ECS417U - Fundamentals of Web Technology

Tao Xiang
t.xiang@qmul.ac.uk

Lecture 2, Tuesday 17/01/17

What we learned last week

- By now, you should be able to:
  - Explain what is Internet and how Internet works
  - Explain what is the Web and how it works
  - Explain the concept of markup language
  - Write simple XHTML codes for text formatting, display them using a web browser, and validate them

Revision: XHTML

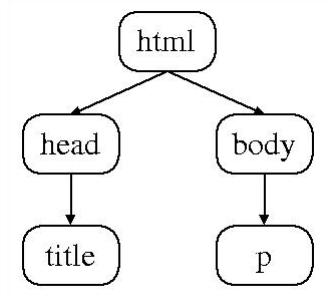
Document Type Declaration

```
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
```

Document Instance

```
<!-- helloworld.html
     A trivial document
     -->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> Our first document: Hello World!
</title>
  </head>
  <body>
    <p>
      Hello World!
    </p>
  </body>
</html>
```


Revision: XHTML



Revision: XHTML

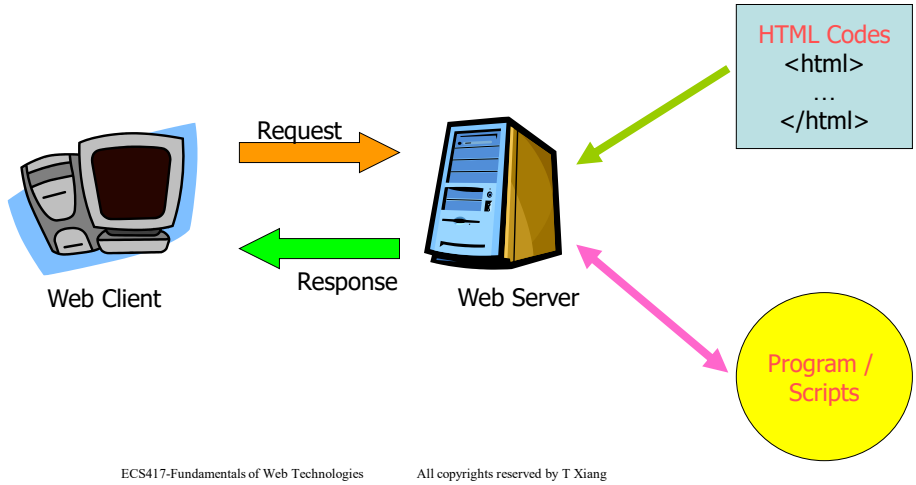
- Basic syntax: elements defined by tags
  - Paragraph:
  - Headings: <h1></h1>
  - Retained text format: <pre></pre>
  - Font:
    - Style:
    - Size: <big></big><small></small>
  - Hypertext link: three types
  - Line break:

Practical Server Basics

Outline

- **Introduction to web client operation**
- Hypertext Transfer Protocol (HTTP)
- Introduction to web server operation
- How to set up a server
- How to use the webprojects server

The request and response model



Web Client

- A web client is a software that accesses a web server by:
    1. Sending an HTTP request message
    2. Processing the resulting HTTP response
- Web browser is one type of web client

Web client functionalities

1. Reformat the URL entered as an HTTP request
2. Convert host name to IP address using DNS
3. Establish TCP connection to the server using IP address
4. Send the HTTP request over TCP and wait for server response
5. Once receiving the response, display the document (in HTML format) contained in the response

Outline

- Introduction to web client operation
- **Hypertext Transfer Protocol (HTTP)**
- Introduction to web server operation
- How to set up a server
- How to use the webprojects server

Hypertext Transport Protocol (HTTP)

- HTTP is a communication protocol specifying how web clients and servers should communicate
- It is based on the request-response communication model:
  - Client sends a request
  - Server sends a response
- One can use the Internet's Telnet protocol to simulate browser request and view server response
  - Under Window, one can use the free SSH/Telnet client PuTTY

HTTP

Connect

```
$ telnet www.google.co.uk 80
Trying 64.233.167.94...
Connected to www.google.co.uk.
Escape character is '^]'.
GET / HTTP/1.1
Host:www.google.co.uk

HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html
…
```

