce following result:
- The following word uses a **bold** typeface.

# Presentational Tags

- Italic Text - The `<i>` Element:
- Anything that appears in a `<i>...</i>` element is displayed in italicized, like the word italicized here:
- `<p>The following word uses a <i>italicized</i> typeface.</p>`
- This will produce following result:
- The following word uses a *italicized* typeface.

# Presentational Tags

- Underlined Text - The `<u>` Element:
- Anything that appears in a `<u>...</u>` element is displayed with underline, like the word underlined here:
- `<p>The following word uses a <u>underlined</u> typeface.</p>`
- This will produce following result:
- The following word uses a underlined typeface.

# Presentational Tags

- Strike Text - The `<strike>` Element:
- Anything that appears in a `<strike>...</strike>` element is displayed with strikethrough, which is a thin line through the text:
- `<p>The following word uses a <strike>strikethrough</strike> typeface.</p>`
- This will produce following result:
- The following word uses a ~~strikethrough~~ typeface.

# Presentational Tags

- Monospaced font - The `<tt>` Element:
- The content of a `<tt>` element is written in monospaced font.
- Most fonts are known as variable-width fonts because different letters are of different widths (for example, the letter m is wider than the letter i). In a monospaced font, however, each letter is the same width.
- `<p>The following word uses a <tt>monospaced</tt> typeface.</p>`
- This will produce following result:
- The following word uses a monospaced typeface

# Presentational Tags

- Superscript Text - The `<sup>` Element:
- The content of a `<sup>` element is written in superscript; the font size used is the same size as the characters surrounding it but is displayed half a characters height above the other characters.
- `<p>The following word uses a <sup>superscript</sup> typeface.</p>`
- This will produce following result:
- The following word uses a <sup>superscript</sup> typeface.

# Presentational Tags

- Subscript Text - The `<sub>` Element:
- The content of a `<sub>` element is written in subscript; the font size used is the same as the characters surrounding it, but is displayed half a character's height beneath the other characters.
- `<p>The following word uses a <sub>subscript</sub> typeface.</p>`
- This will produce following result:
- The following word uses a subscript typeface.

# Presentational Tags

- Larger Text - The `<big>` Element:
- The content of the `<big>` element is displayed one font size larger than the rest of the text surrounding it.
- `<p>The following word uses a <big>big</big> typeface.</p>`
- This will produce following result:
- The following word uses a **big** typeface.

# Presentational Tags

- Smaller Text - The `<small>` Element:
- The content of the `<small>` element is displayed one font size smaller than the rest of the text surrounding it.
- `<p>The following word uses a <small>small</small> typeface.</p>`
- This will produce following result:
- The following word uses a small typeface.

# Phrase Elements

- The following elements are not used as widely. As the element names indicate, they are designed to describe their content:
  - <em> and <strong> for emphasis
  - <blockquote>, <cite>, and <q> for quotations and citations
  - <abbr>, <acronym>, and <dfn> for abbreviations, acronyms, and key terms
  - <code>, <kbd>, <var>, and <samp> for computer code and information
  - <address> for addresses

# Phrase Elements

- Emphasized Text - The `<em>` Element:
- The content of an `<em>` element is intended to be a point of emphasis in the document, and it is usually displayed in italicized text. The kind of emphasis intended is on words such as "must" in the following sentence:
- `<p>You <em>must</em> remember to close elements in HTML.</p>`
- This will produce following result:
- You *must* remember to close elements in HTML.

# Phrase Elements

- Strong Text - The `<strong>` Element:
- The `<strong>` element is intended to show strong emphasis for its content; stronger emphasis than the `<em>` element. As with the `<em>` element, the `<strong>` element should be used only when you want to add strong emphasis to part of a document.
- `<p>You <strong>must</strong> remember to close elements in HTML.</p>`
- This will produce following result:
- You **must** remember to close elements in XHTML.

# Phrase Elements

- Text Abbreviation - The `<abbr>` Element
- You can indicate when you are using an abbreviated form by placing the abbreviation between opening `<abbr>` and closing `</abbr>` tags.
- `<p>I have a Friend called <abbr title = "Abhishek" > Abhy</abbr>.</p>`
- This will produce following result:
- I have a friend called **Abhy**.
-

# Phrase Elements

- Using Acronym - The `<acronym>` Element
- The `<acronym>` element allows to indicate that the text between an opening `<acronym>` and closing `</acronym>` element is an acronym.
- When possible use a title attribute whose value is the full version of the acronyms on the `<acronym>` element.
- `<p>This chapter covers marking up text in <acronym title="Hypertext Markup Language"> HTML </acronym>.</p>`
- This will produce following result:
- This chapter covers marking up text in **HTML**.

# Phrase Elements

- The `<dfn>` Element :
- The `<dfn>` element allows to specify a special term.
- Its use is similar to the words that are in italics in the midst of paragraphs.
- Most recent browsers render the content of a `<dfn>` element in an italic font.
- `<p>This tutorial teaches you how mark up your documents for the web using <dfn> HTML </dfn></p>`
- This will produce following result:
- This tutorial teaches you how mark up your<sup>51</sup> documents for the web using *HTML*.

# Phrase Elements

- The `<blockquote>` Element :
- When to quote a passage from another source, you should use the `<blockquote>` element.
- Text inside a `<blockquote>` element is usually indented from the left and right edges of the surrounding text, and sometimes uses an italicized font.
- `<p>`The following description of HTML is taken from the W3C Web site:`</p> <blockquote>` HTML 1.0 is the W3C's first Recommendation for XHTML, `</blockquote>`
- This will produce following result:
- The following description of HTML is taken from the W3C Web site:
- HTML 1.0 is the W3C's first Recommendation for XHTML 52

# Phrase Elements

- Short Quotations - The `<q>` Element :
- The `<q>` element is intended to be used to add a quote within a sentence rather than as an indented block on its own.
- `<p>Amit is in Spain, <q>He is their at my home. I think I am wrong</q>.</p>`
- This will produce following result:
- Amit is in Spain, "He is their at my home. I think I am wrong."

# Phrase Elements

- Citations - The `<cite>` Element :
- If you are quoting a text, you can indicate the source placing it between an opening `<cite>` tag and closing `</cite>` tag.
- The content of the `<cite>` element is rendered in italicized text by default.
- `<p>This HTML Tutorial is derived from <cite>World Wide Web Standard for HTML</cite>.</p>`
- This will produce following result:
- This HTML Tutorial is derived from *World Wide Web Standard for HTML*.

# Phrase Elements

- The `<code>` Element :
- Any code to appear on a Web page should be placed inside a `<code>` element. Usually the content of the `<code>` element is presented in a monospaced font, just like the code in most programming books.
- `<h1><code>This` is inside code element `</code></h1>`
- This will produce following result:
- This is inside code element

# Phrase Elements

- Keyboard Text - The `<kbd>` Element :
- When you are talking about computers, if you want to tell a reader to enter some text, you can use the `<kbd>` element to indicate what should be typed in, as in this example.
- The content of a `<kbd>` element is usually represented in a monospaced font rather like the content of the `<code>` element.
- `<h1> <kbd>This is inside kbd element</kbd></h1>`  
This will produce following result:
- This is inside kbd element

# Phrase Elements

- Programming Variables - The `<var>` Element :
- This element is usually used in conjunction with the `<pre>` and `<code>` elements to indicate that the content of that element is a variable that can be supplied by a user.
- `<p><code>document.write("<var>user-name</var>")</code></p>` This will produce following result:
- `document.write("user-name")`

# Phrase Elements

- Addresses - The <address> Element :
- The <address> element is used to contain any address. For example:
- <address>304, Menna Colony, Hyderabad - INDIA, 500032</address> This will produce following result:
- 304, Menna Colony, Hyderabad - INDIA, 500032

# Phrase Elements

- Program Output - The `<samp>` Element :
- The `<samp>` element indicates sample output from a program, script, or the like. Again, it is mainly used when documenting programming concepts. For example:
- `<p>Result produced by the program is <samp>Hello World</samp></p>` This will produce following result:
- Result produced by the program is Hello World

# Comments

- Comments are piece of code which is ignored by any web browser. It is good practice to comment your code, especially in complex documents, to indicate sections of a document, and any other notes to anyone looking at the code. Comments help you and others understand your code.
- HTML Comment lines are indicated by the special beginning tag `<!--` and ending tag `-->` placed at the beginning and end of EVERY line to be treated as a comment.

# Comments

- Comments do not nest, and the double-dash sequence "--" may not appear inside a comment except as part of the closing --> tag. You must also make sure that there are no spaces in the start-of-comment string.
- For example: Given line is a valid comment in HTML
- <!-- This is commented out -->

# Comments

- T Multiline Comments:
  - You have seen how to comment a single line in HTML. You can comment multiple lines by the special beginning tag `<!--` and ending tag `-->` placed before the first line and end of the lastline to be treated as a comment.
  - For example:
  - `<!--` -
    - This is a multiline comment `<br />` and can span through as many as lines you like.
- >

# HTML Fonts

- Font face and color depends entirely on the computer and browser that is being used to view your page. But the `<font>` tag is used to add **style, size, and color** to the text on the site.
- it can use a `<basefont>` tag to set all of your text to the same size, face, and color.
- The font tag is having three attributes called size, color, and face to customize your fonts.
- To change any of the font attributes at any time within your page, simply use the `<font>` tag.
- The text that follows will remain changed until you close with the `</font>` tag.

# HTML Fonts

- Font Size attribute:
- You can set the size of your font with size attribute. The range of accepted values is from 1(smallest) to 7(largest). The default size of a font is 3.
- Example:
- `<font size="1">Font size="1"</font>`
- `<font size="2">Font size="2"</font>`
- `<font size="3">Font size="3"</font>`
- `<font size="4">Font size="4"</font>`
- `<font size="5">Font size="5"</font>`
- `<font size="6">Font size="6"</font>`
- `<font size="7">Font size="7"</font>`

# HTML Fonts

- SPECIFY THE RELATIVE FONT SIZE.
- `<font size="+n">` or `<font size="-n">`:  
You can specify how many sizes larger or how many sizes smaller than the preset font size should be.
- Example:
- `<font size="-1">Font size="-1"</font>`
- `<font size="+1">Font size="+1"</font>`
- `<font size="+2">Font size="+2"</font>`
- `<font size="+3">Font size="+3"</font>`
- `<font size="+4">Font size="+4"</font>`

# HTML Fonts

- Font Face:
- You can set any font you like using face attribute but be aware that if the user viewing the page doesn't have the font installed, they will not be able to see it.
- Instead they will default to Times New Roman of your font with size attribute.
- Example:
- `<font face="Times New Roman" size="5">Times New Roman</font>`
- `<font face="Verdana" size="5">Verdana</font>`
- `<font face="Comic sans MS" size="5">Comic Sans MS</font>`
- `<font face="WildWest" size="5">WildWest</font>`
- `<font face="Bedrock" size="5">Bedrock</font>`

# **HTML Fonts**

- Specify alternate font faces:
- A visitor will only be able to see your font if they have that font installed on their computer.
- So, it is possible to specify two or more font face alternatives by listing the font face names, separated by a comma.
- Example:
- `<font face="arial,helvetica">`
- `<font face="Lucida Calligraphy,Comic Sans MS,Lucida Console>`
- When your page is loaded, their browser will display the first font face that it has available.
- If none of your selections are installed....then it will display the default font face Times New Roman.

# HTML Fonts

- Font Color:
- You can set any font color you like using color attribute.
- You can specify the color that you want by either the color name or hexadecimal code for that color.
- Example:
- `<font color="#FF00FF">This text is hexcolor #FF00FF</font>`
- `<font color="red">This text is red</font>`

# **HTML marquee**

- A HTML marquee is a scrolling piece of text displayed either horizontally across or vertically down your web site page depending on the settings. This is created by using HTML tag <marquees>.
- **Syntax:**
- A simple syntax to use marquee is as follows:
- <**marquee attribute\_name="attribute\_value"....more attributes**>
- One or more lines or text message or image  
**</marquee>**

# HTML marquee

- Attrubutes:
- A HTML marquee can have following attributes:
  1. **width**: how wide the marquee is. This will have a value like 10 or 20% etc.
  2. **height**: how tall the marquee is. This will have a value like 10 or 20% etc.
  3. **direction**: which direction the marquee should scroll. This will have value either up,down, left or right.
  4. **behavior**: what type of scrolling. This will have value scroll, slid and alternate.
  5. **scrolldelay**: how long to delay between each jump. This will have a value like 10 etc.

# HTML **marquee**

6. **scrollamount**: how far to jump. This will have a value like 10 etc.
7. **loop**: how many times to loop. The default value is INFINITE, which means that the marquee loops endlessly.
8. **bgcolor**: background color. This will have any color name or color hex value.
9. **hspace**: horizontal space around the marquee. This will have a value like 10 or 20%etc.
10. **vspace**: vertical space around the marquee. This will have a value like 10 or 20%etc.

# HTML Images

- Images are very important to beautify as well as to depicts many concepts on your web page. Its is true that one single image is worth than thousands of words.
- Insert Image - The `<img>` Element:
- You will insert any image in your web page by using `<img>` tag.
- Following is the simple syntax to use this tag.
- ``

# **HTML Images**

- <IMG> This element defines a graphic image on the page.
- Image File (SRC:source):
- This value will be a URL (location of the image) E.g. <http://www.domain.com/dir/file.ext> or </dir/file.txt>.

# HTML Images

- **Image Attributes:**
- Following are most frequently used attributes for <img> tag.
  1. **width:** sets width of the image. This will have a value like 10 or 20% etc.
  2. **height:** sets height of the image. This will have a value like 10 or 20% etc.
  3. **border:** sets a border around the image. This will have a value like 1 or 2 etc.
  4. **src:** specifies URL of the image file.
  5. **alt:** this is an alternate text which will be displayed if image is missing.

# HTML Images

6. **align**: this sets horizontal alignment of the image and takes value either left, right or center.
7. **valign**: this sets vertical alignment of the image and takes value either top, bottom or center.
8. **hspace**: horizontal space around the image. This will have a value like 10 or 20% etc.
9. **vspace**: vertical space around the image. This will have a value like 10 or 20% etc.
10. **name**: name of the image with in the document.
11. **id**: id of the image with in the document.

# **HTML Images**

12. **style**: this will be used if you are using CSS.
13. **stitle**: specifies a text title. The browser, perhaps flashing the title when the mouse passes over the link.

# HTML Images

- Some Examples on images
- 1) <IMG SRC="graden.gif" border=4>
  - 2) <IMG SRC="graden.gif " width="60" height="60">
  - 3) <IMG SRC="graden.gif " ALT="This is a text that goes with the image">
  - 4) <IMG SRC="graden.gif" Hspace="30"  
Vspace="10" border=20>
  - 5) <IMG SRC =" graden.gif" align="left">

# Lists

- In this chapter you will learn how to create a variety of lists.
- Objectives
- Upon completing this section, you should be able to
  - Unordered lists, which are like lists of bullet points
  - Ordered lists, which use a sequence of numbers or letters instead of bullet points
  - Definition lists, which allow you to specify a term and its definition
- Nest Lists.

# List Elements

- HTML supplies several list elements. Most list elements are composed of one or more <LI> (List Item) elements.
- **UL** : Unordered List. Items in this list start with a list mark such as a bullet. Browsers will usually change the list mark in nested lists.
- <UL>
- <LI> List item ...</LI>              List item ...
- <LI> List item ...</LI>              List item ...
- </UL>

# List Elements

- You have the choice of three bullet types: disc(default), circle, square.
- These are controlled in Netscape Navigator by the “TYPE” attribute for the <UL> element.
- <UL TYPE=“square”>
- <LI> List item ...</LI>
- <LI> List item ...</LI>
- <LI> List item ...</LI>
- </UL>

# List Elements

- OL: Ordered List. Items in this list are numbered automatically by the browser.
- <OL>
- <LI> List item One...</LI>
- <LI> List item Two ...</LI>
- <LI> List item Three...</LI>
- </OL>
- You have the choice of setting the TYPE Attribute to one of five numbering styles.

# List Elements

TYPE	Numbering Styles	
1	Arabic numbers	1,2,3, .....
a	Lower alpha	a, b, c, .....
A	Upper alpha	A, B, C, .....
i	Lower roman	i, ii, iii, .....
I	Upper roman	I, II, III, .....

# List Elements

- You can specify a starting number for an ordered list.
- <OL TYPE =“i”>
- <LI> List item ...</LI>
- <LI> List item ...</LI>
- </OL>
- <P> text ....</P>
- <OL TYPE=“i” START=“3”>
- <LI> List item ...</LI>
- </OL>

# List Elements

- **DL**: Definition List. This kind of list is different from the others. Each item in a DL consists of one or more Definition Terms (DT elements), followed by one or more Definition Description (DD elements).
- <DL>
- <DT> HTML </DT>
- <DD> Hyper Text Markup Language </DD>
- <DT> DOG </DT>
- <DD> A human's best friend!</DD>
- </DL>

# Nesting Lists

- You can nest lists by inserting a UL, OL, etc., inside a list item (LI).

- EXample

- <UL TYPE = “square”>
- <LI> List item ...</LI>
- <LI> List item ...
- <OL TYPE=“i” START=“3”>
- <LI> List item 1 ...</LI>
- <LI> List item 2...</LI>
- <LI> List item 3...</LI>
- <LI> List item 4...</LI>
- <LI> List item ...</LI>
- </OL>
- </LI>
- <LI> List item ...</LI>
- </UL>

# What will be the output?

```
<H1 ALIGN="CENTER">SAFETY TIPS FOR CANOEISTS</H1>
<OL TYPE="a" START="2">
<LI>Be able to swim </LI>
<LI>Wear a life jacket at all times </LI>
<LI>Don't stand up or move around. If canoe tips,
    <UL>
        <LI>Hang on to the canoe </LI>
        <LI>Use the canoe for support and </LI>
        <LI>Swim to shore
    </UL> </LI>
<LI>Don't overexert yourself </LI>
<LI>Use a bow light at night </LI>
</OL>
```

# The output....

## **SAFETY TIPS FOR CANOEISTS**

- b. Be able to swim
- c. Wear a life jacket at all times
- d. Don't stand up or move around. If canoe tips,
  - o Hang on to the canoe
  - o Use the canoe for support and
  - o Swim to shore
- e. Don't overexert yourself
- f. Use a bow light at night

# <H1 ALIGN="CENTER">SAFETY TIPS FOR CANOEISTS</H1>

<OL TYPE="a" START="2">

<LI>Be able to swim </LI>

<LI>Wear a life jacket at all times </LI>

<LI>Don't stand up or move around. If canoe tips,

<UL>

<LI>Hang on to the canoe </LI>

<LI>Use the canoe for support

<OL type="I" start="4">

<LI> Be careful </LI>

<LI> Do not look around</LI>

</LI> </OL>

<LI>Swim to shore

</UL> </LI>

<LI>Don't overexert yourself </LI>

<LI>Use a bow light at night </LI>

</OL>

What  
will  
be the  
output?

# The output....

## **SAFETY TIPS FOR CANOEISTS**

- b. Be able to swim
- c. Wear a life jacket at all times
- d. Don't stand up or move around. If canoe tips,
  - Hang on to the canoe
  - Use the canoe for support
- IV. Be careful
- V. Do not look around
  - Swim to shore
- e. Don't overexert yourself
- f. Use a bow light at night

# Tables

- Tables provide a means of organising the layout of data
- A table is divided into **rows** and **columns**: these specify the **cells** of the table
- Cells can contain text, images, links, other tables...
- Tables can also be used for organising the layout of the web page itself.

