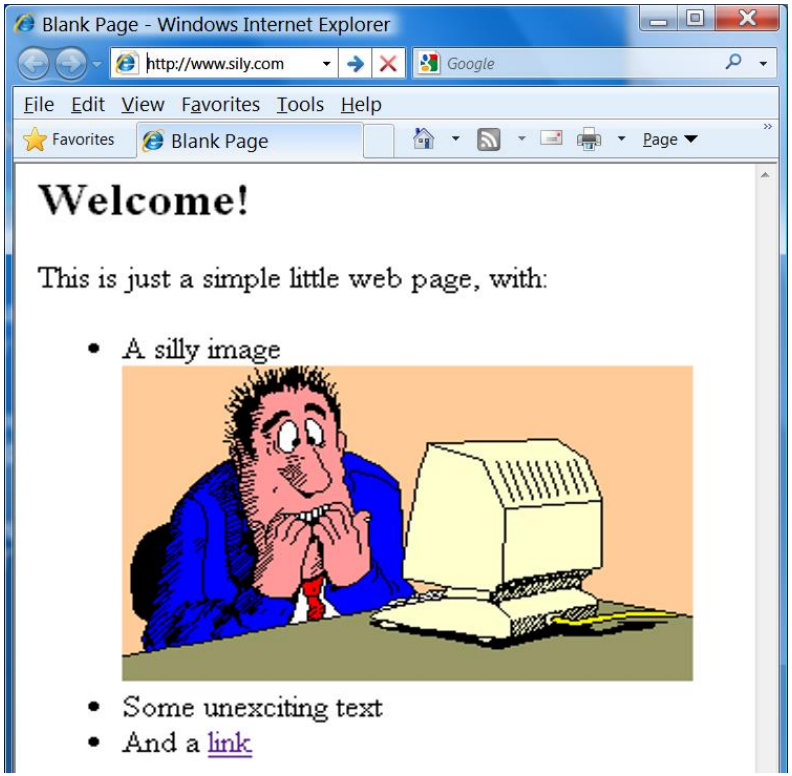
COMP 4021
Internet Computing

The Browser Process / HTTP

David Rossiter

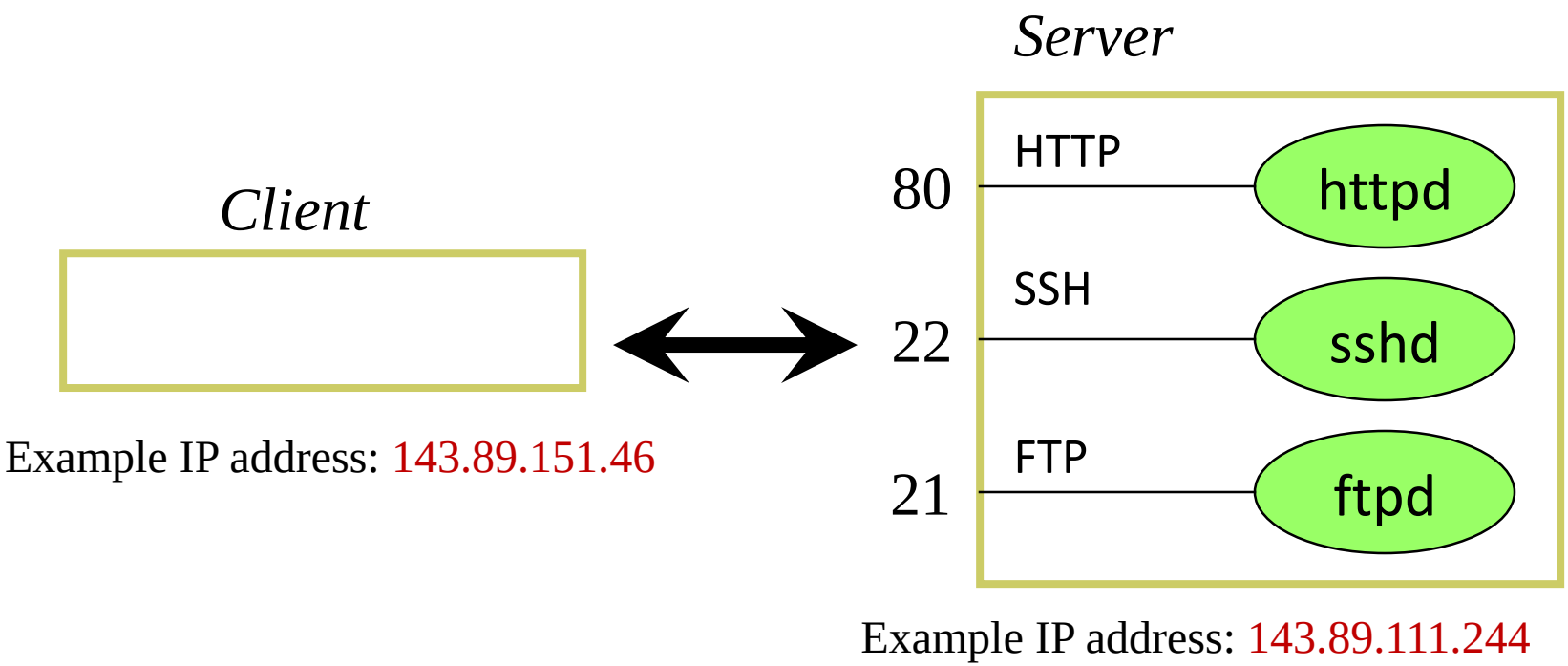
A Browser Uses HTTP

- Here is a simple web page
- Assume this web page is at `http://www.silly.com`
- To get the page, you type the URL in the browser and press Enter
- The browser requests the web page using *HTTP*



IP Address and Ports

- May processes may be running on a server
- They each use a different *port* (=door)



Example Ports

Protocol	*Default Port*
FTP	21
SSH	22
Telnet	23
HTTP	80 ⇐
NNTP (Usenet news)	119
ICQ	5190
Quake game	26000
Half-life game	27010

Client-Server Communication

- The browser connects to the machine silly.com using the HTTP protocol
- No port was specified by the user so the browser assumes port 80

Client

HTTP Protocol

Server

Client's Request

- The message (called a *request*) that the browser sends to the silly.com server at port 80 is:

```
GET / HTTP/1.1
```

You could simply send this line only.

Could send this also

```
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
```

The program is IE, running on Windows XP

Server's Response

Response header

```
HTTP/1.1 200 OK
Date: Mon, 1 Nov 2010 09:09:47 HKT
Server: Apache/1.3.6 (Unix) mod_ssl/2.2.8 OpenSSL/0.9.2b
Last-Modified: Mon, 14 Apr 2008 09:39:08 HKT
Accept-Ranges: bytes
Content-Length: 507
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html
```

The file follows the header

