

COS10011

Creating Web Applications

Lecture 1

Unit Overview

Introduction to the Web



Introductions: Teaching Staff



Jeffery Sijore(Unit Convener & Lecturer)

jsijore@swinburne.edu.my

Office: G504, Building G

Outline



■ Unit Overview

■ Introduction to the Web

- ☐ The Internet and the Web
- ☐ IP addresses and Domain names
- ☐ The Internet Stack
- ☐ Client requests and Server responses
- ☐ HTTP
- ☐ HTTPS

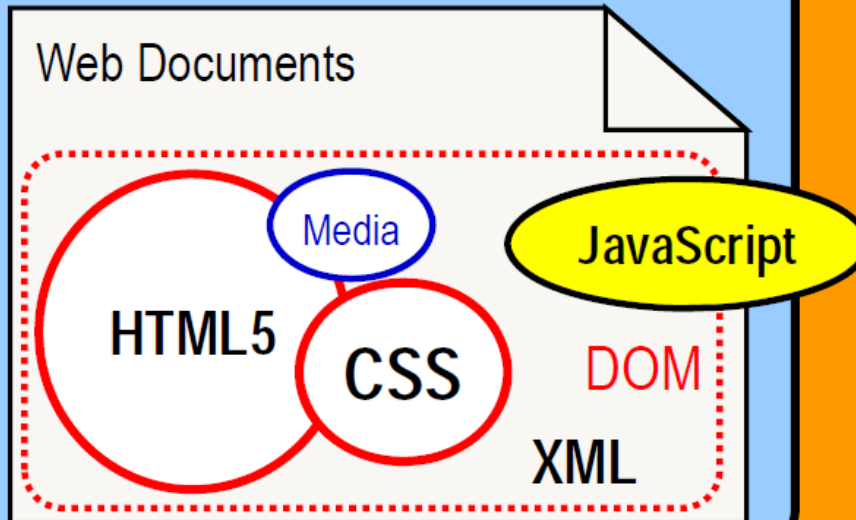
Unit of Study Outline



Internet Technologies: TCP/IP, URLs, URIs, DNS, MIME, SSL

Web Technologies: HTTP, HTTPS, Web Architectural Principles

Client Side Technologies:
Web Applications, Markup Languages



Server Side Technologies:
PHP, SSL, ...
Server-Side Data:
MySQL

Standards
Quality Assurance
Accessibility
Usability
Security

Web Languages you will learn (or revise)



- HTML5
- XML
- CSS
- JavaScript
- DOM
- PHP
- MySQL
- ...

Web Languages you will learn (or revise)



■ Tips

- ☐ **Be proactive learners.** Take advantage of online tutorials.
- ☐ Learn a few core features of each language
- ☐ You won't be an expert in any of the above languages (unless you already are)
- ☐ Emphasis on what the languages do, how they are used and how they fit together
- ☐ You will learn how to create simple applications by using and extending standard-based '**templates**'



Templates - Language

- Standards compliant
- A form of 'best practice'
 - ☐ Well-formed code that can be read by any standards compliant browser
 - ☐ Embody accessibility principles
 - ☐ Based on separation of content from presentation
 - ☐ Avoid deprecated language
 - ☐ Properly documented



Design ‘templates’

- Information layout design and flow within a page
 - ☐ Context (“where am I?”)
 - ☐ Navigation (“where can I go?”)
 - ☐ Usability (“Can others use it easily?”)
- Information design between web pages
 - ☐ Web site layouts

Extending templates



- You should be able to *pass* the assignments by adding content and a bit of logic to the templates, integrating and deploying them
- **HOWEVER**
 - ☐ We encourage you to go beyond the templates with other language features.
- **BUT** – language extensions **MUST**
 - ☐ Be standards-compliant, if included in your web page,
 - ☐ Acknowledge the author/source of code in comments, if you haven't written it yourself
 - ☐ Run on the prescribed version of Firefox
 - ☐ Must run on the standard setup of **local Apache Server**



Using tools

■ Tools we will use

- ☐ Validators
- ☐ Editor (Notepad ++)
- ☐ Browser plugins – e.g DOM inspectors, validators
- ☐ Did I mention validators?

■ Tools we *won't* use

- ☐ Web development frameworks e.g. Dreamweaver
- ☐ Mobile app development frameworks
- ☐ Content management systems
- ☐ Server-side Web application frameworks e.g ASP.NET, JSP, Ruby on Rails, etc...
- ☐ Javascript Frameworks / Libraries e.g jQuery, Angular JS, MooTools,....

Tutorials



■ Very important

- ☐ Most of the coursework assessments will be evaluated during tutorial sessions.
- ☐ Weekly online quizzes and checking of completion of lab work. Quizzes are only available at a given time to those students officially enrolled in the tutorial.

OR

- ☐ *Demonstration* of your Assignments. Assignments are hurdles. You must demonstrate your assignment to have it assessed.

■ Stick to your own tutorial session.

Online Quizzes



- Periodically (Week 5 and 11)
- *Aim* – refresh the concepts covered in the lecture of the previous week or two.
- Largely multiple-choice (good practice for the exam)
- Closed book. No computing or hard copy resources allowed.
- Short - usually 10-15 minutes. The Blackboard system will not record your answers after that.
- Available **only** for the first *20 minutes* of your **allocated** tutorial (no exceptions). The link will disappear after that. You can *only* do the quiz in your allocated tutorial.

Checking of lab exercises



- **Aim** – early diagnosis of any difficulties you are having with the practical tasks (before you need to use these skills in your assignments)
- Certain key Lab exercises are marked with an asterisk *. When you complete these show them to you tutor *in your tutorial*
- **It is a good idea to do the lab exercises *prior to* you tutorial session.**
 - **More time to discuss problems with your tutor You can leave as soon as the work is checked off**
- These exercises count towards your final assessment (10%) and need to be **completed** by the **end of the lab** in which they are assigned.

Group Assignments



- Three (3) Assignments – or more accurately *3 assessable incremental instalments* for developing the same Web site.
- **Assignment 1** (due Week 6) you will demonstrate your ability to develop and deploy on a server well-structured, linked Web pages with text, graphics, and tables, that are styled with CSS.
- **Assignment 2** (due Week 9) further extends the use of HTML elements and makes use of JavaScript to validate forms and add dynamic behaviour to the Web pages.
- **Assignment 3** (due Week 12) you will submit and retrieve data from a server using PHP and MySQL.

Assignments - assessing



- Assignments submitted via Blackboard and assessed by demonstrating your work in specific tutorial sessions.
- Assignments are assessed using a range of criteria:
 - ☐ Your assignment *must* fulfill all **essential requirements** (e.g. validation) in order to pass the assignment.
 - ☐ Up to 75% available for meeting the **specified requirements** (less any deductions)
 - ☐ > 75% can be obtained by enhancing your assignment with **significant extra techniques/features** not covered in the tutorials
- Meeting the *essential requirements* of Assignments 1 and 2 are **hurdles**. As the development task is incremental you need to fix any errors in an assignment **before** submitting the next one.
- Your tutor will mark off the fixes you make. Fixing the errors will mean you are eligible to have the next assignment assessed – but it will not alter the grade you received for your original submission.

Outline



■ Unit Overview

■ Introduction to the Web

- ☐ The Internet and the Web
- ☐ IP addresses and Domain names
- ☐ The Internet Stack
- ☐ Client requests and Server responses
- ☐ HTTP
- ☐ HTTPS

Internet versus the Web



What's the difference?





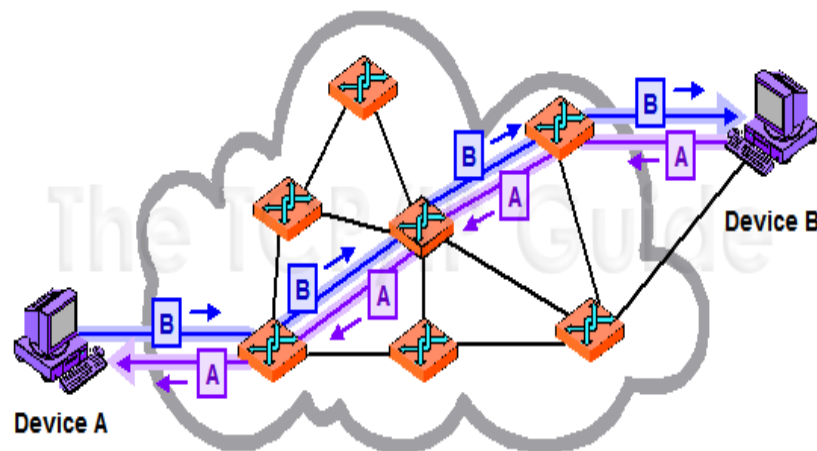
Internet – What is it?