# Tables

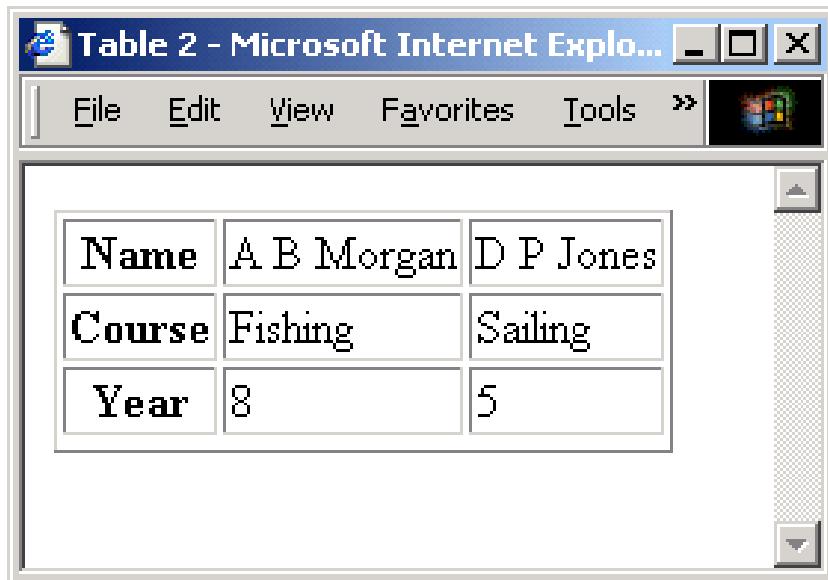
- <table> main element
- <tr> table row
- <th> table header
- <td> table data

Name	Course	Year
A B Morgan	Fishing	5
D P Jones	Sailing	8

```
<table border="1">
<tr>
<th>Name</th>
<th>Course</th>
<th>Year</th>
</tr>
<tr>
<td>A B Morgan</td>
<td>Fishing</td>
<td>5</td>
</tr>
<tr>
<td>D P Jones</td>
<td>Sailing</td>
<td>8</td>
</tr>
<tr>
</table>
```

# Tables

- <table> main element
- <tr> table row
- <th> table header
- <td> table data



A screenshot of Microsoft Internet Explorer window titled "Table 2 - Microsoft Internet Expl...". The menu bar includes File, Edit, View, Favorites, Tools, and a Help icon. The table has three rows and three columns:

Name	A B Morgan	D P Jones
Course	Fishing	Sailing
Year	8	5

```
<table border="1">
  <tr>
    <th>Name</th>
    <td>A B Morgan</td>
    <td>D P Jones</td>
  </tr>
  <tr>
    <th>Course</th>
    <td>Fishing</td>
    <td>Sailing</td>
  </tr>
  <tr>
    <th>Year</th>
    <td>8</td>
    <td>5</td>
  </tr>
  <tr>
  </table>
```

# Rows and Columns

- Cells can span multiple columns and multiple rows with the **colspan** and **rowspan** attributes

A screenshot of Microsoft Internet Explorer window titled "Table 3 - Microsoft Internet ...". The menu bar includes File, Edit, View, Favorites, and Help. The table has three columns: Name, Course, and Year. The first row contains "A B" in the Name column, "Morgan" in the Course column, and "5" in the Year column. The second row contains "D J" in the Name column, "Jones" in the Course column, and "8" in the Year column. The "Course" and "Year" headers span two rows, while the "Name" header spans two columns.

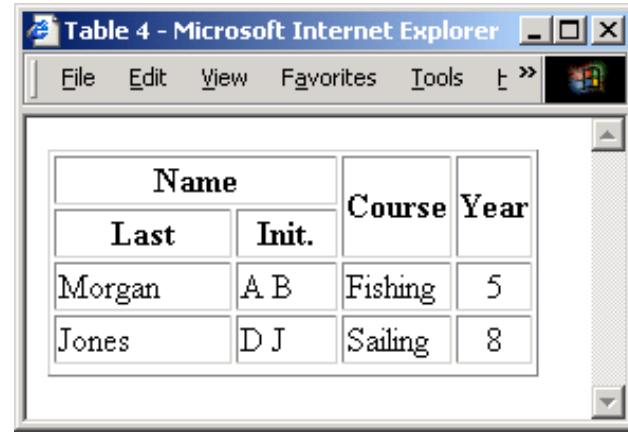
Name	Course	Year
A B	Morgan	5
D J	Jones	8
Fishing	5	Fishing
		8

```
<table border="1">
<tr>
  <th colspan="2">Name</th>
  <th>Course</th>
  <th>Year</th>
</tr>
<tr>
  <td>A B</td>
  <td>Morgan</td>
  <td rowspan="2">Fishing</td>
  <td>5</td>
</tr>
<tr>
  <td>D J</td>
  <td>Jones</td>
  <td>Sailing</td>
  <td>8</td>
</tr>
<tr>
</table>
```

# The align and width attributes

- The **align** attribute determines the position of the text within a cell
- The **width** attribute determines the width of the row relative to the table

```
<table border="1" align="center">
  <tr>
    <th colspan="2" width="60%">Name</th>
    <th rowspan="2">Course</th>
    <th rowspan="2">Year</th>
  </tr>
  <tr>
    <th>Last</th>
    <th>Init.</th>
  </tr>
  <tr>
    <td>Morgan</td>
    <td>AB</td>
    <td>Fishing</td>
    <td align="center">5</td>
  </tr>
  <!-- etc -->
```



Name		Course	Year
Last	Init.		
Morgan	A B	Fishing	5
Jones	D J	Sailing	8

# Table attributes

## Table attributes

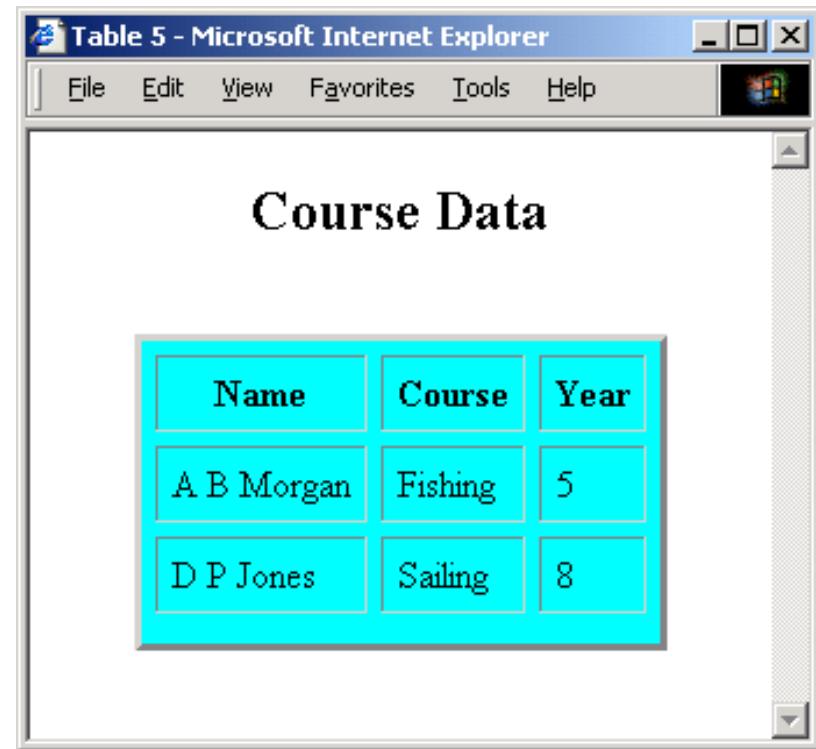
- **align** alignment relative to the page
- **width** in pixels or percentage of page width
- **border** - width of border (pixels)
- **cellspacing** separation between cells (pixels)
- **cellpadding** - space around data inside cell (pixels)
- **bgcolor** - background colour (inside cells)

Furthermore

- The **<caption>** element puts a title above the table

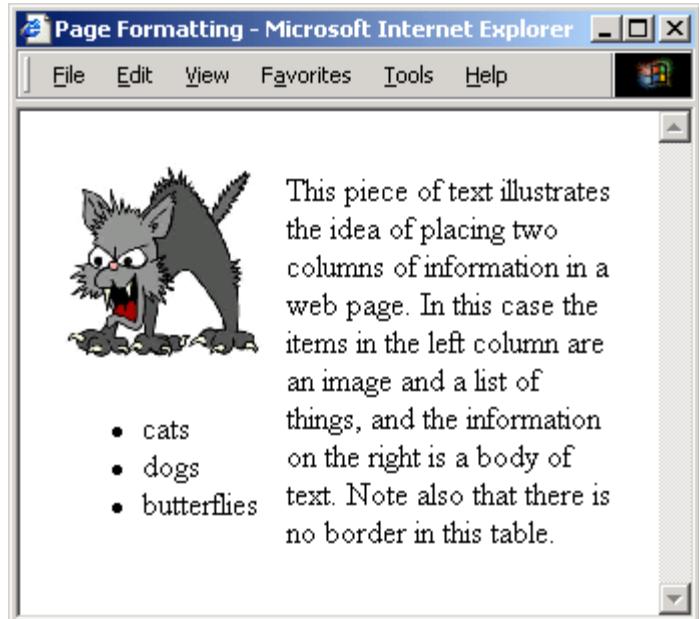
# Table attributes

```
<table border="3" align="center" cellspacing="6"
 cellpadding="6" bgcolor="cyan">
<caption>
  <h2>Course Data</h2>
</caption>
<tr>
  <th>Name</th>
  <th>Course</th>
  <th>Year</th>
</tr>
<tr>
  <td>A B Morgan</td>
  <td>Fishing</td>
  <td>5</td>
</tr>
<!-- etc -->
```



# Page formatting

- Tables can be used to organize the layout of the web page itself



```
</body>
<table border="0" cellspacing="10">
<tr>
<td>

<ul>
<li>cats</li>
<li>dogs</li>
<li>butterflies</li>
</ul>
</td>
<td>
This piece of text illustrates
the idea of placing two columns
of information in a web page...
Note also that there is no
border in this table.
</td>
</tr>
</table>
</body>
```

# **HTML Links**

- Web pages can contain links that take you directly to other pages and even specific parts of a given page.
- These links are known as **hyperlinks**.
- **Hyperlinks** allow visitors to navigate between Web sites by clicking on words, phrases, and images.

# **HTML Links**

- Linking Documents - The `<a>` Element:
- A link is specified using the `<a>` element.
- This element is called **anchor tag** as well.
- Anything between the opening `<a>` tag and the closing `</a>` tag becomes part of the link and a user can click that part to reach to the linked document.
- Following is the simple syntax to use this tag.
- `<a href="Document URL" attr_name = "attr_value"  
..more attributes />`

# More on LINKs

- <body      **LINK="#C0C0C0"**      **VLINK="#808080"**  
**ALINK ="#FF0000">**
- **LINK** - standard link - to a page the visitor hasn't been to yet. (standard color is blue - #0000FF).
- **VLINK** - visited link - to a page the visitor has been to before. (standard color is purple - #800080).
- **ALINK** - active link - the color of the link when the mouse is on it. (standard color is red - #FF0000).
- If the programmer what to change the color
- Click      <a      href="http://www.yahoo.com"><font  
color="FF00CC">here</font></a> to go to yahoo.

# **HTML Links**

- **Anchor Attributes:**
- Following are most frequently used attributes for <a> tag.
  1. **href:** specifies the URL of the target of a hyperlink. Its value is any valid document URL, absolute or relative, including a fragment identifier or a JavaScript code fragment.
  2. **target:** specify where to display the contents of a selected hyperlink.
    1. If set to "\_blank" then a new window will be opened to display the loaded page,

# **HTML Links**

- **Anchor Attributes:**
  - if set to "\_top" or "\_parent" then same window will be used to display the loaded document,
  - if set to "\_self" then loads the new page in current window.  
By default its "\_self".
- 3. **name & id:** attributes places a label within a document. When that label is used in a link to that document, it is the equivalent of telling the browser to goto that label.
- 4. **event:** attributes like onClick, onMouseOver etc. are used to trigger any Javascript or VBscript code.

# **HTML Links**

5. **title**: attribute lets you specify a title for the document to which you are linking. The value of the attribute is any string, enclosed in quotation marks. The browser might use it when displaying the link, perhaps flashing the title when the mouse passes over the link.
6. **accesskey**: attribute provides a keyboard shortcut that can be used to activate a link. For example, you could make the T key an access key so that when the user presses either the Alt or Ctrl key on his keyboard (depending on his operating system) along with the T key, the link gets activated

# **Internal Links**

- Internal Links : Links can also be created inside large documents to simplify navigation.
- Select some text at a place in the document that you would like to create a link to, then add an anchor to link to like this:
- `<A NAME="bookmark_name"></A>`
-

# Internal Links

- The Name attribute of an anchor element specifies a location in the document that we link to shortly. All NAME attributes in a document must be unique.
- Next select the text that you would like to create as a link to the location created above.
- `<A HREF="#bookmark_name">Go To Book  
Mark</A>`

# Special Characters & Symbols

- These Characters are recognized in HTML as they begin with an ampersand and end with with a semi-colon e.g. &value; The value will either be an entity name or a standard ASCII character number. They are called escape sequences.
- The next table represents some of the more commonly used special characters.

# Special Characters & Symbols

Special Character	Entity Name	Special Character	Entity Name
Ampersand	&amp; &	Greater-than sign	&gt; >
Asterisk	&lowast; **	Less-than sign	&lt; <
Cent sign	&cent; ¢	Non-breaking space	&nbs;
Copyright	&copy; ©	Quotation mark	&quot; "
Fraction one qtr	&frac14; 1/4	Registration mark	&reg; ®
Fraction one half	&frac12; 1/2	Trademark sign	&trade; TM

# Special Characters & Symbols

- Additional escape sequences support accented characters, such as:
- &ouml;
  - a lowercase o with an umlaut: ö
- &ntilde;
  - a lowercase n with a tilde: ñ
- &Egrave;
  - an uppercase E with a grave accent: È
- NOTE: Unlike the rest of HTML, the escape sequences are case sensitive. You cannot, for instance, use &LT; instead of &lt;.

# E-Mail (Electronic Mail)

- E.g. mailto:kmf@yahoo.com
- The type of service is identified as the mail client program. This type of link will launch the users mail client.
- The recipient of the message is kmf@yahoo.com
- <A HREF=“mailto:kmf@yahoo.com”>Send me
- More Information </A>

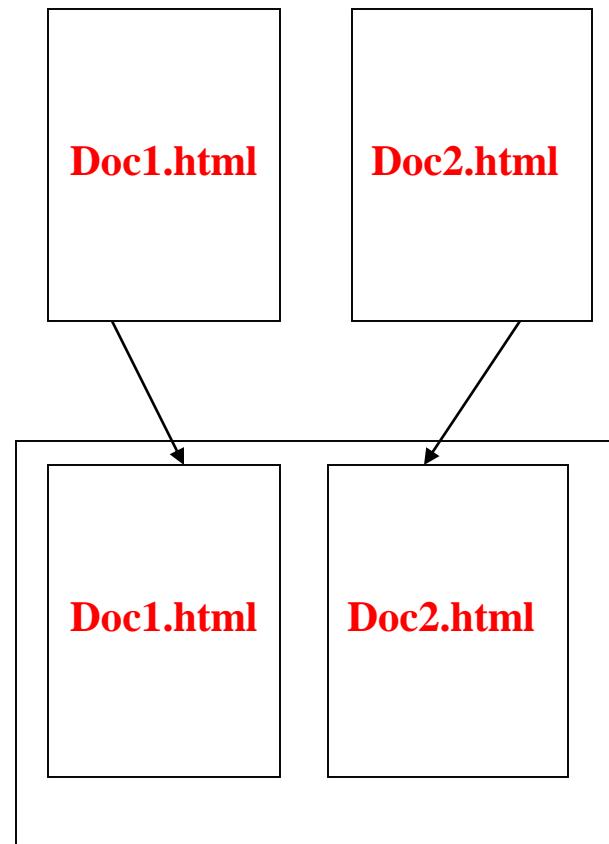
# Frames

- Frames are a relatively new addition to the HTML standard. First introduced in Netscape Navigator 2.0.
- Objectives:
- Upon completing this section, you should be able to:
- Create a Frame based page.
- Work with the Frameset, Frame, and Noframes elements.
- Use the attributes of the Frames elements to control the display.
- Set Targets appropriately.

# Frames

- A framed page is actually made up of multiple HTML pages. There is one HTML document that describes how to break up the single browser window into multiple windowpanes. Each windowpane is filled with an HTML document.
- For Example to make a framed page with a windowpane on the left and one on the right requires three HTML pages. Doc1.html and Doc2.html are the pages that contain content. Frames.html is the page that describes the division of the single browser window into two windowpanes.

# Frames



# Frame Page Architecture

- A <FRAMESET> element is placed in the html document before the <BODY> element. The <FRAMESET> describes the amount of screen real estate given to each windowpane by dividing the screen into ROWS or COLS.
- The <FRAMESET> will then contain <FRAME> elements, one per division of the browser window.
- **Note:** Because there is no BODY container, FRAMESET pages can't have background images and background colors associated with them.

# Frame Page Architecture

- <HTML>
- <HEAD>
- <TITLE> Framed Page </TITLE>
- <FRAMESET COLS=“23%,77%”>
- <FRAME SRC=“Doc1.html”>
- <FRAME SRC=“Doc2.html”>
- </FRAMESET >
- </HEAD>
- </HTML>

# Graphical view

**FRAMESET COLS="23%, 77%"**

**FRAME**

NAME= left\_pane  
SRC= Doc1.html

**FRAME**

NAME=right\_pane  
SRC= Doc2.html

# <FRAMESET> Container

- <FRAMESET> : The FRAMESET element creates divisions in the browser window in a single direction. This allows you to define divisions as either rows or columns.
- ROWS : Determines the size and number of rectangular rows within a <FRAMESET>. They are set from top of the display area to the bottom.
- Possible values are:
- Absolute pixel units, I.e. “360,120”.
- A percentage of screen height, e.g. “75%,25%”.
- Proportional values using the asterisk (\*). This is often combined with a value in pixels , e.g. “360,\*”.
- <Frameset cols=“200,20%,\*,2\*”>

# Creating a Frames Page

- **COLS:** Determines the size and number of rectangular columns within a <FRAMESET>. They are set from left to right of the display area.
- Possible values are:
- Absolute pixel units, I.e. “480,160”.
- A percentage of screen width, e.g. “75%,25%”.
- Proportional values using the asterisk (\*). This is often combined with a value in pixels , e.g. “480,\*”.

# Creating a Frames Page

- **FRAMEBORDER** : Possible values 0, 1, YES, NO. A setting of zero will create a borderless frame.
- **FRAMESPACING**: This attribute is specified in pixels. If you go to borderless frames you will need to set this value to zero as well, or you will have a gap between your frames where the border used to be.
- **BORDER** (thickness of the Frame): This attribute specified in pixels. A setting of zero will create a borderless frame. Default value is 5.
- **BORDERCOLOR**: This attribute is allows you choose a color for your border. This attribute is rarely used.

# <FRAME>

- <FRAME>: This element defines a single frame within a frameset. There will be a FRAME element for each division created by the FRAMESET element. This tag has the following attributes:
- SRC: Required, as it provides the URL for the page that will be displayed in the frame.
- NAME: Required for frames that will allow targeting by other HTML documents. Works in conjunction with the target attribute of the <A>, <AREA>, <BASE>, and <FORM> tags.

# **<FRAME>**

- **MARGINWIDTH**: Optional attribute stated in pixels. Determines horizontal space between the <FRAME> contents and the frame's borders.
- **MARGINHEIGHT**: Optional attribute stated in pixels. Determines vertical space between the <FRAME> contents and the frame's borders.
- **SCROLLING**: Displays a scroll bar(s) in the frame. Possible values are:
  - Yes – always display scroll bar(s).
  - No – never display scroll bar(s).
  - Auto – browser will decide based on frame contents.  
120
  - By default: scrolling is auto.

# <FRAME>

- **NORESIZE**: Optional – prevents viewers from resizing the frame. By default the user can stretch or shrink the frame's display by selecting the frame's border and moving it up, down, left, or right.

# <NOFRAMES>

- <NOFRAMES>: Frame – capable browsers ignore all HTML within this tag including the contents of the BODY element. This element does not have any attributes.
- <HTML>
- <HEAD>
- <TITLE> Framed Page </TITLE>
- </HEAD>

# <NOFRAMES>

- <FRAMESET COLS="23%,77%">
- <FRAME SRC="" NAME="left\_pane">
- <FRAME SRC="" NAME="right\_pane">
- <NOFRAMES>
- <P> This is a Framed Page. Upgrade your browser to support frames.</P>
- </NOFRAMES></FRAMESET>

# **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.
- Objectives:
- Upon completing this section, you should be able to
- Create a FORM.
- Add elements to a FORM.
- Define CGI (Common Gateway Interface).
- Describe the purpose of a CGI Application.
- Specify an action for the FORM.
- Forms work in all browsers.
- Forms are Platform Independent.

# 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.xnu.com/formtest.asp>
- <P> First Name: <INPUT TYPE=“TEXT” NAME=“fname” MAXLENGTH=“50”> </P>
- <P> <INPUT TYPE=“SUBMIT” NAME=“submit1” 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>**.

Name:

Sami Ali

Student No.:

123456789

Address:

Al al-Bayt University  
CIS Department  
Faculty of IT



City:

Amman



- Amman
- Irbid
- Karak

is foreign?



Male:



Female:



Submit

Reset

# 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.

# 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.

# Example on Text Box

- <TITLE>Form\_Text\_Type</TITLE>  
</HEAD> <BODY>
- <h1> <font color=blue>Please enter the following  
bioData</font></h1>
- <FORM name="fome1" Method= " get " Action= "  
URL " >
- First Name: <INPUT TYPE="TEXT"  
NAME="FName"
- SIZE="15" MAXLENGTH="25"><BR>
- Last Name: <INPUT TYPE="TEXT"  
NAME="Lname"

# **Example on Text Box**

- 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>

# Output

The screenshot shows a Microsoft Internet Explorer window with two tabs both titled "form\_text\_type". The active tab displays a form with the following content:

**please enter the following biodata**

first name:

last name:

nationality:

the phone number:

# 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.

# Password

- **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.