Send Request

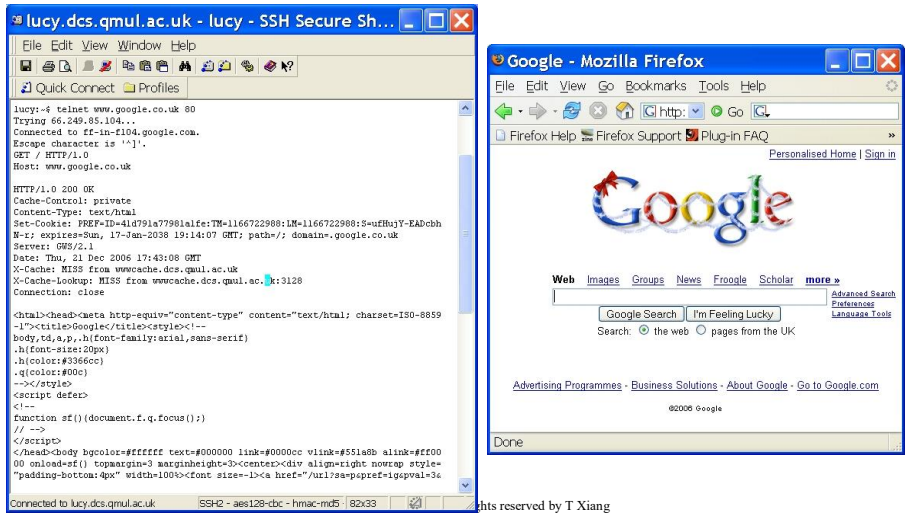
Receive Response

HTTP

Given the example URL (www.google.co.uk), the browser will send a request containing the lines:

```
GET / HTTP/1.1
…
Host: www.google.co.uk
…
```


HTTP



HTTP Request

- Structure of the request:
  - **start line**
  - header field(s)
  - blank line
  - optional body

HTTP Request

- Start line
  - Example: `GET / HTTP/1.1`
- Three space-separated parts:
  - HTTP request method
  - Request-URI
  - HTTP version

HTTP Request

- Start line
  - Example: `GET / HTTP/1.1`
- Three space-separated parts:
  - HTTP request method
  - **Request-URI**
  - HTTP version

HTTP Request

Uniform Resource Identifier (URI)

- Syntax: *scheme : scheme-depend-part*
  - e.g. In http://www.example.com/ the scheme is `http`
- Request-URI is the portion of the requested URI that follows the host name (which is supplied by the required Host header field)
  - e.g. `/` is the Request-URI portion of `http://www.example.com/`

Uniform Resource Identifier (URI)

- URI's are of two types:
  - Uniform Resource Name (URN)
    - Can be used to identify resources with unique names, such as books (which have unique ISBN's)
    - Scheme is `urn`
  - Uniform Resource Locator (URL)
    - Specifies location at which a resource can be found
    - In addition to `http`, some other URL schemes are `https`, `ftp`, `mailto`, and `file`

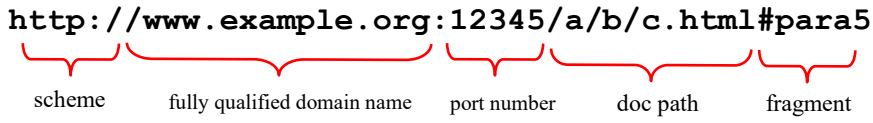
Uniform Resource Locator (URL)

- General form:

  scheme:object-address

- The scheme is often a communications protocol, such as http or ftp
  - For the http protocol, the object-address is: //fully qualified domain name/doc path
  - e.g. http://webprojects.eecs.qmul.ac.uk/txiang/helloWorld.html

HTTP-scheme URL



- Fully qualified domain name can be replaced by an IP address
- Port number is by default 80
- Path is partial path
- Fragment is used together with in-document hypertext links

Locating files using URL paths

- Two types of paths are used in URL
  - Completed path: one that includes all directories along the way; used with *file* scheme
  - Partial path: one that is relative to some base path; used with *http* scheme
- The server concatenate the partial path in the http-scheme URL with a base file path (normally document root) to create the complete path
- URLs cannot include spaces or any of a collection of other special characters (semicolons, colons, ...)

Locating files using URL paths

- If the doc path ends with a slash, it means it is a directory
- If not path is given, the server searches at the top level of the document root for something that is recognised at a homepage (normally index.html).
  - e.g. http://www.bbc.co.uk
- If no homepage is found, directory listing is returned to the browser.