- It is a large group of interconnected networks
 - The networks consist of **servers**, **routers**, computers, **mobile phones** and a range of other devices.
 - Computer needs to understand how to talk to each other (**“protocols”**)
 - Uses **Packet Switching**
 - Protocols have different functions e.g.
 - **Internet Protocol (IP)** routes packets to destination
 - **TCP** breaks messages up into packets and reassembles them
 - Many application protocols FTP, HTTP, SMTP, etc run over it
 - It is highly fault tolerant
 - An interesting video that explains the Internet:
<https://goo.gl/ZeKF0H>

Circuit switching

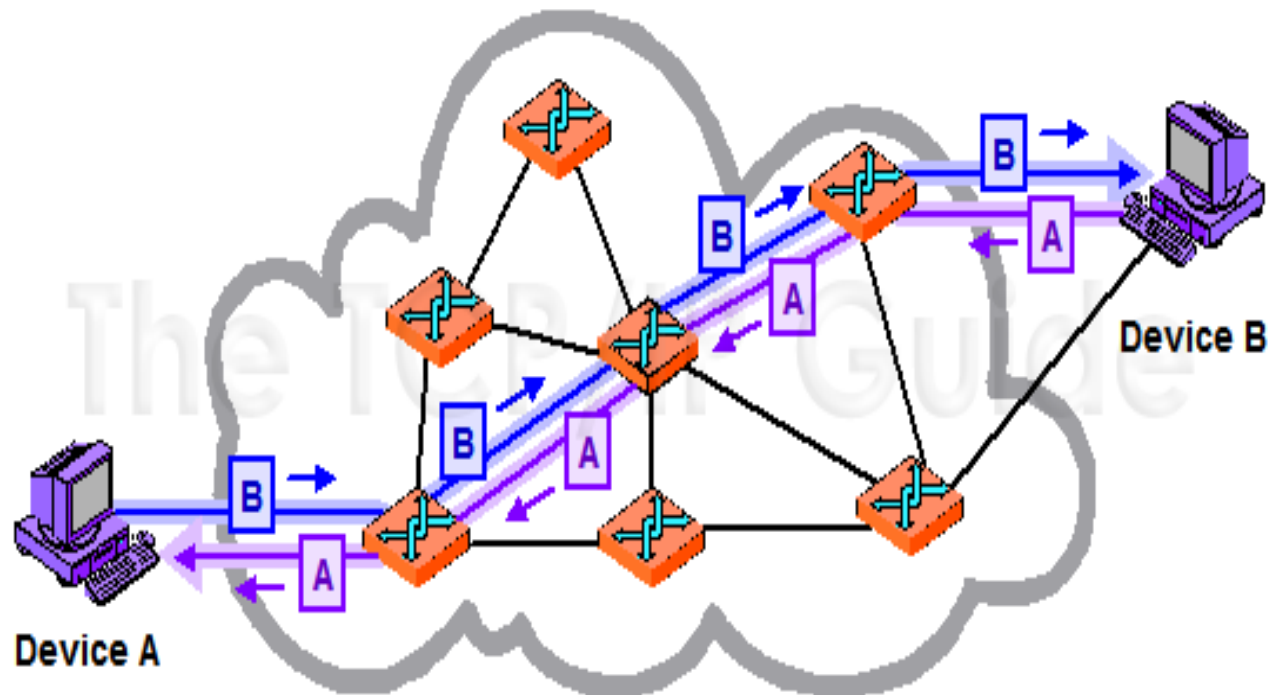


We create a dedicated circuit to connect two devices



- Great for telephone networks because it is reliable and you only have a few thousands people talking to each other (at any given point in time)
- Unfortunately, not efficient when you have 200 million computers that
- Need to be always connected

Circuit switching

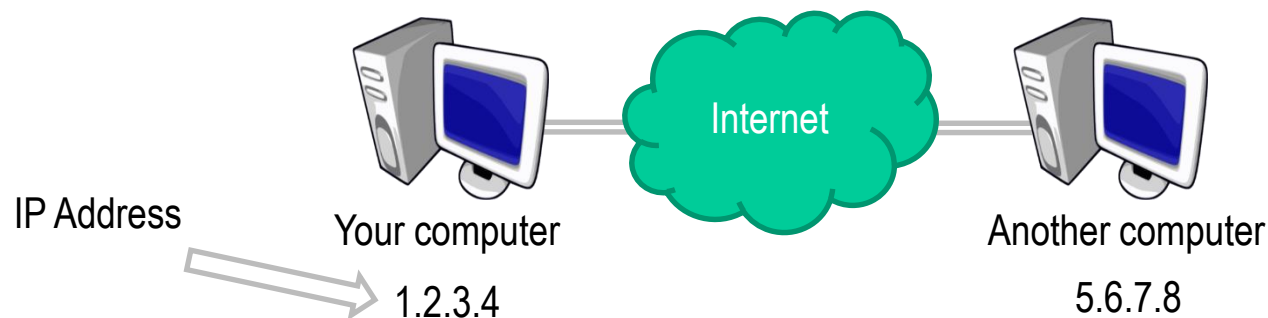


- We take a block of information and then break it up into small packets
- Each packet is then sent off to the destination device
- The packets are joined back at the destination
- Highly scalable and makes efficient use of resources

Internet – How does it work?



- Every node (can be a single computer or a network) is assigned a unique IP address
- The nodes are connected to each other using switches and routers
- Path is not fixed but determined by routers at runtime
 - Each message (or part of a message) may take a different route through the networks.
 - E.g. <http://visualroute.visualware.com>



Outline



■ Unit Overview

■ Introduction to the Web

- ☐ The Internet and the Web
- ☐ IP addresses and Domain names
- ☐ The Internet Stack
- ☐ Client requests and Server responses
- ☐ HTTP
- ☐ HTTPS



IP Addresses

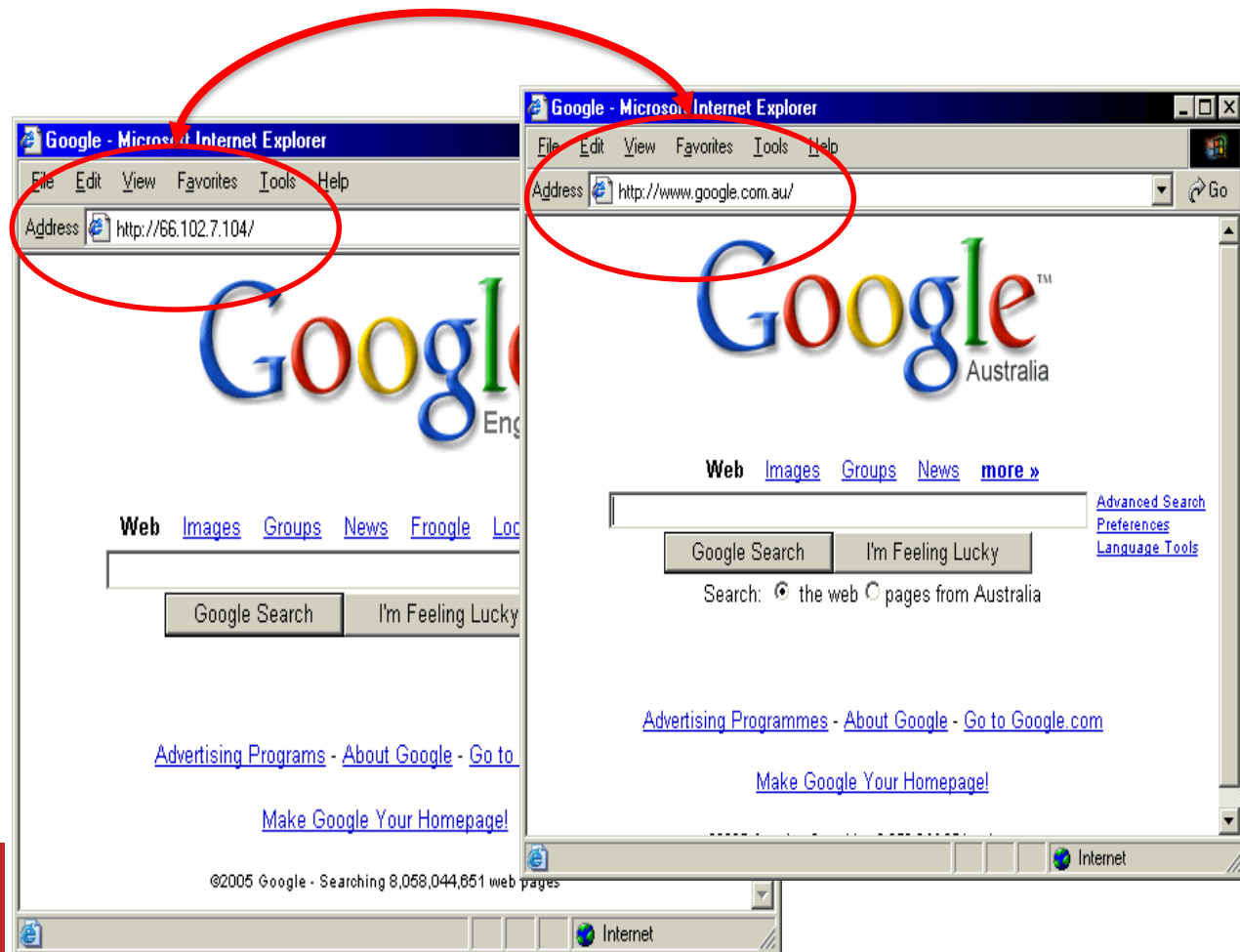
- Current IP addresses **IPv4** are *32 bit integers* (4 bytes),
 - e.g. 186.126.2.1 (2^{32} = *about 4 billion addresses*)
- Addresses are divided into hierarchical blocks:
 - Class A ~16 million ---.xxx.xxx.xxx
 - Class B ~65,000 ---.---.xxx.xxx
 - Class C 255 ---.---.---.xxx
- New **IPv6** uses *128 bit addresses* (16 bytes) & offers additional features (2^{128} *that is about 340 trillion, trillion, trillion addresses*)
 - e.g. 2001:db8:1f70:999:de8:7648:6e8
- Differences between IPv4 and IPv6, and FAQ:

http://www.isoc.org/internet/issues/ipv6_faq.shtml

Domain name system



■ Name or Number!

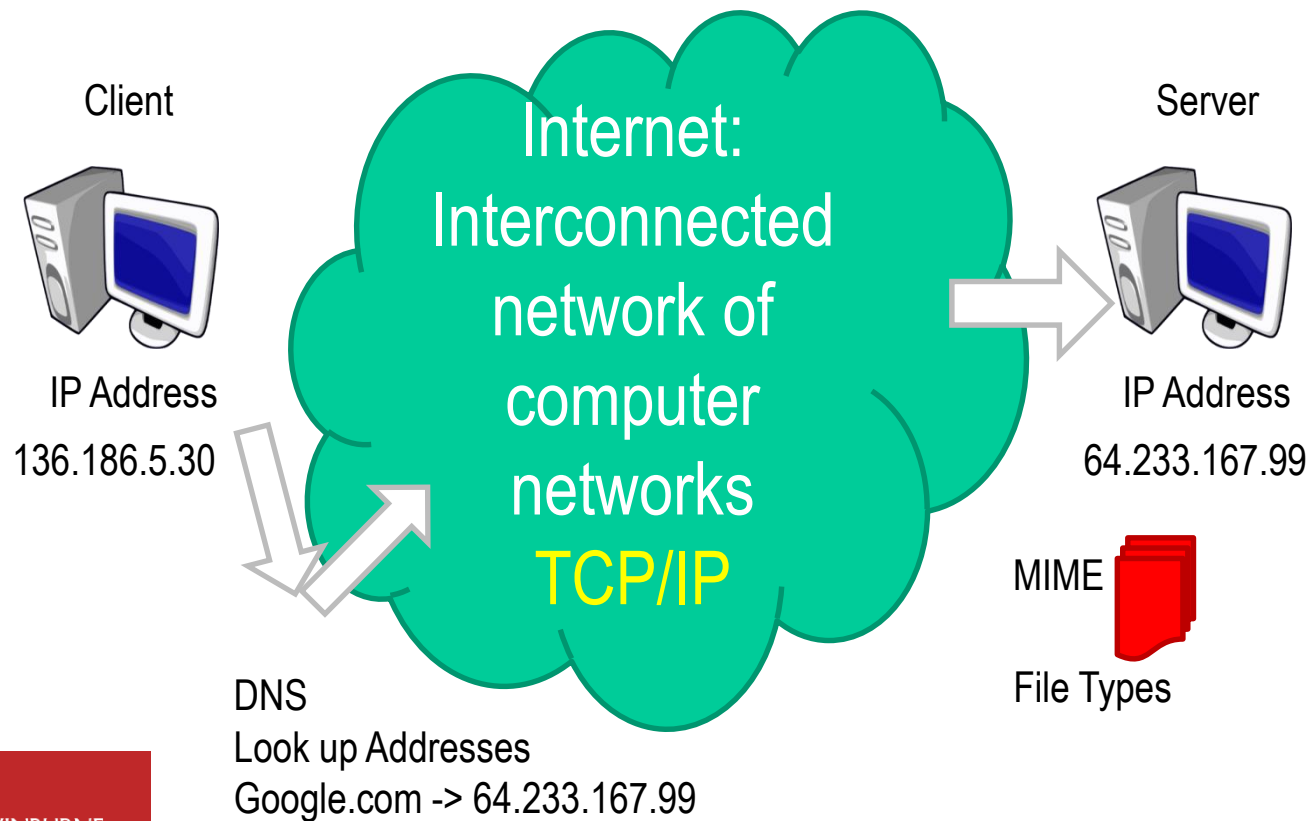


Domain name system



- **Domain Name System** (DNS) creates a “human interface” that allows **symbolic domain names** to be mapped to numerical **IP Addresses** of hosts across the Internet.
 - ☐ The DNS is implemented as a **distributed** database.
 - ☐ DNS is one of the critical operational elements of the Internet
 - ☐ There are many **Domain Name System Servers** to provide the DNS.
- Example:
 - ☐ Type in an “easy to remember” name
www.google.com.au
 - ☐ This name is sent to the local DNS and it looks up IP Address
66.102.7.104

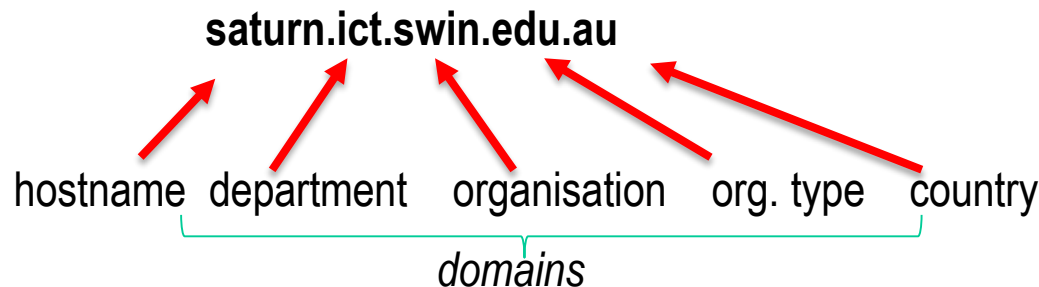
How does it know where to find google?



Domain name system



- In the DNS, each machine (“host”) has a unique name, or “**hostname**”, used to identify it.
 - Some machines have more than one name...
 - A simple “hostname” is a single word like “saturn” or “www” etc.
 - An internal DNS server will provide the IP from these “suffix” details
- We use a long “**fully qualified domain name**” (**FQDN**) in URLs that includes “domains” that a host belongs to.
- A **FQDN** consists of multiple parts separated by periods.



URLs



- A **Uniform Resource Locator (URL)** is a naming technique for **requests**. It can specify the **protocol**, **host**, **port** (optional) and **location** of an Internet resource.