# 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="fome2" 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>

# Output



# 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.

# 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.

# 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>
India:<INPUT TYPE="CheckBox" Name="country"
CHECKED><BR>
```

# Check Box

- Srilanka<INPUT TYPE="CheckBox" Name="country"><BR>  
Qatar:<INPUT TYPE="CheckBox" Name="country"><BR><BR>  
<font color=blue>Select Language:</font><BR>  
Hindi:<INPUT TYPE="CheckBox" Name="lang" CHECKED><BR>  
English:<INPUT TYPE="CheckBox" Name="lang">
- Arabic:<INPUT TYPE="CheckBox" Name="lang"> <BR>
- </FORM>
- </BODY>
- </Html>

# Output

The screenshot shows a web browser window with the title bar "CheckB". The address bar displays the URL "file:///E:/HTML/Code/CheckBox.html". The main content area contains the following text and form elements:

**Please check one of the following**

Select Country:

India:

Srilanka

Qatar:

Select Language:

Hindi:

English:

Arabic:

# 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.

```
<HTML> <HEAD><TITLE>Check Box Type</TITLE> </HEAD>
<BODY>
<h1> <font color=green>Please check one of the
following</font></h1>
<FORM name="fome3" 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

The screenshot shows a web browser window with three tabs, all titled "CheckBoxTyp". The active tab displays the URL "file:///E:/HTML/Code/Radio.html". The page content includes a large green header "Please check one of the following" and two sections of form fields.

**Select Country:**

jordan:

Yemen:

Qatar:

**Select Language:**

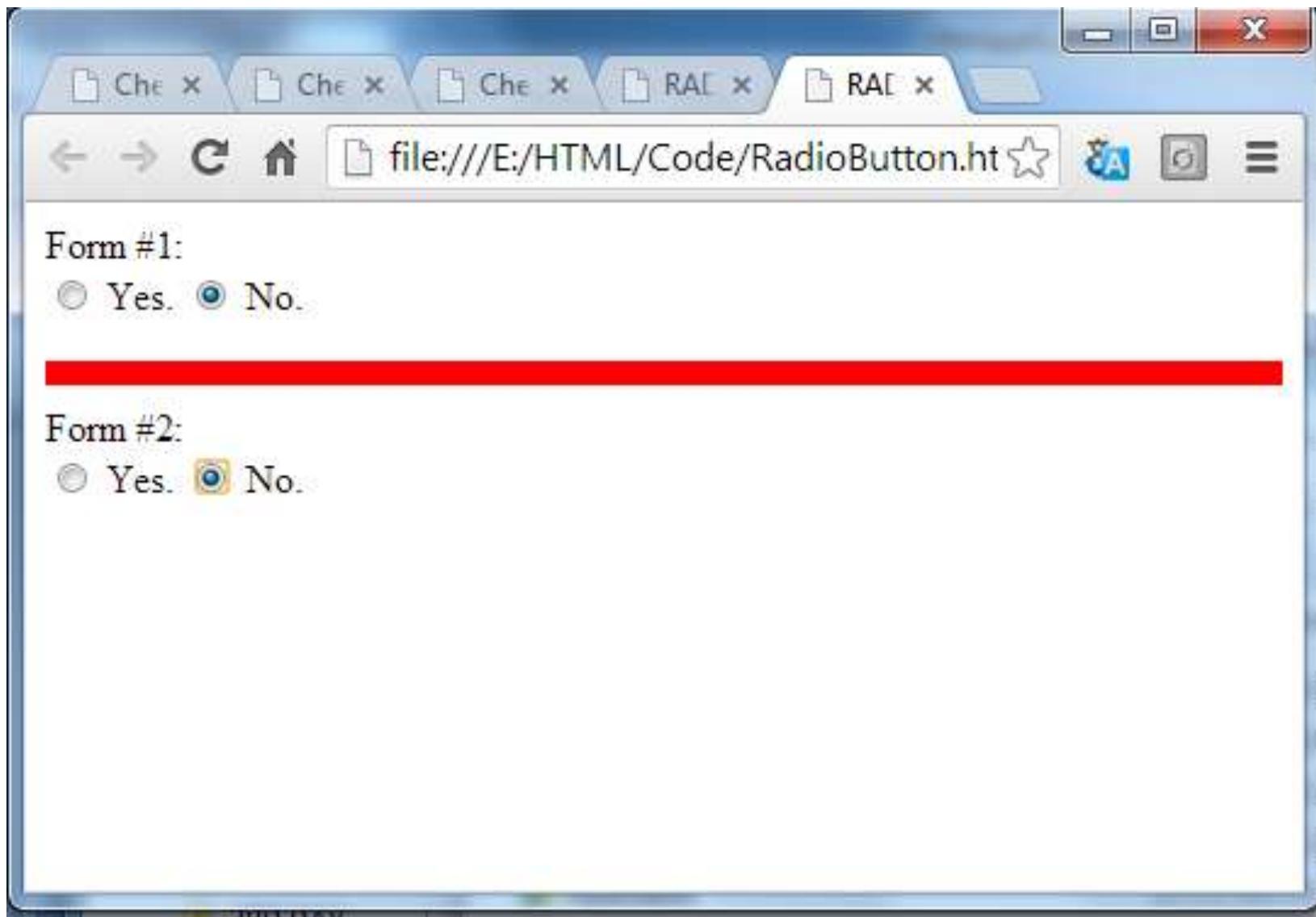
Arabic:

English:

French:

```
<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>
```

# Output



# 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.

# Push Button

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

# OUTPUT



# 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”> A dark grey rectangular button with the text "Submit Query" in white, centered horizontally within the button's width.
- 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.

```
<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="SubmitData " >
</FORM>
```

# OUTPUT

The screenshot shows a web browser window with two tabs both titled "BUTTON". The URL bar displays "file:///E:/HTML/Code/SUBMIT.HTML". The main content area contains the following form elements:

First Name:

Family Name:

Press Here to submit the data:

# 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.

# 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>

# Output

A screenshot of a web browser window titled "file:///E:/HTML/Code/R". The browser interface includes a toolbar with icons for back, forward, search, and file operations. Below the toolbar, there are two input fields: "First Name:" and "Family Name:", each followed by a text input box. Below these fields is a large red text instruction: "Press Here to submit the data:". Underneath this instruction are two buttons: "SubmitData" and "Reset".

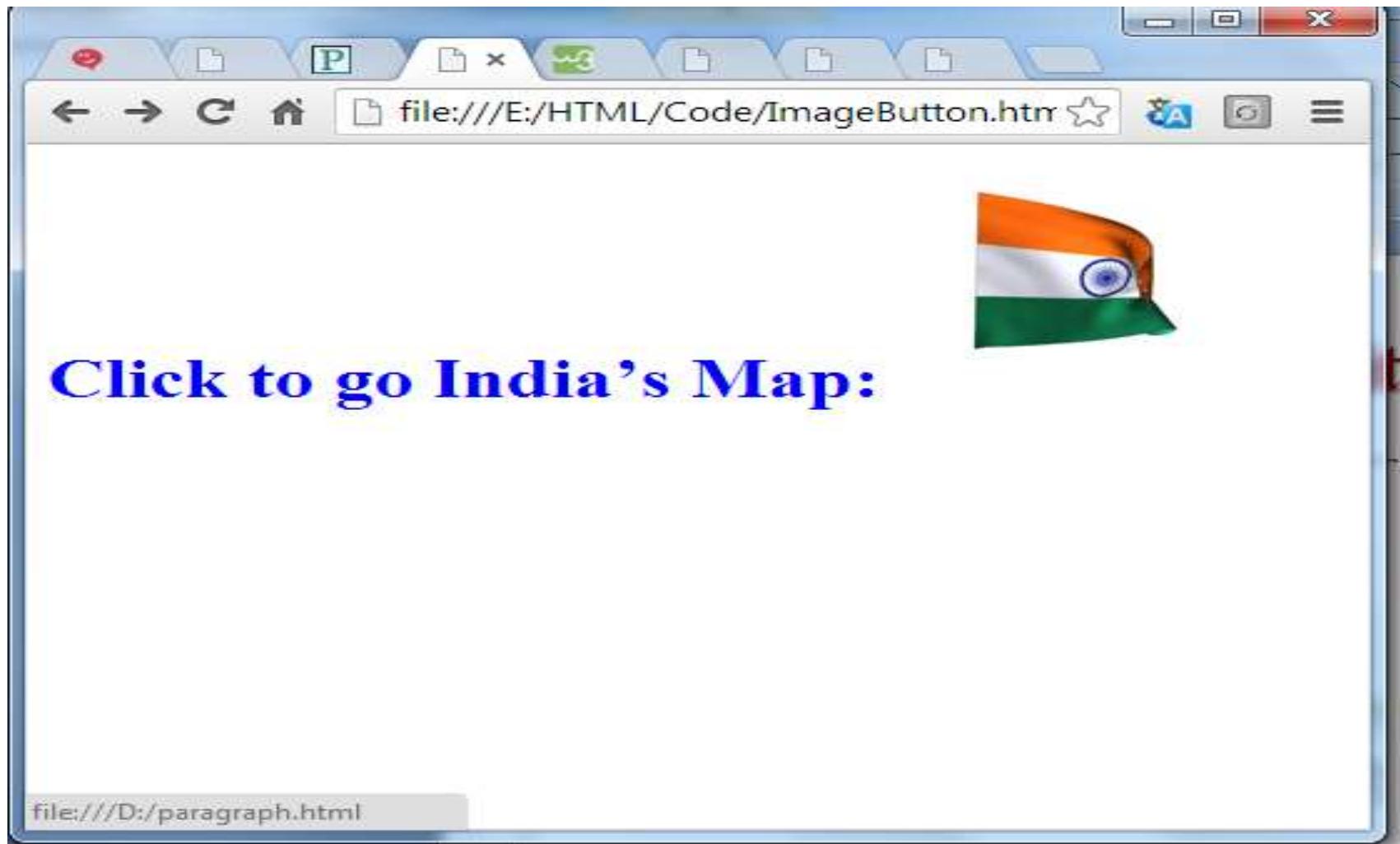
# Image Submit Button

- Image Submit Button: Allows you to substitute an image for the standard submit button.
- `<INPUT TYPE="IMAGE" SRC="jordan.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.

# Image Submit Button

- <form>  
<H1><font color=blue>  
Click to go Jordan's Map:  
<INPUT TYPE="IMAGE" SRC="India.gif">  
</form>

# Image Submit Button



# File

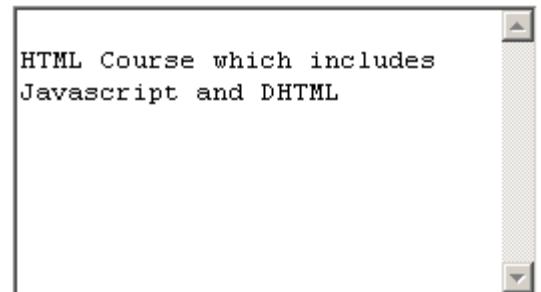
- File Upload: You can use a file upload to allow surfers to upload files to your web server.
- <INPUT TYPE=“FILE”>
- Browser will display
- File Upload has the following attributes:
- **TYPE**: file.
- **SIZE**: is the size of the text box in characters.
- **NAME**: is the name of the variable to be sent to the
- CGI application.
- **MAXLENGTH**: is the maximum size of the input in the textbox in characters.

# 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="SubmitFile">
- </form>
- </BODY>

# Other Elements used in Forms

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

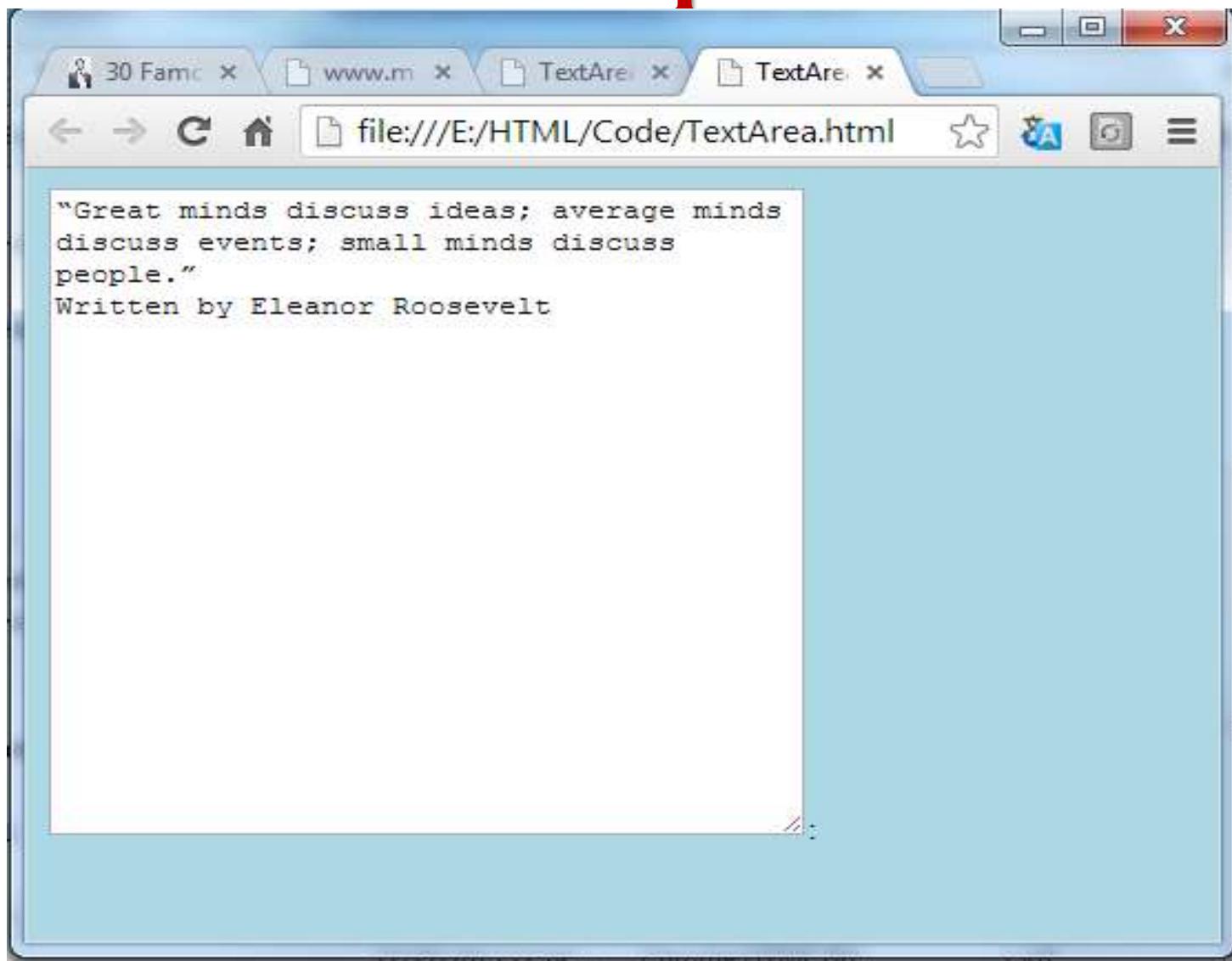


- Browser will display
- Textarea has the following attributes:
- **NAME**: is the name of the variable to be sent to the CGI application.
- **ROWS**: the number of rows to the textbox.
- **COLS**: the number of columns to the textbox.

# Other Elements used in Forms

- <BODY bgcolor=lightblue>
- <form>
- <TEXTAREA COLS=40 ROWS=20 Name="comments" >
- “Great minds discuss ideas; average minds discuss events; small minds discuss people.”
- Written by Eleanor Roosevelt
- </TEXTAREA>:
- </form>
- </BODY>

# Output



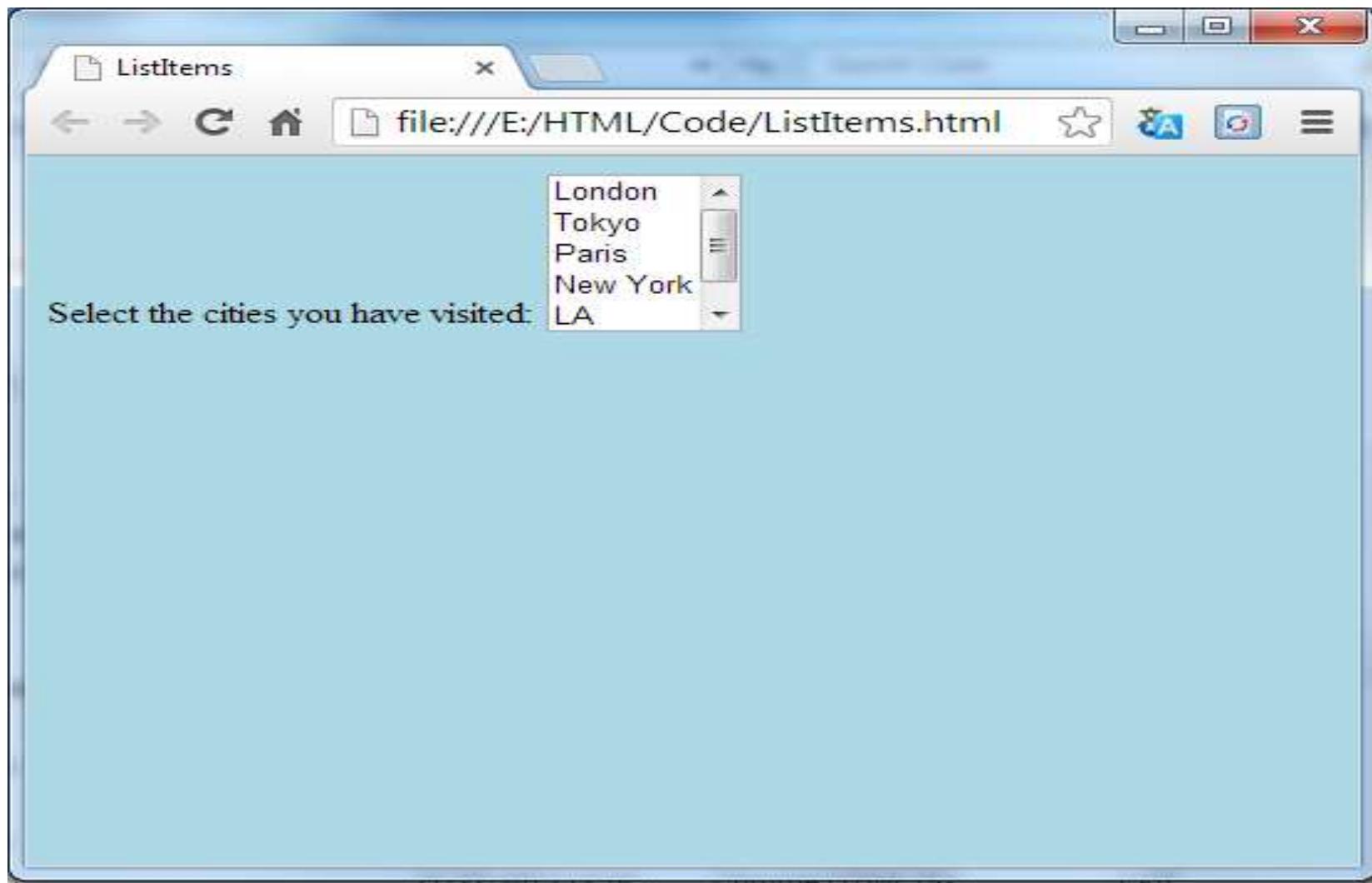
# Other Elements used in Forms

- 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.

# Other Elements used in Forms

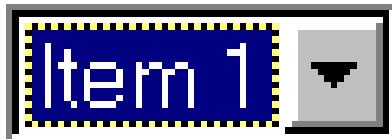
- <BODY bgcolor=lightblue>
- <form> Select the cities you have visited:
- <SELECT name="list" size=5>
- <option> London</option>
- <option> Tokyo</option>
- <option> Paris</option>
- <option> New York</option>
- <option> LA</option>
- <option> KL</option>
- </SELECT>
- </form> </BODY>

# Other Elements used in Forms



# Other Elements used in Forms

- Drop Down List:



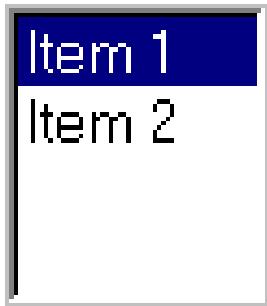
- **Name**: is the name of the variable to be sent to the CGI application.
- **Size**: 1.

