IT430 E-Commerce Short Notes Lecture `1 to 18`

By ⁺˚⋆｡ Mʀ Ֆĥāĥłʒāʐ°⁺˚⋆｡

Lecture 1

E-COMMERCE

E-Commerce Classification:

1. Business-to-Business (B2B): Electronic transactions between organizations.
2. Business-to-Consumer (B2C): Retail transactions with individual shoppers.
3. Consumer-to-Consumer (C2C): Direct sales between consumers, often through classified ads or auction sites.
4. Consumer-to-Business (C2B): Individuals selling products or services to organizations.
5. Intra-Business (Organizational) EC: Internal activities within an organization, such as selling corporate products to employees or online training.
6. Non-Business EC: Academic institutions, not-for-profit organizations, religious/social organizations, and government agencies using EC to improve operations and customer service.

Basic Definitions:

1. Web client: A machine that initiates internet requests.
2. Web server: A machine that services internet requests.
3. Browser: Software on the client side used to interact with web data.
4. Intranet: An internal network of computers confined to a single place.
5. Extranet: Connection of two or more intranets, forming an Extranet (e.g., Virtual Private Network).
6. Internet: A global network of networks.

Client-Server Model:

The internet follows a two-way client-server communication model.

The Web:

1. The Web is a protocol that uses the internet for communication.
2. It links documents stored in computers communicating on the internet.
3. It is based on Hypertext Transfer Protocol (HTTP) for making web page requests.
4. HTTP follows a four-step process per transaction: Client makes an HTTP request, server accepts the request, server sends the page at HTTP, and client downloads the page.

Side Effects of HTTP Transfers:

1. All web transactions are recorded in a file called a common log file.
2. User data and site access information are recorded in log files.
3. User privacy may not be maintained.

Utilizing Data from Log Files or what can you do with this data

1. Rearrange your site based on popularly accessed and ignored portions.
2. Adjust marketing strategies based on user behavior.
3. Create a mailing list for targeted marketing based on user location data.

Lecture 2

WHAT IS A NETWORK

Network

A network can range from a simple collection of computers connected through a connectivity media to the internet, which is a global network of networks.

Local Area Network (LAN) is a server-based network confined to a specific area or place.

Reasons for Networking Computers:

1. File sharing.
2. Hardware sharing, such as printer sharing.
3. Program sharing.
4. User communication through an email server.

Network Protocol:

- Network protocols are standard rules that enable computers on a network to communicate and exchange data.
- A group of protocols that prepare data for communication is called a protocol stack.

ISO OSI Model:

- ISO's OSI model is a conceptual model for network communications.
- It proposes a 7-layer architecture: Application, Presentation, Session, Transport, Network, Data Link, and Physical.000000000

The OSI (Open Systems Interconnection) model is a conceptual model proposed by the International Organization for Standardization (ISO) for network communication. It consists of seven layers:

1. Application layer:

Handles high-level protocols and interfaces for user applications, such as file transfer (FTP) and web browsing (HTTP).

2. Presentation layer:

Translates, encrypts, and formats data for effective communication between different systems. It defines file formats and data types using protocols like MIME (Multipurpose Internet Mail Extensions).

3. Session layer:

Establishes, maintains, and terminates sessions between devices. It manages the communication link and provides synchronization and check pointing mechanisms.

4. Transport layer:

Ensures reliable and orderly delivery of data between devices. It provides services like segmentation, flow control, and error correction using protocols such as TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

5. Network layer:

Handles logical addressing, routing, and packet forwarding. It assigns IP addresses to data packets and uses protocols like IP (Internet Protocol) and routing protocols (e.g., RIP - Routing Information Protocol).

6. Data link layer:

Organizes data packets into frames and manages physical addressing and error detection. It uses protocols like Ethernet and MAC (Media Access Control).

7. Physical layer:

Transmits raw data bits over the physical medium, such as cables or wireless signals. It deals with the electrical, mechanical, and functional aspects of data transmission.

8. TCP/IP Stack:

The TCP/IP stack is an internet communication model with a 4-layer architecture.

Equivalent layers: Application (combining Application, Presentation, and Session), Host-to-Host (Transport), Internet (Network), Network Access (Data Link).

9. IP Addressing:

- IP addressing is the logical addressing scheme used to identify computer machines on the internet.
- Each computer has a unique IP address represented by four decimal numbers separated by periods (e.g., 140.57.220.200).

10. Classes of IP Address:

- There are five classes: A, B, C, D, and E.
- Classes A, B, and C are for general public use, while D and E are used by special groups.
- The class is determined by looking at the number in the first byte of the IP address.

Lecture 3

HOW MANY CLASS A, B, C NETWORKS AND HOSTS ARE POSSIBLE?

- Subnet Mask:
  A subnet mask accompanies an IP address and determines the network address and host machine address. The default subnet masks for Class A, B, and C IP addresses are:

- Class A: 255.0.0.0

- Class B: 255.255.0.0

- Class C: 255.255.255.0

- IP Version:
  IP version 4 (IPv4) is likely to be replaced by IP version 6 (IPv6). IPv6 provides 128-bit IP addresses in hexadecimal format, offering a significantly larger address space of approximately 3.4 x 10^38 addresses.

- Domain Name System (DNS):
  DNS is a system that translates user-friendly domain names (e.g., vu.edu) into their corresponding IP addresses. It provides the structure and strategy for referring to computers on the internet using domain names. Domain names are unique and assigned upon registration. Fully qualified domain names consist of a top-level domain (TLD), second-level domain (SLD), and sub domains.

- Name Resolution:
  The translation of a fully qualified domain name into an IP address is done using Domain Name Servers (DNS). A DNS server maintains a database/table of domain names and their corresponding IP addresses. The name resolution process involves the browser's resolver sending a request to the local name server, which contacts the root name server and traverses the hierarchy to obtain the IP address of the requested web server.

- Domain Name Registration:
  Domain names are administered in a hierarchy supervised by the Internet Corporation for Assigned Names and Numbers (ICAAN). Registration/administration is carried out by organizations such as APNIC, ARIN, and RIPE-NCC in different regions. Internet Service Providers (ISPs) at the local level can also register domain names and lease IP addresses. DHCP servers can dynamically assign IP addresses to clients for a session.

- Media Access Control (MAC) Address:
  A MAC address is a 48-bit unique hardware address assigned to a network interface card (NIC) installed in a computer. It is represented by 12 hexadecimal characters (e.g.,

09:00:17:A9:B2:EF). Unlike IP addresses, MAC addresses remain the same unless the NIC is replaced. The Institute of Electrical and Electronic Engineers (IEEE) administers the allocation of MAC addresses globally to prevent duplication.

Lecture 4

NETWORKING DEVICES

Networking devices:

Hubs:
They provide a central connection point for a LAN, organizing cables and relaying data signals to all computers.