<http://www.it.swin.edu.au:80/staff/webdev/>

- There are different kinds of URLs for each protocol:

Local file: `file:///usr/local/pics/face.gif`

HTTP: `http://www.it.swin.edu.au/subjects/mm`

FTP: `ftp://ftp.it.swin.edu.au/pub/test.doc`

Telnet: `telnet://a.remote.host/`

SMTP : `mailto:jblow@it.swin.edu.au`

- A URL is actually a special case of the Uniform Resource Identifier (**URI**) scheme.
- References:

□ <http://www.iana.org>

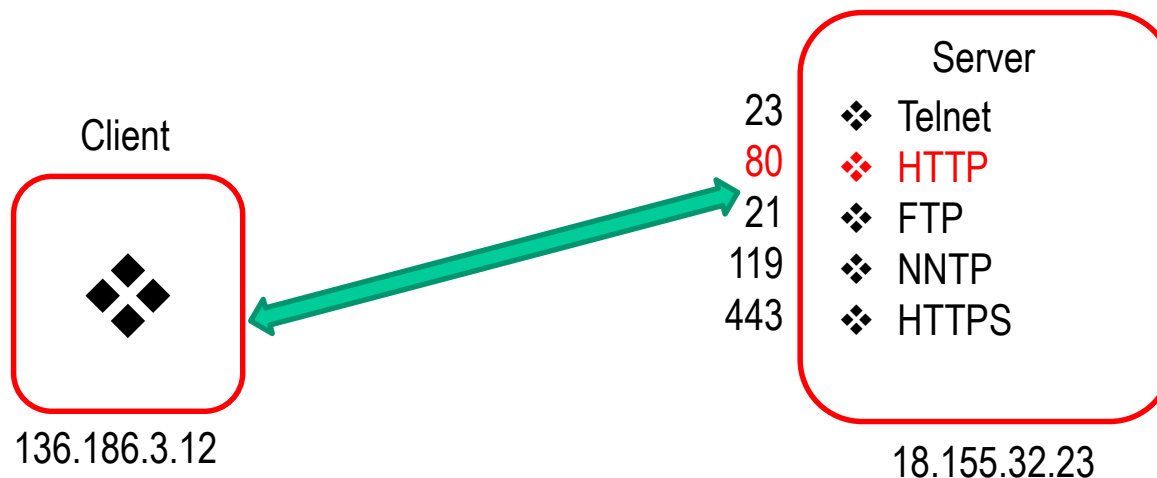
(then /protocols search for MIME)

□ <http://www.w3.org/Addressing/>



Ports

- IP addresses identify machines, but **ports** identify a particular program (server) running on the machine.
 - Ports are a number from 0 to 65,535.
 - Ports from 0 to 1024 are called “reserved” (mainly for historical reasons).
- A single machine may run several different types of servers:
 - *e.g. a Telnet service, an FTP service, HTTP, Gopher etc.*
- “Well-known ports” are port numbers used for a specific network **protocol**.



Outline



■ Unit Overview

■ Introduction to the Web

- ☐ The Internet and the Web
- ☐ IP addresses and Domain names
- ☐ The Internet Stack
- ☐ Client requests and Server responses
- ☐ HTTP
- ☐ HTTPS

The Internet Stack – TCP/IP



■ Communication between computers

- ☐ Requires a lot of different functions to be performed, e.g. addressing, creating packets, routing packets, sending packets 'Over the wire', ensuring correct delivery, making applications understand each other, etc.
- ☐ These Function need to be able to be understood and executed by each computers involved in the communication. A **protocol** ensure computer understand each other.
- ☐ These protocols are separated into layers of a '**stack**'

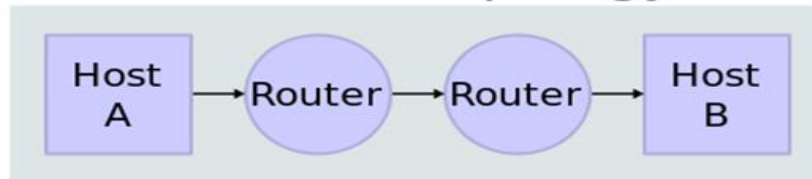
■ The Internet relies on many protocols but 2 in particular

- ☐ Internet Protocol or IP
- ☐ Transmission Control Protocol or TCP

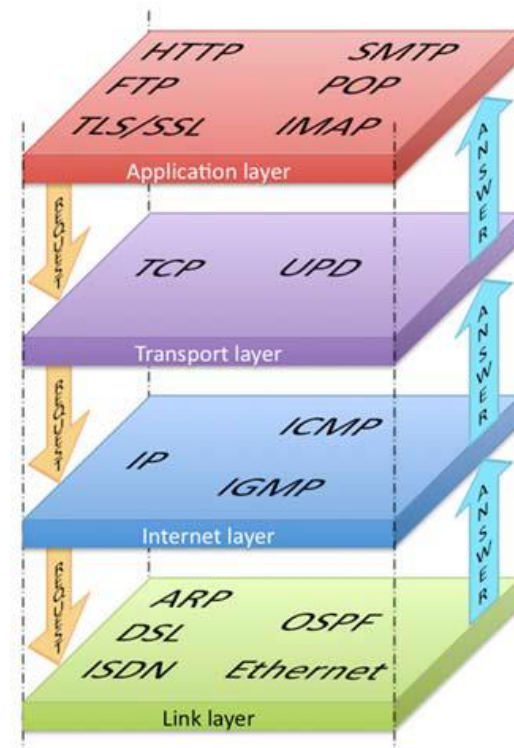
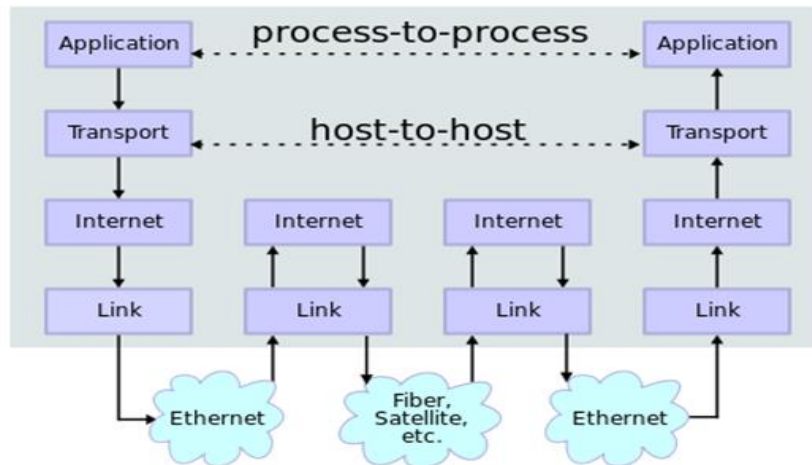
Internet protocol suite



Network Topology



Data Flow



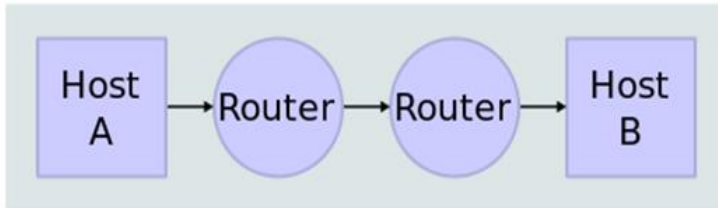
Example protocols

<http://infoacrs.com/nm/ipprotocol.html>

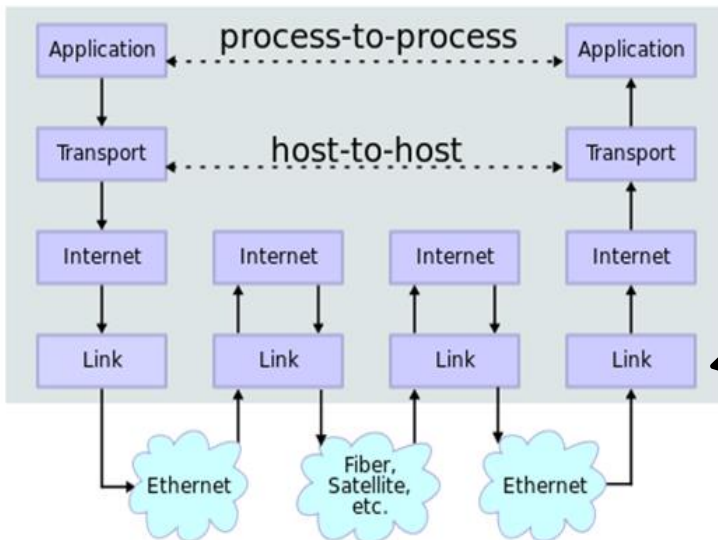
Internet protocol stack – link / physical layers