```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
   <title>Simple page</title>
</head>
<body>
<h2>Welcome!</h2>
This is just a simple little web page, with:
<ul>
<li>
A silly image <img src="man.gif" height="169" width="272" />
</li>
<li>Some unexciting text</li>
<li>
And a <a href="http://www.winniethepooh.com">link</a></li>
</ul>
</body>
</html>
```

The silly.com server responds with everything shown here

Server's Response - Header

```
HTTP/1.1 200 OK
```

This line tells the client what version of the HTTP protocol the server uses and says that the document has been found and is going to be transmitted.

```
Date: Mon, 1 Nov 2010 09:09:47 HKT
```

Current date on the server in Greenwich Mean Time (GMT)

```
Server: Apache/1.3.6 (Unix) mod_ssl/2.2.8 OpenSSL/0.9.2b
```

Tells the client what type of software the server is running, in this case Apache, version 1.3.6, running under Unix

```
Last-Modified: Mon, 14 Apr 2008 09:39:08 HKT
```

Tells the client the last time that the document was modified

```
Content-Length: 507
```

Tells the client how many bytes are coming

```
Content-Type: text/html
```

Tells the client the type of the document

Client's Request

```
. . .
A silly image <img src="man.gif" height="169" width="272" />
. . .
```

The browser sees this in the HTML, and understands that the web page also needs an image file. So it sends a request for the image:

```
GET /man.gif HTTP/1.1

User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
```

