Web technology

The World Wide Web (or "Web") is a hypertext and hypermedia information system built on top of the Internet. Web clients and Web servers communicate via HTTP and exchange documents and information that is formatted with HTML and XML. The Web browser interface has made a world of information available to everyone. Hypermedia and hypertext is nonlinear information, presented to users in a way that lets them jump from one reference to another with the click of a button. This has created a global library of instantly available information.

On the application development side, distributed object computing technologies and Web-based languages such as Java have changed the way that applications are designed and developed. Wireless technologies are making the Web more available to people wherever they are. Bluetooth and wireless PANs (personal area networks) let people in the same vicinity create spontaneous wireless networks to exchange information. This is "edge-of-the-Internet" networking. Collaborative computing, messaging, groupware systems, unified messaging, instant messaging, videoconferencing and related technologies help people stay in touch and work together over long distances.

Web appliances, Webcasting, embedded systems, and voice telephony, along with improved bandwidth and many other new technologies are making the Web more exciting. Refer to the following topics for more information.

Browsers

It seems a little ridiculous to define a Web browser since they are as well known as television or radio, so this section contains mostly references to related topics and an extensive list of Web sites that have information on all the Web browsers available. Here are the basic facts about Web browsers:

- Web browsers run on TCP/IP networks.
- HTTP (Hypertext Transfer Protocol) is the protocol that a Web browser uses to access a Web server.
- HTML is the traditional document formatting (markup) language used to display information inside Web browsers.
- Hyper linking is the most unique feature of Web browsers. Users click hyperlinks to move to related Web sites and documents. This is where the term "browser" comes from.
- Web browsers are considered a "universal front end" for accessing information on almost any server, including Web servers, data base servers, and so on.
- Web browsers are "containers" that are capable of displaying graphics and video content and running all sorts of applications.
- Component technologies such as ActiveX and Java were designed to take advantage of the container metaphor of the browser.
- Plug-ins such as Shockwave give Web browsers additional capabilities

The World Wide Web (WWW) is a collection of servers distributed all over the world that respond to various clients. The WWW allows you to click on links to text, pictures, music, or video located on these servers and then to play the selected files on your local client PC, workstation, or terminal, along with more links to related information. You never need to know where the information is located or to learn any obscure commands to access it.

The on-line version of this paper is available as a linked set of files or as a large single file. Downloading this paper as a single file may take some time, but has the advantage of making it convenient to save or print the entire paper with your Web browser.

TO UNDERSTAND THE WWW, IT HELPS IF YOU UNDERSTAND SOME BASIC WEB CONCEPTS. FUNDAMENTAL TO THIS UNDERSTANDING IS THE CONCEPT OF CLIENT/SERVER COMPUTING ON A GLOBAL SCALE.

Whether you're reading WWW documents or creating your own, it helps if you understand the basic components of the WWW language.

One powerful feature of the WWW is that the information you publish on your server can be read by many different clients. In this section, we provide a quick introduction to some of the popular WWW clients.

If you want to make your own information available to WWW clients, you'll want to set up your own server. In this section, we discuss some common WWW server software and give our suggestions for how WWW server information should be designed.

INTRODUCTION

The WWW is a new way of viewing information -- and a rather different one. If, for example, you are viewing this paper as a WWW document, you will view it with a browser, in which case you can immediately access hypertext links. If you are reading this on paper, you will see the links indicated in parentheses and in a different font. Keep in mind that the WWW is constantly evolving. We have tried to pick stable links, but sites reorganize and sometimes they even move. By the time you read the printed version of this paper, some WWW links may have changed.

THE WORLD WIDE WEB

The WWW project has the potential to do for the Internet what Graphical User Interfaces (GUIs) have done for personal computers -- make the Net useful to end users. The Internet contains vast resources in many fields of study (not just in computer and technical information). In the past, finding and using these resources has been difficult. The Web provides consistency: Servers provide information in a consistent way and clients show information in a consistent way. To add a further thread of consistency, many users view the Web through graphical browsers which are like other windows (Microsoft Windows, Macintosh windows, or X-Windows) applications that they use. A principal feature of the Web is its links between one document and another. These links, described in the section on hypertext, allow you to move from one document to another. Hypertext links can point to any server connected to the Internet and to any type of file. These links are what transform the Internet into a web.

WHAT IS HYPERTEXT?

Hypertext provides the links between different documents and different document types. If you have used Microsoft Windows Win Help system or the Macintosh HyperCard application, you likely know how to use hypertext. In a hypertext document, links from one place in the document to another are included with the text. By selecting a link, you are able to jump immediately to another part of the document or even to a different document. In the WWW, links can go not only from one document to another, but from one computer to another.

THE HYPERTEXT TRANSFER PROTOCOL

When you use a WWW client, it communicates with a WWW server using the Hypertext Transfer Protocol (HTTP). When you select a WWW link, the following things happen:

1. The client looks up the hostname and makes a connection with the WWW server.
2. The HTTP software on the server responds to the client's request.
3. The client and the server close the connection.

Compare this with traditional terminal/host computing. Users usually logon (connect) to the server and remain connected until they logoff (disconnect). An HTTP connection, on the other hand, is made only for as long as it takes for the server to respond to a request. Once the request is completed, the client and the server are no longer in communication.

THE INTERNET

The Internet is the world's largest interconnected computer network. Computers on the Internet communicate using the Internet Protocol (IP) and the Transmission Control Protocol (TCP). You identify individual computers

by their IP-address. This address is a 32-bit number that is usually represented by four octets (e.g., 192.40.254.0). Fortunately, you can usually refer to a computer by its name.

THE LANGUAGE OF THE WEB

In order to use the WWW, you must know something about the language used to communicate in the Web. There are three main components to this language:

Uniform Resource Locators (URLs)

> URLs provide the hypertext links between one document and another. These links can access a variety of protocols (e.g., ftp, gopher, or http) on different machines (or your own machine).

Hypertext Markup Language (HTML)

> WWW documents contain a mixture of directives (markup), and text or graphics. The markup directives do such things as make a word appear in bold type. This is similar to the way UNIX users write nroff or troff documents, and MPE users write with Galley, TDP, or Prose. For PC users, this is completely different from WYSIWYG editing. However, a number of tools are now available on the market that hides the actual HTML.

Common Gateway Interfaces (CGI)

> Servers use the CGI interface to execute local programs. CGIs provide a gateway between the HTTP server software and the host machine.

UNIFORM RESOURCE LOCATORS (URLS)

Uniform Resource Locators (URLs) specify the access-method (how), the server name (where), and the location (what) needed for a WWW client to find and access a WWW object. The general form of a URL is

Access-method: //server-name[:port]/location

ACCESS METHODS

The three most popular access methods are

Http:

> This is the method provided by WWW servers. It includes hypertext linking, the hypertext markup language, and server scripts.

Gopher:

> Gopher was developed at the University of Minnesota as a distributed campus information service. There are gopher servers everywhere -- many of them provide campus-wide information systems.

Gopher information is organized into menus. Because hypertext provides the same services as gopher and more, many sites are moving from gopher-supplied information to WWW-supplied information.

Ftp:

The File Transfer Protocol is one of the oldest and most popular of all Internet services. You can access millions of files, documentation, source code, and other useful objects on anonymous FTP archives. You can use a WWW browser to view and to retrieve information from FTP archives.

SERVER NAME

The server name is an IP host name or an IP address. WWW servers often start with the name "www" as in www.robelle.com or www.mayfield.hp.com. The port number is usually not needed. If there are many servers on one machine (e.g., two different WWW servers on the same host), you would use a port number to select one of them. By default, WWW servers are on port 80. Other protocols have different ports (e.g., the default for FTP is 21). Most users never need to know about port numbers.

WELCOME PAGE

Most WWW servers provide a welcome or home page. This is the document that you see if you specify a machine name, but not a document name (see all the examples above under "Server Name"). Good WWW welcome pages provide a short description of the information the WWW server provides, as well as links to all the other information available on the server. The welcome page must be explicitly configured for each WWW server. If you access a WWW server without giving a document name, and receive the error message "no document found", you should try one of the following common document names: welcome.html, index.html, or default.html.