# Other Elements used in Forms

- <body>
- <select>
- <option value="volvo">Volvo</option>
- <option value="saab">Saab</option>
- <option value="opel">Opel</option>
- <option value="audi">Audi</option>
- </select>
- 
- </body>

# 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.
-

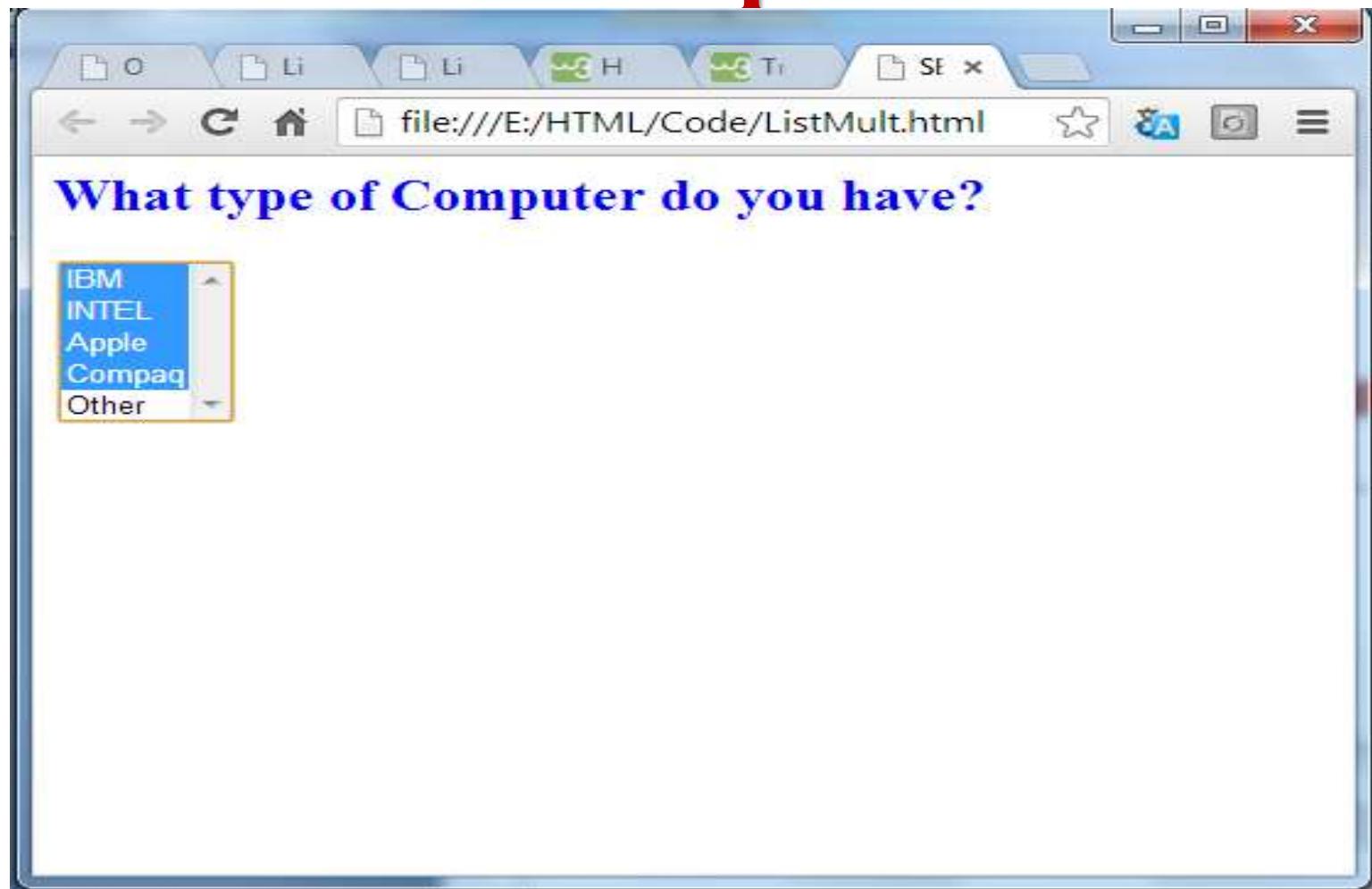
# Other Elements used in Forms

- <BODY>
- <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>

# Other Elements used in Forms

- <FORM>
- <SELECT NAME="ComputerType" size=5 multiple>
  - <OPTION value="IBM" > IBM</OPTION>
  - <OPTION value="INTEL"> INTEL</OPTION>
  - <OPTION value=" Apple"> Apple</OPTION>
  - <OPTION value="Compaq" SELECTED> Compaq</OPTION>
  - <OPTION value=" other"> Other</OPTION>
- </SELECT>
- </FORM>

# Output



There are eleven different types of form elements:

Button

Checkbox

FileUpload

Hidden

Password

Radio

Reset object

Select object

Submit object

Text

Textarea


# Cascading Style Sheets

- Cascading Style Sheets, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable.
- **CSS** handles the look and feel part of a web page.
- Using **CSS**, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, as well as a variety of other effects.
- **CSS** is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

# Understanding the Styles

- A **style** is a consistent, recognizable pattern
- A **CSS** style is a collection of one or more rules
- A **style sheet** is a collection of styles
- A rule is the combination of a **selector**, a **property**, and a **value**
- The selector identifies the element to which you are applying a style.
  - Element, type, class, and id selectors

# Advantages of CSS

- CSS saves time - it write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- Pages load faster - If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply to all the occurrences of that tag. So less code means faster download times.

# Advantages of CSS

- Easy maintenance - To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Superior styles to HTML - CSS has a much wider array of attributes than HTML so you can give far better look to your HTML page in comparison of HTML attributes.
- Multiple Device Compatibility - Style sheets allow content to be optimized for more than one type of device.

# Advantages of CSS

- By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- **Global web standards** - Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
-

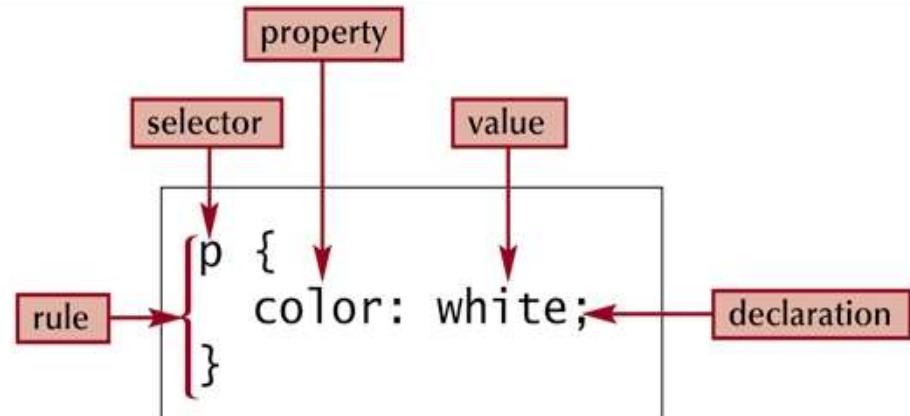
# Creating a CSS Style

- Type a right brace on its own line (but not indented) below the declaration list. The following code shows the complete syntax for a rule:

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

---

The components of a CSS style



# Creating a CSS Style

- **Selector:** A selector is an HTML tag at which style will be applied. This could be any tag like `<h1>` or `<table>` etc.
- **Property:** A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties.
- They could be color or border etc.
- **Value:** Values are assigned to properties.
- For example color property can have value either red or `#F1F1F1` etc.

# Creating a CSS Style

## ► The CSS terminology quick glossary

### CSS Terminology Quick Glossary

<b>CSS style</b>	A collection of one or more rules
<b>style sheet</b>	A collection of styles
<b>rule</b>	The combination of a selector and one or more properties and values
<b>selector</b>	What is being styled
<b>property</b>	How the selector will be modified
<b>value</b>	The manner or extent to which the property will be modified
<b>declaration</b>	A property and value pair
<b>declaration list</b>	All the declarations for the same selector

# CSS Units

- CSS supports a number of measurements including absolute units such as **inches**, **centimeters**, **points**, and so on, as well as relative measures such as percentages and em units.
- You need these values while specifying various measurements in your Style rules e.g border="1px solid red".

# CSS Units

Unit	Description	Example
in	Defines a measurement in inches.	p {word-spacing: .15in;}
mm	Defines a measurement in millimeters.	p {word-spacing: 15mm;}
pc	Defines a measurement in picas. A pica is equivalent to 12 points; thus, there are 6 picas per inch.	p {font-size: 20pc;}
pt	Defines a measurement in points. A point is defined as 1/72nd of an inch.	body {font-size: 18pt;}
px	Defines a measurement in screen pixels.	p {padding: 25px;}

# CSS Units

Unit	Description	Example
%	Defines a measurement as a percentage relative to another value, typically an enclosing element.	p {font-size: 16pt; line-height: 125%;}
cm	Defines a measurement in centimeters.	div {margin-bottom: 2cm;}
em	A relative measurement for the height of a font in em spaces. Because an em unit is equivalent to the size of a given font, if you assign a font to 12pt, each "em" unit would be 12pt; thus, 2em would be 24pt.	p {letter-spacing: 7em;}

# Style Precedence

## 1. Inline styles

- Add styles to each tag within the HTML file
- Use it when you need to format just a single section in a web page
- Example
  - `<h1 style="color:red; font-family: sans-serif">Hello</h1>`
- Here you can see that the properties are added as the value of the style attribute.
- There is no need for a selector here and there are no curly braces.
- To separate each property from its value with a colon and each of the property-value pairs from each other with a semicolon.

# Style Precedence

## 2. Embedded or internal styles

- A style is applied to the entire HTML file
- Use it when you need to modify all instances of particular element (e.g., h1) in a web page
- Example
  - <style>
    - h1 {color:red; font-family:sans-serif}
  - </style>

# Style Precedence

- <head>
  - <title>Embedded Example</title>
  - <style> (default is “text/css”)
    - Style declarations
    - </style>
  - </head>
- A style declaration:
  - Selector {attribute1:value1; attribute2:value2; ...}
  - Selector = an element in a document (e.g., a header or paragraph)

# **Example**

- <head>  
<title>Getting Started</title>  
<style type="text/css">  
    h1 {font-family: sans-serif; color: organge}  
</style>
- </head>

# Style Precedence

## 3. External style sheets

- An external style sheet is a text file containing the style definition (declaration)
- Use it when you need to control the style for an entire web site
- Example
  - `h1, h2, h3, h4, h5, h6 {color:red; font-family :sans-serif}`
  - Save this in a new document using a `.css` extension

# Creating an External Style Sheet

- Open a new blank document in Notepad
- Type style declarations
  - `h1 {color:red; font-family:sans-serif;}`
- Do not include `<style>` tags
- Save the document as `filename.css`
- Open an HTML file
- Between `<head>` and `</head>` add
  - `<link href= URL rel =“relation_type” type =“link_type”>`
- `URL` is the `.css`
- `Relation_type =“stylesheet”`
- `Link_type =“text/css”`
- Save this file and the `.css` file in the same web server directory

# An example of an external style sheet with an original html file

```
<head>
<title>Getting
    Started</title>
<link href="scraps.css"
      rel="stylesheet"
      type="text/css" />
</head>
```

html file

```
h1 {font-family: sans-serif;
    color: orange}
b {color: blue}
```

Text file of css named “scraps”

# The <style> Element

- The <style> element is used inside the <head> element to contain style sheet rules within a document, rather than linking to an external document.
- It is also sometimes used when a document needs to contain just a few extra rules that do not apply to the other documents that share the same style sheet.

# The <style> Element

- <head>

```
<style type="text/css">  
    h1 {color:#FF0000;}
```

```
</style>
```

```
</head>
```

# Advantages of External CSS Style Sheets

- There are several advantages to using external CSS style sheets rather than internal style sheets or inline style rules, including the following:
  1. The same style sheet can be reused by all of the web pages in your site.
  2. Because the style rules are written only once, rather than appearing on every element or in every document, the source documents are smaller.

This means that, once the CSS style sheet has been downloaded with the first document that uses it, subsequent documents will be quicker to download

# **Advantages of External CSS Style Sheets**

3. It is easy to change the appearance of several pages only by altering the style sheet rather than each individual pages.
4. The style sheet can act as a style template to help different authors achieve the same style of document without learning all of the individual style settings.
5. Because the source document does not contain the style rules, different style sheets can be attached to the same document.
6. A style sheet can import and use styles from other style sheets, making for modular development and good reuse.

# Selectors

- Selectors
- You should be starting to get the hang of writing rules in style sheets that indicate how an element should appear.
- You can create selectors that are a lot more specific. In addition to providing the element name as a selector, you can use the following as selectors.

# Selectors

- **Universal Selector**
- The universal selector is an asterisk; it is like a wildcard and matches all element types in the document.

`*{ }`

- If you want a rule to apply to all elements, you can use this selector.

# Selectors

- Sometimes it is used for default values, such as a `font-family` and `font-size`, that will apply to the whole of the document (unless another more specific selector indicates an element should use different values for these same properties).
- It is slightly different from applying default styles to the `<body>` element, as the universal selector applies to every element, and does not rely on the property being inherited from the rules that apply to the `<body>` element.

# Selectors

## 2. The Type Selector

- The **type selector** matches all of the elements specified in the comma-delimited list.
- It allows to apply the same rules to several elements. For example, the following would match all h1, h2, and p elements.

**h1, h2, p {}**

- If you have the same rules that apply to several elements, this technique can lead to a smaller style sheet, saving bandwidth and load on your server

# Selectors

## 3. The Class Selector

- The class selector allows to match a rule with an element carrying a class attribute whose value that specify in the class selector.
- For example, imagine that a `<p>` element with a class attribute whose value was `BackgroundNote`, like so:
- `<p class=“specialcolor”> This paragraph contains an aside.</p>`

# Selectors

- A class selector can be used in two ways
  - I. simply assign a rule that applies to any element that has a class attribute whose value is specialcolor, like so, simply preceding the value of the class attribute with a **period** or **full stop**:
  - **.specialcolor { }**
  - II. you can create a selector that selects only the <p> elements that carry a class attribute with a value of BackgroundNote (not other elements) like so:
  - **p. .specialcolor { }**

# Selectors

## 4. The ID Selector

- The **id selector** works just like a class selector, but works on the value of id attributes. But rather than using a period or full stop before the value of the id attribute, use a hash or pound sign (#).
- So, a `<p>` element with an id attribute whose value is abstract can be identified with this selector.
- `p#abstract`
- Because the value of an id attribute should be unique within a document, this selector should apply only to the content of one element.

# Selectors

## 5. The Child Selector

- The child selector matches an element that is a direct child of another.
- In this case it matches any `<p>` elements that are direct children of `<body>` elements:

```
body > p {  
    color: #000000;  
}
```

- This rule will render all the paragraphs in black if they are direct child of `<body>` element.
- Other paragraphs put inside other elements like `<div>` or `<td>` etc. would not have any effect of this rule.
- The less-than symbol (`>`) is referred to as a `combinator`.

# Selectors

## 6. The Descendent Selector

- The descendent selector matches an element type that is a descendent of another specified element, at any level of nesting, not just a direct child.
- While the less-than symbol was the **combinator** for the child selector, for the descendent selector the combinator is the space. Take a look at this example:

`table b {}`

- In this case, the selector matches any `<b>` element that is a child of the `<table>` element, which means it would apply to `<b>` elements both in `<td>` and `<th>` elements.
- This is a contrast to the child selector because it applies to all of the children of the `<table>` element, rather than just the direct children.

# Selectors

## 7. The Adjacent Sibling Selector

- An adjacent sibling selector matches an element type that is the next sibling of another.
- For example, if you want to make the first paragraph after any level 1 heading a different style you can use the adjacent sibling selector like so:
- `h1+p {}`
- Both elements must have the same element, and this will apply only to the `<p>` element directly after a heading.

# Selectors

- CSS File
- p {font-family:arial, verdana, sans-serif;}
- div>p {border:1px solid #000000;}
- p+p+p {background-color:#999999;}

# Selectors

- Html file
- <p>Here is an example of some adjacent sibling and child selectors.</p>
- <div>
- <p>One</p>
- <p>Two</p>
- <p>Three</p>
- <p>Four</p>
- <p>Five</p>
- </div>

# Selectors

The screenshot shows a web browser window titled "SelectType.html". The address bar displays the URL "file:///E:/HTML/Css/Code/selectors/SelectType.html". The main content area contains the following text:

Here is an example of some adjacent sibling and child selectors.

One

Two

Three

Four

Five

The words "One" and "Two" are displayed in white text on a white background. The words "Three", "Four", and "Five" are displayed in white text on a dark gray background.

# Selectors

- The three different paragraph styles are as follows:
  1. The first paragraph has no border or background color.
  2. The paragraphs inside the <div> element all have borders.
  3. The last three paragraphs have a gray background.

# Selectors

- Here not used three different classes to specify different paragraph styles; rather, only one rule that controls the font used for all paragraphs.
- Second rule for any paragraph that is a child of a <div> element. (Because the first paragraph is not inside a <div> element, the rule does not apply to the first paragraph.)
- The third rule matches any paragraph and is the third consecutive <p> element. (Because the fourth and fifth <p> elements have two previous <p> elements, this rule applies to them, too.)
- p+p+p {background-color:#999999;}

# CSS Properties

## 1. Controlling Fonts

- Several properties allow you to control the appearance of text in your documents. These can be split into two groups:
  - ❑ Those that directly affect the font and its appearance
  - ❑ Those that have other formatting effects upon the text

# 1. Fonts

Property	Purpose
Font	Allows you to combine several of the following properties into one
font-family	Specifies the family of font to be used (the user must have this installed on his or her computer)
font-size	Specifies the size of a font font-weight Specifies whether the font should be normal, bold, or bolder than the containing element

# 1. Fonts

font-style	Specifies whether the font should be normal, italic, or oblique (an oblique font is the normal font on a slant rather than a separate italic version of the font)
font-Stretch	Allows you to control the width of the actual letters in a font (not spaces between them)
font-variant	Specifies whether the font should be normal or small caps
font-size	-adjust Allows you to alter the aspect ratio of the size of characters of the font

# 1. Fonts

- A **typeface** is a family of fonts, such as the Arial family.
- A **font** is a specific member of that family, such as Arial 12-point bold.

## 1. The font-family Property

- The font-family property allows you to specify the typeface that should be used.
- The big drawback with this property is that those viewing the page must have this font on their computers; otherwise they will not see the page in that font.
- You can, however, specify more than one font so that, if the user does not have your first choice of font, the browser looks for the next font in the list

# 1. Fonts

- Example
- p.one {font-family:arial, verdana, sans-serif;}
- p.two {font-family:times, “times new roman”, serif;}
- p.three {font-family:courier, “courier new”, serif;}

## 2. The font-size Property

- The font-size property enables you to specify a size for the font. You can specify a value for this property in several ways:
  - Absolute size
  - Relative size
  - Length
  - Percentage (in relation to parent element)

# 1. Fonts

- The following values are absolute sizes:
  - xx-small x-small small medium large x-large xx-large
- The following two values are relative sizes:
  - smaller larger
- Length can be expressed in one of the following units of length:
  - px em ex pt in cm pc mm
- A percentage is calculated as a proportion of the element that contains the text:
  - 2% 10% 25% 50% 100%

# 1. Fonts

- For example:
- p.one {font-size:xx-small;}
- p.two {font-size:12px;}
- p.three {font-size:3pc;}
- p.four {font-size:10%;}

# 1. Fonts

## 3. The font-weight Property

- Most fonts have different variations, such as bold and italic.
- While many well-made fonts have completely different versions of each character for bold text, browsers tend to use an algorithm to calculate and add to the character's thickness when it is supposed to be bold.