Network Topology



Data Flow



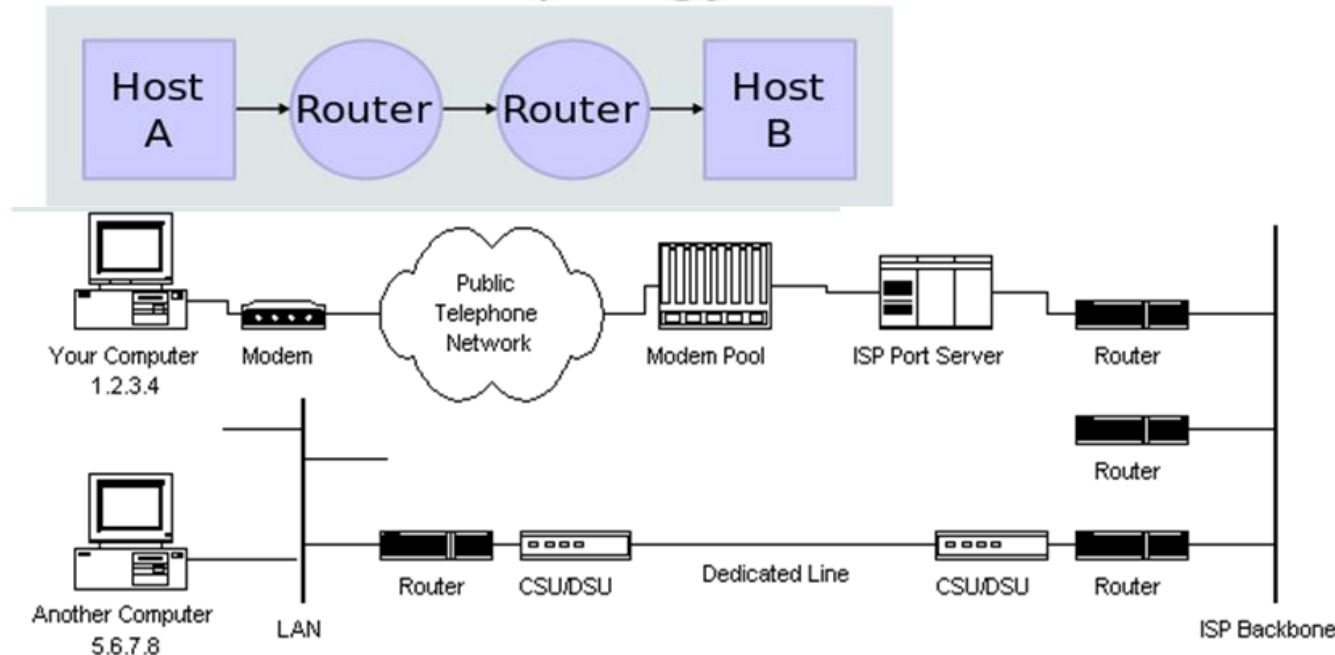
The **link layer** handles the transfer of data between adjacent network nodes in a local area network (LAN) segment or in a wide area network (WAN)
e.g. Ethernet, HDLC

This layer is transparent to application developers, and not the focus of this course.
(Covered in your networking subjects)

Internet protocol stack – link / physical layers



Network Topology

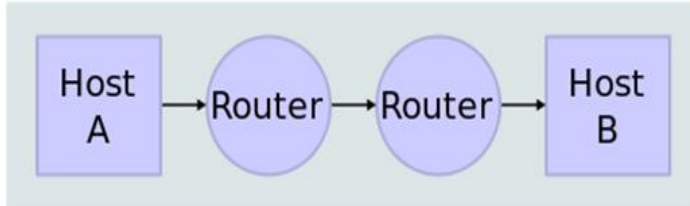


- To get to the destination, we often need to go through a number of different switches, routers and gateways
- Messages also need to travel over multiple networks running different link protocols

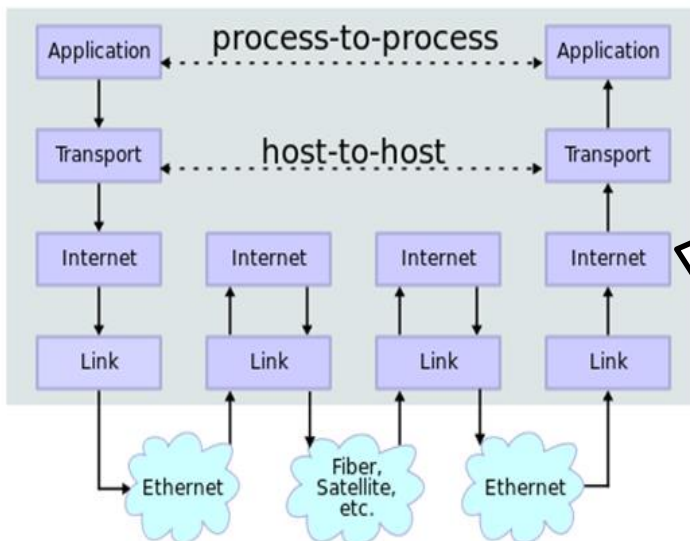
Internet protocol stack – internet (internetwork) layer



Network Topology



Data Flow



The **Internet layer** is responsible for sending packets across multiple networks, **routing** data from the source network to the destination network.

e.g. **IP (Internet Protocol)**

IP performs two basic functions:

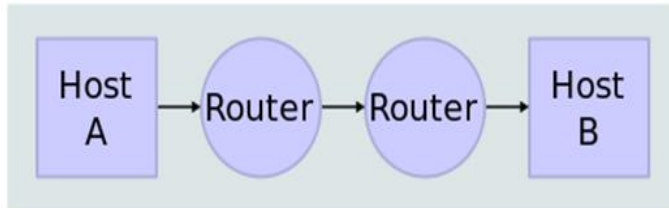
1. Host addressing and identification
2. Packet routing: This is the basic task of sending packets of data (datagrams) from source to destination by forwarding them to the next network router closer to the final destination.

Other internet layer protocols like **ICMP** (Internet Control Message Protocol) are used by network devices (e.g. routers) to send management messages to each other (e.g. error messages)

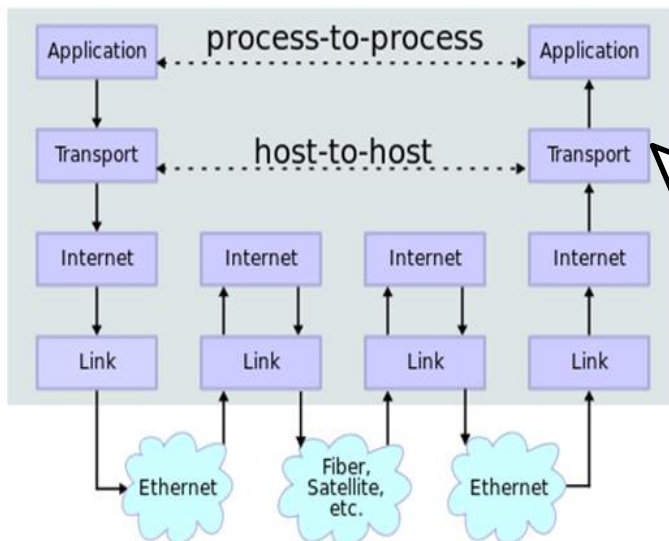
Internet protocol stack – Transport layer



Network Topology



Data Flow



The **Transport layer** is responsible for establishing a host-to-host 'channel' for passing messages

e.g. **TCP (Transmission Control Protocol)** ensures:

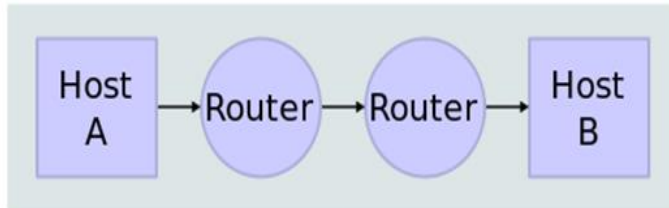
- data arrives in-order
- data has minimal error (i.e. correctness)
- duplicate data is discarded
- lost or discarded packets are resent

Other transportation layer protocols like **UDP** (User Datagram Protocol) provide fast 'unreliable' communication that does not care about lost or out-of-order packets. This is useful for real-time applications like VoIP.