LOCATION

The location can be a filename, a directory, a directory and filename, a server-script name, or something specific to the access-method. Filenames and directory structure often change, so don't be surprised if a URL that worked a few months ago no longer works now.

NETWORK INFRASTRUCTURE

How you connect to the Internet affects how you view the WWW. If you connect via a modem, you won't be able to view large WWW pages, images, sounds, or video; if you have a T1 connection (1.544M bits/second), you will be able to enjoy these features. Some WWW pages assume that you have a fast connection to the Internet.

LOCAL AREA NETWORKS

If your Local Area Network has a gateway to the Internet (there are several different methods to do this), you should be able to use a graphical browser on your own workstation to cruise the WWW. If you are using a PC with Microsoft Windows, you'll need to have a Winsock interface installed (in addition to the regular networking configuration). Macintosh users already have network support via MacTCP. UNIX workstation users should also have built-in support for networking.

DIAL-IN ACCESS

There are two methods of dialing into a machine to get access to the Internet. If you dial in and log on as usual (on UNIX you see "login:" and shell prompt or on MPE you type "HELLO" and get a colon prompt), your computer is not directly connected to the Internet, so it cannot send network packets from your PC to the Internet. In this case, you will have to use Lynx to access the WWW.

WWW SERVERS

WWW servers provide information to the Web. Server software is available for many computer platforms, but setting up a server isn't always easy.

Why Set Up a WWW Server?

Even if you don't have an Internet connection, there are lots of uses for an internal WWW server.

WWW Server Design

Setting up a server to provide information to the many different Internet clients requires extra thought, but the effort is worth it.

Setting Up Your WWW Server

Server software exits for UNIX, MPE, Windows NT, Microsoft Windows, and even MS-DOS.

Maintaining Your WWW Server

Like most applications, your WWW server will need a little help from time to time.

CLIENT–SERVER MODEL

The **client–server model** of computing is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients.[1] Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. A server machine is a host that is running one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests.

ADVANTAGES

- In most cases, client–server architecture enables the roles and responsibilities of a computing system to be distributed among several independent computers that are known to each other only through a

network. This creates an additional advantage to this architecture: greater ease of maintenance. For example, it is possible to replace, repair, upgrade, or even relocate a server while its clients remain both unaware and unaffected by that change.
- All data is stored on the servers, which generally have far greater security controls than most clients.[citation needed] Servers can better control access and resources, to guarantee that only those clients with the appropriate permissions may access and change data.
- Since data storage is centralized, updates to that data are far easier to administer in comparison to a P2P paradigm. In the latter, data updates may need to be distributed and applied to each peer in the network, which is time-consuming as there can be thousands or even millions of peers.
- Many mature client–server technologies are already available which were designed to ensure security, friendliness of the user interface, and ease of use.[citation needed]
- It functions with multiple different clients of different capabilities.

DISADVANTAGES

- As the number of simultaneous client requests to a given server increases, the server can become overloaded.[citation needed] Contrast that to a P2P network, where its aggregated bandwidth actually increases as nodes are added, since the P2P network's overall bandwidth can be roughly computed as the sum of the bandwidths of every node in that network.
- The client–server paradigm lacks the robustness of a good P2P network.[citation needed] Under client–server, should a critical server fail, clients' requests cannot be fulfilled. In P2P networks, resources are usually distributed among many nodes. Even if one or more nodes depart and abandon a downloading file, for example, the remaining nodes should still have the data needed to complete the download.

BASIC CONCEPTS OF HTML

Before getting into the details of making HTML documents accessible, an understanding of basic HTML concepts is needed. If you are familiar with the general structure and coding of HTML, you can skip this page and move onto the first common HTML element to be covered, Making Images Accessible.

TAGS, ATTRIBUTES, AND ELEMENTS

There are several versions of HTML that have evolved since its inception in 1989; HTML 4.0 is the latest and most prevalent version. HTML code, or mark-up, consists of tags, which are a set of symbols defined in HTML to have special meaning.

Tags start with a less-than sign (<) followed by a keyword, and conclude with a greater-than sign (>). (These signs are known as angle brackets.) For example: <img>, <form>, <table> are all tags and respectively represent images, forms, and tables.

Note: Tags are not case-sensitive, so <img>, <Img>, and <IMG> all have the same meaning.

There are two types of tags:

- Start tags always begin the effect, such as for a paragraph and may take certain attributes which affect the tag's behavior.
- End tags close the effect and repeat the keyword with a slash in front, such as .

An attribute is a word separated from the keyword by a space which generally requires a value in quotes preceded by an equal sign, such as <table width="90%"> (which calls for a table that is 90 percent of the width of the screen display.)

When many web developers talk about tags, their preferred term is "element". An HTML element defines the structures and behaviors of the different parts of a document. Most elements consist of three parts:

- a start tag,
- the content,
- and an end tag.

For example: This is a paragraph creates a paragraph element.

Empty elements have no content and never have end tags. Some common empty elements are:

- <img> (image),
-
 (line-break),
- <hr> (horizontal rule).

In this module, you will be taught the proper use of the most common HTML elements along with any necessary accessibility-related attributes of the start tag.

BASIC STRUCTURE OF AN HTML DOCUMENT

Every HTML document consists of four basic structure elements: html, head, title, and body. Each of these is explored in detail below:

- Html Element
    - Start tag: <html> - First tag in the document which declares you are writing an HTML element.
    - End tag: </html> - Last tag in the document.
- Head Element
    - Start tag: <head> - Second tag, follows the <html> tag, and starts the head section, which describes the document.
    - End tag: </head> - Fifth tag, follows the </title> tag and closes the head section.
- Title Element
    - Start tag: <title> - Third tag, follows the <head> tag, and contains the title you want for the document. This information will be displayed in the title bar at the top of the browser window.
    - End tag: </title> - Fourth tag, immediately follows, without any spaces, the title you want for the document and precedes the </head> tag.
- Body Element

1. Start tag: <body> - Sixth tag, follows the </head> tag to denote starting the content of the document.

2. End tag: </body> - Next to last tag in the document, follows the end of the document content and precedes the </html> tag.

CODE SAMPLE FOR BASIC STRUCTURE OF AN HTML DOCUMENT:

```
<html>
 <head>
  <title> Put the title of the document here. Making certain there are no extra spaces between the title itself and the title tags. </title>
 </head>

 <body>
    Put the document content here.
 </body>
</html>
```

3. Primary Tags

To build any web page you will need four primary tags: <html>, <head>, <title> and <body>. These are all container tags and **must appear as pairs with a beginning and an ending.**

<html>…</html>

Every HTML document begins and ends with the <html> tag. This tells the browser that the following document is an html file. Remember, tags tell the browsers how to display information.

<head>…</head>

The <head> tag contains the title of the document along with general information about the file, like the author, copyright, keywords and/or a description of what appears on the page.

<title>…</title>

Appears within the <head> tag and gives the title of the page. Try to make your titles descriptive, but not more than 20 words in length. The title appears at the very top of the browser page on the title bar.

<body>…</body>

The main content of your page is placed within the body tags: your text, images, links, tables and so on.

4. Creating your first web page

Using the primary HTML tags, mentioned in Chapter 3, you are now ready to create your first Web page.