# 1. Fonts

- The possible values for font-weight are:
- normal bold bolder lighter 100 200 300 400 500 600 700 800 900
- For Example
- p.one {font-weight:normal;}
- p.two {font-weight:bold;}
- p.three {font-weight:bolder;}
- p.four {font-weight:lighter;}
- p.five {font-weight:100;}
- p.six {font-weight:200;}

# 1. Fonts

## 4.The font-style Property

- The font-style property allows you to specify that a font should be normal, italic, or oblique, and these are the values of the font-style property; for example:
- p.one {font-style:normal;}
- p.two {font-style:italic;}
- p.three {font-style:oblique;}

# 1. Fonts

## 5. The font-variant Property

- There are two possible values for the font-variant property: normal and small-caps.
- A small caps font looks like a smaller version of the uppercase letterset.
- For example,
- <p>This is a normal font, but then <span class="smallcaps">there are some small caps</span> in the middle.</p>
- Now look at the style sheet:
- p {font-variant:normal;}
- span.smallcaps {font-variant:small-caps;}

# 1. Fonts

## 6. The font-stretch Property

- The font-stretch property sets the width of the actual letters in a font (not the space between them).
- It can take either relative or fixed values. The relative values are as follows:
  - normal wider narrower
- The fixed values are as follows:
  - ultra-condensed extra-condensed condensed semi-condensed semi-expanded expanded extra-expanded ultra-expanded
- For example, you can make a condensed Arial font using the following syntax:
  - `p {font-family:arial; font-stretch:condensed;}`

# Text Formatting

Property	Purpose
Color	Specifies the color of the text
text-align	Specifies the alignment of the text within its containing element
vertical-align	Vertical alignment of text within containing element and in relation to containing element
text-decoration	Specifies whether the text should be underlined, overlined, strikethrough, or blinking text

# Text Formatting

Property	Purpose
text-indent	Specifies an indent from the left border for the text
text-transform	Specifies that the content of the element should all be uppercase, lowercase, or capitalized
text-shadow	Specifies that the text should have a drop shadow
letter-spacing	Controls the width between letters (known to print designers as kerning)

# Text Formatting

Property	Purpose
word-spacing	Controls the amount of space between each word
white-space	Specifies whether the white space should be collapsed, preserved, or prevented from wrapping
direction	Specifies the direction of text (similar to the dir attribute)
unicode-bidi	Allows you to create bidirectional text

# Text Formatting

## 1. The color Property

- The `color` property allows you to specify the color of the text.
- The value of this property can either be a `hex code` for a `color` or a `color name`.
- For example, the following rule would make the content of paragraph elements red:
- `p {color:#ff0000;}`

# Text Formatting

- <html>
- <head>
- <link rel="stylesheet" type="text/css" href = "TextColor.css" />
- </head>
- <body>
- <p>Welcome To MSc Computer Science.</p>
- </body>
- </html>



# Text Formatting

## 2. The text-align Property

- The text-align property works like the deprecated align attribute would with text. It aligns the text within its containing element or the browser window.

# Text Formatting

Value	Purpose
left	Aligns the text with the left border of the containing element
right	Aligns the text with the right border of the containing element
center	Centers the content in the middle of the containing element
justify	Spreads the width across the whole width of the containing element

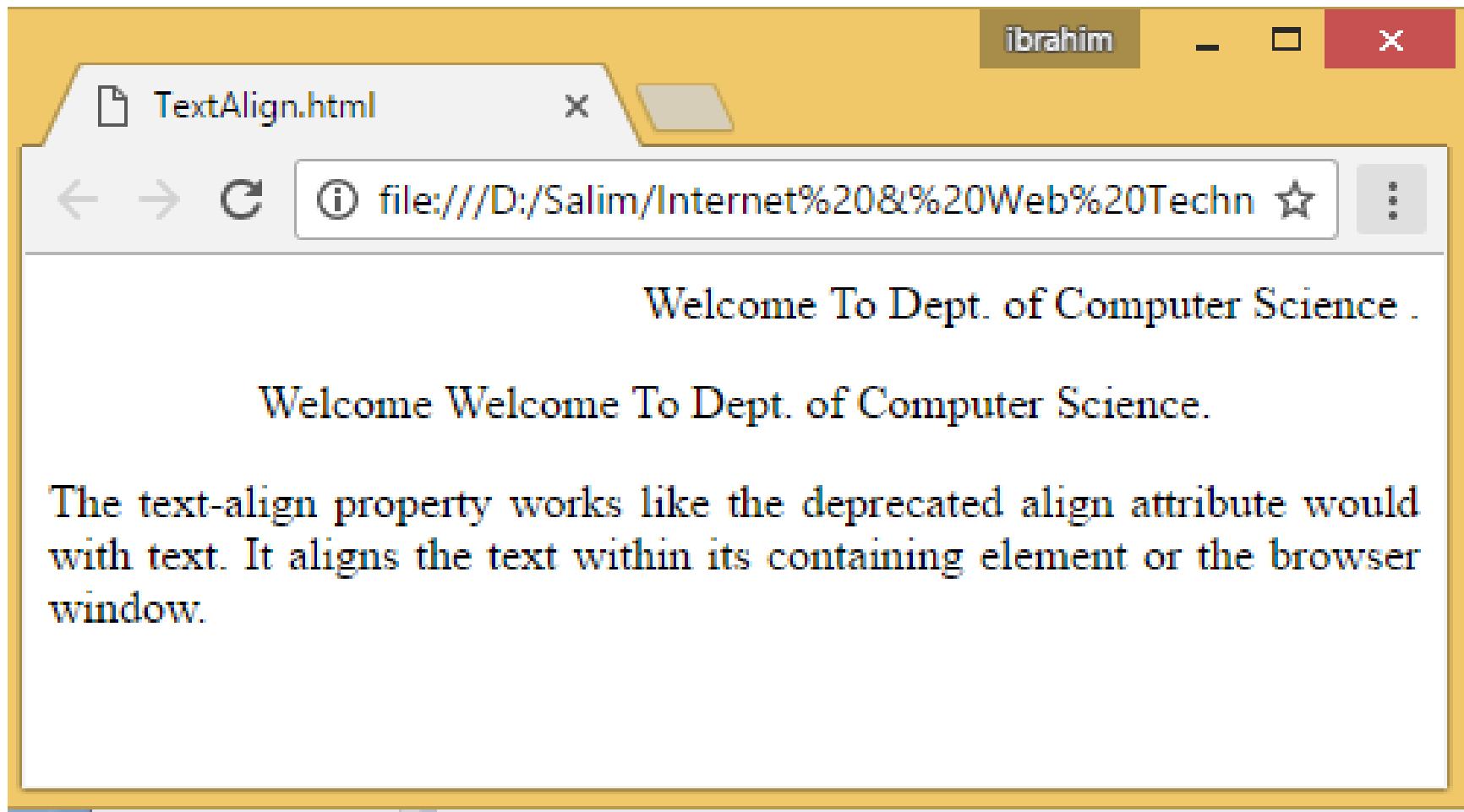
# Text Formatting

- Example of CSS file
- p.one {text-align:right;}
- p.two {text-align:center;}
- p.three {text-align:justify;}

# Text Formatting

- Example
- <html>
- <head>
- <link rel="stylesheet" type="text/css" href=".TextAlign.css" />
- </head>
- <body>
- <p class ="one">Welcome To Dept. of Computer Science.</p>
- <p class ="two">Welcome To Dept of Computer Science.</p>
- <p class ="three">The text-align property works like the deprecated align attribute would with text. It aligns the text within its containing element or the browser window.</p>
- </body>
- </html>

# Text Formatting



# Text Formatting

## 3. The vertical-align Property

- The vertical-align property is useful when working with inline elements, in particular images and portions of text.
- It allows you to control their vertical positioning within the containing element.
- `span.footnote {vertical-align:sub;}`

# Text Formatting

Value	Purpose
baseline	Everything should be aligned on the baseline of the parent element (this is the default setting).
sub	Makes the element subscript. With images, the top of the image should be on the baseline. With text, the top of the font body should be on the baseline.
super	Makes the element superscript. With images, the bottom of the image should be level with the top of the font.
top	The top of the text and the top of the image should align with the top of the tallest element on the line.

# Text Formatting

## 4. The text-decoration Property

- The text-decoration property allows you to specify the values shown in the table that follows.
- | Value                        | Purpose   |
|------------------------------|---|
| 1. <code>underline</code>    | Adds a line under the content.  |
| 2. <code>overline</code>     | Adds a line over the top of the content.  |
| 3. <code>line-through</code> | Like strikethrough text, with a line through the middle. In general, this should be used only to indicate text that is marked for deletion. |
| 4. <code>blink</code>        | Creates blinking text (which is generally frowned upon and considered annoying).  |

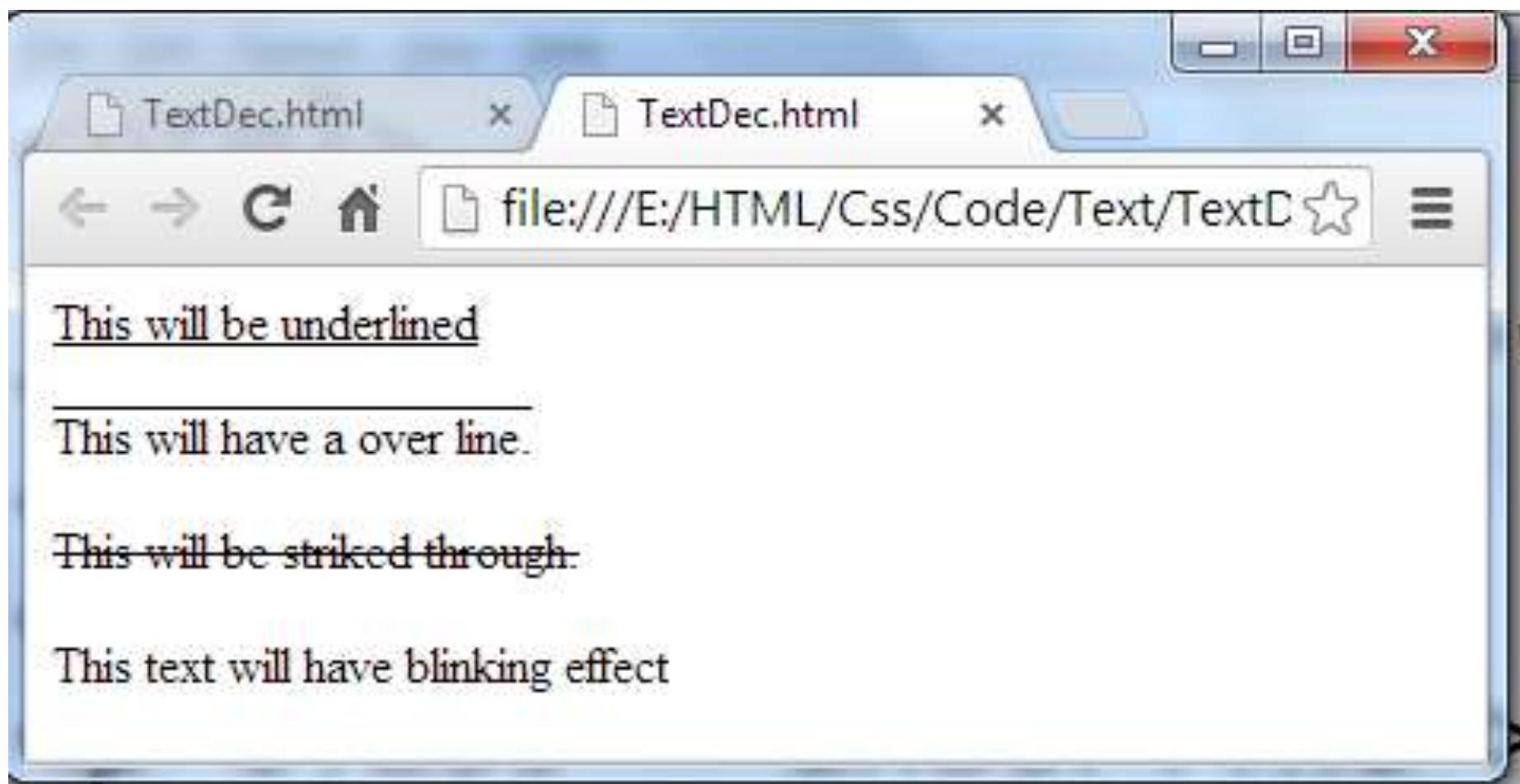
# Text Formatting

- For example, here are these properties used on separate paragraphs:
  - p.underline {text-decoration:underline;}
  - p.overline {text-decoration:overline;}
  - p.line-through {text-decoration:line-through;}
  - p.blink {text-decoration:blink;}

# Text Formatting

- <html>
- <head>
- <link rel="stylesheet" type="text/css" href="TextDec.css" />
- </head>
- <body>
- <p class = "underline">This will be underlined</p>
- <p class="overline"> This will have a over line.</p>
- <p class="line-through"> This will be striked through. </p>
- <p class="blink"> This text will have blinking effect </p>
- </body>
- </html>

# Text Formatting



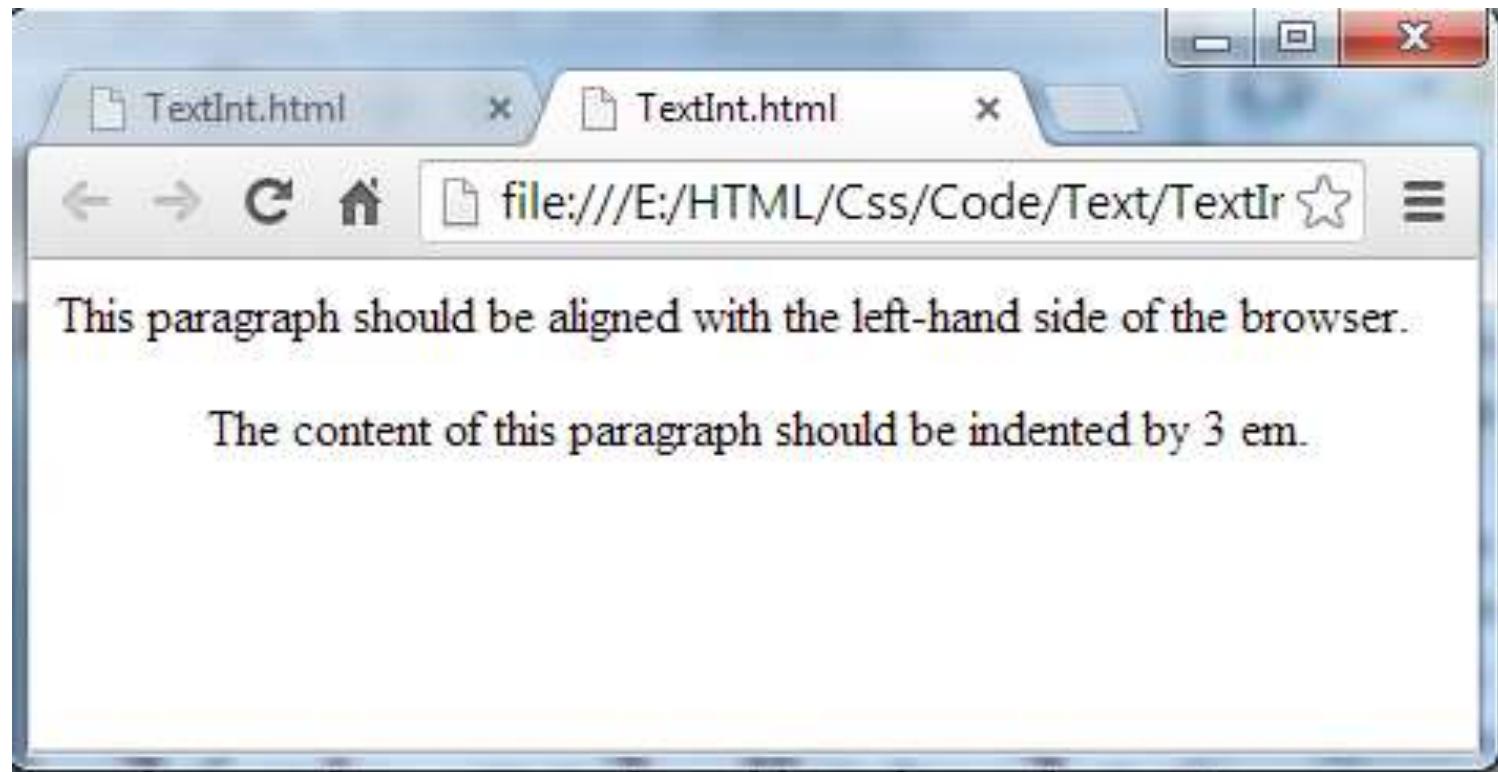
# Text Formatting

## 5. The text-indent Property

- The text-indent property allows you to indent the first line of text within an element.
- For example, here you can see that the first line of the second paragraph has been indented.
- Html Program
- <p>This paragraph should be aligned with the left-hand side of the browser. </p>
- <p class="indent">The content of this paragraph should be indented by 3 em. </p>

# Text Formatting

- Css Example
- .indent {text-indent:3em;}



# Text Formatting

## 6. The text-shadow Property

- The text-shadow property is supposed to create a drop shadow, which is a dark version of the word just behind it and slightly offset.
- This has often been used in print media, and its popularity has meant that it has gained its own CSS property in CSS2.
- The value for this property is quite complicated because it can take three lengths, optionally followed by a color:
  - `.dropShadow { text-shadow: 0.3em 0.3em 0.5em black }`
  - The first two lengths specify X and Y coordinates for the offset of the drop shadow, while the third specifies a blur effect.
  - This is then followed by a color, which can be a name or a hex value.

# Text Formatting

- <html>
- <head>
- <link rel="stylesheet" type="text/css" href="TextShad.css" />
- </head>
- <body>
- <p class="drop">This paragraph should be aligned with the left-hand side of the browser. </p>
- </body>
- </html>

# Text Formatting



# Text Formatting

## 7. The text-transform Property

- The text-transform property allows you to specify the case for the content of an element.
- Value                          Purpose
  - 1. **none**                          No change should take place.
  - 2. **capitalize**                      The first letter of every word should be capitalized.
  - 3. **uppercase**                      The entire content of the element should be uppercase.
  - 4. **lowercase**                      The entire content of the element should be lowercase.

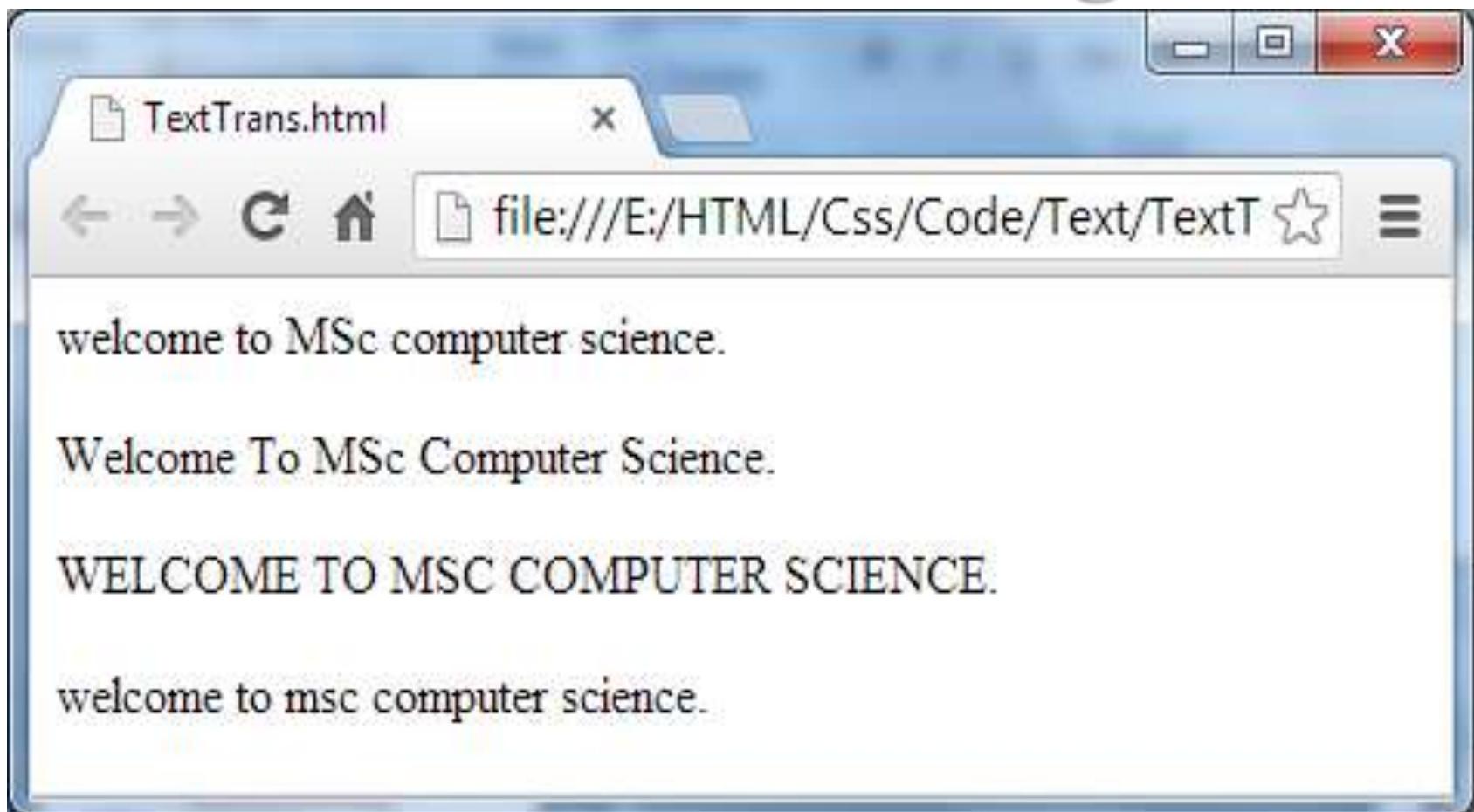
# Text Formatting

- Here you can see the four different values for the text-transform property
- p.none {text-transform:none;}
- p.Capital {text-transform:Capitalize;}
- p.upper {text-transform:UPPERCASE;}
- p.lower{text-transform:lowercase;}

# Text Formatting

- <html>
- <head>
- <link rel="stylesheet" type="text/css" href="TextTrans.css" />
- </head>
- <body>
- <p class ="none">welcome to MSc computer science.</p>
- <p class ="capital">welcome to MSc computer science.</p>
- <p class ="upper">welcome to MSc computer science.</p>
- <p class ="lower">Welcome To MSc Computer Science.</p>
- </body>
- </html>

# Text Formatting



# Text Formatting

## 8. The letter-spacing Property

- The letter-spacing property is supposed to control something that print designers refer to as tracking: the gap between letters.
- Loose tracking indicates that there is a lot of space between letters, whereas tight tracking refers to letters being squeezed together.
- No tracking refers to the normal gap between letters for that font.
- `p.wider {letter-spacing:10px;}`

# **Text Formatting**



# Text Formatting

## 9. The word-spacing Property

- The word-spacing property is supposed to set the gap between words. Its value should be a unit of length.
- span.wider { word-spacing:20px; }



# Text Formatting

## 10. The white-space Property

- The white-space property controls whether or not white space is preserved within and between block level elements.
- By default, a browser changes any two or more spaces next to each other into a single space, and makes any carriage returns a single space, too.
- The white-space property offers the same results as the HTML <pre> element and nowrap attribute.