URL document type

- A URL sent to a web server can specify two types of documents:
  - A data file stored on the server that is to be sent to the client
    - XHTML files, plain text files
    - Video, image, audio files
  - A program stored on the server that the client wants to execute, with the output of the program returned to the client (e.g. online shopping)
    - .cgi, .pl, .php, .asp

Multipurpose Internet Mail Extensions (MIME)

- Originally developed for email
- Used to specify to the browser the form of a file returned by the server (attached by the server to the beginning of the document)
- Type specifications
  - Form: type/subtype
  - Examples: text/plain, text/html, image/gif, image/jpeg

MIME (cont'd)

- Server gets type from the requested file name's suffix (.html implies text/html)
- Browser gets the type explicitly from the server
- If the document type is not supported by the browser, helper applications or plug-ins are needed

HTTP Request

- Start line
  - Example: `GET / HTTP/1.1`
- Three space-separated parts:
  - **HTTP request method**
  - Request-URI
  - HTTP version

HTTP Request

- Common request methods:
  - GET
    - Used if link is clicked or address typed in browser
    - No body in request with GET method
  - POST
    - Used when submit button is clicked on a form
    - Form information contained in body of request

HTTP Request

- Structure of the request:
  - start line
  - **header field(s)**
  - blank line
  - optional body

HTTP Request

- Header field structure:
  - *field name* : *field value*
- Syntax
  - Field name is not case sensitive
  - Field value may continue on multiple lines by starting continuation lines with white space
  - Field values may contain MIME types, quality values, and wildcard characters (*'s)

HTTP Request

- Common header fields:
  - Host: host name from URL (required)
  - User-Agent: type of browser sending request
  - Accept: MIME types of acceptable documents
  - Connection: value `close` tells server to close connection after single request/response
  - Content-Type: MIME type of (POST) body, normally application/x-www-form-urlencoded
  - Content-Length: bytes in body
  - Referer: URL of document containing link that supplied URI for this HTTP request

HTTP Response

- Structure of the response:
  - **status line**
  - header field(s)
  - blank line
  - optional body

HTTP Response

- Status line
  - Example: HTTP/1.1 200 OK
- Three space-separated parts:
  - HTTP version
  - status code
  - reason phrase (intended for human use)

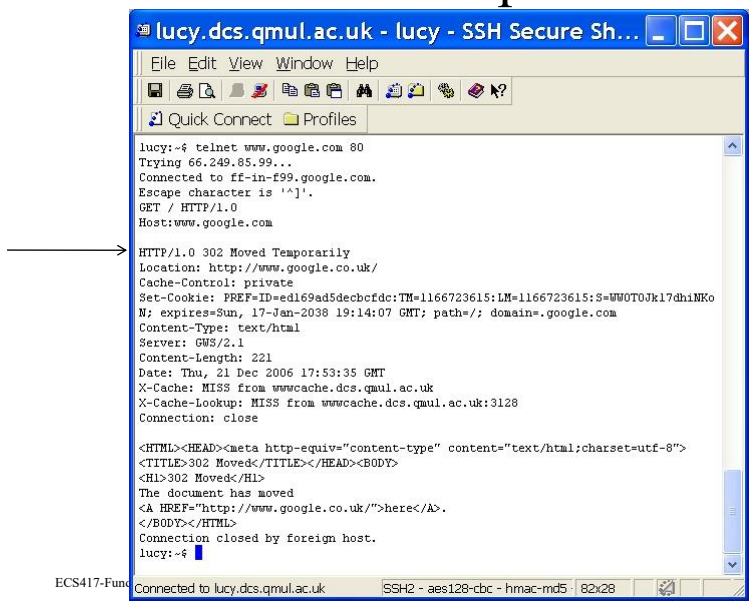
HTTP Response

- Status code
  - It is a three-digit number
  - The first digit is the class of the status code:
    - 1=Informational
    - 2=Success
    - 3=Redirection (alternate URL is supplied)
    - 4=Client Error
    - 5=Server Error
  - Other two digits provide additional information

HTTP Response

- Some common HTTP/1.1 status codes:
  - 301 Moved Permanently
  - 302 Moved Temporarily
  - 307 Temporary redirect
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
  - 500 Internal Server Error

HTTP Response



HTTP Response

- Structure of the response:
  - status line
  - **header field(s)**
  - blank line
  - optional body

HTTP Response

- Common header fields:
  - Connection, Content-Type, Content-Length
  - Date: date and time at which response was generated (required)
  - Location: alternate URI if status is redirection
  - Last-Modified: date and time the requested resource was last modified on the server
  - Expires: date and time after which the client's copy of the resource will be out-of-date
  - ETag: a unique identifier for this version of the requested resource (changes if resource changes)