Step 1 Open up a text editor (SimpleText for Mac or Notepad for Windows)

Step 2 Enter the following:

```
<html>
<head>
<title> This is my first web page</title>
</head>
<body>
Hello world. This is my first web page. There's more to come.
</body>
</html>
```

HTML Introduction

```
<html>
<body>

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

What is HTML?

HTML is a language for describing web pages.

- HTML stands for **H**yper **T**ext **M**arkup **L**anguage
- HTML is not a programming language, it is a **markup language**
- A markup language is a set of **markup tags**
- HTML uses **markup tags** to describe web pages

HTML Tags

HTML markup tags are usually called HTML tags

- HTML tags are keywords surrounded by **angle brackets** like <html>
- HTML tags normally **come in pairs** like and
- The first tag in a pair is the **start tag,** the second tag is the **end tag**
- Start and end tags are also called **opening tags** and **closing tags**

HTML Documents = Web Pages

- HTML documents **describe web pages**
- HTML documents **contain HTML tags** and plain text

- HTML documents are also **called web pages**

The purpose of a web browser (like Internet Explorer or Firefox) is to read HTML documents and display them as web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page:

HTML Element Syntax

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

NESTED HTML ELEMENTS

Most HTML elements can be nested (can contain other HTML elements).

HTML documents consist of nested HTML elements.

EMPTY HTML ELEMENTS

HTML elements with no content are called empty elements.

 is an empty element without a closing tag (the
 tag defines a line break).

HTML Attributes

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

HTML Attributes Reference

A complete list of legal attributes for each HTML element is listed in our:

Complete HTML Reference

Below is a list of some attributes that are standard for most HTML elements:

Attribute	Value	Description
class	*classname*	Specifies a classname for an element
id	*id*	Specifies a unique id for an element
style	*style_definition*	Specifies an inline style for an element

title	*tooltip_text*	Specifies extra information about an element (displayed as a tool tip)

CSS STANDS FOR CASCADING STYLE SHEETS.

Css is one of the important factor in designing website page. Now all web developer give importance to this css property while designing website. Css plays a role about how web site page will display. CSS includes many of property which can effect to website page.

The **page extention of css page** where css styleet sheet are stored is .css. It becomes very easy to change layout of website if web developer use this style sheet. CSS includes propery which effect differently to browser to browser. That means all property may not work in all browser.

There are three method to insert to apply css style sheet into html page.

(1) External CSS
(2) Internal CSS
(3) in-line CSS

External CSS Method:

External style sheets are styles that can be applied to as many web pages as you like that are on the same website. The external css file which you want to include in html page shold be write between <head> </head> tags.

<link rel="stylesheet" type="text/css" href="style.css" />

In above example style.css is external valid css file. External css file should not include html tag like <tile>.<head>,<body> etc.

Internal CSS Method :

Generally an internal style sheet is used or placed within the web page where you want to apply style sheet.

```
<style type="text/css">
p
{
color: red;
}
</style>
```

The styles are placed within the openning and closing <head> tags.

In-line CSS Method:

You have need to use "style" attribute within the relevant HTML tag where you want in-line css method.

```
<font style="color:red">Beyondmart.com</font>
```

Now, it is cleared from above that how to insert css stylesheet into web page or how css style is called from page.

GROUPING AND NESTING SELECTORS IN CSS

Sometimes in CSS style sheets there are some elements that have the same style, writing separate block for every element consume space, so in order to get optimized CSS you should consider using grouping and nesting selectors.

These days, lots of programming involving in the website development process, resulting in the large file sizes. Hence the need of code optimization is greater than before. Specifically talking about CSS code optimization, grouping selectors and using nesting selector practice can be helpful in doing so. If different elements have same style you can group the selectors like this

```
h1,h2,p
{
color:red;
}
```

Grouping Selectors

In style sheets there are often elements with the same style.

```
h1
{
color:green;
}
h2
{
color:green;
}
p
{
color:green;
```

To minimize the code, you can group selectors.

Separate each selector with a comma.

In the example below we have grouped the selectors from the code above:

```
h1,h2,p
{
color:green;
}
```

Nesting Selectors

It is possible to apply a style for a selector within a selector.

In the example below, one style is specified for all p elements, and a separate style is specified for p elements nested within the "marked" class:

```
P
{
COLOR:BLUE;
TEXT-ALIGN:CENTER;
}
.MARKED
```

CSS PSEUDO-CLASSES

CSS pseudo-classes are used to add special effects to some selectors.

Syntax

The syntax of pseudo-classes:

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

CSS classes can also be used with pseudo-classes:

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

Anchor Pseudo-classes

Links can be displayed in different ways in a CSS-supporting browser:

Example

```
a:link {color:#FF0000;}     /* unvisited link */
a:visited {color:#00FF00;}  /* visited link */
a:hover {color:#FF00FF;}  /* mouse over link */
a:active {color:#0000FF;}  /* selected link */
```

Try it yourself »

Pseudo-classes and CSS Classes

Pseudo-classes can be combined with CSS classes:

```
a.red:visited {color:#FF0000;}

<a class="red" href="css_syntax.asp">CSS Syntax</a>
```

JavaScript Introduction

JavaScript is the most popular scripting language on the internet, and works in all major browsers, such as Internet Explorer, Firefox, Chrome, Opera, and Safari.

What is JavaScript?

- JavaScript was designed to add interactivity to HTML pages
- JavaScript is a scripting language
- A scripting language is a lightweight programming language
- JavaScript is usually embedded directly into HTML pages
- JavaScript is an interpreted language (means that scripts execute without preliminary compilation)
- Everyone can use JavaScript without purchasing a license

Are Java and JavaScript the same?

NO!Java and JavaScript are two completely different languages in both concept and design!

Java (developed by Sun Microsystems) is a powerful and much more complex programming language - in the same category as C and C++.

What can a JavaScript do?

- **JavaScript gives HTML designers a programming tool -** HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small "snippets" of code into their HTML pages
- **JavaScript can put dynamic text into an HTML page -** A JavaScript statement like this: document.write("<h1>" + name + "</h1>") can write a variable text into an HTML page

JAVASCRIPT CAN REACT TO EVENTS - A JAVASCRIPT CAN BE SET TO EXECUTE WHEN SOMETHING HAPPENS, LIKE WHEN A PAGE HAS FINISHED LOADING OR WHEN A USER CLICKS ON AN HTML ELEMENT

- **JavaScript can read and write HTML elements -** A JavaScript can read and change the content of an HTML element
- **JavaScript can be used to validate data -** A JavaScript can be used to validate form data before it is submitted to a server. This saves the server from extra processing
- **JavaScript can be used to detect the visitor's browser** - A JavaScript can be used to detect the visitor's browser, and - depending on the browser - load another page specifically designed for that browser
- **JavaScript can be used to create cookies** - A JavaScript can be used to store and retrieve information on the visitor's computer
- JavaScript Comments JavaScript comments can be used to make the code more readable.

JavaScript Comments

- Comments can be added to explain the JavaScript, or to make the code more readable.
- Single line comments start with //.
- The following example uses single line comments to explain the code:

```
<script type="text/javascript">
// Write a heading
document.write("<h1>This is a heading</h1>");
// Write two paragraphs:
document.write("<p>This is a paragraph.</p>");
document.write("<p>This is another paragraph.</p>");
</script>
```

- JavaScript Multi-Line Comments
- Multi line comments start with /* and end with */.
- JavaScript Statements

JavaScript is a sequence of statements to be executed by the browser.

JavaScript is Case Sensitive

Unlike HTML, JavaScript is case sensitive - therefore watch your capitalization closely when you write JavaScript statements, create or call variables, objects and functions.

JavaScript Statements

- A JavaScript statement is a command to a browser. The purpose of the command is to tell the browser what to do.
- This JavaScript statement tells the browser to write "Hello Dolly" to the web page:
  document.write("Hello Dolly");