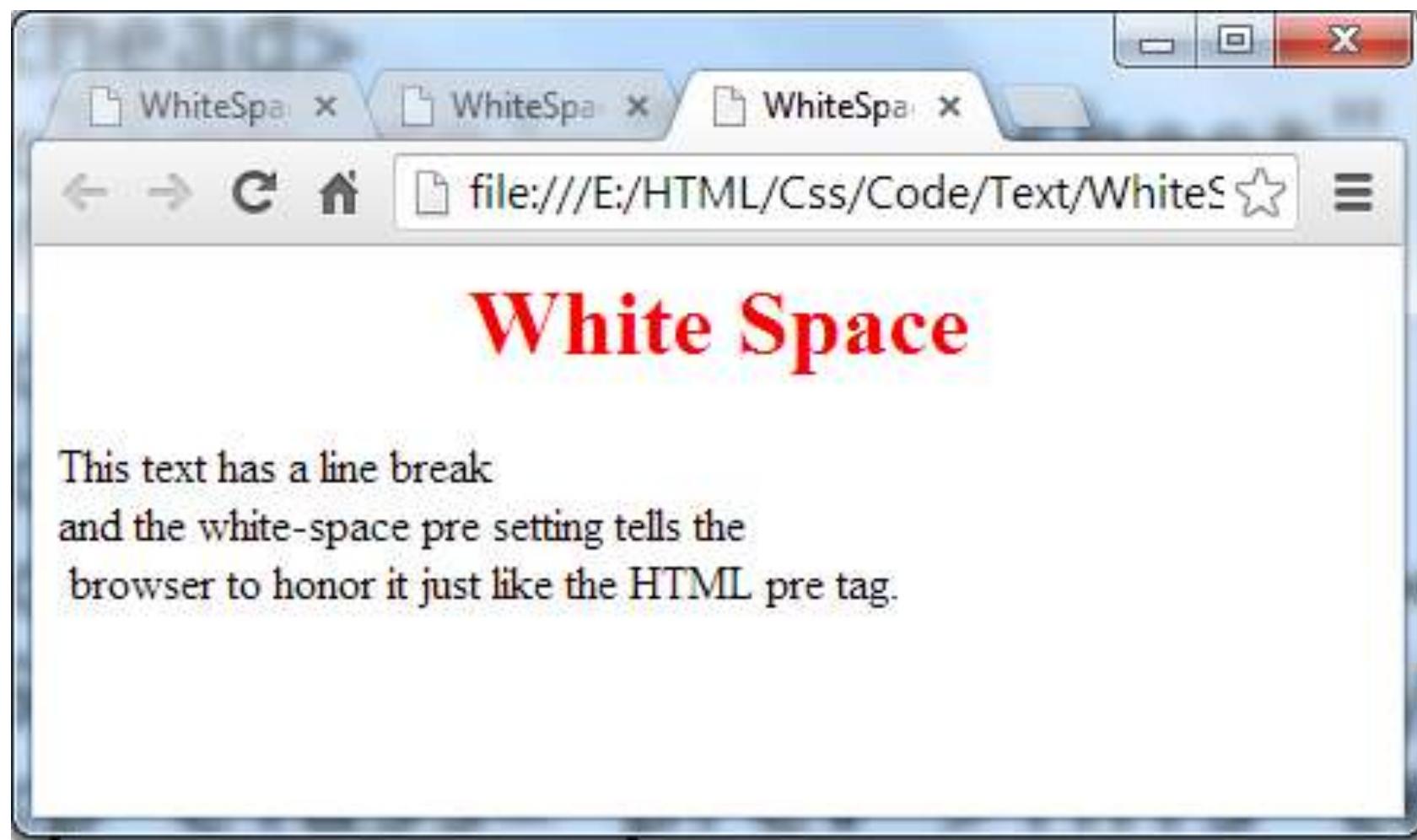
# Text Formatting

- Value
- 1. **normal**      Meaning  
Normal white space collapsing rules are followed.
- 2. **pre**      Meaning  
White space is preserved just as in the `<pre>` element of XHTML, but the formatting is whatever is indicated for that element, not just a monospaced font.
- 3. **nowrap**      Meaning  
Text is broken onto a new line only if explicitly told to with a `<br />` element; otherwise text does not wrap.

# Text Formatting

- For example, you can use the white-space property like so :
- `.prev {white-space:pre;}`
- `.nowrap {white-space:nowrap;}`
- Example
- `<p class="prev">This text has a line break`
- and the white-space pre setting tells the browser to honor it just like the HTML pre tag.`</p>`

# Text Formatting



# Text Formating

## 11. The direction Property

- The direction property is rather like the dir attribute and specifies the direction in which the text should flow.

Value	Meaning
1. ltr	The text flows from left to right.
2. rtl	The text flows from right to left.
3. inherit	The text flows in the same direction as its parent element. The following table shows the possible values.

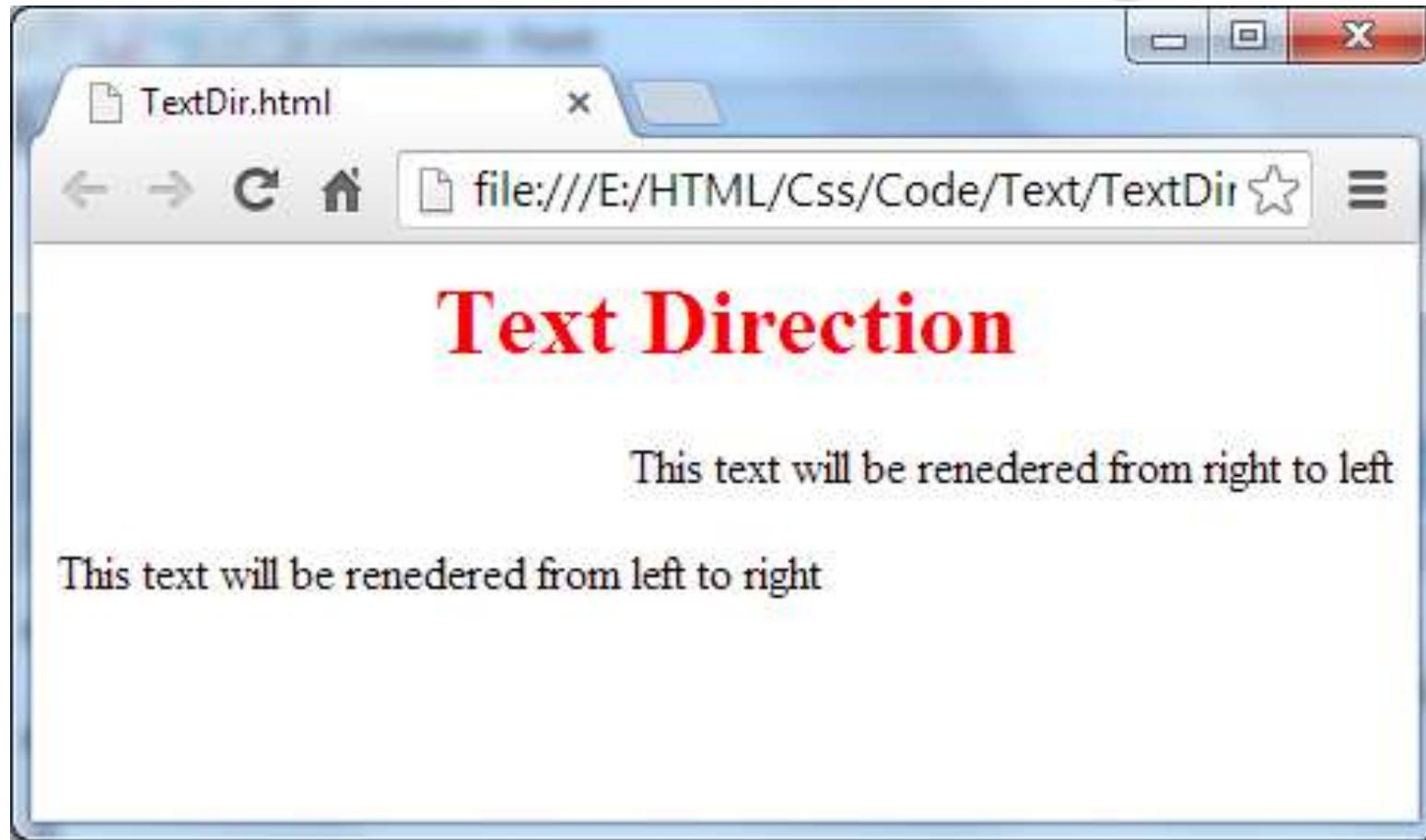
# Text Formating

- p.ltr {direction:ltr;}
- p.rtl {direction:rtl;}

# Text Formating

- Css Example
- p.ltr {direction:ltr;}
- p.rtl {direction:rtl;}
- Html code
- <p class="rtl"> This text will be renedered from right to left </p>
- <p class="ltr"> This text will be renedered from left to right </p>

# Text Formating



# Text Pseudo-Classes

- These pseudo-classes allow you to render either the first letter or the first line of an element in a different way than the rest of that element.
- Both of these are commonly used when laying out text.

## 1. **The first-letter Pseudo-Class**

- The first-letter pseudo-class allows you to specify a rule just for the first letter of an element.
- This is most commonly used on the first character of a new page, either in some magazine articles or in books.

# Text Pseudo-Classes

- <p class="pageOne"> These pseudo-classes allow you to render either the first letter or the first line of an element in a different way than the rest of that element. </p>
- Css File
- p.pageOne:first-letter {font-size:42px;}

# Text Pseudo-Classes

The screenshot shows a web browser window with the following details:

- Title Bar:** Shows three tabs: "TextDir.htm", "SeudoFir...", and "SeudoFirs...".
- Address Bar:** Displays the URL "file:///E:/HTML/Css/Code/Sudo/Seudo...".
- Content Area:** Contains the following text:

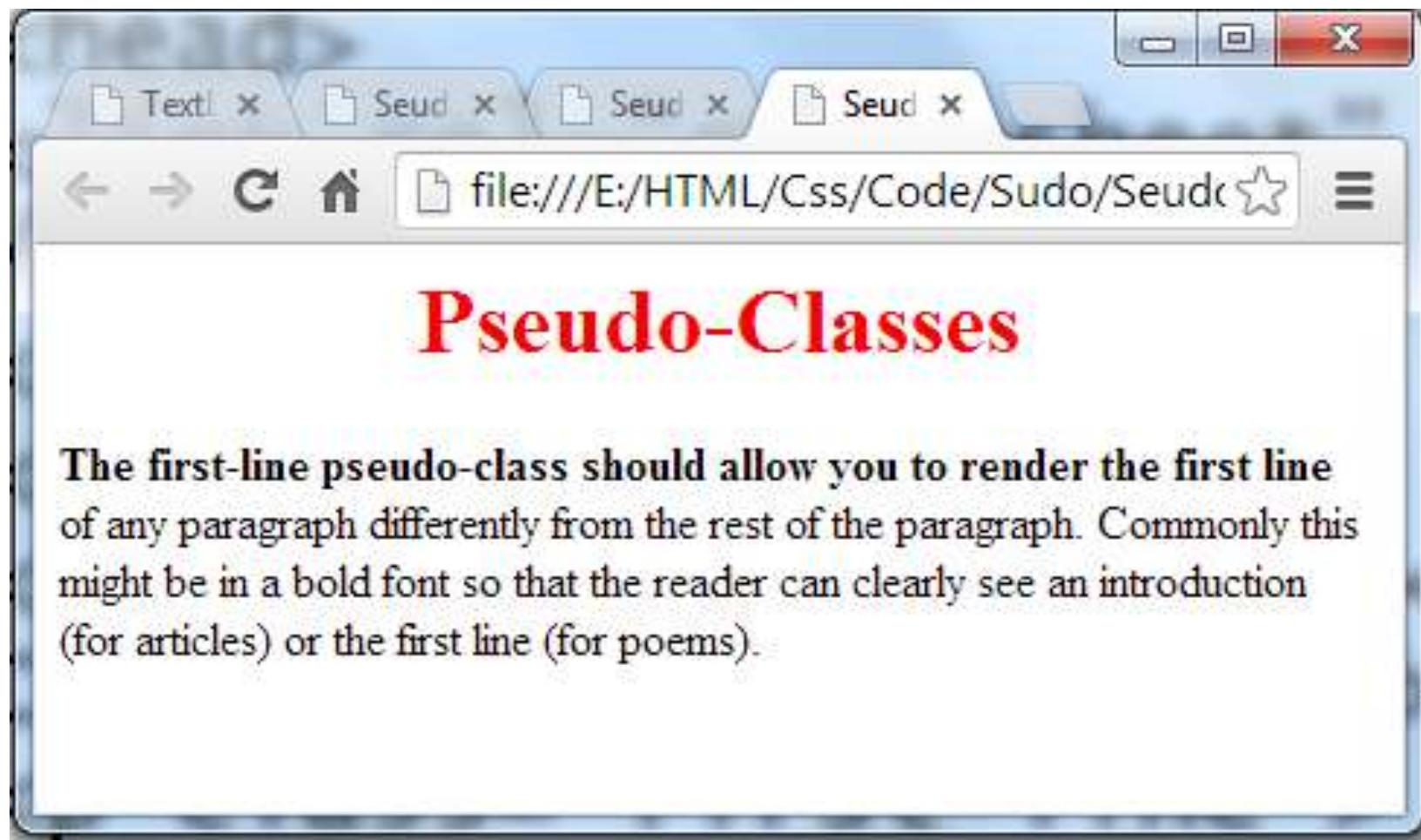
**Pseudo-Classes**

These pseudo-classes allow you to render either the first letter or the first line of an element in a different way than the rest of that element.

# Text Pseudo-Classes

- **The first-line Pseudo-Class**
- The first-line pseudo-class should allow you to render the first line of any paragraph differently from the rest of the paragraph.
- Commonly this might be in a bold font so that the reader can clearly see an introduction (for articles) or the first line (for poems).
- The name of the pseudo-class is separated from the element it should appear on by a colon:
- `p:first-line {font-weight:bold;}`

# Text Pseudo-Classes



# Links

- Link is a connection from one web page to another web pages. CSS property can be used to style the links in various different ways.
- The ability to change slightly the color of links that have visited can help users navigate your site, and changing the color when someone hovers over a link can encourage the user to click it.
- Color property can change the color of the text inside any element, and web designers commonly use this property in rules that apply to `<a>` elements in order to change the colors of links.

# Links

- Pseudo-class      Purpose
- link                Styles for links in general
- visited             Styles for links that have already been visited
- active              Styles for links that are currently active (being clicked)
- hover               Styles for when someone is hovering over a link

# Links

1. **color**: Often used to change the colors of the links. As mentioned, it is helpful to differentiate slightly between different links that have already been visited and those not yet visited, as this helps users see where they've been.
2. **text-decoration**: Often used to control whether the link is underlined or not.
  - Using the text-decoration property, you can specify that your links should not be underlined, and you can even set them to be underlined only when the user hovers over the link or selects it.

# Links

3. **background-color:** Highlights the link, as if it had been highlighted with a highlighter pen.
- It is most commonly used when the user hovers over a link, just offering a slight change in color.
- Syntax:

```
a:link {  
    color:color_name;  
}
```

- color\_name can be given in any format like color name (green), HEX value (#5570f0) or RGB value rgb(25, 255, 2). There is another state ‘a:focus’ which is used to focused when a user uses tab key to navigate through the links.

# Backgrounds

- The table that follows lists the six properties in CSS that allow you to specify how the background of either the whole browser window or any individual box should appear.
- **Property** **Purpose**
- 1. **background-color** Specifies a color that should be used for the background of the page or box
- 2. **background-image** Sets an image to be in the background of a page or box
- 3. **background-repeat** Indicates whether the background image should be repeated across the page or box

# Backgrounds

## ■ Property

4. **background-attachment**

## Purpose

Indicates a background image should be fixed in one position on the page, and whether it should stay in that position when the user scrolls down the page or not

5. **background-position**

Indicates where an image should be positioned in either the window or the containing box

6. **background**

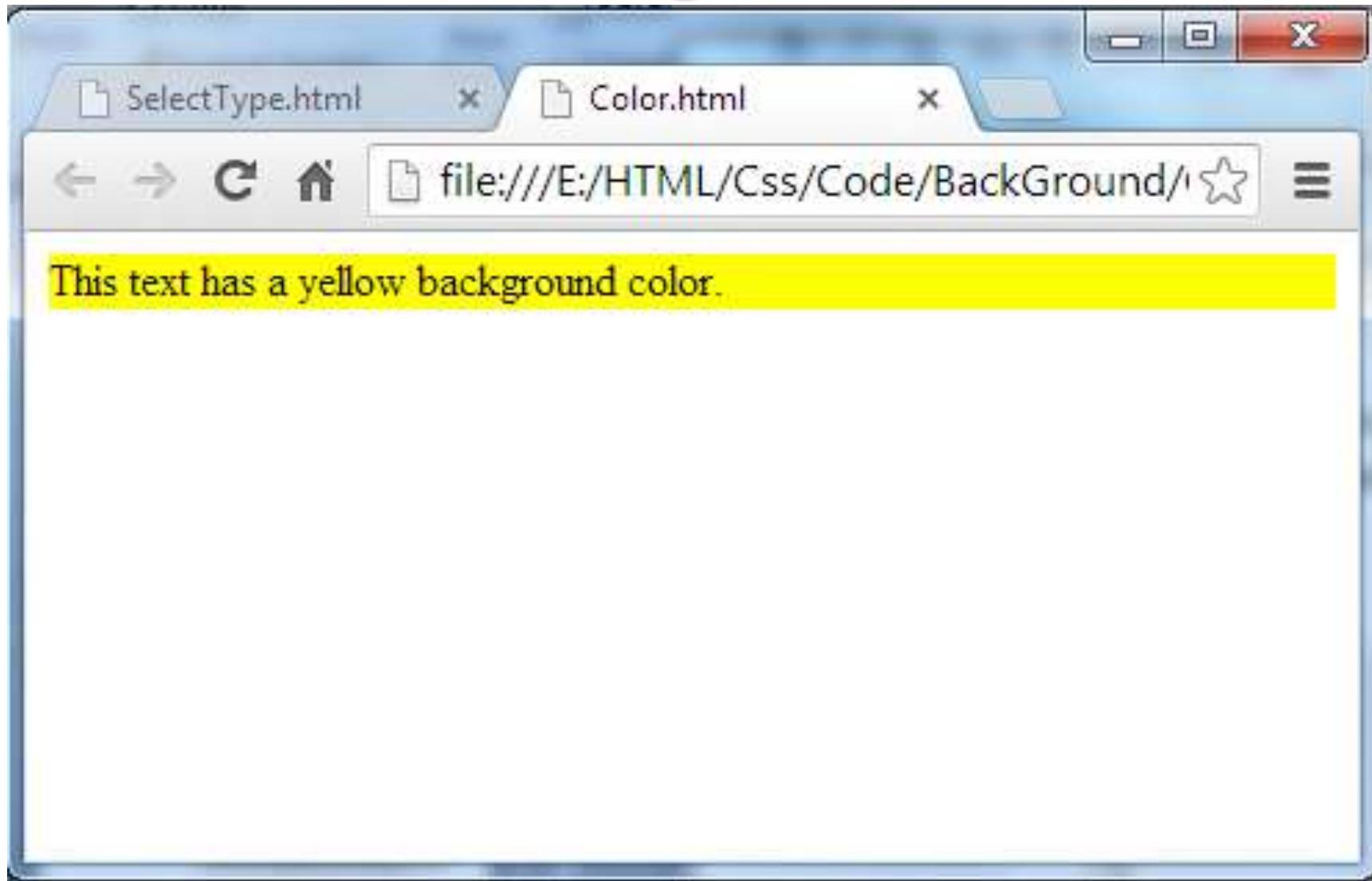
A shorthand form that allows you to specify all of these properties