Repeaters:
They regenerate signals, amplifying the entire electric signal they receive. However, they have no capabilities of directing network traffic.

Bridges:

They help conserve bandwidth by dividing larger networks into smaller segments. Bridges read the MAC address of a computer on data packets, match it with the stored MAC address in their table, and send the packet to the corresponding segment.

Switches:
Like bridges, switches increase the bandwidth of a network. They virtually divide a network into smaller segments called Virtual LANs (VLANs) and send data packets only to the intended machines within the corresponding VLAN.
Routers:
Routers are more efficient and sophisticated than bridges and switches. They can divide large networks into logical segments called subnets based on IP addressing. Routers build a routing table to determine the shortest possible path for delivering data packets to their destinations.

Cabling options:

Cooper-based cables:
Commonly used for connecting computers, with two main types: coaxial and twisted pair (further divided into unshielded twisted pair - UTP, and shielded twisted pair - STP).

Fiber optic cables:
Used on the internet, delivering data at high speeds using glass or plastic filaments and pulses of light as the data transfer method.

Telephone and electrical wire networks
These can also be used for connectivity purposes.

Wireless options:
Includes radio connectivity (Wi-Fi, Bluetooth), infrared connectivity, and satellite microwave transmissions.

Address Resolution Protocols (ARP) and Reverse Address Resolution Protocol (RARP)
Each computer on the internet prepares a list of its IP address and corresponding MAC address using ARP. RARP forwards this information to a network server. When a data packet arrives at a destination router, it inquires about the corresponding MAC address from the network server, inserts it, and delivers the packet to that MAC address. Both IP and MAC addresses are required for a data packet to reach its destination.

Role of ISPs on the internet:
ISPs provide internet connections, web hosting, email services, and newsgroups. They operate in a hierarchical structure, with the top-level ISP connected to the internet backbone. ISPs are connected to the internet backbone at Network Access Points (NAPs), allowing them to move data. The routing of data packets takes place on the internet backbone, which interconnects various ISPs and network providers worldwide.

Lecture 5

BASICS OF HTML

HTML (Hypertext Markup Language) is a language used for creating web documents. It uses tags enclosed in angular brackets to define the structure, formatting, and content of a web page. Here are some basics of HTML:

Email Address:
An email address like "john@hotmail.com" is an example of an email address. The domain part of the email address (hotmail.com) is translated into the IP address of the email server through DNS (Domain Name System).

Email Protocols:
There are three commonly used protocols for emails. SMTP (Simple Mail Transfer Protocol) is used for sending email messages between servers. POP3 (Post Office Protocol 3) and IMAP (Internet Message Access Protocol) are used to retrieve messages from the server. POP3 downloads the email to the client machine and deletes it from the server, while IMAP allows accessing the messages from multiple locations without deleting them from the server.

HTML Structure:
An HTML document starts with the `<HTML>` tag and ends with the `</HTML>` tag. The content of the web page is contained within the `<body>` tags. The `<head>` tag can be used to provide additional information about the web page, such as the title.

Text Formatting:
HTML provides various tags for formatting text. Examples include the `<p>` tag for paragraphs, `<br>` tag for line breaks, `<b>` tag for bold text, `<i>` tag for italic text, `<u>` tag for underlined text, `<big>` and `<small>` tags for font sizes, and `<font>` tag for specifying font properties.

Headings:
HTML provides six levels of headings, from `<H1>` (largest) to `<H6>` (smallest). Headings are used to structure and organize content on a web page.

List in HTML: HTML allows creating both ordered and unordered lists. The `<ul>` and `<ol>` tags are used for unordered and ordered lists, respectively. List items are defined using the `<li>` tag.

Images:
Images can be added to an HTML document using the `<img>` tag. The `src` attribute is used to specify the image file's location, and other attributes like `width`, `height`, and `align` can be used to control the image's display.

Downloading Graphics:

To download an image from the web, right-click on the image and choose the "Save Picture" or "Save Picture As" option. The image can then be referenced in an HTML document using the `<img>` tag.

Lecture 6

BASICS OF HTML

Hypertext links are used to connect HTML documents. Text and images can be made into links. The <a> tag is used for creating links, and the "href" attribute specifies the URL or file name to be opened when the link is clicked.

Example:

```
<HTML>
 <BODY>
  <H1>Computer Science</H1>
  Welcome to <A HREF="http://www.vu.edu">Virtual University</A> in Pakistan
 </BODY>
</HTML>
```

To create an email link

Use the "mailto" syntax within the <a> tag's "href" attribute.

Example:

```
<A HREF="mailto:vtv@hotmail.com">email address</A>
```

Changing colors in a page

Colors in a web page can be specified using the "bgcolor" attribute in the <body> tag. The"text"attribute sets the font color, and the "link" attribute sets the color of hyperlinks.

Example:

```
<BODY bgcolor="Green" text="white" link="red">
```

- Colors can also be specified using RGB values in hexadecimal format, such as "#00FF00" for pure green.
- The <table> tag is used to create tables in HTML. Tables are built row-by-row, and the basic tags used are <table>, <tr> (table row), <td> (table data/cell), and <th> (table header). The <caption> tag can be used to provide a caption for the table.

Example:

```
<HTML>
  <BODY>
    <TABLE Border=1>
      <TR>
        <TD>Cell1</TD>
        <TD>Cell2</TD>
     </TR>
      <TR>
        <TD>Cell3</TD>
        <TD>Cell4</TD>
      </TR>
    </TABLE>
  </BODY>
</HTML>
```

6. The <td> tag can have attributes such as "colspan" and "rowspan" to span cells across multiple columns or rows, respectively.

7. Various attributes can be used to customize tables, such as "border" to set cell boundaries, "width" and "height" to control the table dimensions, "cellpadding" to specify the space between cells, and "bgcolor" or "background" to set the background color or image for the table.

8. Attributes can be used with <td> tags to control the width, alignment, and vertical alignment of cells.

9. Tables can be used for page layout. For example, a left-hand margin can be created using a table with two columns, where the first column serves as the margin and the second column contains the page content.

Example:

```
<HTML>
  <BODY>
    <TABLE>
      <TR>
        <TD BACKGROUND="image2.gif" WIDTH="100">&nbsp;</TD>
        <TD VALIGN="TOP">This section contains the contents of your web page.</TD>
      </TR>
```

```
   </TABLE>
  </BODY>
</HTML>
```

10. Forms in HTML are used to gather information from users. The <form> tag is used to create a form, and the "action" and "method" attributes specify where the form data should be sent.

11. Different types of form elements include submit and reset buttons, text boxes, text areas, checkboxes, radio buttons, and lists.

12. Submit and reset buttons are created using the <input> tag with the "type" attribute set to "submit" or "reset", respectively.

13. Text boxes are created using the <input> tag with the "type" attribute set to "text". Text areas are created using the <textarea> tag.