```
document.write("Hello Dolly");
```

- It is normal to add a semicolon at the end of each executable statement. Most people think this is a good programming practice, and most often you will see this in JavaScript examples on the web.
- The semicolon is optional (according to the JavaScript standard), and the browser is supposed to interpret the end of the line as the end of the statement. Because of this you will often see examples without the semicolon at the end.
- **Note:** Using semicolons makes it possible to write multiple statements on one line.
-

JavaScript Code

- JavaScript code (or just JavaScript) is a sequence of JavaScript statements.
- Each statement is executed by the browser in the sequence they are written.
- This example will write a heading and two paragraphs to a web page:
  <script type="text/javascript">
  document.write("<h1>This is a heading</h1>");
  document.write(" This is a paragraph. ");
  document.write(" This is another paragraph. ");
  </script>

Unit 2: Issues of Web Technology (4 Hrs.)

The term Architecture in IT generally refers to Client Server Architecture. A client is defined as a requester of services and a server is defined as the provider of services. A single machine can be both a client and a server depending on the software configuration. There are different types of Client-Server architecture available. Some of them are 2-tier architecture, 3 tier architecture, 4 tier architecture and n tier architecture.

2-tier architecture

In 2-tier, the application logic is either buried inside the User Interface on the client or within the database on the server (or both). With two tier client/server architectures (see Two Tier Software Architectures), the user system interface is usually located in the user's desktop environment and the database management services are usually in a server that is a more powerful machine that services many clients. Processing management is split between the user system interface environment and the database management server environment. The database management server provides stored procedures and triggers

3-tier architecture

In 3-tier, the application logic (or) process lives in the middle-tier, it is separated from the data and the user interface. 3-tier systems are more
, robust and flexible. In addition, they can integrate data from multiple sources. In the three tier architecture, a middle tier was added between the user system interface client environment and the database management server environment. There are a variety of ways of implementing this middle tier, such as transaction processing monitors, message servers, or application servers. The middle tier can perform queuing, application execution, and database staging. For example, if the middle tier provides queuing, the client can deliver its request to the middle layer and disengage because the middle tier will access the data and return the answer to the client. In addition the middle layer adds scheduling and prioritization for work in progress. The three tier client/server architecture has been shown to improve performance for groups with a large number of users (in the thousands) and improves flexibility when compared to the two tier approach. Flexibility in partitioning can be a simple as "dragging and dropping" application code modules onto different computers in some three tier architectures. A limitation with three tier architectures is that the development environment is reportedly more difficult to use than the visually-oriented development of two tier applications. The most basic type of three tier architecture has a middle layer consisting of Transaction Processing (TP) monitor technology. The TP monitor technology is a type of message queuing, transaction scheduling, and prioritization service where the client connects to the TP monitor (middle tier) instead of the database server. The transaction is accepted by the monitor, which queues it and then takes responsibility for managing it to completion, thus freeing up the client.

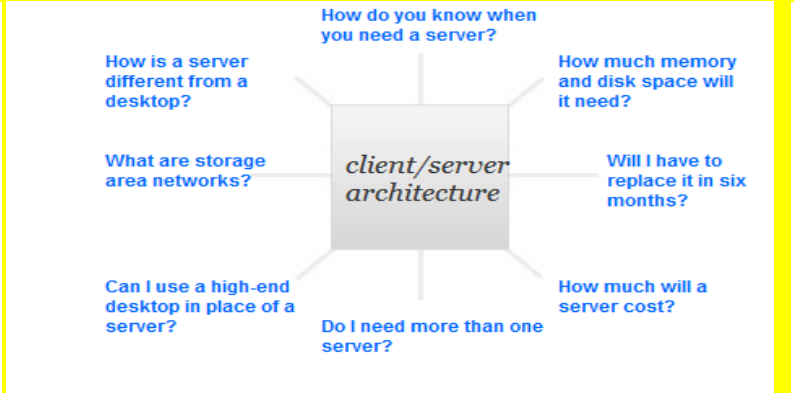
3-Tier Architecture..
There are three layers in 3-tier architecture.
1. GUI Layer
2.Object Layer
3.Database Layer
Verify these links for 3-Tier Architecture..

Client/server architecture

A network architecture in which each computer or process on the network is either a *client* or a *server*. Servers are powerful computers or processes dedicated to managing disk drives (*file servers*), printers (*print servers*), or network traffic (*network servers*). Clients are PCs or workstations on which users run applications. Clients rely on servers for resources, such as files, devices, and even processing power.

Another type of network architecture is known as a *peer-to-peer* architecture because each *node* has equivalent responsibilities. Both client/server and *peer-to-peer architectures* are widely used, and each has unique advantages and disadvantages.

Client-server architectures are sometimes called *two-tier architectures.*



A **Tier 2 Network** is an Internet service provider who engages in the practice of peering with other networks, but who still purchases IP transit to reach some portion of the Internet.Tier 2 providers are the most common providers on the Internet as it is much easier to purchase transit from a Tier 1 network than it is to peer with them and then attempt to push into becoming a Tier 1 carrier.

DIFFERENCE BETWEEN CLIENT SIDE & SERVER SIDE PROGRAMMING

1. CLIENT-SIDE

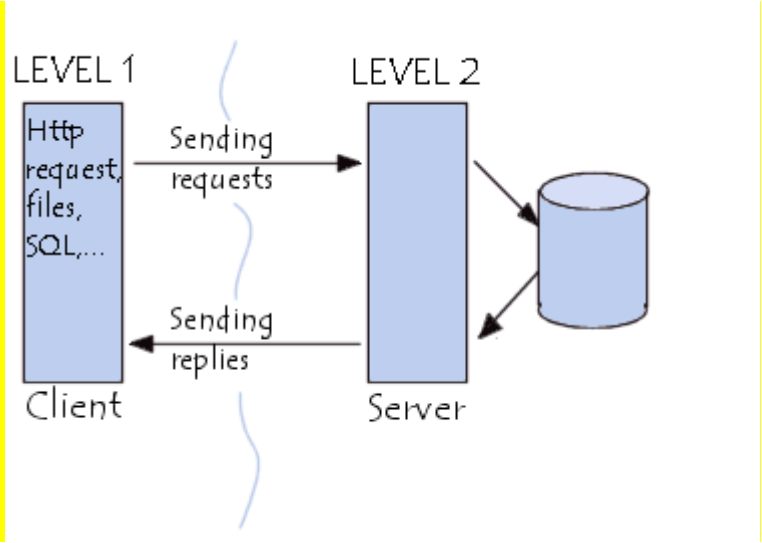
- Client-side programming is run on the user's computer. An example of client-side programming is Javascript. Javascript can be used to run checks on form values and send alerts to the user's browser. The problem with client-side scripts is the limit of control and problems with operating systems and web browsers. Since programming a website involves users with several options of computer software, it's difficult for programmers to account for any bugs in the code or compatibility issues with browsers.

SERVER-SIDE

- Server-side scripts are run on the server. It reduces the amount of bugs or compatibility issues since the code is run on one server using one language and hosting software. Server-side programming can also be encrypted when users send form variables, protecting users against any hack attempts. Some examples of server-side programming languages are C#, VB.NET, and PHP.

Introduction to 2-Tier Architecture

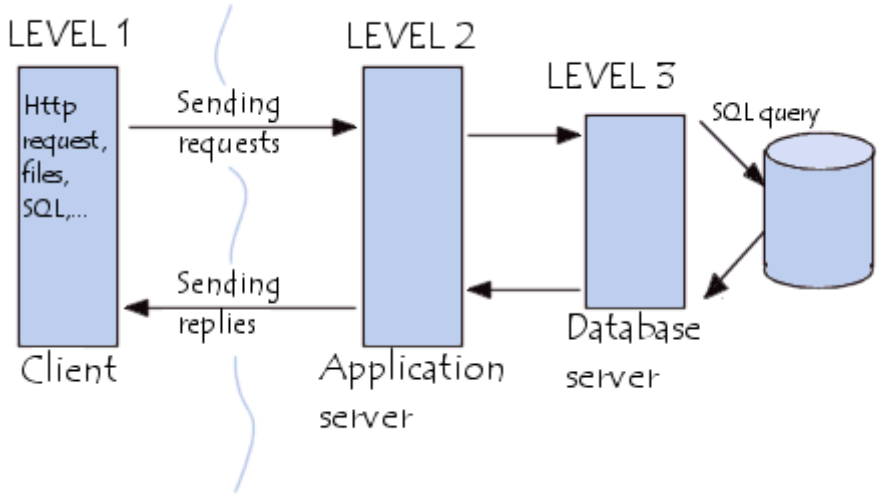
2-tier architecture is used to describe client/server systems where the client requests resources and the server responds directly to the request, using its own resources. This means that the server does not call on another application in order to provide part of the service.



Introduction to 3-Tier Architecture

In 3-tier architecture, there is an intermediary level, meaning the architecture is generally split up between:

1. A client, i.e. the computer, which requests the resources, equipped with a user interface (usually a web browser) for presentation purposes
2. The application server (also called **middleware**), whose task it is to provide the requested resources, but by calling on another server
3. The data server, which provides the application server with the data it requires



Unit 3: The Client Tier (10 Hrs.)

Introduction to XML

XML was designed to transport and store data. HTML was designed to display data.

What is XML?

- XML stands for Extensible Markup Language
- XML is a markup language much like HTML
- XML was designed to carry data, not to display data
- XML tags are not predefined. You must define your own tags
- XML is designed to be self-descriptive
- XML is a W3C Recommendation

The Difference between XML and HTML

XML is not a replacement for HTML.

XML and HTML were designed with different goals:

- XML was designed to transport and store data, with focus on what data is
- HTML was designed to display data, with focus on how data looks

HTML is about displaying information, while XML is about carrying information.

XML Does Not DO Anything

Maybe it is a little hard to understand, but XML does not DO anything. XML was created to structure, store, and transport information.

The following example is a note to Tove, from Jani, stored as XML:

```
<NOTE>
<TO>TOVE</TO>
<FROM>JANI</FROM>
<HEADING>REMINDER</HEADING>
<BODY>DON'T FORGET ME THIS WEEKEND!</BODY>
</NOTE>
```

The note above is quite self descriptive. It has sender and receiver information, it also has a heading and a message body.

But still, this XML document does not DO anything. It is just information wrapped in tags. Someone must write a piece of software to send, receive or display it.

XML is Not a Replacement for HTML

XML is a complement to HTML.

It is important to understand that XML is not a replacement for HTML. In most web applications, XML is used to transport data, while HTML is used to format and display the data.

My best description of XML is this:

XML is a software- and hardware-independent tool for carrying information

WHAT IS AN XML ELEMENT?

An XML element is everything from (including) the element's start tag to (including) the element's end tag.

An element can contain:

- other elements
- text
- attributes
- or a mix of all of the above...

```
<bookstore>
 <book category="CHILDREN">
  <title>Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
 </book>
 <book category="WEB">
  <title>Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
 </book>
</bookstore>
```

In the example above, <bookstore> and <book> have **element contents**, because they contain other elements. <book> also has an **attribute** (category="CHILDREN"). <title>, <author>, <year>, and <price> have **text content** because they contain text.
XML Naming Rules
XML elements must follow these naming rules:
Names can contain letters, numbers, and other characters
Names cannot start with a number or punctuation character
Names cannot start with the letters xml (or XML, or Xml, etc)
Names cannot contain spaces

Any name can be used, no words are reserved.

BEST NAMING PRACTICES

Make names descriptive. Names with an underscore separator are nice: <first_name>, <last_name>.

Names should be short and simple, like this: <book_title> not like this: <the_title_of_the_book>.

Avoid "-" characters. If you name something "first-name," some software may think you want to subtract name from first.

Avoid "." characters. If you name something "first.name," some software may think that "name" is a property of the object "first."

Avoid ":" characters. Colons are reserved to be used for something called namespaces (more later).

XML documents often have a corresponding database. A good practice is to use the naming rules of your database for the elements in the XML documents.

XML Attributes

XML elements can have attributes, just like HTML.Attributes provide additional information about an element.

XML Attributes

In HTML, attributes provide additional information about elements:

```
<img src="computer.gif">
<a href="demo.asp">
```

Attributes often provide information that is not a part of the data. In the example below, the file type is irrelevant to the data, but can be important to the software that wants to manipulate the element:

```
<file type="gif">computer.gif</file>
```

XML Attributes Must be Quoted

Attribute values must always be quoted. Either single or double quotes can be used. For a person's sex, the person element can be written like this:

```
<person sex="female">
```

or like this:

```
<person sex='female'>
```

If the attribute value itself contains double quotes you can use single quotes, like in this example:

```
<gangster name='George "Shotgun" Ziegler'>
```

or you can use character entities:

```
<gangster name="George &quot;Shotgun&quot; Ziegler">
```

XML Elements vs. Attributes .Take a look at these examples:

```
<person sex="female">
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

In the first example sex is an attribute. In the last, sex is an element. Both examples provide the same information.

There are no rules about when to use attributes or when to use elements. Attributes are handy in HTML. In XML my advice is to avoid them. Use elements instead.

XML Syntax Rules

All XML Elements Must Have a Closing Tag

In HTML, some elements do not have to have a closing tag:

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

In XML, it is illegal to omit the closing tag. All elements **must** have a closing tag:

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

Note: You might have noticed from the previous example that the XML declaration did not have a closing tag. This is not an error. The declaration is not a part of the XML document itself, and it has no closing tag.

XML Tags are Case Sensitive

XML tags are case sensitive. The tag <Letter> is different from the tag <letter>.

Opening and closing tags must be written with the same case:

```
<Message>This is incorrect</message>
<message>This is correct</message>
```

Note: "Opening and closing tags" are often referred to as "Start and end tags". Use whatever you prefer. It is exactly the same thing.

XML Elements Must be Properly Nested

In HTML, you might see improperly nested elements:

```
<b><i>This text is bold and italic</b></i>
```

In XML, all elements **must** be properly nested within each other:

```
<b><i>This text is bold and italic</i></b>
```

In the example above, "Properly nested" simply means that since the <i> element is opened inside the <b> element, it must be closed inside the <b> element.

XML Documents Must Have a Root Element

XML documents must contain one element that is the **parent** of all other elements. This element is called the **root** element.

Unit 5: Introduction to Advanced Server Side Issues (4 Hrs.)

A **database connection** is a facility in computer science that allows client software to communicate with database server software, whether on the same machine or not. A connection is required to send commands and receive answers.

Connections are a key concept in data-centric programming. Since some DBMS engines require considerable time to connect connection pooling was invented to improve performance. No command can be performed against a database without an "open and available" connection to it.

Connections are built by supplying an underlying driver or provider with a connection string, which is a way of addressing a specific database or server and instance as well as user authentication credentials (for example, **Server=**sql_box;**Database=**Common; **User ID=**uid;**Pwd=**password;). Once a connection has been built it can be opened and closed at will, and properties (such as the command time-out length, or transaction, if one exists) can be set. The Connection String is composed of a set of key/value pairs as dictated by the data access interface and data provider being used.

SQL CREATE DATABASE Statement

The CREATE DATABASE Statement

The CREATE DATABASE statement is used to create a database.

SQL CREATE DATABASE Syntax

```
CREATE DATABASE database_name
```

CREATE DATABASE Example

Now we want to create a database called "my_db". We use the following CREATE DATABASE statement:

```
CREATE DATABASE my_db
```

Database tables can be added with the CREATE TABLE statement.

QL Syntax

Database Tables

A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

Below is an example of a table called "Persons":

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The table above contains three records (one for each person) and five columns (P_Id, LastName, FirstName, Address, and City).

SQL Statements

Most of the actions you need to perform on a database are done with SQL statements.

The following SQL statement will select all the records in the "Persons" table:

```
SELECT * FROM Persons
```

Semicolon after SQL Statements?

Some database systems require a semicolon at the end of each SQL statement.

Semicolon is the standard way to separate each SQL statement in database systems that allow more than one SQL statement to be executed in the same call to the server.

We are using MS Access and SQL Server 2000 and we do not have to put a semicolon after each SQL statement, but some database programs force you to use it.

SQL DML and DDL

SQL can be divided into two parts: The Data Manipulation Language (DML) and the Data Definition Language (DDL).