Outline

- Introduction to web client operation
- Hypertext Transfer Protocol (HTTP)
- **Introduction to web server operation**
- How to set up a server
- How to use the webprojects server

Web Server Functionalities

- Provide responses to browser requests (serving information), via returning either existing documents or dynamically built documents
- Monitor a communications port on the host, accepting HTTP messages when they appear
- Controlling access to the server
- Log information about the request and response

Web server operation procedure

1. Receive HTTP request via TCP
2. Map host header to specific virtual host (one of many host names sharing an IP address)
3. Map requested URL to specific resource associated with the virtual host
   1. File: Return file in HTTP response
   2. Program: Run program and return output in HTTP response
4. Map type of resource to appropriate MIME type and use to set Content-Type header in HTTP response
5. Log information about the request and response in a plain-text file

Virtual Host

A server nowadays can support more than one site on a single computer. For instance, www.dcs.a_small_univ.ac.uk and www.elec.a_small_univ.ac.uk can reside on the same machine and share the same IP address

Virtual Host

- Is this the case in Queen Mary? telnet/nslookup the following addresses to find out:
  - http://german.sllf.qmul.ac.uk/
  - http://french.sllf.qmul.ac.uk/
  - www.eecs.qmul.ac.uk

Web server structure

- A web server is a computer with a server software installed in one place and hosted information stored in another place
- Web servers have two main directories:
  - Document root (servable documents)
  - Server root (server system software)
- Document root is accessed indirectly by clients
  - Its actual location is set by the server configuration file
  - Requests are mapped to the actual location

Quiz

`http://www.example.org:12345/a/b/c.html#para5`

If document root is /user/tony/websever/doc/, where the requested file is located?

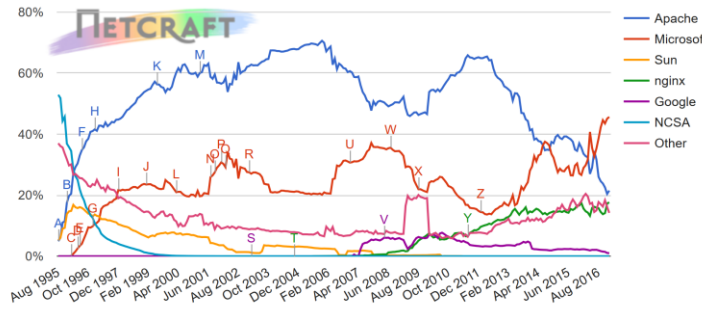
Web server structure

- Many servers allow part of the servable documents to be stored outside the directory at the document root.
- These alternative locations are called *virtual document trees*
- To support this, the server needs to be configured accordingly

Types of web servers

- httpd: UIUC, primary web server, discontinued in mid-90s
- Apache: “A patchy” version of httpd, popular on Linux platforms
- IIS: Microsoft Internet Information Server (only works on Windows platform)
- Nginx (pronounced "engine-x") is an open source reverse proxy server for HTTP, HTTPS, SMTP, POP3, and IMAP protocols, as well as a web server, on multiple platforms (including Windows)

Types of web servers



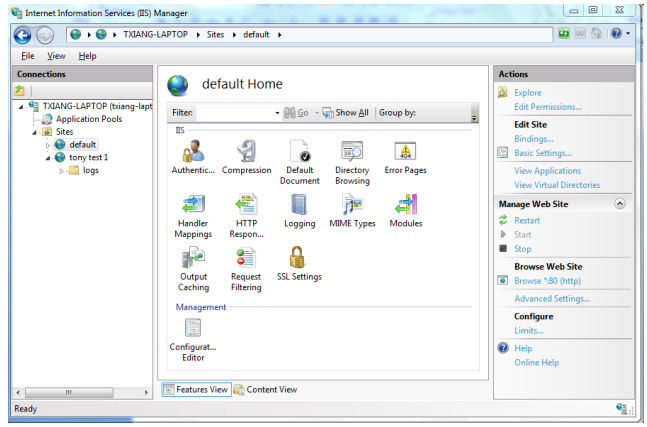
Developer	December 2016	Percent	January 2017	Percent	Change
Microsoft	783,790,492	45.07%	821,905,283	45.66%	0.59
Apache	354,949,196	20.41%	387,211,503	21.51%	1.10
nginx	300,839,507	17.30%	317,398,317	17.63%	0.33
Google	18,602,544	1.07%	17,933,762	1.00%	-0.07