Example:

```
<FORM ACTION="http://www.forms.com" METHOD="post">
  <INPUT TYPE="submit" value="label">
  <INPUT TYPE="reset" value="label">
</FORM>
```

Lecture 7

TEXT BOXES, CHECK BOXES, RADIO BUTTONS

Text Boxes:

- Text boxes are input fields where users can enter text or information.
- The 'value' attribute in the input tag pre-fills the text box with the specified value.

Example:

```
`<input type="text" name="URL" value="http://">`
```

Checkboxes:

- Checkboxes are used for yes/no or true/false selections.
- The 'type' attribute is set as "checkbox" in the input tag.
- The 'checked' attribute can be used to pre-select certain checkboxes.
- Each checkbox should have a unique 'name' attribute to indicate the field name.

Example:

```
<input type="checkbox" name="Water">Fear of water<br>
<input type="checkbox" name="Bald">Fear of becoming bald<br>
<input type="checkbox" name="Lock">Fear of being locked inside<br>
<input type="checkbox" name="Flying" checked>Fear of flying<br>
```

Radio Buttons:

- Radio buttons allow users to select one option from a set of choices.
- All radio buttons belonging to the same group should have the same 'name' attribute.
- The 'value' attribute specifies the value of the option when it is selected.
- The 'checked' attribute can be used to pre-select a radio button.

Example:

```
<input type="radio" name="Income" value="Poverty">Living below the poverty line<br>
<input type="radio" name="Income" value="Middle" checked>Living at the level of the middle
class<br>
<input type="radio" name="Income" value="Upper">Living at the level of the upper class<br>
```

Selection Lists:

- Selection lists allow users to choose an option from a dropdown list.
- The 'select' tag is used, with the 'name' attribute specifying the field name.
- The 'size' attribute sets the number of visible items in the list.
- The 'option' tags define the list items within the 'select' tag.
- The 'selected' attribute can be used to pre-select an item.
- The 'multiple' attribute allows users to select multiple items.

- Example:

```
<select name="Nationality" size="4">
  <option>American</option>
  <option>Australian</option>
  <option>Hungarian</option>
  <option>Indian</option>
  <option selected>Pakistani</option>
  <option>French</option>
</select>
```

Lecture 8

FRAMES AND IMAGES IN HTML

Frames in HTML allow you to divide the browser window into multiple sections and display different web pages within each section.
There are two types of pages involved in frames
Frameset pages Frameset pages define the size and arrangement of frames
Content pages Content pages are regular HTML pages displayed within frames.

To divide the screen horizontally, you can use the following code:

```
<HTML>
<HEAD>
<TITLE>Horizontal Frames</TITLE>
</HEAD>
<FRAMESET ROWS="25%,75%">
<FRAME>
<FRAME>
</FRAMESET>
</HTML>
```

This code creates a frameset with two rows, where the first frame takes up 25% of the screen height and the second frame takes up 75%. The `<FRAME>` tags represent the individual frames.

To divide the screen vertically, you can use the following code:

```
<HTML>
<HEAD>
<TITLE>Vertical Frames</TITLE>
</HEAD>
<FRAMESET COLS="20%,60%,20%">
<FRAME>
<FRAME>
<FRAME>
</FRAMESET>
</HTML>
```

This code creates a frameset with three columns, where the first and third frames each take up 20% of the screen width, and the second frame takes up 60%.
Frames can also contain content. You can specify the content pages to be displayed within frames using the `SRC` attribute of the `<FRAME>` tag. Here's an example:

```
<HTML>
<HEAD>
```

```
<TITLE>Horizontal Frames with Content</TITLE>
</HEAD>
<FRAMESET ROWS="25%,75%">
<FRAME SRC="1.htm" Name="upper">
<FRAME SRC="2.htm" Name="lower">
</FRAMESET>
</HTML>
```

In this example, the frameset contains two frames named "upper" and "lower." The `SRC` attribute specifies the content pages to be displayed in each frame.

Frames can also be nested, where a frame contains another set of frames. Here's an example:

```
<HTML>
<HEAD>
<TITLE>Nested Frames</TITLE>
</HEAD>
<FRAMESET ROWS="25%,75%">
<FRAME SRC="1.htm" NAME="upper">
<FRAMESET COLS="50%,50%">
<FRAME SRC="2.htm" NAME="lower">
<FRAME SRC="3.htm" NAME="right">
</FRAMESET>
</FRAMESET>
</HTML>
```

In this example, the "upper" frame is divided into two columns, and the "lower" frame is further divided into two rows.

Images in HTML can also be made clickable. You can turn an image into a link by wrapping it with an `<A>` tag. Here's an example:

```
<HTML>
<HEAD>
<TITLE>Images Can Be Links, Too</TITLE>
</HEAD>
<BODY>
Click this house <A HREF="main.htm"><IMG SRC="home.gif"></A> to return to my home page.
</BODY>
</HTML>
```

In this example, the image of a house is wrapped in an `<A>` tag, and clicking on the image will take you to the "main.htm" page.

Additionally, HTML supports image maps, which allow you to define clickable areas within an image. You can assign different links to specific regions of an image. Image maps use the `<MAP>` and `<AREA>` tags. Here's an example:

```
<HTML>
<HEAD>
<TITLE>Image Map Example</TITLE>
</HEAD>
<BODY>
<IMG SRC="image.gif" usemap="#Testmap">
<MAP Name="Testmap">
<AREA shape="Rect" coords="0,0,199,99" href="a.htm">
<AREA shape="Rect" coords

="199,0,399,99" href="b.htm">
<AREA shape="Rect" coords="399,0,599,99" href="c.htm">
</MAP>
</BODY>
</HTML>
```

In this example, the image is divided into three rectangular areas using the `<AREA>` tag, and each area is assigned a different link.

You can experiment with different shapes like circles and polygons using the `shape` attribute of the `<AREA>` tag. The `coords` attribute specifies the coordinates of the shape's boundary.

Lecture 9

TAG ATTRIBUTES, SOUNDS FILES, ANIMATIONS

<Area> tag attributes
- The 'alt' attribute is used to label the clickable region.
- The 'target' attribute can be used to display the result in a given frame.
- The 'nohref' attribute can be used to make a region non-clickable.

Sound Files

- Sound file formats commonly used are AU, WAV, MP3, and MIDI.
- Sound files can be used in the anchor tag (<a>) by specifying the file in the 'href' attribute.
- -Sound files can also be embedded using the <embed> tag.

Animated GIF images

- Animated GIFs are created by displaying a sequence of individual image files.
- Programs like GIF Animator can be used to specify the sequential display of images as an animation.

- Animated GIFs are added to web pages using the <img> tag, where the 'src' attribute points to the GIF file.

<Marquee> tag

- The <marquee> tag is used to create scrolling or sliding text effects.
- Attributes such as 'align', 'direction', 'behavior', 'loop', 'scrollamount', 'width', 'height', and 'bgcolor' can be used to customize the marquee.

