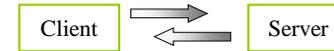# Web Technology

COMP476 – Networked
Computer Systems

## Client-Server Paradigm

The method of interaction used when two application programs communicate over a network. A server application waits at a known address and a client application contacts the server.

The World Wide Web is an example of a client-server system using TCP/IP.

Client ⇄ Server

## Hypertext and Hypermedia

- The Web is a **distributed hypermedia** system that supports interactive access.
- A hypermedia system provides extensibility to a hypertext system by incorporating elements other than only text.
- A non-distributed system would be a closed network system (usually a single computer) where every link could be verified. Distributed systems are spread across many independent systems where links may become broken.

## Document Representation

- Web Documents use the HTML (Hyper Text Markup Language) representation.
- Instead of specifying a detailed document format, HTML allows a document to contain general guidelines for display and allows a browser to choose the details.
- Two browsers may display an HTML document differently.

## HTML Format

- HTML pages are made up of HTML tags which give the page format and structure.

  The general format of a tag is:
  - **`<TAGNAME>`** to start and
  - **`</TAGNAME>`** to end

## Basic HTML format

```
<HTML>
  <HEAD>
    <TITLE> a title </TITLE>
  </HEAD>
  <BODY>
  body of document here
  </BODY>
</HTML>
```

## HTML Tags

- Links in HTML can appear as text or images that can be selected to redirect the browser.

The general format for a link is:

**`<A HREF =`** "linkname.htm"**`>`** to start and **`</A>`** to end

- Media can be incorporated into HTML using tags.

  The general format to insert an image is:
  - **`<IMG SRC = "`** imagename.gif**`">`**

- Other HTML tags can be used to control how these elements are formatted and related to one another.

## HTML is a

1. Protocol for sending web pages
2. Executable program
3. Markup language
4. Graphical programming language



25% 25% 25% 25%

Protocol for s...   Executable pro...   Markup languag...   Graphical prog...
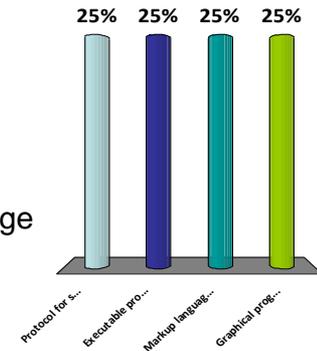
## Types of Web Documents

Web pages can come in a variety of forms. In general, any of these forms can be categorized as one of the following.

1. <u>Static Web Documents</u> - same file sent every request.

2. <u>Dynamic Web Documents</u> - program at the server creates a document and sends it to client.

3. <u>Active Web Documents</u> - program sent from server and run on the client.

## Static Documents

- A static Web document resides in a file that is associated with a web server.
- The author of a static document determines the contents at the time the document is written.
- Because the contents do not change, each request for a static document results in exactly the same response.

Example:

Static web pages, FTP files.

## Dynamic Documents

- A dynamic Web document does not exist in a predefined form. They created by a Web server whenever a browser requests the document.
- When a request arrives, the Web server runs a program that creates the dynamic document.
- The server returns the output of the program as a response to the browser that requested the document. Because a fresh document is created for each request, the contents of a dynamic document can vary from one request to another.

Example: CGI

## Dynamic Documents (CGI)

- Common Gateway Interface (CGI) is a standard defining how a program will interact with a server to generate dynamic documents.
- CGI can use any programming language and can generate any type of output.

## CGI output

- The CGI header indicates the type of document, such as:

    **Content-type: text/html**

    **Location: /new/file.txt**

- Parameters are passed in environment variables. Forms are web pages that send the contents of fields to an active document. Each field has a name. The browser sends the entered values to the server as:

**fieldname = value,field2=value2**

## Active Documents

- Active documents contain programs downloaded from the server to the browser and run on the clients machine.
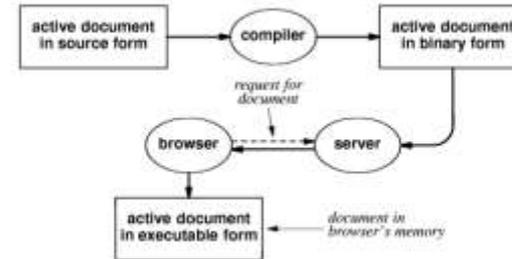


Examples: ActiveX and Java.