The query and update commands form the DML part of SQL:

- **SELECT** - extracts data from a database
- **UPDATE** - updates data in a database
- **DELETE** - deletes data from a database
- **INSERT INTO** - inserts new data into a database

The DDL part of SQL permits database tables to be created or deleted. It also define indexes (keys), specify links between tables, and impose constraints between tables. The most important DDL statements in SQL are:

- **CREATE DATABASE** - creates a new database
- **ALTER DATABASE** - modifies a database
- **CREATE TABLE** - creates a new table
- **ALTER TABLE** - modifies a table
- **DROP TABLE** - deletes a table
- **CREATE INDEX** - creates an index (search key)
- **DROP INDEX** - deletes an index

SQL SELECT Statement

This chapter will explain the SELECT and the SELECT * statements.

The SQL SELECT Statement

The SELECT statement is used to select data from a database.

The result is stored in a result table, called the result-set.

SQL SELECT Syntax

```
SELECT column_name(s)
FROM table_name
```

```
SELECT * FROM table_name
```

An SQL SELECT Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

Now we want to select the content of the columns named "LastName" and "FirstName" from the table above.

We use the following SELECT statement:

```
SELECT LastName,FirstName FROM Persons
```

The result-set will look like this:

LastName	FirstName
Hansen	Ola
Svendson	Tove
Pettersen	Kari

SELECT * Example

Now we want to select all the columns from the "Persons" table.

We use the following SELECT statement:

```
SELECT * FROM Persons
```

Tip: The asterisk (*) is a quick way of selecting all columns!

The result-set will look like this:

P_Id	LastName	FirstName	Address	City

1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The INSERT INTO Statement

The INSERT INTO statement is used to insert a new row in a table.

SQL INSERT INTO Syntax

It is possible to write the INSERT INTO statement in two forms.

The first form doesn't specify the column names where the data will be inserted, only their values:

```
INSERT INTO table_name
VALUES (value1, value2, value3,...)
```

The second form specifies both the column names and the values to be inserted:

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)
```

SQL INSERT INTO Example

We have the following "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The UPDATE Statement

The UPDATE statement is used to update existing records in a table.

SQL UPDATE Syntax

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

Note: Notice the WHERE clause in the UPDATE syntax. The WHERE clause specifies which record or records that should be updated. If you omit the WHERE clause, all records will be updated!

SQL UPDATE Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob		

The DELETE Statement

The DELETE statement is used to delete rows in a table.

SQL DELETE Syntax

```
DELETE FROM table_name
WHERE some_column=some_value
```

Note: Notice the WHERE clause in the DELETE syntax. The WHERE clause specifies which record or records that should be deleted. If you omit the WHERE clause, all records will be deleted!

SQL DELETE Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob	Nissestien 67	Sandnes

The WHERE Clause

The WHERE clause is used to extract only those records that fulfill a specified criterion.

SQL WHERE Syntax

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
```

WHERE Clause Example

The "Persons" table:

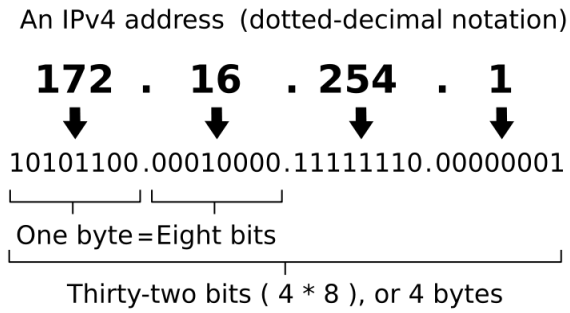
P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

Now we want to select only the persons living in the city "Sandnes" from the table above.

We use the following SELECT statement:

```
SELECT * FROM Persons
WHERE City='Sandnes'
```

IP address authentication considered harmful



The practice of using IP addresses to identify institutional users is commonplace in most sectors of the online publishing marketplace, as it allows users to seamlessly access resources without needing to go through an explicit log-in process. But this practice has serious pitfalls which lead to significant customer service costs for publishers, and frustration for institutional customers.

The increasing realization by publishers that their users need community and social networking features in publishing platforms, but at the same time have little or no tolerance for additional sign-on procedures, leads to terminal pressure on this once tried and trusted method of authentication for institutional users.

Integrated Windows Authentication

Integrated Windows Authentication (**IWA**) is a term associated with Microsoft products that refers to the SPNEGO, Kerberos, and NTLMSSP authentication protocols with respect toSSPI functionality introduced with Microsoft Windows 2000 and included with later Windows NT-based operating systems. The term is used more commonly for the automatically authenticated connections between Microsoft Internet Information Services, Internet Explorer, and other Active Directory aware applications.

IWA is also known by several names like *HTTP Negotiate authentication*, *NT Authentication*, *NTLM Authentication*, *Domain authentication*, *Windows Integrated Authentication*, *Windows NT Challenge/Response authentication*,[1] or simply *Windows Authentication*.

SUPPORTED BROWSERS

Integrated Windows Authentication works with most modern browsers,[4] but does not work over HTTP proxy servers.[3] Therefore, it is best for use in intranets where all the clients are within a

single domain. It may work with other Web browsers if they have been configured to pass the user's logon credentials to the server that is requesting authentication.

In Mozilla Firefox on Windows operating systems, the names of the domains/websites to which the authentication is to be passed can be entered (comma delimited for multiple domains) for the "*network.negotiate-auth.trusted-uris*" (for Kerberos) or in the "*network.automatic-ntlm-auth.trusted-uris*" (NTLM) Preference Name on the *about:config* page. On the Macintosh operating systems this works if you have a kerberos ticket (use negotiate).
Some websites may also require configuring the "*network.negotiate-auth.delegation-uris*".

Opera 9.01 and later versions can use NTLM/Negotiate, but will use Basic or Digest authentication if that is offered by the server.

Chrome works as of 8.0.

Safari works, once you have a Kerberos ticke

SQL injection

SQL injection is a code injection technique that exploits a security vulnerability occurring in the database layer of an application. The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. It is an instance of a more general class of vulnerabilities that can occur whenever one programming or scripting language is embedded inside another. SQL injection attacks are also known as SQL insertion attacks.[1]

HTTP cookie

A **cookie**, also known as a **web cookie**, **browser cookie**, and **HTTP cookie**, is a piece of text stored on a user's computer by their web browser. A cookie can be used for authentication, storing site preferences, shopping cart contents, the identifier for a server-based session, or anything else that can be accomplished through storing text data.

A cookie consists of one or more name-value pairs containing bits of information, which may be encrypted for information privacy and data securitypurposes. The cookie is sent as a field in the header of the HTTP response by a web server to a web browser and then sent back unchanged by the browser each time it accesses that server.

Cookies may be set by the server with or without an expiration date. Cookies without an expiration date exist until the browser terminates, while cookies with an expiration date may be stored by the browser until the expiration date passes. Users may also manually delete cookies in order to save space or to address privacy issues.

As text, cookies are not executable. Because they are not executed, they cannot replicate themselves and are not viruses. However, due to the browser mechanism to set and read cookies, they can be used as spyware (see zombie cookie and evercookie for more details). Anti-spyware products may warn users about some cookies because cookies can be used to track computer activity—a privacy concern, later causing possiblemalware.

Most modern browsers allow users to decide whether to accept cookies, and the time frame to keep them, but rejecting cookies makes some

TERMINOLOGIES

SESSION COOKIE

A session cookie only lasts for the duration of users using the website. It will expire if a user closes his/her browser, or if a user hasn't visited the server for certain period of time (called session idle timeout, in which case, the server will expire/invalidate the user session).

PERSISTENT COOKIE

A persistent cookie will outlast user sessions. If a persistent cookie has its Max-Age set to 1 year, then, within the year, the initial value set in that cookie would be sent back to server every time the user visits the server. This could be used to record a vital piece of information such as how the user initially came to this website. For this reason, persistent cookies are also called tracking cookies.

SECURE COOKIE

A secure cookie is only used when a browser is visiting a server via HTTPS, ensuring that the cookie is always encrypted when transmitting from client to server. This makes the cookie less likely to be exposed to cookie theft via eavesdropping.

HTTPONLY COOKIE

The HttpOnly cookie is supported by most modern browsers.[8] On a supported browser, a HttpOnly cookie will only be used when transmitting HTTP (or HTTPS) requests. In addition, the cookie value is not available to client side script (such as Javascript), thereby mitigating the threat of cookie theft via Cross-site scripting.