Server's Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Nov 2010 09:09:48 HKT
Server: Apache/1.3.6 (Unix) mod_ssl/2.2.8 OpenSSL/0.9.2b
Last-Modified: Mon, 14 Apr 2008 09:39:12 HKT
Content-Length: 4627
Content-Type: image/gif

v({CCPP P]]]kkkxxx¡¡¡®®®»»»»ÉÉÉÖÖÖäääñññÿÿÿÿÿÿS?Ul¶D@âU
nQÿÿ{.(/{Rn;/,ÿÿ          H° *\È Ï F(±¡Å3jÜÈ±£Ç 5VD(âÃG
. . .
```

The browser knows that GIF data follows

This is the GIF data (it looks strange because it is not meant for viewing as text)

Summary

Client

Server

Time

```
GET / HTTP/1.1
. . .
```

```
HTTP/1.1 200 OK
Content-Length: 507
Content-Type: text/html

. . . [HTML data] . . .
```

```
GET /man.gif HTTP/1.1
. . .
```

```
HTTP/1.1 200 OK
Content-Length: 4627
Content-Type: image/gif

. . . [GIF data] . . .
```

Forms

- When you submit a form the browser sends the form data to the server, as well as the name of the program on the server which it needs to be given to

HTML Source Code

```
<html>

<head>
<title>Movie Database!</title>
</head>

<form method="post"
action="http://ihome.ust.hk/~rossiter/cgi-bin/show_environment.php">

<h1>Movie Search</h1>
Select the name and/or the year of the movie you want to search
   for.<p />
Title:<input type="text" name="movie_title" value="" />
Year:<input type="text" name="movie_year" value="" />

<br />
<br />
Press submit when you're ready

<input type="submit" value="Submit">

</form>
</html>
```

Sending Form Data

- After the *Submit* button is pressed, the browser connects to the server shown in the 'action' field, using port 80

```
action="http://ihome.ust.hk/~rossiter/cgi-bin/show_environment.php"
```

- In this case the server is ihome.ust.hk
- The browser then sends:

```
POST /~rossiter/cgi-bin/show_environment.php
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Content-type: application/x-www-form-urlencoded
Content-length: 35