Style sheets

- Cascading Style Sheets (CSS) are used to define the presentation and formatting of HTML elements.
- CSS can be applied in various ways, including embedded method, linking to an external style sheet, and inline styles.

Method #1: Embedded method

- Style definitions are placed between <style> and </style> tags within the HTML document.

Method #2: Linking to an external style sheet

Style definitions are stored in a separate .css file, and the <link> tag is used to reference the external style sheet.

Method #3: Inline styles

Styles are applied directly to specific HTML tags using the 'style' attribute.

Lecture 10

STYLE SHEETS

Using style sheets in HTML allows you to control the appearance of your web page. Here are some examples of different style sheet techniques:

Inline Style:

You can apply a specific style directly to an element using the `style` attribute. For example:
<h1 style="font-size: 40pt; color: red;">This heading has a font size of 40 points</h1>

Style Classes:

You can define a style class within the `<style>` tag or an external style sheet file. Then, you can apply the class to multiple elements using the `class` attribute. For example:

```
<style>
 .Text1 {
   font-size: 20pt;
   color: red;
   text-align: center;
 }
 .Text2 {
   font-size: 16pt;
   color: green;
   text-align: center;
 }
</style>

<div class="Text1">
 <h1>This text is in red</h1>
</div>
<div class="Text2">
 <h2>This text is in green</h2>
</div>
```

Font Sizes

You can control the font size using the `font-size` property. For example:

```
<span style="font-size: xx-small;">Physics</span>
<span style="font-size: x-small;">Math</span>
<span style="font-size: small;">Computer Science</span>
<span style="font-size: medium;">Literature</span>
<span style="font-size: large;">Ecommerce</span>
<span style="font-size: x-large;">History</span>
<span style="font-size: xx-large;">Islam</span>
```

Font Families

You can specify different font families using the `font-family` property. For example:

```
<style>
 .fonttype1 {
   font-size: 20pt;
   font-family: "Times New Roman";
 }
 .fonttype2 {
   font-size: 20pt;
   font-family: Arial;
 }
 .fonttype3 {
   font-size: 18pt;
   font-family: "Courier New";
 }
</style>

<div class="fonttype1">This sentence is in Times New Roman</div>
```

<div class="fonttype2">This one is in Arial Font Type</div>
<div class="fonttype3">This is in Courier New Font Type</div>

Text Decoration

You can apply different decorations to text using the `text-decoration` property. For example:
```html
<h2 style="text-decoration: none;">Introduction to Ecommerce</h2>
<h2 style="text-decoration: line-through;">Introduction to E-commerce</h2>
<h2 style="text-decoration: overline;">Introduction to E-commerce</h2>
<h2 style="text-decoration: underline;">Introduction to E-commerce</h2>
```

Text Transform

You can transform the case of text using the `text-transform` property. For example:
```html
<h2 style="text-transform: capitalize;">We love Pakistan</h2>
<h2 style="text-transform: lowercase;">We love Pakistan</h2>
<h2 style="text-transform: none;">We love Pakistan</h2>
<h2 style="text-transform: uppercase;">We love Pakistan</h2>
```

Background Color and Image

You can set the background color or image using the `background-color` and `background-image` properties. For example:
<div style="background-color: yellow; color: blue;">This is an example of some blue text on a yellow background using style sheets.</div>

<div style="background-color:

Lecture 11

STYLE SHEETS

Style Sheet Box Model:
- The box model is a concept that represents each block element (e.g., `<p>`, `<h1>`) as having an invisible box around it.
- The box consists of content, padding, border, and margin.
- By applying specific properties, such as dimensions, padding, border, and margin, you can control the appearance of the elements.

Dimension Example:
- The `<p>` element can be styled using the `border` property with the value of `thin solid` to add a border around it.
- Using the `height` and `width` properties, you can define the height and width of the `<p>` element to create a box of specific dimensions.

Padding:
- Padding adds space around the content inside the box.
- There are specific properties for each side of the box: `padding-top`, `padding-right`, `padding-bottom`, and `padding-left`.
- You can also use the shorthand property `padding` to apply padding to all four sides in the order of top, right, bottom, and left.

Border:
- Borders can be customized with properties such as `border-width`, `border-style`, and `border-color`.
- You can specify the width, style, and color of each side of the border or use the shorthand property `border` to set all properties at once.

Margins:
- Margins define the space outside the box.
- Similar to padding, you can use properties like `margin-top`, `margin-right`, `margin-bottom`, and `margin-left` to set margins for each side.
- The shorthand property `margin` can be used to set all margins in the order of top, right, bottom, and left.

Position Styles:
- The position property allows you to control the positioning of elements.
- There are different position values, including `absolute`, `relative`, and `static`.
- You can use properties like `top`, `right`, `bottom`, and `left` to specify the position of an element from different sides.
- Absolute positioning allows you to place the element anywhere on the page, relative positioning positions the element relative to its normal position, and static positioning keeps the position unchanged.

Lecture 12

SOME USEFUL STYLE SHEETS PROPERTIES

Introduction to java script
JavaScript is a programming language that is used to create dynamic and interactive web pages. It is a client-side scripting language, which means that it is executed by the user's browser. JavaScript can be used to add functionality to web pages, such as animations, games, and forms.

Variables are used to store data in JavaScript. A variable is a named location in memory that can be used to store a value. Variable names can be any combination of letters, numbers, and underscores.

In example, the variable is declared and initialized with the value 3. The value 3 is then stored at the location 65895 in memory.

Rules for variables
A variable is a named location in memory that can be used to store data. In JavaScript, variable names can begin with an uppercase letter (A-Z), lower case letter (a-z), an underscore character (_), or dollar sign

($). Remaining characters can be any of the above or from digits (0-9). In JavaScript variables are case sensitive. It means that if you have a variable ‘money’ you cannot write ‘Money’ or ‘mONEY’. You don’t need to define the variable with the data type (rather it is not allowed in JavaScript)

General data types
There are five general data types in JavaScript:

Integer: An integer is a whole number, such as 1, 2, 3, 4, etc.

Float: A float is a decimal number, such as 1.5, 2.3, 3.14, etc.

Char: A char is a single character, such as a letter, number, or symbol.

String: A string is a sequence of characters, such as "Hello world!"

Boolean: A boolean is a value that can be either true or false.

Event handler

An event handler is a function that is called when a specific event occurs. For example, an event handler could be called when a user clicks a button, enters text into a form, or moves the mouse over an image. Event handlers are used to make web pages interactive.

Changing images using event handlers
The following code will change the image that is displayed when the user moves the mouse over the image.

```
<img src="image1.jpg" onmouseover="this.src='image2.jpg'">
```
Use code with caution. Learn more
When the user moves the mouse over the image, the onmouseover event will be triggered. The this keyword refers to the current object, which in this case is the image. The src property of the image is then set to image2.jpg, which is the new image that will be displayed.