THIRD-PARTY COOKIE

First-party cookies are cookies set with the same domain (or its subdomain) in your browser's address bar. Third-party cookies are cookies being set with different domains than the one shown on the address bar.

For example: Suppose a user visits `www.example1.com`, which sets a cookie with the domain `ad.foxytracking.com`. When the user later visits `www.example2.com`, another cookie is set with the domain `ad.foxytracking.com`. Eventually, both of these cookies will be sent to the advertiser when loading their ads or visiting their website. The advertiser can then use these cookies to build up a browsing history of the user across all the websites this advertiser has footprints on.

SUPER COOKIE

A Super cookie is a cookie with a Public Suffix[9] domain, like `.com`, `.co.uk` or `k12.ca.us`.

Most browsers, by default, allow first-party cookies—a cookie with domain to be the same or sub-domain of the requesting host. For example, a user visiting `www.example.com` can have a cookie set with domain `www.example.com` or `.example.com`, but not `.com`. A super cookie with domain `.com` would be blocked by browsers; otherwise,

a malicious website, like attacker.com, could set a super cookie with domain .com and potentially disrupt or impersonate legitimate user requests to example.com. Unfortunately, the Public Suffix List keeps changing. Older versions of browsers will not have the most up-to-date list, and will therefore be vulnerable to certain super cookies.

ZOMBIE COOKIE

*Main article: Zombie cookie*A zombie cookie is any cookie that is automatically recreated after a user has deleted it. This is accomplished by a script storing the content of the cookie in some other location, such as the local storage available to Flash content, and then recreating it from

USES

SESSION MANAGEMENT

Cookies may be used to maintain data related to the user during navigation, possibly across multiple visits. Cookies were introduced to provide a way to implement a "shopping cart" (or "shopping basket"),[2][3] a virtual device into which users can store items they want to purchase as they navigate throughout the site.

Shopping basket applications today usually store the list of basket contents in a database on the server side, rather than storing basket items in the cookie itself. A web server typically sends a cookie containing a unique session identifier. The web browser will send back that session identifier with each subsequent request and shopping basket items are stored associated with a unique session identifier.

PERSONALIZATION

Many websites use cookies for personalization based on users' preferences. Users select their preferences by entering them in a web form and submitting the form to the server. The server encodes the preferences in a cookie and sends the cookie back to the browser. This way, every time the user accesses a page, the server is also sent the cookie where the preferences are stored, and can personalize the page according to the user preferences. For example, the Wikipedia website allows authenticated users to choose the webpage skinthey like best; the Google search engine allows users (even non-registered ones) to decide how many search results per page they want to see.

TRACKING

Tracking cookies may be used to track internet users' web browsing habits. This can also be done in part by using the IP address of the computer requesting the page or the referrer field of the HTTP request header, but cookies allow for greater precision. This can be demonstrated as follows:

1. If the user requests a page of the site, but the request contains no cookie, the server presumes that this is the first page visited by the user; the server creates a random string and sends it as a cookie back to the browser together with the requested page;
2. From this point on, the cookie will be automatically sent by the browser to the server every time a new page from the site is requested; the server sends the page as usual, but also stores the URL of the requested page, the date/time of the request, and the cookie in a log file.

Unit 4 Web Server (18 hrs)

Web Technology:-Creating dynamic content:

Creating dynamic content:

Web servers are increasingly being used to deliver dynamic content rather than static HTML pages. In order to generate web pages dynamically, servers
Need to execute a script, which typically connects to a DBMS.

To generate dynamic contents, web servers need to execute a program, through some server-*side scripting mechanism*. This script typically connects to a DBMS, performs a query, retrieves the results, and formats them in HTML in order to be returned to the user.

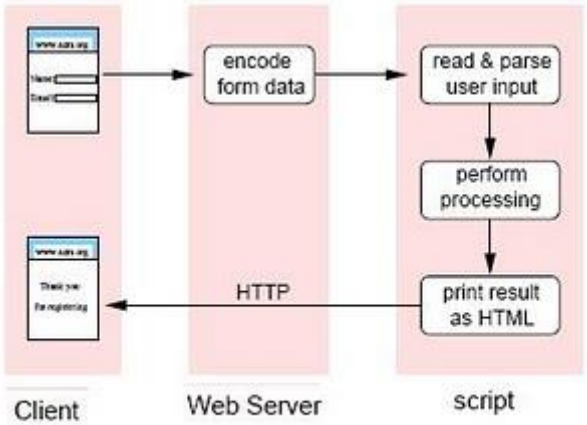


Figure 1:

Using control flow to control dynamic content generation,
(a)Syntax Errors and (b) Runtime Errors (c) Logical Errors:
Architecting Web Application,
Architecture
ASP.NET requires a host. On Windows Server™ 2003, the default host is the Internet Information Services (IIS) 6.0 worker process (W3wp.exe). When you use the ASP.NET Process Model, the host is the ASP.NET worker process (Aspnet_wp.exe).
When a request is received by ASP.NET, the request is handled by the Http Runtime object. The Http Runtime is responsible for application creation and initialization, managing the request queue and thread pool, and dispatching the incoming requests to the correct application. After the request is dispatched to the appropriate application, the request is passed through a pipeline. This pipeline is a staged, event-based execution framework consisting of multiple Http Module objects and a single Http Handler object. This architecture is shown in Figure 6.1.

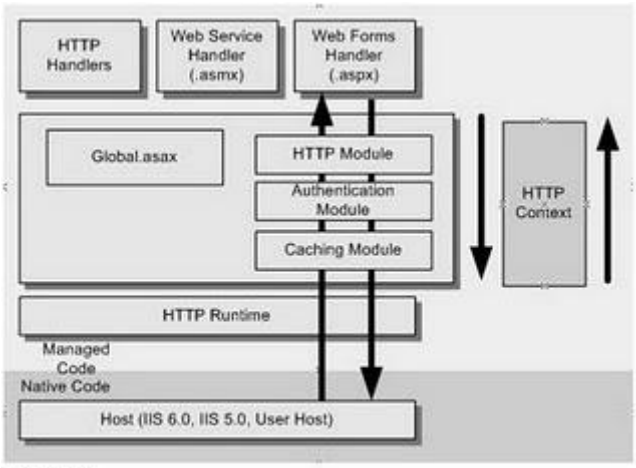


Figure 2

Windows Applications and Web Applications

Developers are often confused about the difference between Windows Applications and Web Applications. Above shown details will provide you brief information about various types of computer applications and provides an introductions to web applications.

DIFFERENT TYPES OF COMPUTER APPLICATIONS

It is very hard to divide applications into any strict categories. There is no clear definition exists to categorize computer applications. However, here is a small list of different types I can think of:

1. Embedded Systems

2. Windows applications (also called 'Desktop applications')

3. Web Applications

Embedded Systems
Have you ever used a digital diary (also called 'digital organizer') or a mobile phone ? Do you know when you save a name and address, how does it get saved in it?

It is a small computer program 'Embedded' in that device. It is similar to any small address book computer program that you can write using your favorite visual basic or c++. The only difference is, it is written using some special language and 'embedded' into a chip in the device inside the mobile phone or digital diary.

If you find a bug in a regular computer program, you can simply write a better program, compile it and copy to your computer. But if you find a bug in your mobile phone's embedded program, you cannot copy it! The manufacturer has to embed the new program in a new chip and replace the chip in your mobile phone!!

WINDOWS APPLICATIONS

If you don't know what a 'Windows Application' is, probably you have never seen a computer. Almost any application you see on a desktop computer is called 'Windows Application'. It is also called 'desktop applications' since they are mostly used in desktop computers.

Some common examples of desktop applications are:

1. Paint Brush program

2. Calculator program

3. MSN Messenger

4. Yahoo Messenger

The first three windows applications are written by some programmers sitting in Microsoft office and they give it free to all who buy Windows operating system. The Yahoo messenger is written by Yahoo programmers and they give it free to download from their web site.