## Active Documents (Java)

- Java is the name of a specific technology (by Sun Microsystems) used to create and run active documents.
- In the case of java these documents are referred to as applets.
- Java applets include three key components.
  - Programming Language – can be used to make conventional programs or applets.
  - Runtime Environment – Provides the facilities to run a Java program.
  - Class Library – Makes applets easier to write.

## ActiveX

- A Microsoft technology for active documents.
- Objects can be written in several languages including C++ and Visual Basic
- Is not transportable to non-Windows platforms.

## Dynamic Document involve

1. Execution on the client
2. Execution on the server
3. Execution on the DNS
4. No execution

**25%** **25%** **25%** **25%**

Exeution on t... | Exeution on t... | Exeution on t... | No execution
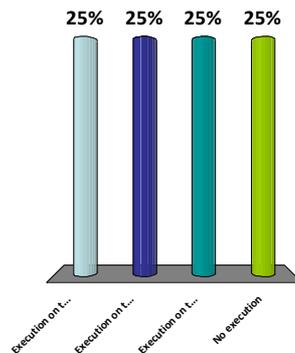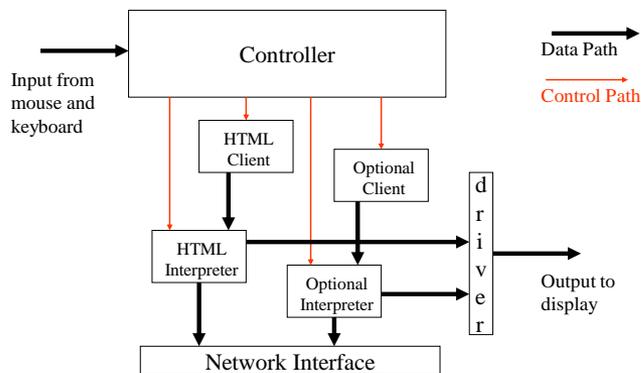
## Browser Architecture

• Web browsers are more complex in structure than Web servers. Servers handle relatively straightforward tasks in a repetitive manner. Browsers handle most of the details of document access and display.

Browser functions

• The browser is made up of a controller, a set of clients and, and a set of interpreters.

• Each function of the browser has its own set of dedicated operations.

## Browser Architecture

Controller

Input from mouse and keyboard

HTML Client

Optional Client

HTML Interpreter

Optional Interpreter

d r i v e r

Network Interface

Data Path

Control Path

Output to display

This example may show a better conceptualization of browser organization.

## Optional Clients

• Browsers can perform other tasks besides interpreting HTML with an HTTP client.

  – Many browsers include a built in FTP client.

    • The client is typically invoked automatically. The protocol header appears as:

      `ftp://ftp.university.edu/dir/file.exe`

  – Email client software is often built into browsers.

    • Here the client is typically opened in a separate application from the browser. However features of the client are still very much intertwined to the browser interface.

## Streaming Media

- Streaming audio or video are "graphics" files embedded in an HTML document.

- Streaming media is displayed as it is received instead of waiting until the entire file is received.

- Most streaming media uses UDP.

## Cache

- Web browsers use a cache to improve document access and retrieval.

- Pages that are retrieved from a server a stored on the local disk.

- The disk cache is checked before retrieving the same item from the server

- Pages can contain an expiration date specifying when they must be refreshed.

## Hyper-Text Transfer Protocol

- Hyper-Text Transfer Protocol is the main request-response (client-server) protocol used to transfer web documents.

- HTTP is an application layer protocol using TCP.

- Other high level protocols for the Web include FTP and Telnet.