movie_title=spiderman+3&movie_year=
```

35 bytes to follow.

Two parameter pairs separated by a '&', with spaces replaced by '+'

The browser now includes this line, specifying the type of data being sent. If no content is to be sent with the request, this line is not necessary.

Response from Server

- After receiving the client request, the server will give all the sent information to the server side program show_environment.php
- The program does whatever it is programmed to do
- Probably, it outputs something back to the client - whatever it outputs (i.e. prints) goes straight back to the browser

COMP 4021
Internet Computing

Browsers

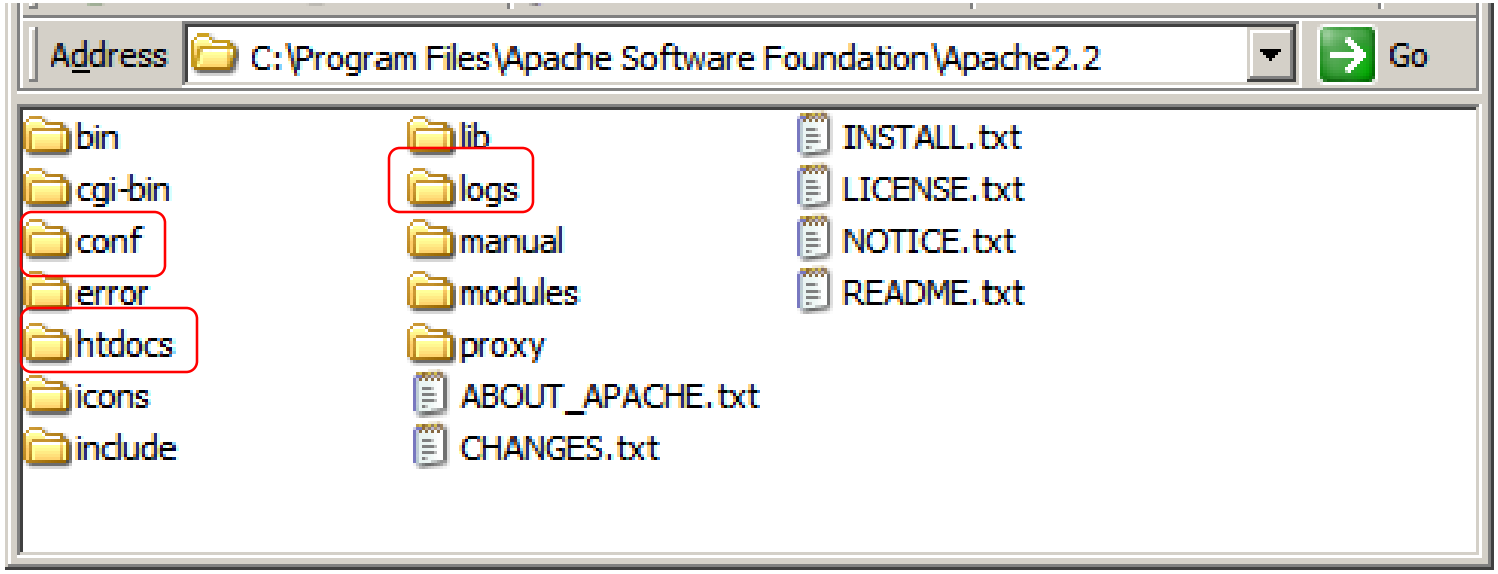
David Rossiter

Browser File Processing

- Browsers must support:
  - HTTP to request pages and respond to server responses
  - Rendering of HTML pages
- After the page is retrieved, the browser will have to do:
  - Retrieve linked external files, e.g., CSS files, JavaScript files, image files, etc.
  - Execute JavaScript and apply CSS
  - Render the page

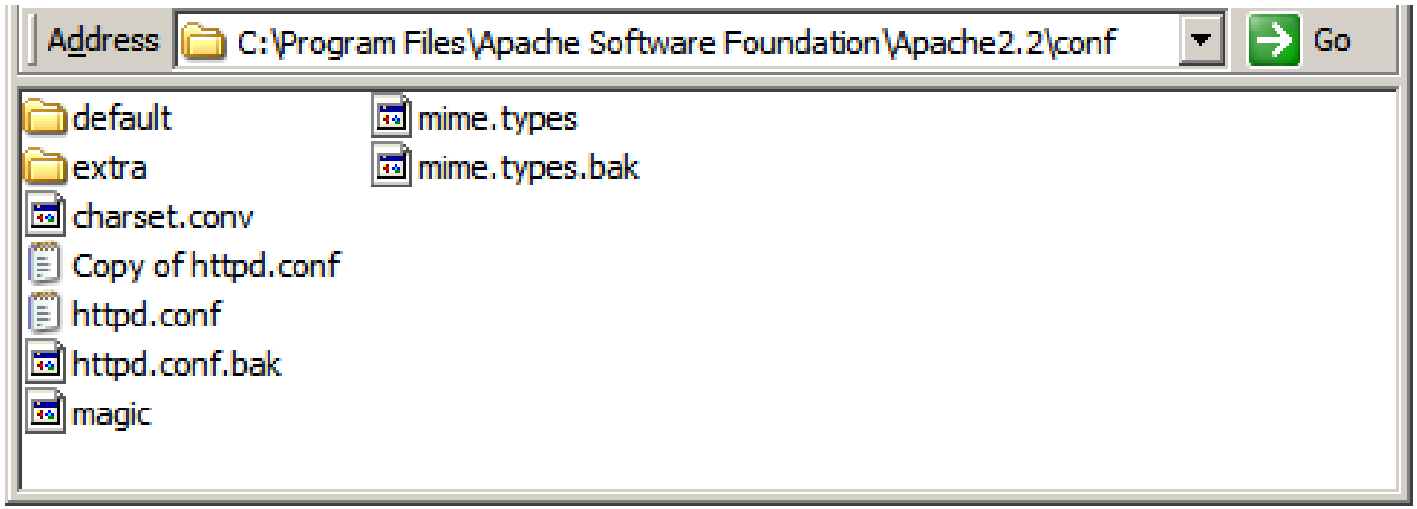
Apache HTTP Server - Directory Structure

- You can install Apache in any directory:
  - E.g., C:\Program Files\Apache Software Foundation\Apache2.1



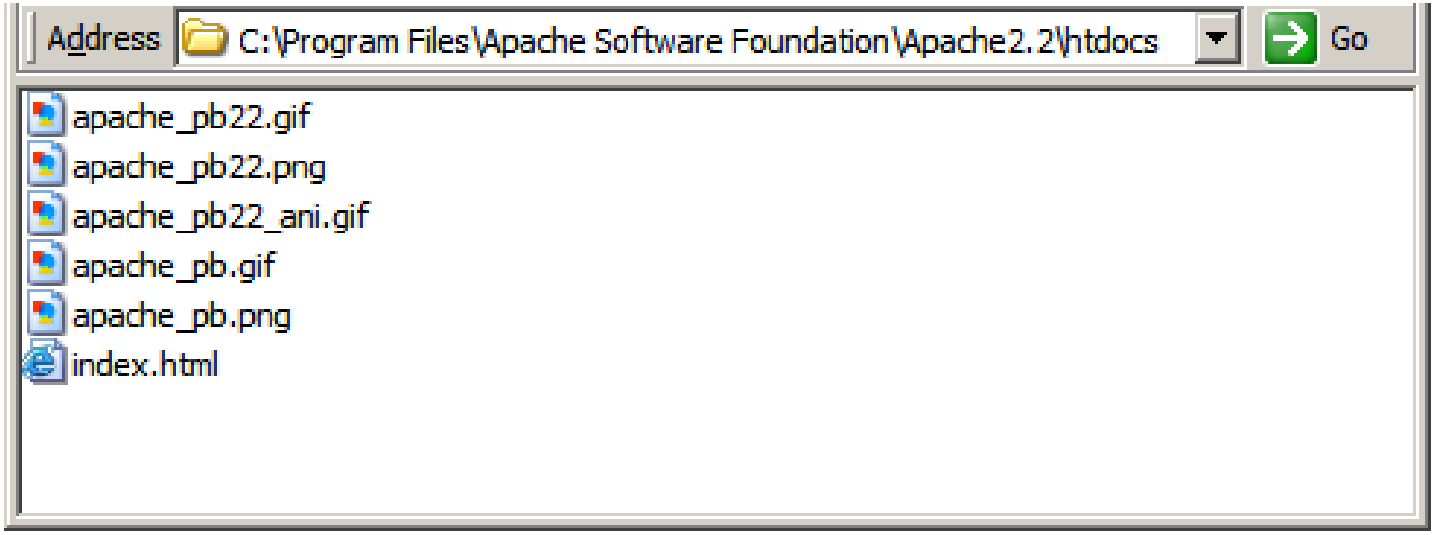
Apache (HTTP Server) – Configuration

- Configuration files are stored in **conf** directory
- Most important function is to configure where your website files are located (i.e., http://www.mysite.com/index.html, where is index.html stored?)



Storing Web Files

- Web files are stored under the **htdocs** directory