If your neighbor ask you to write small 'Address book' application for his personal use, you are going to write a 'windows application'.

Now you must have a good idea of what is windows application.

WEB APPLICATIONS

I am sure you have seen at least one web application! Do you know how I guessed it? It was easy to guess. This tutorial is a web application and you are currently reading this tutorial from our web site (unless you copied it to somewhere...)

So, what is a web application?

A web application is also called 'web site'. A web site is a collection of web pages hosted on a special computer called 'web server'.

Now you are reading this tutorial. This chapter is a page among several other pages part of our web application. The name of our web application is 'AspSpider.com'. This web site (web application) is running in our web server, which is located in a safe place in USA. You are a 'visitor' to our site and you are accessing our web application using a tool called 'Internet Explorer' (or, some other browser like Netscape etc). We don't know where you are (we have several ways to find it, which we will explain in some other chapter)

So, here is some interesting points about a web application:

- **A web application is a collection of web pages.**
- **A web application needs a web server to run.**
- **Web server can be located anywhere and visitors need not be even in the same country of the web server.**
- **Visitors can access the web application using a tool called 'browser'. There are many browsers exists. Most widely used browser is 'Internet Explorer'. This is provided by Microsoft and it is free. Another famous free browser is 'Netscape'.**

Hope this chapter gave a clear picture about web applications and how they are different from windows applications.

Infrastructure to develop a web application (web site)

SO, READY TO DEVELOP A WEB SITE?

ANYONE WHO WANT TO DEVELOP A WEB APPLICATION MUST HAVE THE FOLLOWING SYSTEMS:

1. A WEB SERVER.
2. AN EDITOR TO DEVELOP THE WEB PAGES.
3. A BROWSER TO VIEW THE WEB PAGE YOU DEVELOP.
4. A DATABASE PROGRAM LIKE MS ACCESS, SQL SERVER ETC, IF YOUR WEB SITE NEED TO SAVE DATA INTO A DATABASE.

IN THE REAL WORLD SITUATION, A WEB SERVER WILL BE HOSTED ON A SECURE SERVER, LOCATED IN A SAFE PLACE AND WILL BE ALWAYS CONNECTED TO HIGH SPEED INTERNET. HOWEVER, TO DEVELOP A WEB APPLICATION, YOU DON'T NEED TO WORRY ABOUT SECURITY AND INTERNET CONNECTIVITY. YOU CAN USE YOUR OWN DEVELOPMENT COMPUTER AS THE 'WEB SERVER'.

SO, YOUR DEVELOPMENT COMPUTER MUST HAVE ALL THE 4 SYSTEMS MENTIONED ABOVE.

NOTE: WEB SERVER

There are several types of web servers. But if you like to develop ASP.NET web applications, you need a specific web server called 'Internet Information Server' (IIS).

IIS comes as part of Windows. But it is not installed by default, when you install Windows. Please see the chapter 'Installing IIS' to find more about installing IIS.

EDITOR TO DEVELOP WEB PAGES

Ideally, you do not need any special editor to develop a web application. If you are an expert, you can simply use notepad to type HTML and the code for the web pages. However, who want to hand-wash the vessels when there is a dish washer?!

You don't need to make your hands dirty! Microsoft gives a tool called 'Visual Studio .NET' to edit web pages and write code for ASP.NET.

Visual Studio .NET (VS.NET)

Visual Studio .NET allows to easily create web pages. Some of the benefits in using Visual Studio .Net are:

- **You can simply drag and drop html controls to the web page and VS.NET will automatically write the HTML tags for you.**
- **Start typing an HTML tag and VS.NET will complete it! When you start typing a tag, VS.NET will show you the HTML tags starting with the characters you typed. So, you don't need to even remember all the tags.**
- **If you type any HTML tags wrong, VS.NET will highlight the errors and tell you how to correct it.**

So, even if you are not an expert, VS.NET can help you develop great web pages.

BROWSER

You need a browser to view the web pages you create. If you have any windows operating system in your computer, you will already have a free browser (called 'Internet Explorer')

DATABASE PROGRAM

A database program like MS Access or SQL Server is required only if you need to save data into database. It is not mandatory that all web sites need a database program.

WHAT IS A VIRTUAL DIRECTORY?

A virtual directory represents a web application and it points to a physical folder in your computer.

A web application is accessed using a virtual directory name instead of a physical folder name. For example, if you have a web application called "Shop cart" in your machine; you will have a virtual directory for this web application. You will access your web application using the URL httP://localhost/Shopcart. If your virtual directory name is "Test", then your web application url will be "http://localhost/Test".

Assume you have a web application called "Shop cart", created under the physical folder "C:\MyProjects\Shopcart".

You can go to IIS and see this virtual directory listed. Right click on this virtual directory name in IIS and see the properties. You can see that this virtual directory is pointing to the physical location "C:\MyProjects\Shopcart".

If you have a file called "File1.aspx" under the folder "C:\MyProjects\Shopcart\", then you can access this file using Internet Explorer with the URL "http://localhost/Shopcart/File1.aspx"

HOW TO CREATE A VIRTUAL DIRECTORY?

When you create a new web project using, a new virtual directory will be created automatically for you. This virtual directory will point to a new folder created under C:\Inetpub\wwwroot.

If you like to better organize your projects and files in your favorite folder, you must manually create a new folder for each project in your preferred location and convert it into a virtual folder manually.

There are couples of ways you can do this.

Method 1: Open the IIS. Right click on the node "Default Web Site" and select "new Virtual Directory". When it prompt you to enter the "alias", enter the virtual directory name you want(Eg: Shop cart). In the prompt for "directory", select the folder which you want to make a virtual directory (Eg: C:\MyProjects\Shopcart). Select other default values and press "Finish". Now you should be able to see your new virtual directory in IIS.

Method 2: In the explorer, go to the folder(Eg: C:\MyProjects\Shopcart) which you want to make a "virtual directory". Right click on the folder name and select "Properties". Select the tab "Web sharing" and select teh option "Share this folder". It will prompt you with a default Alias name same as the folder name (Eg: Shop cart). Simply select the default values and press "OK".

How web applications work?

Web applications work quite different from regular windows applications. This chapter explains the fundamentals of a web application and how a web page is served when a client makes a request for a web page.

LIFE CYCLE OF A WEB REQUEST

Viewing a web page is a simple process for a visitor. Just type the URL in a web browser like Internet Explorer or click on a hyper link in any existing web page. The web browser will display the page instantly to you.

But do you know that there are several computers involved in this process? Even though it is a very complex process, we can summarize the process as shown below:

1. You type the web page address (URL) in a browser. For example, consider the current page http://www.aspspider.com/tutorials.aspx. This URL has 3 parts:

- **The protocol - http:**
- **The server name - www.aspspider.com**
- **The file name - tutorials.aspx**

2. Browser communicates with a computer in internet called 'Domain Name Server' to find out the IP Address of the server (Eg: www.aspspider.com).

3. Browser established a connection to the web server at that IP Address.
4. Server composes a 'Request' for the specified URL and sends the request to the web server to which it has established a connection.
5. The web server identifies the type of the page requested. If it is an asp.net web page, then browser knows that needs some processing by the asp.net service running as part of the web server. The request is handed over to the asp.net service. The asp.net service processes the asp.net page and generates the html output.
6. Web server sends the requested page to the browser.
7. When a response is received by the browser, it displays the web page to the user who typed the URL.

After you typed the URL in the browser, the request sent by the browser may go through several computers in the internet before it reaches the actual web server.

You must be surprised to know that so many things happen and several computers are involved before a simple web page is displayed to you. Most of the steps in the above process happens behind the screen. A visitor need not worry about how a web page is processed and served to the browser.

So, by now you must have got a better picture about how a windows application is different from a web application. When you run a windows application (desktop application), only one computer is involved in the whole process. You start an application in your computer and it runs in your computer. But when you request a web page from your home computer, the request goes through several servers in the internet and finally it reaches a computer called 'Web Server'. The actual web page is processed in the web server.