## Web Document Transfer & HTTP

- When a browser interacts with a Web server, the two programs follow the Hyper-Text Transfer Protocol.

- HTTP allows a browser to request a specific item, which the server then returns.

request

| Client (Browser) | | Server |

reply
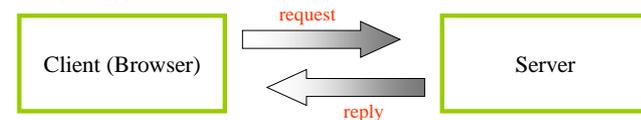
## HTTP Request Format

The protocol sends requests and responses in ASCII characters that can easily be read. The request is always terminated by a blank line. The format of the request sent by a client browser (such as Mozilla or Internet Explorer) to a web server is:

> *Method   filename   HTTP/1.1*
> *options   CRLFCRLF*

## Example HTML **GET**

**GET /mypage.html HTTP/1.1↵**

**HOST: williams.comp.ncat.edu↵↵**

- This example requests the server to send the web page, mypage.html, to the client's browser. The browser has indicated that it is using version 1.1 of the protocol. Note that the request is terminated by two end of line characters(↵↵).

## HTTP Methods

| **GET** | Get a file from the server. |
|---|---|
| **HEAD** | Get information about a file from the server. |
| **POST** | Send information to the server. |
| **PUT** | Send a file to be stored on the server. |
| **DELETE** | Delete a file on the server. |
| **OPTIONS** | Request the available server options. |
| **TRACE** | Invoke a loop-back of the request message |

## Server Response

- The server responds with a status line, including the message's protocol version and a success or error code and possibly message content.

**HTTP/1.1 *statuscode reason*↵**
***response options*↵↵**

**file contents**

## Response Example

```
HTTP/1.1 200 OK
Date: Sun, 26 Nov 2000 23:48:00 GMT
Server: Apache/1.3.6 (Win32)
Last-Modified: 17 Nov 2000 12:51:44
Content-Length: 4683
Connection: close
Content-Type: text/html
<html>
<head> etc.
```

## Server Status Codes

• Some of the status codes returned by the server are:

```
1xx: Informational
2xx: Success
200 OK
3xx: Redirection
301 Moved Permanently
304 Not Modified
```

## Server Status Codes (cont.)

```
4xx: Client Error
400 Bad Request
401 Unauthorized
402 Payment Required
403 Forbidden
404 Not Found
410 Gone
5xx: Server Error
500 Internal Server Error
```

## Multiple Files Per Page

• A web page is usually composed of several files; the original HTML page and embedded graphic files.

• The web browser first retrieves the HTML file which specifies the additional files.

• Each file is retrieved separately using HTTP

## HTTP/1.0 Operations (get file)

**Client**

- Create a socket.
- Connect to the server at port 80.
- Send a GET request to the server for the desired file

*Server*

- *Read request from new connection*
- *Send a header describing the file.*
- *Send the contents of the file.*
- *Close the connection.*

## HTTP/1.0 Operations (get graphic)

**Client**

- Create a socket.
- Connect to the server at port 80.
- Send a GET request to the server for the graphic file.

*Server*

- *Receive request from new connection.*
- *Send header describing graphic file.*
- *Send the contents of the graphic file.*
- *Close the connection.*

\* for each graphic referenced

## HTTP/1.1 Operations

**Client**

- Create a socket.
- Connect to the server at port 80.
- Send a GET request to the server for the desired file

*Server*

- *Send a header describing the file.*
- *Send the contents of the file.*

❖ HTTP/1.1 provides for keeping a connection open to transfer multiple files.

## HTTP/1.1 Operations (get graphic)

**Client**

- Send a GET request on the existing server connection for the graphic file.

*Server*

- *Send a header describing the graphic file over existing connection.*
- *Send the contents of the graphic file.*

\* for each graphic referenced

# HTTP Standards

- The Hyper-Text Transport Protocol is defined by RFC2616 and others of the Internet Engineering Task Force (IETF).