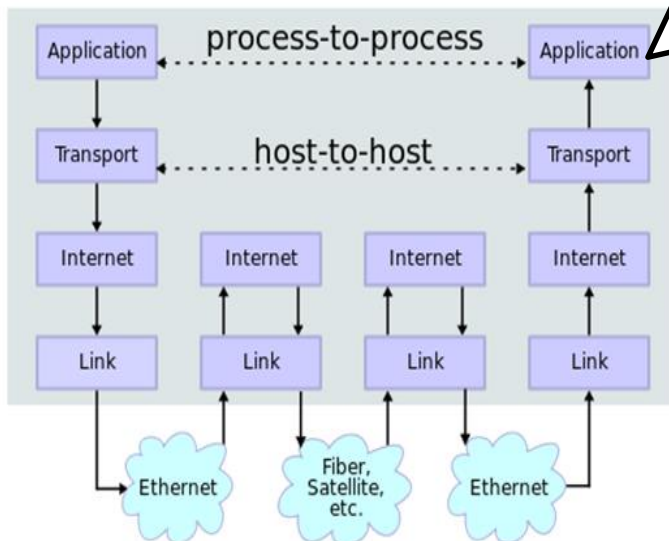
Internet protocol stack – Application layer



Network Topology



Data Flow



The **Application Layer** provides protocols for applications that provide Web pages (**HTTP**), email (POP, SMTP) file transfer (FTP), addressing (DNS), and many more services.

Different transport protocols are better suited to the requirements of different applications protocols. For example file transfer needs to be reliable so it makes sense to use 'reliable' TCP. The WinSCP application you will use in the labs runs FTP and SSH which are all examples application protocols used for file transfer.

Network utilities such as **tracert** (we will play with this in the labs) use UDP running over ICMP

Outline



■ Unit Overview

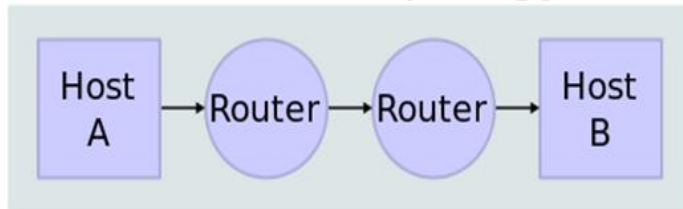
■ Introduction to the Web

- ☐ The Internet and the Web
- ☐ IP addresses and Domain names
- ☐ The Internet Stack
- ☐ Client requests and Server responses
- ☐ HTTP
- ☐ HTTPS

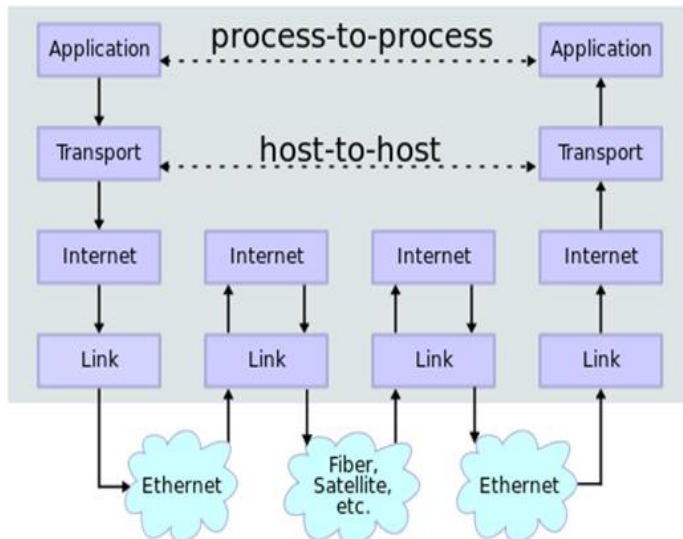
How message data is processed by the stack



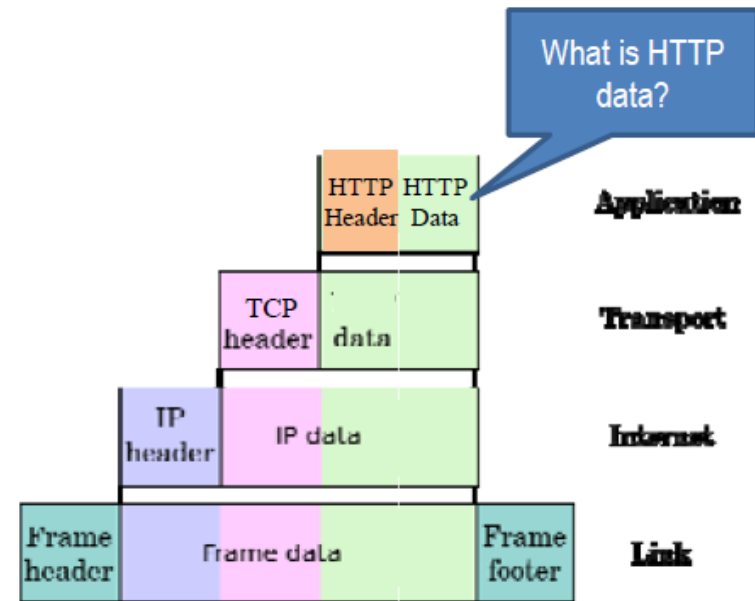
Network Topology



Data Flow



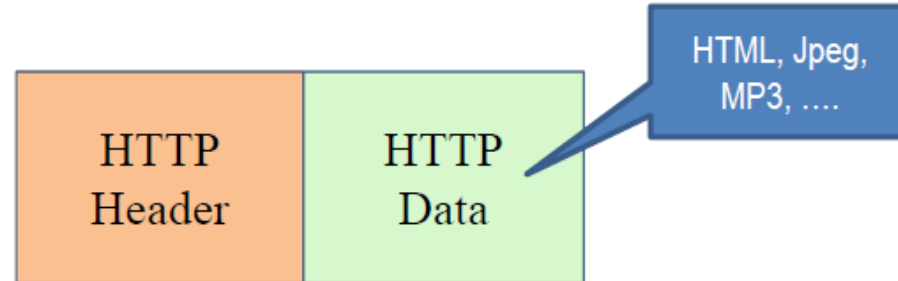
- Each layer encapsulates the message with header information which is then stripped off at the receiving end





The Web and HTTP

- HTTP – Hypertext Transfer Protocol
- Application-level ***client-server*** protocol designed to run on top of TCP/IP



- Originally designed to support HTML payloads but can carry many different media types
 - Hypertext, pictures, audio, movies, ...

The Web – Its History (continued)



- In late 1990 and early 1991, Tim Berners-Lee created the **World Wide Web** at the European Laboratory for Particle Physics (CERN) in Geneva, Switzerland
- The original purpose of the World Wide Web (WWW) was to provide easy access to cross-referenced documents that existed on the CERN computer network
- Hypertext allows you to quickly link to and open other pages.
- **Hypertext Transfer Protocol** (HTTP) enabled HTTP requests / responses over the Internet

The Web – What is it now?



- The Web has evolved into much more than a set of hyperlinked passive documents read by humans → “Web 2.0”
- Basis for re-writing the rule book on almost *everything* based on shared networks of data, services and resources
 - Technically, Economically, Socially, Politically
- Dynamic, Location aware, Mobile, Embedded
- Big Data, Searchable, Programmable, Aware of semantics

The Web – Its Consortium



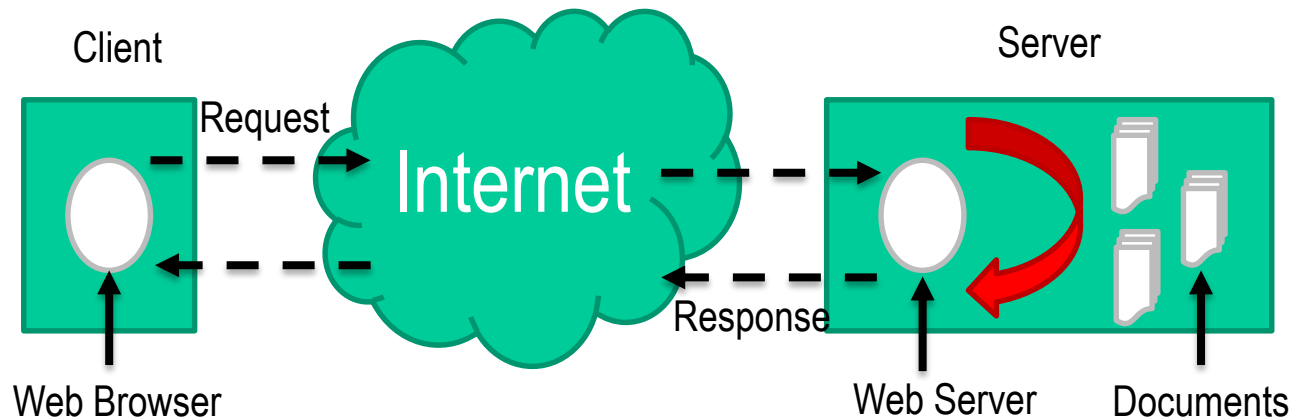
- The **World Wide Web Consortium (W3C)**, is a group of Web developers, programmers, authors, formed in 1994.
 - Purpose of the W3C is to lead, create and recommend **standards** that everyone can use to help bring the web “to its full potential”.
 - The W3C has no enforcement power, however the recommendations of the W3C are usually followed since a uniform approach is in the best interest of everyone.
 - The standards they recommend cover many web areas and include: HTML, CSS, XML, RDF, SVG, SMIL, PNG and more!