  Source: netcraft.com

Outline

- Introduction to web client operation
- Hypertext Transfer Protocol (HTTP)
- Introduction to web server operation
- **How to set up a server**
- How to use the webprojects server

Set up a server using IIS

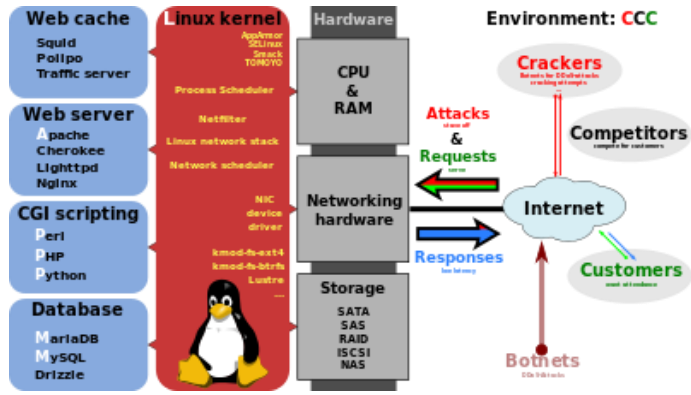


- See http://www.iis.net/ for details on how to install, setup and manage a IIS server
- http://learn.iis.net/page.aspx/28/installing-iis-on-windows-vista-and-windows-7/
- http://support.microsoft.com/kb/323972
- See a link on the course website for instruction on setting up Apache for Mac OS X

Alternative servers

- Setting up Apache on Windows:
  - https://httpd.apache.org/docs/2.2/platform/windows.html
  - Apache, PHP, MySQL package: http://www.wampserver.com/
- Setting up for Mac OS X
  - http://jason.pureconcepts.net/2014/11/install-apache-php-mysql-mac-os-x-yosemite/

Software bundle



LAMP (or WAPM/WIMP)

Outline

- Introduction to web client operation
- Hypertext Transfer Protocol (HTTP)
- Introduction to web server operation
- How to set up a server
- **How to use the webprojects server**

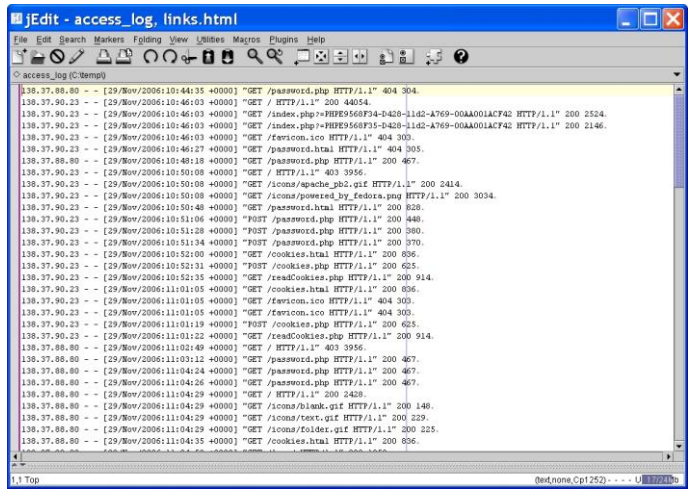
Web server log files

- Web server logs record information about server activities
  - Access log: record every HTTP request processed by the server
  - Message logs: containing a variety of debugging and other information generated by web applications as well as the web server itself
  - Error logs: error streams written by the web server or applications