Lecture 13

JAVA SCRIPTING

Java script examples

```
<HTML>
<HEAD>
<TITLE>Javascript Example</TITLE>
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
alert("Thank you for visiting my web site!")
```

```
//-->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```

In this example, the JavaScript code is enclosed within the `<script>` tags. The `alert()` function is a built-in function provided by the browser's window object. It creates an alert box or popup message with the specified message, in this case, "Thank you for visiting my web site!"

The `<!--` and `//-->` are comment delimiters used in JavaScript. In this case, the `//-->` is added to ensure that old browsers that do not support JavaScript treat the content within the `<script>` tags as comments and do not display any error messages.

When this code is executed, it will display an alert box with the message "Thank you for visiting my web site!" as soon as the HTML document is loaded in the browser.

In second example shows how to write text directly onto the web page using JavaScript:

```
<HTML>
<HEAD>
<TITLE>example-writing on the page</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
document.write("Hello! Thank you for visiting my web site.")
//-->
</SCRIPT>
</BODY>
</HTML>
```

In this example, the `document.write()` function is used to write the specified text directly onto the web page. The text "Hello! Thank you for visiting my web site." will be displayed on the page when the JavaScript code is executed.

The `<script>` tags are used to enclose the JavaScript code. The `<!--` and `//-->` comment delimiters serve the same purpose as in the previous example, to handle compatibility with older browsers.

When the HTML document is loaded and the JavaScript code is executed, the text "Hello! Thank you for visiting my web site." will be written onto the page at the location where the JavaScript code is placed.

Both examples demonstrate the use of JavaScript code embedded within an HTML document to enhance the functionality and interactivity of the web page.

Operators in java script

In JavaScript, there are various operators that can be used to perform mathematical and logical operations. Here's an explanation of the operators.:

Mathematical Operators:

- `+`: Adds two values.
- `-`: Subtracts the second value from the first.
- `*`: Multiplies two values.
- `/`: Divides the first value by the second.
- `%`: Calculates the remainder of the division.
- `++`: Increments a variable by 1.
- `--`: Decrements a variable by 1.

Comparison Operators:

- `==`: Checks if two values are equal.
- `!=`: Checks if two values are not equal.
- `<`: Checks if the first value is less than the second.
- `<=`: Checks if the first value is less than or equal to the second.
- `>`: Checks if the first value is greater than the second.
- `>=`: Checks if the first value is greater than or equal to the second.

Functions in JavaScript:

- A variable is defined using the `var` keyword, which is used to store data.
- A function is defined using the `function` keyword, followed by the function name and code enclosed in curly braces.
- Functions allow you to group and organize code that can be executed when called.

Creating a calculator in JavaScript:

- The HTML code sets up a form with two input fields for numbers and buttons for addition and subtraction.
- The JavaScript code defines two functions, `Addit()` and `minus()`, which are triggered when the buttons are clicked.
- The functions retrieve the values entered in the input fields, perform the respective mathematical operation, and display the result using the `alert()` function.

Lecture 14

JAVA SCRIPTING (CONTINUED….)

```
<HTML>
<script language="JavaScript">
<!--
function Addit()
{

var
num1=document.Calform
.One.value; var
num2=document.Calform
.Two.value;
document.write("The result of this addition is " +

(parseFloat(num1)+parse
Float(num2))); } function
minus()
{ var num1=document.Calform.One.value; var num2=document.Calform.Two.value;
document.write("The result of this subtraction is " + (parseFloat(num1)
parseFloat(num2)));
}
//-->
</script>
<BODY>
<FORM name="Calform">

        <P>
First Number:<INPUT TYPE="TEXT" NAME="One" maxlength="3">

        <P>
Second Number:<INPUT TYPE="TEXT" NAME="Two" maxlength="3">

        <P>
<INPUT TYPE="button" NAME="Add" VALUE="Add Them!!" onclick="Addit()">
<INPUT TYPE="button" NAME="Minus" VALUE="Subtract
Them!!" onclick="minus()">
<INPUT TYPE="RESET" VALUE="Reset!">
</FORM> </BODY>
</HTML>
```

- The code represents a basic web page with two text input fields, two buttons, and a reset button.
- The user can enter numbers in the text boxes and perform addition or subtraction operations.
- The `Addit()` function is called when the "Add Them!!" button is clicked.
- It retrieves the values entered in the text boxes using `document.Calform.One.value` and `document.Calform.Two.value`.
- The values are converted to floating-point numbers using `parseFloat()`.
- The addition of the two numbers is performed, and the result is displayed on the web page using `document.write()`.

- The `minus()` function is called when the "Subtract Them!!" button is clicked.
- It follows a similar process as `Addit()` but performs subtraction instead of addition.
- The results of the addition and subtraction operations are displayed on the web page after the button clicks.
- The `parseFloat()` function is used to convert the input values from strings to numbers before performing mathematical operations.
- The `document.write()` method is used to write content on the web page. It concatenates strings with the addition/subtraction results using the `+` sign.

Multiply and divide calculator

See the following code

```
<HTML>
<script language="JavaScript">
<!--
function Multiplyit()
{

var num1=document.Calform.One.value; var num2=document.Calform.Two.value;
alert(parseFloat(num1)*parseFloat(num2));
}
function Divideit()
{ var num1=document.Calform.One.value; var num2=document.Calform.Two.value;
alert(parseFloat(num1)/parseFloat(num2));
}
//-->
</script>
<BODY><h2> Multiply and Divide Calculator</h2>
<FORM name="Calform">

<P>
First Number:<INPUT TYPE="TEXT" NAME="One" maxlength="3">
<P>
Second Number:<INPUT TYPE="TEXT" NAME="Two" maxlength="3">
<P>
<INPUT TYPE="button" NAME="Multiply" VALUE="Multiply" onclick="Multiplyit()">
<INPUT TYPE="button" NAME="Divide" VALUE="Divide" onclick="Divideit()">
<INPUT TYPE="RESET" VALUE="Reset!">
</FORM>
</BODY>
</HTML>
```

Example - A Drop-Down List of Links

```
<HTML>
<HEAD>
<TITLE>A Drop-Down List of Links</TITLE>
</HEAD>
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
function GoToIt(list)
{ var selection = list.options[list.selectedIndex].value        if (selection != "None") location.href = selection
}
```