See <http://www.w3.org>

HTTP is a Client – Server Protocol



- **HyperText Transfer Protocol** (HTTP) is a fast, *stateless* protocol for the exchange of textual information.



- HTTP uses **Request – Response** model.
 - ☐ Clients **request** a document from the server
 - ☐ The server generates a **response** (usually containing the document).

Clients and Servers



■ Web Browsers (or user agents)

- ☐ Manage and make HTTP requests
- ☐ Receive HTTP responses
- ☐ Interpret and render/display completed Web Documents

■ Web Servers (HTTP Servers)

- ☐ Receive HTTP requests
- ☐ Retrieve Web Documents
- ☐ Manage and make HTTP responses

Web Browsers



- Web Browser software is available for most platforms.
- The appearance of a Web page may differ between browsers.
- Commonly used Web Browsers:



Microsoft **Internet Explorer**



Mozilla **Firefox**



Apple **Safari**



Google **Chrome**

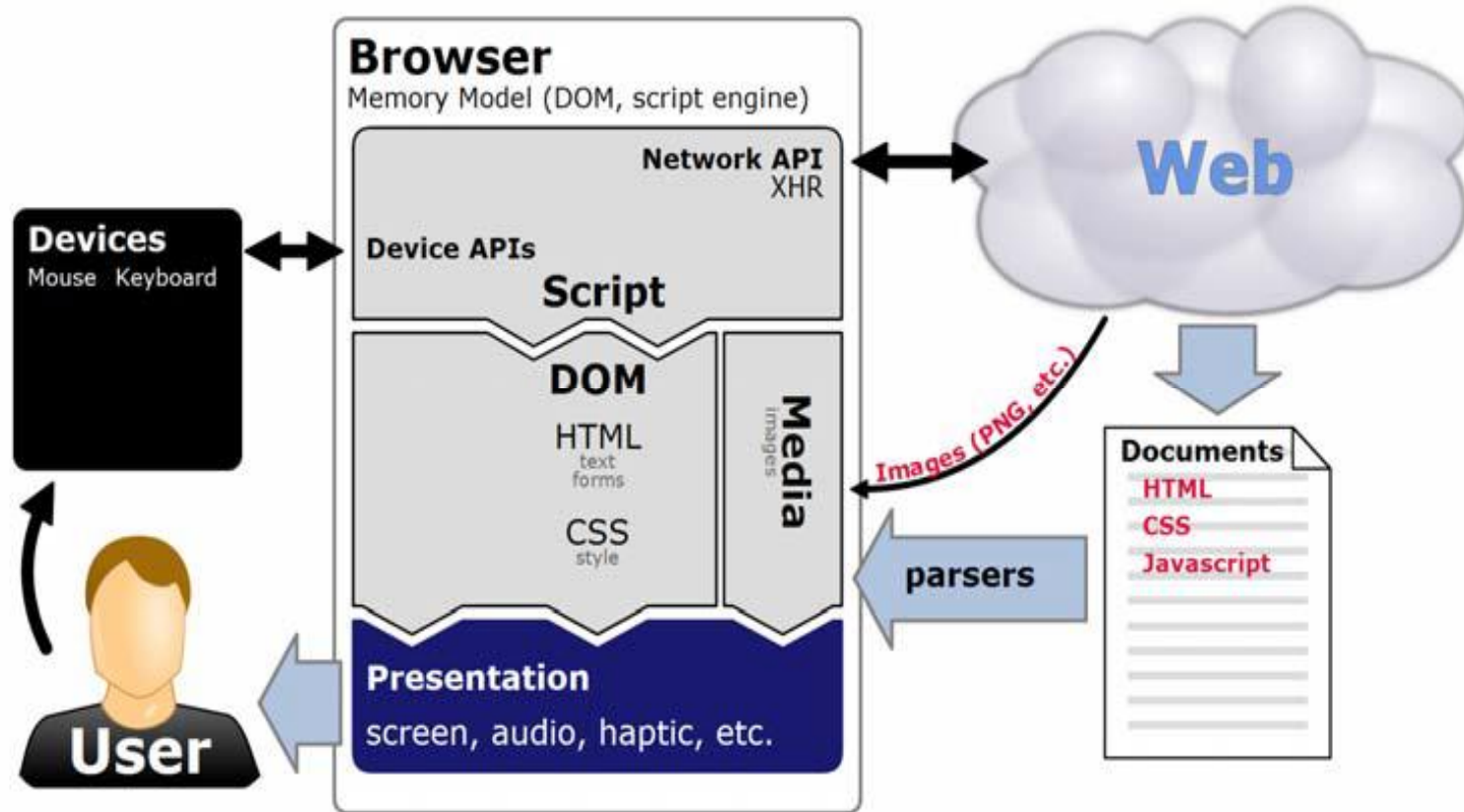


Opera

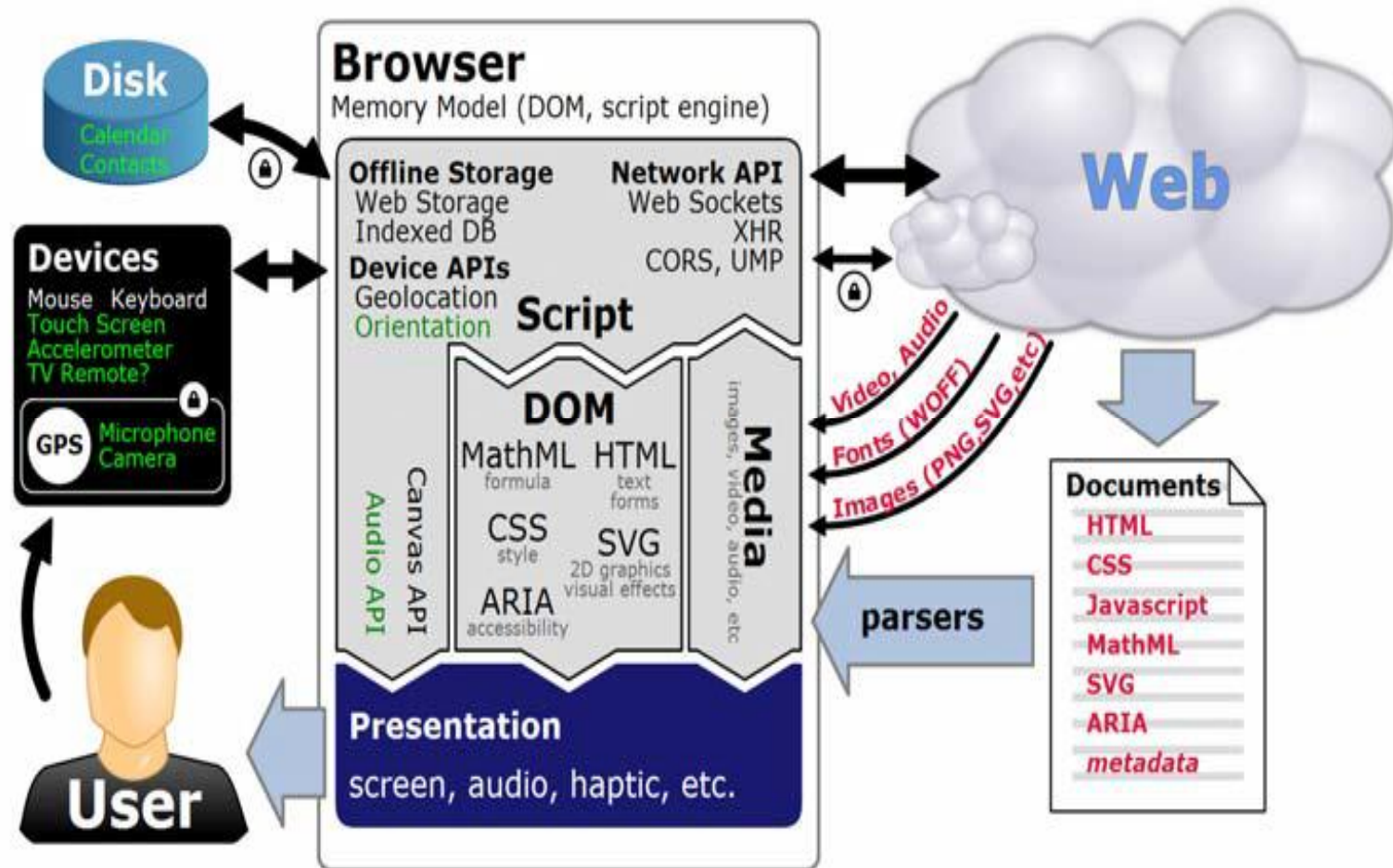
... and many others ...