# Backgrounds

## 1. The background-color Property

- The background-color property allows you to specify a single solid color for the background of your pages and the inside of any box created by CSS.
- The value of this property can be a hex code, a color name, or an RGB value.
- Css File
- `p {background-color:yellow;}`
- [Html file](#)
- `<p> This text has a yellow background color. </p>`

# Backgrounds

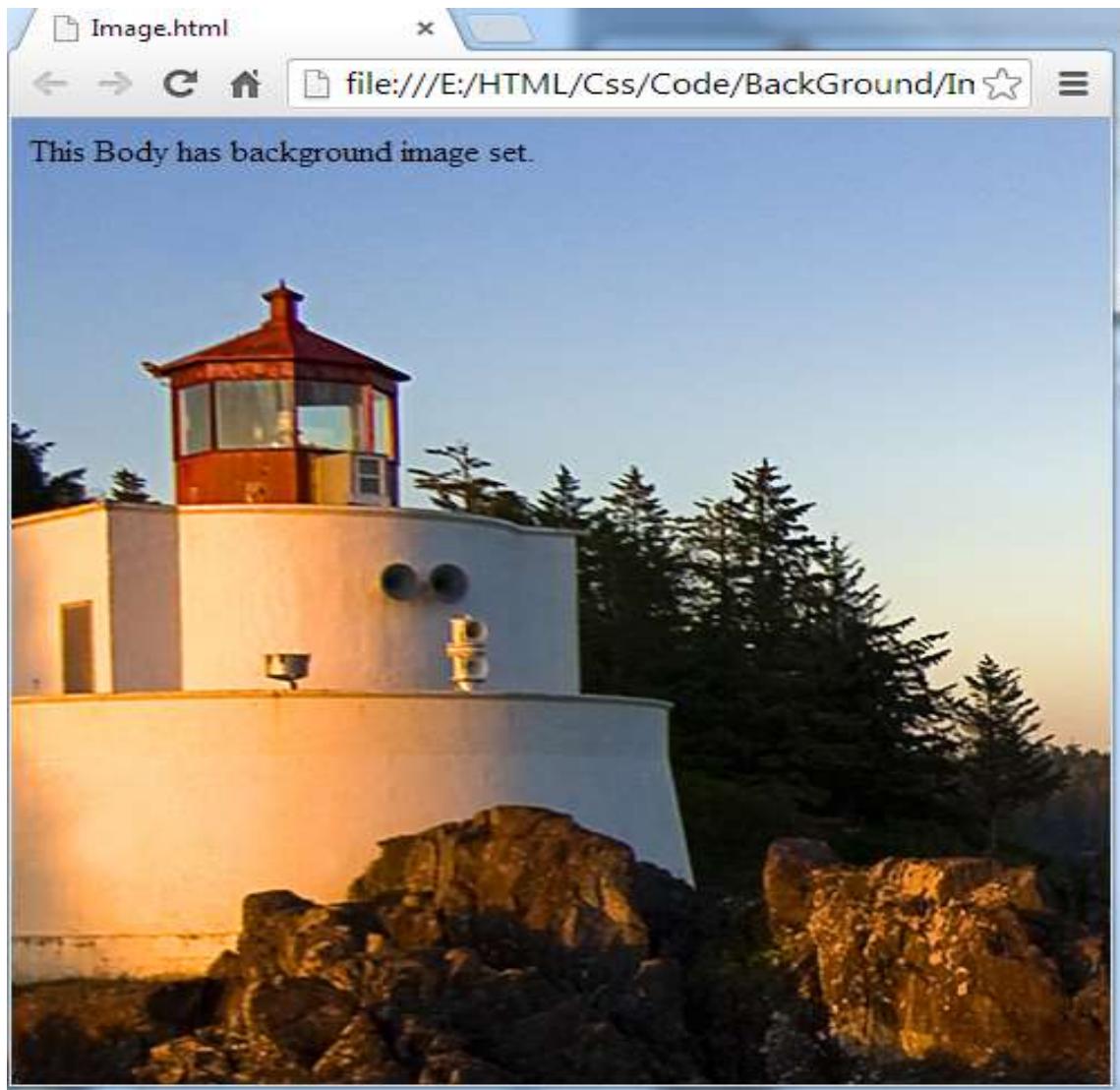


# Backgrounds

## 2. The background-image Property

- As its name suggests, the background-image property allows you to add an image to the background of any box in CSS, and its effect can be quite powerful.
- The value it takes is as follows, starting with the letters url, and then holding the URL for the image in brackets and quotes:
- `body {background-image: url ("images / background .gif) ;" }`

# Backgrounds



# Backgrounds

## 3. The background-repeat Property

- By default, the background-image property repeats across the whole page, creating what is affectionately known as wallpaper.
- The wallpaper is made up of one image that is repeated over and over again, and which (if the image is designed well) you will not see the edges of. Therefore, it is important that any patterns should tessellate, or fit together, well.

# Backgrounds

- Value

1. repeat

## Purpose

This causes the image to repeat to cover the whole page.

2. repeat-x

The image will be repeated horizontally across the page (not down the whole page vertically).

3. repeat-y

The image will be repeated vertically down the page (not across horizontally).

4. no-repeat

The image is displayed only once.

# Backgrounds

- Html file
- <body>
- <p><br><br> This table has background image which repeats multiple times.
- </body>
- CSS File
- body {background-image:url(e:/Leaf.gif);
- background-repeat:repeat-x;}

# Background

- The background-position Property
- you may want to alter the position of this image, and you can do so using the background-position property, which takes the values shown in the table that follows
- Value                              Meaning  
1. x% y%                         Percentages along the x (horizontal) and y (vertical) axis  
2. x y                                Absolute lengths along the x (horizontal) and y (vertical) axis

# Background

- |           |   |
|-----------|---|
| 3. left   | Shown to the left of the page or containing element   |
| 4. center | Shown to the center of the page or containing element |
| 5. right  | Shown to the right of the page or containing element  |
| 6. top    | Shown at the top of the page or containing element    |
| 7. center | Shown at the center of the page or containing element |
| 8. bottom | Shown at the bottom of the page or containing element |

# Background

- CSS File
- body {background-image:url(e:/Leaf.gif);
- background-position:100px;
- background-repeat:no-repeat; }
- HTML File
- <body>
- <p>Background image positioned 100 pixels away from the left.
- </body>

# Background



# Background

- The **background-attachment** Property (for watermarks)
- The background-attachment property allows you to specify an image known as a watermark.
- The key difference with this setting is that the background image can stay in the same position even when the user scrolls up and down a page or scrolls with all of the other elements of the page.
- The background-attachment property can take two values. They are

# Background

- Value
  - 1. **fixed**
  - 2. **scroll**

## Purpose

The image will not move if the user scrolls up and down the page.

The image stays in the same place on the background of the page. If the user scrolls up or down the page, the image moves too.

# Lists

- Lists are very helpful in conveying a set of either numbered or bullet points. This chapter teaches you how to control list type, position, style, etc., using CSS.
- We have the following five CSS properties, which can be used to control lists –
  1. The list-style-type allows you to control the shape or appearance of the marker.
  2. The list-style-position specifies whether a long point that wraps to a second line should align with the first line or start underneath the start of the marker.
  3. The list-style-image specifies an image for the marker rather than a bullet point or number.
  4. The list-style serves as shorthand for the preceding properties.
  5. The marker-offset specifies the distance between a marker and the text in the list.

# Lists

- The list-style-type Property
- The list-style-type property allows you to control the shape or style of bullet point (also known as a marker) in the case of unordered lists and the style of numbering characters in ordered lists.
- By default, items in an ordered list are numbered with Arabic numerals (1, 2, 3, 5, and so on), whereas in an unordered list, items are marked with round bullets (•).
- But, you can change this default list marker type to any other type such as roman numerals, latin letters, circle, square, and so on using the list-style-type property.

# Lists

- The list-style-position Property
- The list-style-position property indicates whether the marker should appear inside or outside of the box containing the bullet points. It can have one of the two values –

1	<b>none</b> NA
2	<b>inside</b> If the text goes onto a second line, the text will wrap underneath the marker. It will also appear indented to where the text would have started if the list had a value of outside.
3	<b>outside</b> If the text goes onto a second line, the text will be aligned with the start of the first line (to the right of the bullet).

# Lists

- The list-style-image Property
- The list-style-image allows you to specify an image so that you can use your own bullet style.
- The syntax is similar to the background-image property with the letters url starting the value of the property followed by the URL in brackets.
- If it does not find the given image then default bullets are used.



# INTRODUCTION TO PHP

- Server-Side Scripting
- MySQL
- PHP
- Apache
- Variables
- Control Structure
- Looping

# Server-Side Scripting

- A script is a set of programming instructions that is interpreted at runtime.
- The purpose of the scripts is usually to enhance the performance or perform routine tasks for an application.
  - Client-side
  - Server-side
- In server-side scripting, (such as PHP, ASP) the script is processed by the server Like: Apache, ColdFusion, ISAPI and Microsoft's IIS on Windows.
- Client-side scripting such as JavaScript runs on the web browser.

# Server-Side Scripting – Continued

- Advantages of Server-Side Scripting
  - Dynamic content.
  - Computational capability.
  - Database and file system access.
  - Network access (from the server only).
  - Built-in libraries and functions.
  - Known platform for execution (as opposed to client- side, where the platform is uncontrolled.)
  - Security improvements

# Introduction to PHP

## *What is PHP?*

- PHP stands for PHP: Hypertext Preprocessor
- Developed by Rasmus Lerdorf in 1994
- It is a powerful server-side scripting language for creating dynamic and interactive websites.
- It is an open source software, which is widely used and free to download and use (PHP is FREE to download from the official PHP resource: [www.php.net](http://www.php.net)).
- It is an efficient alternative to competitors such as Microsoft's ASP.

# Introduction to PHP

PHP scripts are executed on the server

- PHP supports many databases (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.)
- PHP is an open source software
- PHP is free to download and use

# Introduction to PHP

- PHP is perfectly suited for Web development and can be embedded directly into the HTML code.
- The PHP syntax is very similar to JavaScript, Perl and C.
- PHP is often used together with Apache (web server) on various operating systems. It also supports ISAPI and can be used with Microsoft's IIS on Windows.
- PHP supports many databases (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.)

# Introduction to PHP

- What is a PHP File?
- PHP files may contain text, HTML tags and scripts
- PHP files are returned to the browser as plain HTML
- PHP files have a file extension of ".php", ".php3", or ".phtml"
- What is MySQL?
  - MySQL is a database server
  - MySQL is ideal for both small and large applications
  - MySQL supports standard SQL
  - MySQL compiles on a number of platforms
  - MySQL is free to download and use

# Introduction to PHP

- **What you need to develop PHP Application:**
  - Install Apache (or IIS) on your own server, install PHP, and MySQL  
OR
  - Install Wampserver2 (a bundle of PHP, Apache, and MySql server) on your own server/machine

# Advantages of PHP

- **Easy to learn:** PHP is easy to learn and use. For beginner programmers who just started out in web development, PHP is often considered as the preferable choice of language to learn.
- **Open source:** PHP is an open-source project. It is developed and maintained by a worldwide community of developers who make its source code freely available to download and use.
- **Portability:** PHP runs on various platforms such as Microsoft Windows, Linux, Mac OS, etc. and it is compatible with almost all servers used today such Apache, IIS, etc.

# Advantages of PHP

- **Fast Performance:** Scripts written in PHP usually execute or runs faster than those written in other scripting languages like ASP, Ruby, Python, Java, etc.
- **Vast Community:** Since PHP is supported by the worldwide community, finding help or documentation related to PHP online is extremely easy.

# PHP Installation Downloads

## Free Download

- PHP: <http://www.php.net/downloads.php>
- MySQL Database:  
<http://www.mysql.com/downloads/index.html>
- Apache Server: <http://httpd.apache.org/download.cgi>
  - How to install and configure [apache](#)
  - Here is a link to a good tutorial from PHP.net on how to install PHP5:  
<http://www.php.net/manual/en/install.php>

# How to Download & Install XAMPP on Windows

- What is XAMPP?
- XAMPP is an open-source, cross-platform web server that consists of a web server, MySQL database engine, and PHP and Perl programming packages. It is compiled and maintained by Apache.
- It allows users to create WordPress websites online using a local web server on their computer. It supports Windows, Linux, and Mac.
- It is compiled and maintained by apache

# How to Download & Install XAMPP on Windows

- The acronym **XAMPP** stands for;
- **X** – [cross platform operating systems] meaning it can run on any OS Mac OX , Windows , Linux etc.
- **A** – Apache - this is the web server software.
- **M** – MySQL - Database.
- **P** – PHP
- **P** – Perl – scripting language

# How PHP is Processed

- When a PHP document is requested of a server, the server will send the document first to a PHP processor
- Two modes of operation
  - **Copy mode** in which plain HTML is copied to the output
  - **Interpret mode** in which PHP code is interpreted and the output from that code sent to output
  - The client never sees PHP code, only the output produced by the code

# Basic PHP Syntax

- The PHP script can be embedded within HTML web pages.
- A PHP script starts with the <?php and ends with the ?> tag.
- The PHP delimiter <?php and ?> in the following example simply tells the PHP engine to treat the enclosed code block as PHP code, rather than simple HTML.
- 

<?php

- // Some code to be executed
- echo "Hello, world!";
- ?>

# Basic PHP Syntax

- Embedding PHP within HTML
- PHP files are plain text files with .php extension.
- Inside a PHP file you can write HTML like you do in regular HTML pages as well as embed PHP codes for server side execution.
- `<html> <head><title>A Simple PHP File</title>`  
`</head>`
- `<body>`
- `<h1><?php echo "Hello, world!"; ?></h1> </body>`
- `</html>`
- when you run this code the PHP engine executed the instructions between the `<?php ... ?>` tags and leave rest of the thing as it is.
- At the end the web server send the final output back to your browser which is completely in HTML.

# Basic PHP Syntax

- **PHP statements are terminated with semicolons ;**
- Curly braces, { } are used to create compound statements
- Variables cannot be defined in a compound statement unless it is the body of a function
- PHP has typical scripting language characteristics
  - Dynamic typing, un-typed variables
  - Associative arrays
  - Pattern matching
  - Extensive libraries
- **Primitives, Operations, Expressions**
  - Four scalar types: boolean, integer, double, string
  - Two compound types: array, object
  - Two special types: resource and NULL

# Basic PHP Syntax

- A PHP scripting block always starts with <?php and ends with ?>

<?php ..... ?>

- Other options are:
  1. <? ..... ?>
  2. <script> ... </script>
- There are three basic statements to output text with PHP:

echo, print, and printf.

Example: echo 'This is a <b>test</b>!';

# PHP Comments

- A comment in PHP code is a line that is not executed as a part of the program.
- Its only purpose is to be read by someone who is looking at the code.
- <?php
- // This is a single-line comment
- # This is also a single-line comment
- /\*
- This is a multiple-lines comment block
- that spans over multiple
- lines
- \*/
- ?>

# Basic PHP Syntax

## Example 1

```
<html >
<head> <title>Simple PHP Example</title>
<body>
<?php
echo "Hello Class of 2021. This is my first PHP Script"; echo "<br
/>";
print "<b><i>What have you
    learnt and how many friends have you made?</i></b>";
?>
</body>
</html>
```

# Echo and Print Statements

- **Echo**
- The echo statement can output one or more strings.
- echo can be used with or without parentheses
  - e.g. echo or echo().
- echo does not return any value.
- We can pass multiple strings separated by comma (,) in echo.
- echo is faster than print statement.

# Echo and Print Statements

- <?php
- // Displaying string of text
- echo "Hello World!";
- ?>

# Echo and Print Statements

- Display HTML Code
- <?php
- // Displaying HTML code
- echo "<h4>This is a simple heading.</h4>";
- echo "<h4 style='color: red;'>This is heading with style.</h4>";
- ?>

Output : **This is a simple heading.**  
**This is heading with style.**

# Echo and Print Statements

- The print statement (an alternative to echo) is set to display output to the browser.
- print can be used with or without parentheses.
- Using print, we cannot pass multiple arguments.
- print is slower than echo statement.
- Both echo and print statement works exactly the same way except that the print statement can only output one string, and always returns 1.
- That's why the echo statement considered marginally faster than the print statement since it doesn't return any value.

# Echo and Print Statements

- Display Strings of Text
- <?php
- // Displaying string of text
- print "Hello World!";
- ?>
- Display HTML Code
- <?php // Displaying HTML code print “
- <h4>This is a simple heading.</h4>”;
- print "<h4 style='color: red;'>This is heading with style.</h4>";
- ?>

# Variables

- Variables are used to store data, like string of text, numbers, etc.
- Variable values can change over the course of a script.
- Here're some important things to know about variables:
- In PHP, a variable does not need to be declared before adding a value to it.
- PHP automatically converts the variable to the correct data type, depending on its value.
- After declaring a variable it can be reused throughout the code.
- The assignment operator (=) used to assign value to a variable.

# Variables

- Example
- <?php
- // Declaring variables
- \$txt = "Hello World!";
- \$number = 10;
- 
- // Displaying variables value
- echo \$txt; // Output: Hello World!
- echo \$number; // Output: 10
- ?>

# Variable Naming Rules

- All variables in PHP start with a \$ sign, followed by the name of the variable.
- A variable name must start with a letter or the underscore character \_.
- A variable name cannot start with a number.
- A variable name in PHP can only contain alphanumeric characters and underscores (A-z, 0-9, and \_).
- A variable name cannot contain spaces

# PHP Data Types

- PHP supports total eight primitive data types:
  - Integer
  - Floating point number or Float
  - String
  - Booleans
  - Array
  - Object
  - Resource
  - NULL.
  - These data types are used to construct variables.

# PHP Data Types

## □ Integer

- Integer means numeric data with a negative or positive sign.
- It holds only whole numbers, i.e., numbers without fractional part or decimal points.
- Rules for integer:
  1. An integer can be either positive or negative.
  2. An integer must not contain decimal point.
  3. Integer can be decimal (base 10), octal (base 8), or hexadecimal (base 16).
  4. The range of an integer must be lie between 2,147,483,648 and 2,147,483,647 i.e.,  $-2^{31}$  to  $2^{31}$

# PHP Data Types

□ <?php

```
$dec1 = 34;
```

```
$oct1 = 0243;
```

```
$hexa1 = 0x45;
```

```
echo "Decimal number: " . $dec1. "<br>";
```

```
echo "Octal number: " . $oct1. "<br>";
```

```
echo "HexaDecimal number: " . $hexa1.
```

```
"<br>";
```

```
?>
```

## OutPut

Decimal number: 34

Octal number: 163

HexaDecimal number: 69

# PHP Data Types

- **Float**
- A floating-point number is a number with a decimal point.
- Unlike integer, it can hold numbers with a fractional or decimal point, including a negative or positive sign.
- <?php
- \$n1 = 19.34;
- \$n2 = 54.472;
- \$sum = \$n1 + \$n2;
- echo "Addition of floating numbers: " . \$sum;
- ?> **Output** Addition of floating numbers: 73.812

# PHP Data Types

## □ Strings

- Strings are sequences of characters, where every character is the same as a byte.
- A string can hold letters, numbers, and special characters and it can be as large as up to 2GB (2147483647 bytes maximum).
- The simplest way to specify a string is to enclose it in single quotes (e.g. 'Hello world!'), however you can also use double quotes ("Hello world!").

# PHP Data Types

## □ Strings

- Strings are sequences of characters, where every character is the same as a byte.
- A string can hold letters, numbers, and special characters and it can be as large as up to 2GB (2147483647 bytes maximum).
- The simplest way to specify a string is to enclose it in single quotes (e.g. 'Hello world!'), however you can also use double quotes ("Hello world!").

# PHP Data Types

## Strings

- <?php
- \$a = 'Hello world!';
- echo \$a;
- echo "<br>";
- 
- \$b = "Hello world!";
- echo \$b;
- echo "<br>";
- 
- ?>

# PHP Data Types

- Boolean
- Hold only two values, either TRUE or FALSE.
- Successful events will return true and unsuccessful events return false. NULL type values are also treated as false in Boolean.
- Apart from NULL, 0 is also consider as false in boolean.
- If a string is empty then it is also considered as false in boolean data type.
- \$x = true;
- \$y = false;

# PHP Data Types

- Arrays
- An array is a variable that can hold more than one value at a time.
- It is useful to aggregate a series of related items together, for example a set of country or city names.
- An array is formally defined as an indexed collection of data values.
- Each index (also known as the key) of an array is unique and references a corresponding value.