```
//-->
</SCRIPT>
<BODY>
<FORM>
<SELECT WIDTH="20" onChange="GoToIt(this)">
<OPTION VALUE="None">Select a page previously done --->
<OPTION VALUE="calculator.htm">Calculator
<OPTION VALUE="styles.htm">Style Sheet
<OPTION VALUE="forms.htm">Web Forms
<OPTION VALUE="tablemargin.htm">Table Margins
<OPTION VALUE="frames.htm">Frames
</SELECT>
</FORM>
</BODY>
</HTML>
```

Explanation of above example

1. Multiply and Divide Calculator:

- This code represents a web page with two text input fields, two buttons for multiplication and division, and a reset button.
- The user can enter numbers in the text boxes and perform multiplication or division operations.
- The `Multiplyit()` function is called when the "Multiply" button is clicked.
- It retrieves the values entered in the text boxes using `document.Calform.One.value` and `document.Calform.Two.value`.
- The values are converted to floating-point numbers using `parseFloat()`.
- The multiplication of the two numbers is performed, and the result is displayed in an alert using `alert()`.
- The `Divideit()` function is called when the "Divide" button is clicked.
- It follows a similar process as `Multiplyit()` but performs division instead of multiplication.
- The results of the multiplication and division operations are shown in alerts.
- The `<FORM>` element is used to encapsulate the input fields and buttons, and the `<INPUT>` tags define the form elements.
- The `onclick` attribute is used to specify the functions to be called when the buttons are clicked.
- The `maxlength` attribute is set to limit the input to three characters.
- The `Reset` button allows resetting the form to its initial state.

2. A Drop-Down List of Links:

- This code represents a web page with a drop-down list of links.
- The user can select an option from the list, and upon selection, the corresponding web page will open.
- The `GoToIt()` function is called when the selection in the drop-down list is changed.
- It retrieves the selected value using `list.options[list.selectedIndex].value`.

- If the selected value is not "None", the location of the web page is changed using `location.href` to the selected option's value.
- The drop-down list is defined using the `<SELECT>` element, and each option is specified using the `<OPTION>` tag.
- The `onChange` attribute is used to specify the function to be called when the selection changes.

Example - If Statement

IF statement in programming is used to alter the course of execution of code depending upon the value of a condition. See the following example:

```
<HTML>
<script language="JavaScript">
<!--
function minus()
{

 var
 num1=document.Calf
 orm.One.value;      var
 num2=document.Calf
 orm.Two.value;
 if(parseFloat(num1)<
 parseFloat(num2))

{ alert("negative");
}

else
{
  alert(parseFloat(num1)-parseFloat(num2));

}
}
//-->
</script>
<BODY> <FORM name="Calform"> <P> First Number:<INPUT
TYPE="TEXT" NAME="One" maxlength="3"> <P> Second
Number:<INPUT TYPE="TEXT" NAME="Two" maxlength="3"> <P>
<INPUT TYPE="button" NAME="Minus" VALUE="Subtract" onclick="minus()">
<INPUT TYPE="RESET" VALUE="Reset!">
</FORM>
</BODY>
</HTML>
```

Explanation

- This code represents a web page with two text input fields, a button for subtraction, and a reset button.
- The user can enter numbers in the text boxes and perform a subtraction operation.
- The `minus()` function is called when the "Subtract" button is clicked.
- It retrieves the values entered in the text boxes using `document.Calform.One.value` and `document.Calform.Two.value`.
- The values are converted to floating-point numbers using `parseFloat()`.
- An `if` statement is used to check if the first number is less than the second number.
- If the condition is true, i.e., the first number is less than the second number, an alert is displayed with the message "negative".
- If the condition is false, the subtraction of the two numbers is performed, and the result is displayed in an alert.

- The `else` block is executed when the condition in the `if` statement is false.
- The results of the subtraction operation or the "negative" message are shown in alerts.
- The `<FORM>` element is used to encapsulate the input fields and buttons, and the `<INPUT>` tags define the form elements.
- The `onclick` attribute is used to specify the function to be called when the "Subtract" button is clicked.
- The `maxlength` attribute is set to limit the input to three characters.
- The `Reset` button allows resetting the form to its initial state.

For LOOP

Example