Apache Modules

- Modules add various functions to the Apache server; examples of useful modules:
  - **mod_deflate** compresses content before sending it to the browser using gzip compression.
  - **mod_rewrite** allows Apache to rewrite incoming URLs and rewrites them on the fly according to the needs of your server application.
  - **mod_evasive** detects DoS or DDoS attacks by denying IP addresses when suspicious access patterns are detected
  - **mod_security** is a Web Application Firewall that protects websites from attacks such as Code Injection attacks, SQL injection, etc.
  - **mod_ssl** supports HTTPS, strong cryptography via Secure Sockets Layer and Transport Layer Security protocols

Apache Tomcat

- Tomcat is a container for Servlets and JSP
- Tomcat can act as a simple standalone server for Web applications that use HTML, servlets, and JSP
  - The user submits an HTML form
  - Tomcat finds the servlet based on the URL and the deployment descriptor (web.xml) and passes the request to the servlet
  - The servlet writes an HTML page containing the response
  - Or forwards the response to JSP which embeds the response in an HTML page
  - Tomcat returns the HTML page to the user

Take Home Messages

- Web architecture could be as simple of a client-server system serving static pages to dynamic pages served from data stored in database system (3-tier)
- Both web client and web server are very mature
- Apache HTTP Server is the world's most popular web server (and free)
- HTTP implements many functions not covered in the course (e.g., cache control)