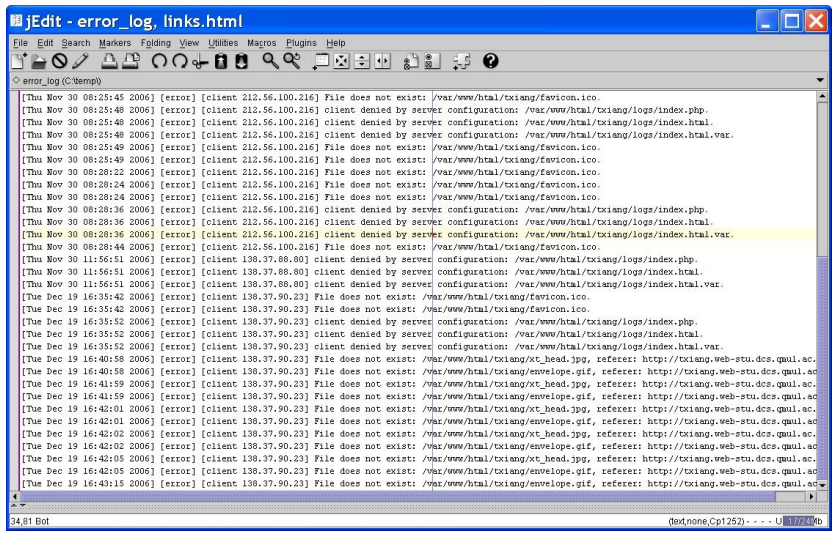
Access log

- Access log often contains the following information
  - Host name/IP address of client machine making the request
  - User name used to log in, if server password protection is enabled
  - Date and time of response, plus the time zone (offset from GMT) of the time
  - Start line of HTTP request (quoted)
  - HTTP status code of response
  - Number of bytes sent in body of response

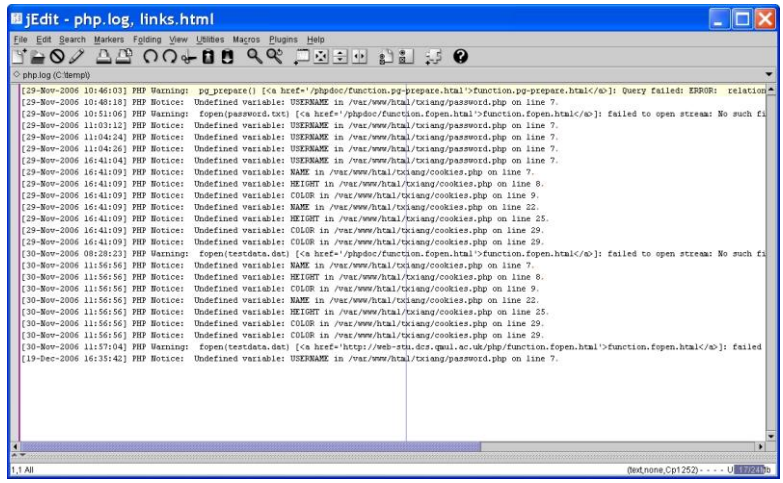
Access log



Error log



PHP log



Using webprojects.eecs.qmul.ac.uk

- Your server is available at:
  - http://webprojects.eecs.qmul.ac.uk/your_user_name/
  - More info (need VPN at home): https://webprojects.eecs.qmul.ac.uk/info/
- File accessible from both ITL and home (you need to install the School open VPN at home)
  - http://support.eecs.qmul.ac.uk/computing-facilities/
- To manage your files on webprojects, see https://webprojects.eecs.qmul.ac.uk/info/webdav.php:
  - Linux:
    - command line = cadaver
    - GUI = konqueror
  - Apple Mac
    - Native support
    - Cyberduck
  - Windows
    - BitKinex (http://www.bitkinex.com/)
    - Cyberduck - recommended
- PHP supported

Access webprojects from home

- You must have EECS VPN installed to start with
- Download CyberDuck from https://cyberduck.io/?l=en
- Open CyberDuck and launch the open connection window,
- Connection method: select "WebDav (HTTP/SSL)"
- Server: webprojects.eecs.qmul.ac.uk
- Untick "Anonymous login"
- Give your EECS login and password
- under "More options", give your EECS user name/login as the path.
- Now the URL would be https://your_login@webprojects.eecs.qmul.ac.uk:433/your_login
- Now connect and see the directory listing of your account/folder on webprojects
- You can now upload and download files

Using webprojects.eecs.qmul.ac.uk

- This server is **self-managed**
- For a comprehensive instruction on how to use the server, check this link (at home, need VPN!):

https://webprojects.eecs.qmul.ac.uk/info/

What you have learned today

- Introduction to web server operation
- Uniform Resource Locators (URL):
- Multipurpose Internet Mail Extensions (MIME)
- Hypertext Transfer Protocol (HTTP)
- Web server log files
- Using the EECS student web server

By the end of the week

- Once you have done the reading and the exercises, you should be able to:
  - Explain the web server functionalities, structure, and operation procedure
  - Explain how the HTTP protocol works
  - Extract information from web server log files
  - Setup your own server at home
  - Put your XHTML files on the webprojects student web server and make it accessible to everyone

Web Resources

- http://www.w3.org/Addressing/URL/
- http://news.netcraft.com/archives/web_server_survey.html
- http://tomcat.apache.org/