```
<HTML>
<HEAD>
<TITLE>Using the For Statement</TITLE>
</HEAD>
<BODY>
<SCRIPT>
<!--
for(i=1;i<7;i++) document.write("<H"+i+">Hello "+i+"!!</H"+i+">");
//-->
</SCRIPT>
</BODY>
</HTML>
```

Explanation

For Loop

- The loop is used to repeat a set of statements until a specific condition is met.

- It consists of three parts: initialization statement, condition, and update statement.
- The code within the loop is executed repeatedly as long as the condition is true.
- In the given example, for loop is used to generate six different levels of headings (H1 to H6) in HTML.
- The loop variable `i` is initialized to 1, the loop continues as long as `i` is less than 7, and `i` is incremented by 1 in each iteration.
- The `document.write()` function is used to output the HTML headings.

Some predefined JavaScript objects

1 Global Object:

The Global object provides globally-accessible variables, properties, and functions.
Examples include `NaN` (not a number) property and the `parseFloat(string)` function/method for parsing a string as a floating-point number.

2. Array Object:

The Array object represents an array in JavaScript.
It has properties like `length`, which indicates the length of the array, and methods/functions like `toString()`, `reverse()`, and `sort()`.
The example demonstrates various operations on an array (`myArray`), such as accessing elements, converting to string, joining elements with a delimiter, reversing, and sorting.

Math Object:
The Math object provides a library of mathematical constants and functions.
Examples of its properties and methods include `Math.PI` (pi), `Math.LN2` (natural logarithm of 2), `Math.sin()` (sine function), `Math.random()` (random number between 0 and 1), `Math.pow()` (power function), and `Math.min()` (minimum value).

Lecture 15

JAVA SCRIPTING (CONTINUED….)

How to display the current date and time in a user-friendly format using JavaScript

- The code starts with the HTML structure, including the `<head>` and `<body>` tags.

- Inside the `<script>` tags, JavaScript code is written to manipulate the date object and display the desired output.

- The line `d = new Date()` creates a new instance of the `Date` object and stores it in the variable `d`. This object represents the current date and time.

- The variable `dateText` is initialized as an empty string. It will be used to store the formatted date and time.

- The code then retrieves the current day using the `getDay()` method of the `Date` object. The value returned represents the day of the week, with Sunday being 0 and Saturday being 6. Based on the day value, the corresponding day name is appended to the `dateText` variable using conditional statements.

- Similarly, the code retrieves the current month using the `getMonth()` method. The month value ranges from 0 to 11, representing January to December. Based on the month value, the corresponding month name is appended to the `dateText` variable.

- The code retrieves the current year using the `getYear()` method. If the year is before 2000, 1900 is added to it. The day of the month is also retrieved using the `getDate()` method.

- The code then retrieves the current minutes and hours using the `getMinutes()` and `getHours()` methods, respectively. It also customizes the greeting message based on the hour value.

- The code writes the greeting, date, and time to the webpage using the `document.write()` method.

- The result is a webpage that displays a greeting message, the current date, and the current time in a user-friendly format.

Using the String Object

This example focuses on demonstrating various string methods in JavaScript.

- The HTML structure is defined, including the `<head>` and `<body>` tags.

- Inside the `<script>` tags, JavaScript code is written to manipulate the string object and perform various operations.

- The code creates a new instance of the `String` object using the `new String()` syntax and assigns it to the variable `str`. The string used is "This is a test of JavaScript string methods".

- An array, `myArray`, is initialized with a size of 10.

- The `str.split(' ')` method is used to split the string into an array of substrings based on the space character. The resulting substrings are then assigned as values to the `myArray` array.

- The code uses different string methods such as `charAt()`, `substring()`, `toLowerCase()`, and `toUpperCase()` to manipulate the `str` string object.

- The results of these operations are displayed using the `document.write()` method.

Applying Checks in a Registration Form

This example showcases how JavaScript can be used to validate form inputs before submission.

- The HTML structure defines a form using the `<form>` element, with various input fields such as name, address, login, password, confirm password, and email.
- The `<script>` section contains JavaScript code to validate the form inputs before submitting.
- The `checkValues()` function is triggered when the form is submitted (`onSubmit="return checkValues()"`). It retrieves the values of different form fields using the `document.regForm.fieldName.value` syntax.
- The function performs several checks on the form inputs. It checks if the login name, address, and name fields are empty and displays an alert if any of them are not filled.
- It also checks if the length of the login name and password fields is within the specified limits. If the length exceeds the limits, it displays an alert with the corresponding message.
- The function further checks if the login name contains invalid characters ("," or ";") using the `charAt()` method.
- Finally, it compares the password and confirm password fields and displays an alert if they don't match.
- If all checks pass, the function returns `true`, allowing the form to be submitted. Otherwise, it returns `false`, preventing the form submission.
- These examples illustrate the usage of string methods and form validation using JavaScript.

Lecture 16

JAVA SCRIPTING AND XML

In a registration form, the "for" loop can be used to check if the user has entered invalid characters in the text box for their login. For example, if the user types a comma (",") in the login text box, an alert box can be displayed to inform them that it is an invalid login.

The "for" loop has three parts: initialization, condition, and increment. The initialization statement is executed at the beginning of the loop, the condition is tested, and if it is true, the statements inside the loop's curly brackets are executed. If the condition is false, the loop is terminated, and the code after the loop is executed.

Another check that can be applied is to compare the passwords entered in two different text boxes. If the passwords do not match, an alert box can inform the user about the inconsistency.

If none of the conditions in the code are violated, the "checkValues" function will return true when the form is submitted. This means that all the information provided by the user is valid, and it can be sent to the server for further processing.

Extensible markup language

XML (Extensible Markup Language) is a flexible language used for structuring and managing data, particularly on the web. It was introduced by the World Wide Web Consortium (W3C) as an alternative to HTML, providing more capabilities for data management.

Following is a simple HTML code for preparing the list of planets: (P75)

In the given example, we have an HTML code that represents a list of planets. The HTML code uses heading tags (<h1>, <h2>, <h3>, etc.) to structure the information. The hierarchy of the headings indicates the level of information. For example, the planet names are represented with <h2>, distance from the Sun with <h3>, number of moons with <h4>, and day length with <h5>.

Consider the example of a list of planets. In HTML, we would use predefined heading tags to structure the information, but HTML has a limitation of only six levels of headings. If we need to display more than six different pieces of information about each planet, HTML becomes less effective.

XML solves this problem by allowing users to create their own tags. XML is not limited to predefined tags like HTML; instead, we can define our own tags to represent different pieces of information. These tags convey the meaning of the data rather than specifying how it should appear on a web page.

For example, in XML, we can create a root element called "PlanetsList" and enclose other elements within it. Each planet can be represented as a child element of "PlanetsList," and each property of a planet can be represented as a child element of the planet element. We can also assign attributes to elements, such as the name of a planet.

XML allows users to define their own tags, there can be variations in naming conventions between different organizations. To ensure consistency and facilitate data sharing, common standards for XML tags have been developed. These standards, known as Data Type Definitions (DTD) or XML schemas, provide a common structure and naming conventions for specific industries or types of documents.

To write XML code, you should follow certain rules and guidelines. Here are the rules for writing XML code:

Rules for writing an XML code

When writing XML code, it is important to follow certain rules and guidelines to ensure that the code is valid and can be correctly parsed. Here are some rules for writing XML code:

1. Proper nesting: All elements must be properly nested within each other. Each opening tag must have a corresponding closing tag. For example:

```
  <outer>
      <inner>content</inner>
  </outer>
```

2. Quoted attribute values: All attribute values must be enclosed in quotes, either single or double quotes. For example:

```
  <FRIES SIZE="LARGE">
```

3. Empty elements: Elements with empty content must be self-closed with a forward slash ("/") before the closing angle bracket. For example:

```
  <BR/>
  <img src="image2.gif" />
```

4. Consistent casing: XML is case-sensitive, so all elements must be consistently cased. Opening and closing tags must match in their casing. For example:

```
  <PART></PART>
```

5. Reserved characters: Certain characters have reserved meanings in XML and cannot be used directly. These characters include ampersand ("&"), less than ("<"), and greater than (">"). They must be replaced with their corresponding character entities. For example:

```
  <text>AT&T</text>
  <tag>&lt;example&gt;</tag>
```

Embedding XML into HTML documents can be done using the `<xml>` element or the `<script>` element with a language attribute set to "xml" and a type attribute set to "text/xml". Here's an example using the `<xml>` element:

```
<html>
<head>
<title>XML-example</title>
</head>
<body>
<h1>HTML text here</h1>
<xml>
<meal>
<burger>
<name>spicy</name>
</burger>
</meal>
</xml>
</body>
</html>
```

XML files can be processed by web servers using XML parsers. These parsers read the XML content and can be written in programming languages like Java. Extensible Stylesheet Language (XSL) is used to define formatting instructions for XML files. The XSL file is then read by the XML parser to transform the XML content into a different format, such as HTML. The process of a web server handling an HTTP request for an XML page involves parsing the XML content and applying the XSL formatting instructions to generate the desired output.

In the given example, the XML code represents a catalog with parts and their details. The XML code includes a reference to an XSL stylesheet (`catalog.xsl`) using the `xml-stylesheet` processing instruction. The XSL code provides formatting instructions for transforming the XML content into HTML.

Here code provided: (P77)

