

# L e c t u r e #



29

# Lecture Goals

- Development of a little Web Server
- Web Server will server HTTP requests sent via a Web Browser using URLs

- URLs:

<http://www.vu.edu.pk/default.html>

<http://www.vu.edu.pk/index.asp>

<http://www.vu.edu.pk/win32.html>

<http://www.vu.edu.pk/courses/win32.html>

# Uniform Resource Locator

- Anatomy of a URL (Uniform Resource Locator):

<http://www.vu.edu.pk/courses/win32.html>

http:// Protocol

[www.vu.edu.pk](http://www.vu.edu.pk) vu.edu.pk server on World Wide Web

/courses/win32.html win32.html file in courses directory on that server

Fetches an HTML file named **win32.html** from **vu.edu.pk** HTTP Server on the World Wide Web

# HTML

- **HTML** – Hyper Text Mark-up Language
- Contains text-formatting information e.g. font faces, font colours, font sizes, alignment etc.
- Contains HyperLinks: text that can be clicked to go to another HTML document on the Internet.
- HTML **tags** are embedded within normal text to make it hypertext

# Web Browser

- **HTTP Client** – a Web Browser e.g. Microsoft Internet Explorer, Netscape Navigator
- Connect to your HTTP web server, requests a document, and displays in its window

# HTTP Protocol

- **HTTP** – Originally developed by Physicists
- Meant to share technical HyperText documents across locations
- Clickable HyperText is much easier to use instead of conventional sidebars and indices
- Text-based protocol: meant to transport printable text-data and NOT binary data

# HTTP Protocol

- HTTP is a **Stateless protocol**
  - No information or “state” is maintained about previous HTTP requests
  - Easier to implement than state-aware protocols

# Encoding and Decoding

- HTTP is a Text Transport Protocol
- Transferring binary data over HTTP needs Data Encoding and Decoding because binary characters are not permitted
- Similarly some characters are not permitted in a URL, e.g. SPACE. Here, URL encoding is used



# Encoding Example: Escape Sequences

Including a Carriage Return / Line feed in a string

```
printf("Line One\nThis is new line");
```

Including a character in a string not found on our normal keyboards

```
printf("The funny character \xB2");
```

# URL

- Anatomy of a URL (Uniform Resource Locator):

<http://www.vu.edu.pk/courses/win32.html>

http:// protocol

[www.vu.edu.pk](http://www.vu.edu.pk) Web Server

courses/win32.html location of file on server

- Or <http://www.vu.edu.pk:80/.../....>  
:80 specifies Port Number to use for connection

# Virtual Directory

- / represents the Home Directory of a Web Server
- IIS (Internet Information Server) has c:\inetpub\wwwroot\ as its default Home Directory
- Here, /courses/ either corresponds to a **Physical Directory**  
c:\inetpub\wwwroot\courses  
OR  
a **Virtual Directoy** (*contd.*)

# Virtual Directory (contd.)

- In a Web Server, we may specify that /courses/ will represent some other physical directory on the Web Server like D:\MyWeb\. Then /courses/ will be a **Virtual Directory**.
- In Windows2000 and IIS 5.0 (Internet Information Server), a folder's "Web Sharing..." is used to create a Virtual Directory for any folder.

# A Web Browser fetches a page...

- `http://www.vu.edu.pk/courses/win32.html`
- Hostname/DNS lookup for `www.vu.edu.pk` to get IP address
- HTTP protocol uses port 80. Connect to port 80 of the IP address discovered above!
- Request the server for `/courses/win32.html`
- How?

# HTTP Client Request

Method

Resource Identifier

HTTP version

GET /courses/win32.html HTTP/1.0 `crlf`

`crlf`

- Request line is followed by 2 Carriage-Return /Line-feed sequences

# HTTP Server Response

HTTP version

Status Code

Description

• **HTTP/1.1 200 OK** } *Status Line*

• **Content-type: text/html**

• **Content-length: 2061**

*Headers delimited by  
CR/LF sequence*

**crlf**

• *Actual data follows the headers*

# File extensions

- File extensions are non-standard across different platforms and can not be used to determine the type of contents of any file.



# MIME

- In an HTTP response, a Web Server tells the browser MIME type of data being sent
- MIME type is used by the browser to handle the data appropriately i.e. show an image, display HTML etc.

# File extensions and MIME

- Different common MIME types

image/gif

GIF image

image/jpeg

JPEG image

text/html

HTML document

text/plain

plain text

# HTTP Request Headers

- HTTP request may also contain quite a few headers sent by the browser
- HTTP Request Headers usually contain information about the browser type, client's IP address, screen resolution etc.

# MIME

- **MIME: Multi-purpose Internet Mail Extensions**
- **MIME Encoding** features were added to enable transfer of binary data, e.g. images (GIF, JPEG etc.) via mail.
- Using MIME encoding HTTP can now transfer complex binary data, e.g. images and video

# RFC

- Short for *Request for Comments*, a series of notes about the Internet, started in 1969 (when the Internet was the ARPANET). An Internet Document can be submitted to the IETF by anyone, but the IETF decides if the document becomes an RFC. Eventually, if it gains enough interest, it may evolve into an Internet standard.
- HTTP version 1.1 is derived from HTTP/1.1, Internet RFC 2616, Fielding, et al.
- Each RFC is designated by an RFC number. Once published, an RFC never changes. Modifications to an original RFC are assigned a new RFC number.

# MIME encoding

- MIME: Short for *M*ultipurpose *I*nternet *M*ail *E*xtensions, a specification for formatting non-ASCII messages so that they can be sent over the Internet.
- Enables us to send and receive graphics, audio, and video files via the Internet mail system.
- There are many predefined MIME types, such as GIF graphics files and PostScript files. It is also possible to define your own MIME types.
- In addition to e-mail applications, Web browsers also support various MIME types. This enables the browser to display or output files that are not in HTML format.
- MIME was defined in 1992 by the Internet Engineering Task Force (IETF). A new version, called S/MIME, supports encrypted messages.



# HTTP Status codes

- **404 Not Found**
  - requested document not found on this server
- **200 OK**
  - request succeeded, requested object later in this message
- **400 Bad Request**
  - request message not understood by server
- **302 Object Moved**
  - requested document has been moved to some other location

# HTTP Redirection

- **HTTP/1.1 302 Object Moved**
- **Location:** <http://www.vu.edu.pk>

`crlf`

- Most browsers will send another HTTP request to the new location, i.e. <http://www.vu.edu.pk>
- This is called Browser Redirection



# 1 HTTP Request per 1 TCP/IP connection

- HTML text is received in one HTTP request from the Web Server
- Browser reads all the HTML web page and paints its client area according to the HTML tags specified.
- Browser generates one fresh HTTP request for each image specified in the HTML file

# Server Architecture

- Ability to serve up to 5 clients simultaneously
- Multi-threaded HTTP Web Server
- 1 thread dedicated to accept client connections
- 1 thread per client to serve HTTP requests
- 1 thread dedicated to perform termination housekeeping of communication threads
- Use of Synchronisation Objects

# Server Architecture: Why threads?

- Many WinSock function calls e.g. `accept()` are blocking calls
- Need to serve up to 5 clients simultaneously using other WinSock blocking calls
- Need to perform termination tasks for asynchronously terminating communication threads