# PHP Data Types

- Arrays
- <?php
- \$intArray = array( 10, 20 , 30);
- echo "First Element: \$intArray[0]\n";  
      echo "Second Element: \$intArray[1]\n";
- echo "Third Element: \$intArray[2]\n";
- ?>
- **Output\**
- **First Element: 10**
- **Second Element: 20**
- **Third Element: 30**

# PHP Data Types

- Objects
- An object is a data type that not only allows storing data but also information on, how to process that data.
- An object is a specific instance of a class which serve as templates for objects.
- Objects are created based on this template via the new keyword.
- Every object has properties and methods corresponding to those of its parent class.
- Every object instance is completely independent, with its own properties and methods, and can thus be manipulated independently of other objects of the same class.

# PHP Data Types

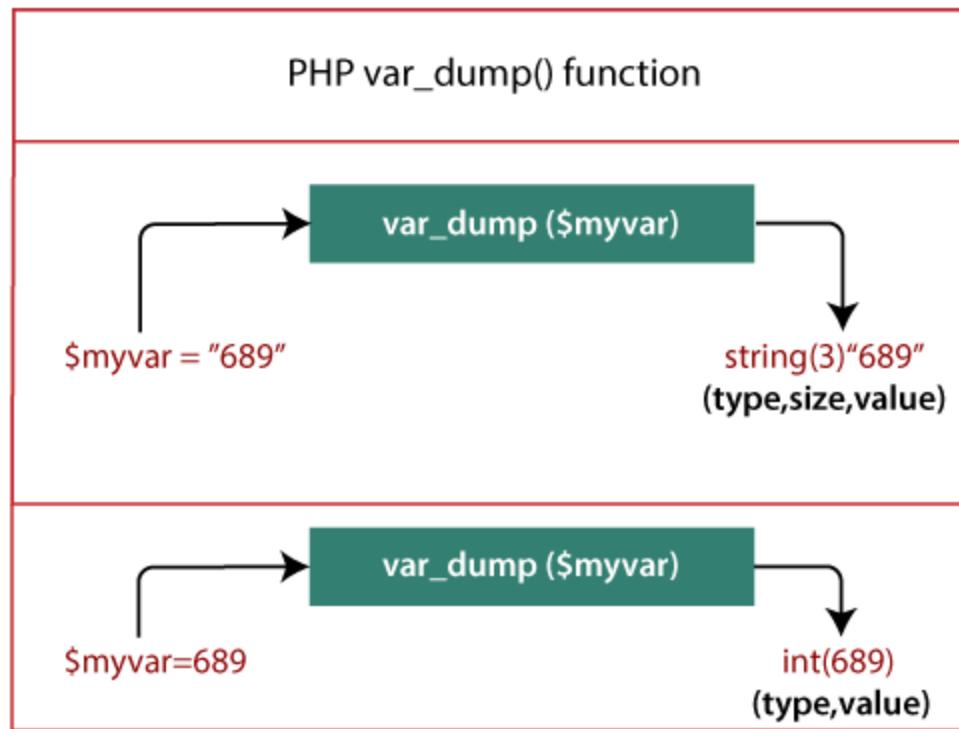
- Objects
- <?php
- class bike {
- function model() {
- \$model\_name = "Royal Enfield";
- echo "Bike Model: " . \$model\_name;
- }
- }
- \$obj = new bike();
- \$obj -> model();
- ?>
- Output:
- Bike Model: Royal Enfield

# PHP var\_dump() function

- The var\_dump() function is a built-in function of PHP that dumps the information about the variables.
- This information includes the data type and value of the variable. In case of string, it also includes the size of the string passed inside the function.
- The array and object are explored recursively with values to show their structure.
- In simple words, this function provides structured information about one or more variables.
- **Syntax**
- **var\_dump(var1, var2, ...);**

# PHP var\_dump() function

- Parameter
- Expression (var1, var2, ...): variable or the value of the variable, which you want to dump.



# PHP var\_dump() function

- <?php
- //PHP program to demonstrate the working of var\_dump function
- \$x = 25;
- //dump integer variable
- var\_dump (\$x);
- echo "</br>\$msg1 = "Hello Alex";
- var\_dump (\$msg1);
- ?>

# PHP Data Types

## Resources

- A resource is a special variable, holding a reference to an external resource.
- Resource variables typically hold special handlers to opened files and database connections.
- <?php
- // Open a file for reading
- \$handle = fopen("note.txt", "r");
- var\_dump(\$handle);
- echo "<br>";
- link = mysql\_connect("localhost", "root", "");
- var\_dump(\$link);
- ?>

# PHP Data Types

- NULL
- The special NULL value is used to represent empty variables in PHP.
- A variable of type NULL is a variable without any data. NULL is the only possible value of type null.
- <?php
- \$nl = NULL;
- echo \$nl; //it will not give any output
- ?>

# PHP Operators

- Arithmetic Operators
- The arithmetic operators are used to perform common arithmetical operations, such as addition, subtraction, multiplication etc. Here's a complete list of PHP's arithmetic operators:

Operator	Description	Example	Result
+	Addition	$\$x + \$y$	Sum of $\$x$ and $\$y$
-	Subtraction	$\$x - \$y$	Difference of $\$x$ and $\$y$ .
*	Multiplication	$\$x * \$y$	Product of $\$x$ and $\$y$ .
/	Division	$\$x / \$y$	Quotient of $\$x$ and $\$y$
%	Modulus	$\$x \% \$y$	Remainder of $\$x$ divided by $\$y$

# Assignment Operators

- The assignment operators are used to assign values to variables.

Operator	Description	Example	Is The Same As
=	Assign	$\$x = \$y$	$\$x = \$y$
+=	Add and assign	$\$x += \$y$	$\$x = \$x + \$y$
-=	Subtract and assign	$\$x -= \$y$	$\$x = \$x - \$y$
*=	Multiply and assign	$\$x *= \$y$	$\$x = \$x * \$y$
/=	Divide and assign quotient	$\$x /= \$y$	$\$x = \$x / \$y$
%=	Divide and assign modulus	$\$x %= \$y$	$\$x = \$x \% \$y$

# Comparison Operators

- The comparison operators are used to compare two values in a Boolean fashion.

Operator	Name	Example	Result
<code>==</code>	Equal	<code>\$x == \$y</code>	True if \$x is equal to \$y
<code>===</code>	Identical	<code>\$x === \$y</code>	True if \$x is equal to \$y, and they are of the same type
<code>!=</code>	Not equal	<code>\$x != \$y</code>	True if \$x is not equal to \$y
<code>&lt;&gt;</code>	Not equal	<code>\$x &lt;&gt; \$y</code>	True if \$x is not equal to \$y
<code>!==</code>	Not identical	<code>\$x !== \$y</code>	True if \$x is not equal to \$y, or they are not of the same type
<code>&lt;</code>	Less than	<code>\$x &lt; \$y</code>	True if \$x is less than \$y
<code>&gt;</code>	Greater than	<code>\$x &gt; \$y</code>	True if \$x is greater than \$y
<code>&gt;=</code>	Greater than or equal to	<code>\$x &gt;= \$y</code>	True if \$x is greater than or equal to \$y
<code>&lt;=</code>	Less than or equal to	<code>\$x &lt;= \$y</code>	True if \$x is less than or equal to \$y

# Incrementing and Decrementing Operators

- The increment/decrement operators are used to increment/decrement a variable's value.

Operator	Name	Effect
<code>++\$x</code>	Pre-increment	Increments \$x by one, then returns \$x
<code>\$x++</code>	Post-increment	Returns \$x, then increments \$x by one
<code>--\$x</code>	Pre-decrement	Decrements \$x by one, then returns \$x
<code>\$x--</code>	Post-decrement	Returns \$x, then decrements \$x by one

# Logical Operators

- The logical operators are typically used to combine conditional statements.

Operator	Name	Example	Result
and	And	<code>\$x and \$y</code>	True if both \$x and \$y are true
or	Or	<code>\$x or \$y</code>	True if either \$x or \$y is true
xor	Xor	<code>\$x xor \$y</code>	True if either \$x or \$y is true, but not both
<code>&amp;&amp;</code>	And	<code>\$x &amp;&amp; \$y</code>	True if both \$x and \$y are true
<code>  </code>	Or	<code>\$x    \$y</code>	True if either \$\$x or \$y is true
!	Not	<code>!\$x</code>	True if \$x is not true

## **== versus ===**

- Two “equality” operators
  - ▣ == tests for “equality” in value but not necessarily type
  - ▣ === tests for “identity” in value AND type
- == ignores the distinction between:
  - ▣ Integers, floating point numbers, and strings containing the same numerical value
  - ▣ Nonzero numbers and boolean TRUE
  - ▣ Zero and boolean FALSE
  - ▣ Empty string, the string ‘0’ and boolean FALSE
  - ▣ Any other non-empty string and boolean TRUE

# Strings

- A sequence of characters
- Single and double quotes:
  - Suppose \$str = 42;
  - echo 'With single quotes, str is \$str';  
→ output: With single quotes, str is \$str
  - echo "With double quotes, str is \$str";  
→ output: With double quotes, str is 42

# Strings in PHP

a string is a sequence of letters, symbols, characters and arithmetic values or combination of all tied together in single or double quotes.

- String literals are enclosed in single or double quotes
- Example:

```
<?php
$sum = 20;
echo 'the sum is:
$sum'; echo "<br
/>";
echo "the sum is: $sum";
echo "<br />";
echo '<input type="text" name="first_name" id="first_name">';
?>
```

- Double quoted strings have escape sequences (such as /n or /r) interpreted and variables interpolated (substituted)
- Single quoted strings have neither escape sequence interpretation nor variable interpolation
- A literal \$ sign in a double quoted string must be escaped with a backslash,

# The Concatenation Operator

- The concatenation operator (.) is used to put two string values together.
- Example:

```
<?php  
    $txt1="Hello Everyone,";  
    $txt2="1234 is Dan's home address";  
    echo $txt1.$txt2;  
?>
```

# Control Structure

- Control structures are the building blocks of any programming language. PHP provides all the control structures that you may have encountered anywhere. The syntax is the same as C or Perl.
- Making computers think has always been the goal of the computer architect and the programmer. Using control structures computers can make simple decisions and when programmed cleverly they can do some complex things.

# Conditional Statements

## 1. The If...Else Statement

### Syntax

*if (condition) code to be executed  
if condition is true;  
  
else code to be executed if  
condition is false;*

```
<?php  
$d=date("D");  
if ($d=="Fri") echo "Have a nice  
weekend!";  
else echo "Have a nice day!";  
?>
```

If more than one line should be executed if a condition is true/false, the lines should be enclosed within curly braces:

```
<?php  
$num=12;  
if($num<100){  
echo "$num is less than 100";  
}  
?>
```

# Conditional Statements

## 2. The ElseIf Statement

- If you want to execute some code if one of several conditions is true use the elseif statement

### Syntax

*if (condition) code to be  
executed if condition is true;  
elseif (condition) code to be  
executed if condition is true; ?>  
else code to be executed if  
condition is false;*

```
<?php
$num=12;
if($num%2==0){
echo "$num is even number";
}else{
echo "$num is odd number";
}
?>
```

# PHP Switch Statement

- If you want to select one of many blocks of code to be executed, use the Switch statement.
- The switch statement is used to avoid long blocks of if..elseif..else code.

## Syntax

```
switch (expression)
{
    case label1: code to be executed if expression = label1;
        break;
    case label2: code to be executed if expression = label2;
        break;
    default: code to be executed if expression is different from both label1 and label2;
}
```

```
switch ($textcolor)
{
    case "black":
        echo "I'm black";
        break;
    case "blue":
        echo "I'm blue";
        break;
    case "red":
        echo "I'm red";
        break;
    default: // It must be something else
        echo "too bad!!, I'm something else";
}
```

# PHP Looping

- Looping statements in PHP are used to execute the same block of code a specified number of times.
- In PHP we have the following looping statements:
  - while - loops through a block of code if and as long as a specified condition is true
  - do...while - loops through a block of code once, and then repeats the loop as long as a special condition is true
  - for - loops through a block of code a specified number of times
  - foreach - loops through a block of code for each element in an array

# The while Statement

## Syntax

**while** (condition)

```
{  
// statements  
}
```

## Example

```
<html> <head> <title>Let us count !!!</title></head>  
<body>  
<?php  
$limit = 10;  
echo "<h2> Let us count from 1 to $limit</h2><br  
/>";  
$count = 1;  
while ($count <= $limit)  
{  
    echo "counting $count of $limit <br>";  
    $count++;  
}  
?>  
</body>
```

# The do...while Statement

- The do...while statement will execute a block of code at least once - it then will repeat the loop as long as a condition is true.

## Syntax

- *do { code to be executed; } while (condition);*

## Example

```
<html> <body>
<?php
$i=0;
do { $i++; echo "The number
is " . $i . "<br />"; }
while ($i<5);
?>
</body> </html>
```

# The for Statement

- It is used when you know how many times you want to execute a statement or a list of statements.

## Syntax

- for (*init*, *cond*, *incr*) { *code to be executed*; }  
Parameters:
  - init:** Is mostly used to set a counter, but can be any code to be executed once at the beginning of the loop statement.
  - cond:** Is evaluated at beginning of each loop iteration. If the condition evaluates to TRUE, the loop continues and the code executes. If it evaluates to FALSE, the execution of the loop ends.
  - incr:** Is mostly used to increment a counter, but can be any code to be executed at the end of each loop.

## Example

```
<html> <body>
<?php
for ($i=1; $i<=5; $i++)
{
echo "Hello World!<br
/>";
}
?>
</body> </html>
```

# The foreach Statement

- The foreach statement is used to loop through arrays.
- The foreach loop works only on arrays, and is used to loop through each key/value pair in an array
- For every loop, the value of the current array element is assigned to \$value (and the array pointer is moved by one)
- - so on the next loop, you'll be looking at the next element.

## Syntax:

```
foreach ($array as $value) {  
    code to be executed;  
}
```

- For every loop iteration, the value of the current array element is assigned to \$value and the array pointer is moved by one, until it reaches the last array element.
- The foreach Statement

# The foreach Statement

## Example

```
<?php  
    $colors = array("red", "green", "blue", "yellow");  
    foreach ($colors as $value) {  
        echo "$value <br>";  
    }  
?>
```

## Out Put

red  
green  
blue  
yellow

# The foreach Statement

## Example

```
<?php  
    $colors = array("red", "green", "blue", "yellow");  
    foreach ($colors as $value) {  
        echo "$value <br>";  
    }  
?>
```

## Out Put

red  
green  
blue  
yellow

# Arrays

- Arrays are complex variables that allow us to store more than one value or a group of values under a single variable name.
- There are three types of arrays that you can create. These are:
  - **Indexed array** — An array with a numeric key.
  - **Associative array** — An array where each key has its own specific value.
  - **Multidimensional array** — An array containing one or more arrays within itself.

# Indexed Arrays

- An indexed or numeric array stores each array element with a numeric index. The following examples shows two ways of creating an indexed array.
- <?php
- // Define an indexed array
- \$colors = array("Red", "Green", "Blue"); ?>
- In an indexed or numeric array, the indexes are automatically assigned and start with 0, and the values can be any data type.

# Indexed Arrays

- <html>
- <body>
- <?php
- /\* First method to create array. \*/
- \$numbers = array( 1, 2, 3, 4, 5);
- foreach( \$numbers as \$value ) {
- echo "Value is \$value <br />";
- }
- ?>
- </body>

</html>Output :

Value is 1

Value is 2

Value is 3

Value is 4

Value is 5

# Indexed Arrays

- <html>
- <body>
- <?php
- /\* Second method to create array. \*/
- \$numbers[0] = "one";
- \$numbers[1] = "two";
- \$numbers[2] = "three";
- \$numbers[3] = "four";
- \$numbers[4] = "five";
- foreach( \$numbers as \$value ) {
- echo "Value is \$value <br />";
- }
- ?> </body>
- </html>

Output:

Value is one  
Value is two  
Value is three  
Value is four  
Value is five

# Associative array

- The associative arrays are very similar to numeric arrays in term of functionality but they are different in terms of their index.
- Associative array will have their index as string so that you can establish a strong association between key and values.
- To store the salaries of employees in an array, a numerically indexed array would not be the best choice. Instead, we could use the employees names as the keys in our associative array, and the value would be their respective salary.
- Don't keep associative array inside double quote while printing otherwise it would not return any value.

# Associative array

- Syntax:
- <?php
- \$variable\_name['key\_name']=value;
- Or
- \$variable\_name = array('keyname' => value); ?>

# Associative array

- “\$variable\_name...” is the name of the variable
- “[‘key\_name’]” is the access index number of the element
- “value” is the value assigned to the array element.
- <?php
- \$persons = array('Mary' => "Female", "John" => "Male", "Mirriam" => "Female");
- print\_r(\$persons);
- echo " ";
- echo "Mary is a " . \$persons["Mary"];
- ?>

# Associative array

- Output:
- Array ( [Mary] => Female [John] => Male [Mirriam] => Female )
- Mary is a Female
- Associative array are also very useful when retrieving data from the database.

```
$persons = array('Mary' => 'Female',  
                 'John' => 'Male',  
                 'Mirriam' => 'Female')
```

Descriptive captions used as array element access key

# Multidimensional Array

- A **multidimensional array** is an array that contains one or more arrays as its elements.
- Each array element in a multidimensional array can also be an array, forming a hierarchy of arrays.
- This can be useful when working with complex data structures such as tables, matrices, or nested lists.
- In PHP, you can create a multidimensional array by defining an array with square brackets and nesting other arrays inside it.

# Multidimensional Array

- For example, the following code creates a two-dimensional array that contains the names and ages of three people:
- `$people = array(`
- `array("Alice", 25),`
- `array("Bob", 30),`
- `array("Charlie", 20)`
- `);`
- Here, `$people` is a two-dimensional array with three sub-arrays, each representing a person. Each sub-array has two elements, the first element being the name of the person and the second element being their age.

# Multidimensional Array

- To access an element of a multidimensional array, you need to specify the index for each level of the array. For example, to access the age of the second person in the array, you would use the following code:
- `echo $people[1][1]; // Output: 30`
- Here, `$people[1]` refers to the second sub-array, which contains the name and age of Bob.
- `$people[1][1]` refers to the second element of this sub-array, which is Bob's age.

# Multidimensional Array

- You can also loop through a multidimensional array using nested loops. For example, the following code loops through the \$people array and displays each person's name and age:

```
• foreach ($people as $person) {  
    echo "Name: " . $person[0] . ", Age: " . $person[1] .  
    "<br>";  
• } This will output:  
• Name: Alice, Age: 25  
• Name: Bob, Age: 30  
• Name: Charlie, Age: 20
```

# Array Functions

1. **count()**: This function is used to count the number of elements in an array.
  - It takes one argument, which is the array to be evaluated, and returns the number of elements in that array. For example:
  - `$fruits = array("apple", "banana", "cherry");`
  - `echo count($fruits); // Output: 3`

# Array Functions

2. **array\_push()**: This function is used to add one or more elements to the end of an array.
  - It takes two or more arguments: the original array and the element(s) to be added to the end of that array.
  - For example:
  - \$fruits = array("apple", "banana", "cherry");
  - array\_push(\$fruits, "orange", "kiwi");
  - print\_r(\$fruits); // Output: Array ( [0] => apple [1] => banana [2] => cherry [3] => orange [4] => kiwi )

# Array Functions

3. **array\_pop()**: This function is used to remove the last element from an array.
  - It takes one argument, which is the array from which the last element is to be removed. For example:
  - \$fruits = array("apple", "banana", "cherry");
  - array\_pop(\$fruits);
  - print\_r(\$fruits); // Output: Array ( [0] => apple [1] => banana )

# Array Functions

- 4) **array\_merge()**: This function is used to merge two or more arrays into a single array.
- The syntax for using the array\_merge() function is `array_merge($array1, $array2, ...)`.
  - Here, \$array1, \$array2, etc. are the arrays that need to be merged.
  - `$fruits1 = array("apple", "banana", "cherry");`
  - `$fruits2 = array("orange", "kiwi");`
  - `$fruits = array_merge($fruits1, $fruits2);`
  - `print_r($fruits); // Output: Array ( [0] => apple [1] => banana [2] => cherry [3] => orange [4] => kiwi )`

# Array Functions

- 5) **array\_search()**: This function is used to search a given array for a specific value and return the corresponding key if found.
- The syntax for using the array\_search() function is `array_search($value, $array)`. Here, `$value` is the value that needs to be searched and `$array` is the array in which the search needs to be performed.

```
$fruits = array("apple", "banana", "cherry");
```

```
$key = array_search("banana", $fruits);
```

```
echo $key; // Output: 1
```