```
<?xml version="1.0"?>
<xsl>
  <rule>
    <root />
    <html>
      <body bgcolor="yellow">
        <children />
      </body>
    </html>
  </rule>
  <rule>
    <target-element type="PART" />
    <div style="margin-bottom:20px">
      <children />
    </div>
  </rule>
  <rule>
    <element type="PART">
      <target-element type="NAME" />
      <b>
        <children />
      </b>
      <br />
    </rule>
    <!-- Other rules -->
</xsl>
```

The XSL code uses `<rule>` elements to define formatting instructions for specific elements in the XML code. The `<root>` element is used to represent the root element of the XML code, and it will be transformed into an HTML structure with a yellow background color.

The `<target-element>` element is used to specify the XML element that the formatting instructions apply to. In the example, a `<rule>` is defined for elements with the type "PART". The corresponding formatting instructions transform these elements into `<div>` elements with a margin-bottom of 20 pixels.

Another `<rule>` is defined for elements with the type "PART", specifically targeting the "NAME" element. The formatting instructions transform the "NAME" element into a `<b>` (bold) element.

The `<br />` element is used to insert a line break within the transformed output.

HTML and XML editors

Regarding HTML and XML editors, you can use general-purpose text editors like Notepad, Wordpad, or any text editor that supports HTML and XML syntax highlighting. However, specialized editors like Macromedia Dreamweaver, Microsoft FrontPage, Epic Editor, or TurboXML can provide additional features and facilitate the editing process by offering a more user-friendly interface, code suggestions, and validation.

Lesson 17

CLIENT AND SERVER SIDEPROCESSING OF DATA

In the Client-Server Model of the internet, data processing occurs on both the client side and the server side.
Client-side processing involves the handling and processing of data on the client's device, typically within a web browser.

1. HTML code: The client-side processing includes interpreting and rendering HTML code by the browser. The browser understands different HTML tags and displays the resulting web page accordingly.

2. JavaScript: JavaScript code is executed on the client side by the browser. It is primarily used for tasks such as form validation, user interaction, dynamic content updates, and simple calculations. JavaScript functions can be embedded within HTML documents and are executed in the browser.

3. Applets: Applets are compiled Java programs that are sent from the server to the browser for processing. Examples of applets include animations, games, and interactive content. Applets require more space on the window screen and are executed within the browser.

4. Cookies: Cookies are text files stored on the client's hard disk. They are generated by JavaScript or CGI scripts on the server side. JavaScript handles cookie processing on the client side. Cookies can be used to store small amounts of data on the client's device and are often used for user tracking, maintaining state between sessions, and personalized recommendations.

```
<HTML>
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javaScript">
<!--
function updateCookie()
{
 document.cookie=document.form1.cookie.value;
location.reload(true); } //--> </SCRIPT> <BODY> <SCRIPT
LANGUAGE="JavaScript"> <!-- document.write("Currently, your cookie is
"+document.cookie); //--> </SCRIPT> <FORM NAME="form1">
<P>
This information would be treated as a Cookie: <INPUT
```

```
TYPE="TEXT" NAME="cookie" size="50">
</P>
<INPUT TYPE="Button" name="setCookie" VALUE="Set Cookie!"
onclick="updateCookie()">
</FORM>
</SCRIPT>
</BODY>
</HTML>
```

In the provided example, a JavaScript function is used to create a cookie. When the user enters information in a text box and clicks the "Set Cookie" button, the function creates a cookie on the client's hard disk and reloads the page with the information of the current cookie displayed.

Cookies can be used to track customer activity, maintain user preferences, and provide personalized experiences. They can store unique IDs or other relevant information, which can be used by the server to identify returning customers and offer tailored recommendations.

By processing data on the client side, web applications can provide dynamic and interactive experiences to users without constant communication with the server. Client-side processing reduces the need for frequent server requests, improving performance and responsiveness.

Fat vs. Thin Client

Fat clients have a lot of processing done on the client side.

Thin clients have very little processing on client side, and most of the processing takes place on the server side.

Lecture 18
APPLETS, CGI SCRIPTS

CGI (Common Gateway Interface) scripts are programs running on a web server that handle client requests and generate dynamic content. CGI provides a standard method/protocol for data exchange between web servers and programs on the server.

In the given example, there are two hyperlinks on a web page. When a user clicks on the hyperlink related to IT Books, an HTTP request is sent to the server at the specified URL, which corresponds to a CGI script called 'hello2.cgi'.

The 'hello2.cgi' script is written in a programming language such as C, C++, or Perl. It is designed to generate a SQL select query to retrieve specific information from the 'IT books' table in a database. The server communicates with the CGI script using the CGI protocol.

Once executed, the CGI script retrieves the required information from the database based on the query and formats it as an HTML document. The server then sends this HTML document as the response to the client's request, and the client's browser displays the retrieved information to the user.

SQL (Structured Query Language) is a programming language used for managing and manipulating relational databases. It provides a standard syntax and set of commands to interact with databases.

Servlets, ASP, JSP, and CGI scripts are server-side technologies used for dynamic web applications. Here are some key differences between them:

- **CGI scripts** are programs written in languages like C, C++, or Perl, executed by the server in response to client requests.
- **ASP (Active Server Pages)** is a Microsoft technology for server-side scripting using VBScript or JScript.
- **JSP (JavaServer Pages)** is a Java-based technology combining HTML and Java code to create dynamic web pages.
- **Servlets are Java programs** that run on a web server and generate dynamic web content.
- **CGI scripts** are interpreted or compiled at runtime, while ASP and JSP can be interpreted or compiled.
- Each CGI request spawns a new process, making it slower and resource-intensive.
- ASP is primarily used on Windows servers, while JSP is platform-independent.
- Servlets provide a more efficient alternative to CGI scripts by eliminating the process spawning overhead.
- Servlets can communicate with databases using JDBC (Java Database Connectivity).
- ODBC (Open Database Connectivity) and JDBC (Java Database Connectivity) are driver technologies for connecting applications to databases.
- ODBC is commonly used in Windows environments, while JDBC is specifically designed for Java applications.

ASP (Active Server Pages) is a server-side scripting technology primarily used on Windows servers with IIS (Internet Information Server). Here are some additional details about ASP:

- ASP runs best on IIS, which is freely available with Windows NT and Windows 2000.
- The ASP engine, part of IIS, interprets and translates ASP code.
- ASP uses VBScript as the default scripting language.
- There are seven predefined objects in ASP called intrinsic objects. Two important ones are the Request and Response objects.
- The Response object is used to send information to the client, while the Request object is used to retrieve information from the client.
- ASP code is enclosed within delimiters <% and %>.
- Variables in ASP are generally defined using the Dim statement.
- The Option Explicit statement is used to enforce variable declaration and avoid potential issues with mistyped variables.
- VBScript offers flexibility for programming, including functions, if statements, and loops.
- ASP can collect data from HTML forms using the Request. Form method.
- VBScript code can be embedded within HTML to generate dynamic content.
- Examples of ASP code include writing information to a web page using the Response. Write function and using loops and conditional statements for dynamic content generation.

In Short
ASP (Active Server Pages) is a server-side scripting technology used with IIS on Windows servers. It uses VBScript as the scripting language and has intrinsic objects like Request and

Response. ASP code is enclosed in <% and %> delimiters. It supports variable declaration with the Dim statement and offers flexibility with functions, if statements, and loops. ASP can collect form data using the Request.Form method and generate dynamic content by embedding VBScript within HTML. Examples include writing to web pages and using loops and conditionals for dynamic content.

Best of Luck