http://en.wikipedia.org/wiki/List_of_web_browsers

Web Browser Technology 2006



Web Browser Technology 2013

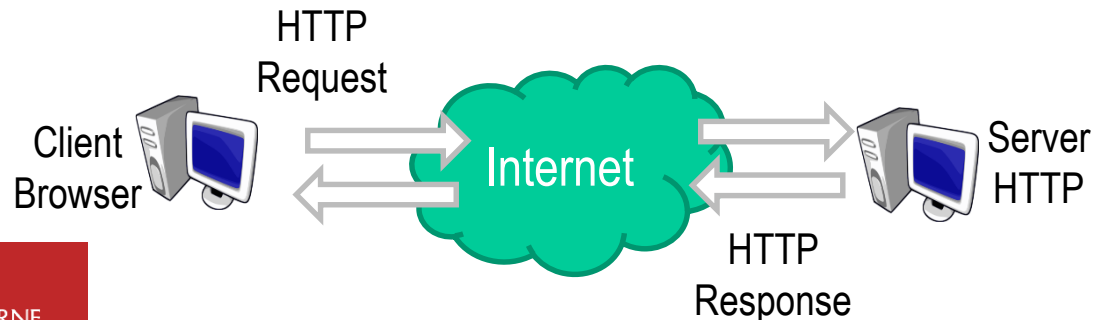


Web Browsers



■ Web browser:

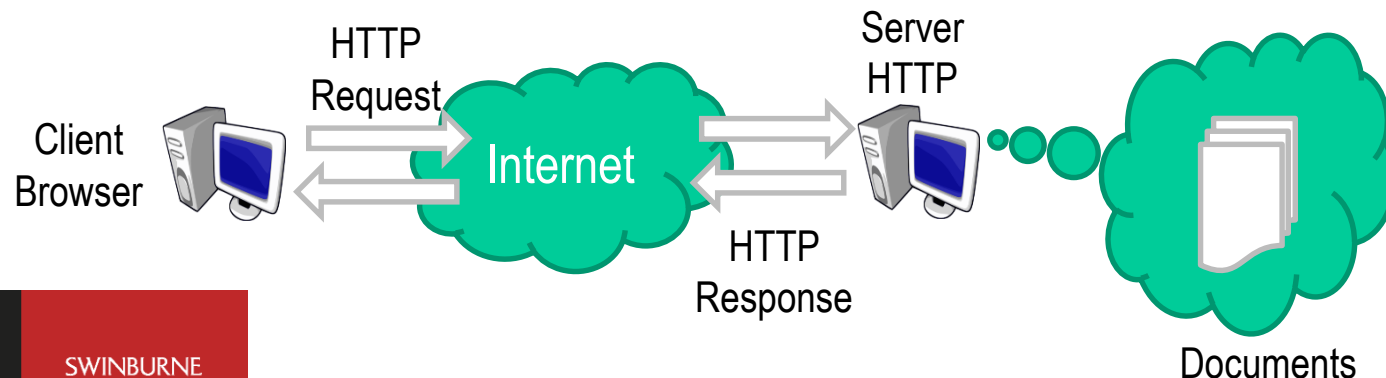
- **Software application** that **lays out** or **renders mark-up**: displaying text, images, and other information typically located in a Web page.
- **Users interact** with the Web browser, *requesting web pages by URL*, clicking on hyperlinks or *submitting forms* within the Web page.
- Web pages are usually located on a **Web Server** on the Internet, but can be located on the local computer, or on a local area network.
- Web browsers format and send HTTP requests, and receive, analyse and layout or render HTTP responses.





Web Server Features

- A Web server is made up of several components:
 - A **computer** with an **Internet connection** and **operating system**.
 - **Web server software** to receive and respond to HTTP requests.
 - **Information**: a collection of documents to be served.
 - Web server software is available for most platforms.
 - The server program usually runs continuously.
 - Handles multiple requests
 - Careful access control to server content should be a feature



Web Servers & Scripting



- Servers can support a ***variety of executable scripts*** so that if a particular URL is requested, the server executes the script and then returns its output to the browser.
 - Examples of this concept:
 - ☐ Built-in interpreters for ***embedded scripting*** – ASP, PHP, Perl, etc
 - ☐ Standard CGI scripts
 - ☐ Server-side includes (SSI)
 - ☐ Database interfaces
 - ☐ Integrated development environments (IDE)
- ... More about CGI, SSI, embedded scripting, later ...*



Most Common Web Servers

■ Apache HTTP Server

- ☐ The 'standard' web server
- ☐ Market Share: **61.5 %** of all websites and **steady** (Netcraft Survey Jul 2012)
- ☐ Free and Open Source, GPL Licence
- ☐ Platform - UNIX, Mac, Windows 95/2000/XP, OS2 etc.

■ MS Internet Information Server

- ☐ Also known as just **IIS**
- ☐ The standard web server for MS Windows platforms
- ☐ Market Share: **14.6 %** of all websites and **steady** (Netcraft Survey Aug2011)
- ☐ Platform - Windows Only (NT / 2000 / XP / Vista)

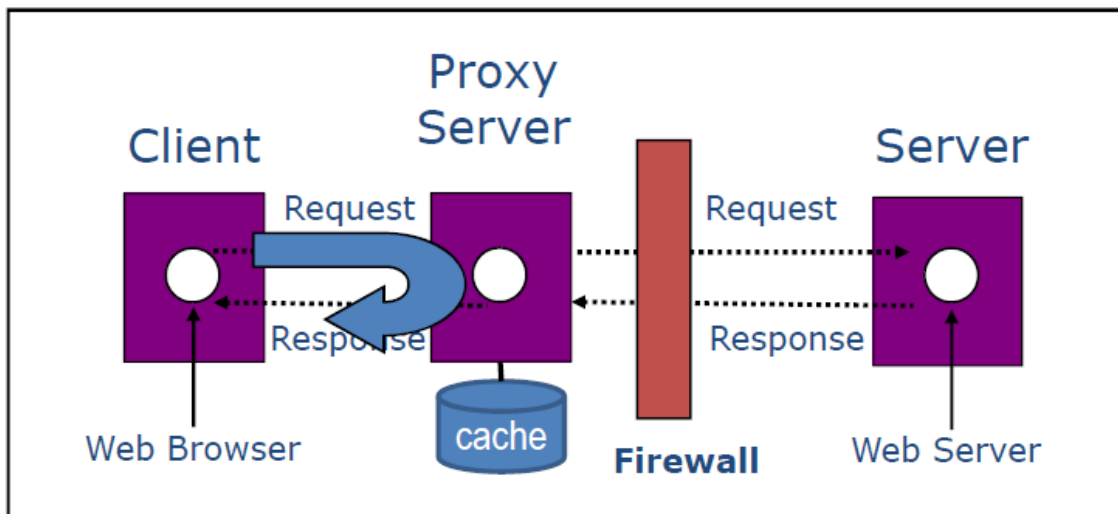
■ Others

- ☐ Nginx ("engineX")
- ☐ Google
- ☐ Lighttpd

Proxy Servers



- A proxy server can sit between a client and a web server.



- Act as a “go between” (protection/“firewall” or management)
- Act as a local “cache” and prevent requests for cached items

Proxy Servers



- A proxy server has a variety of potential purposes, including:
 - ☐ To keep machines behind it anonymous, mainly for **security**
 - ☐ To speed up access to resources (using caching). Web proxies are commonly used to **cache** web pages from a web server
 - ☐ To **log / audit usage**, e.g. to provide company employee Internet usage reporting.
 - ☐ To **scan inbound** content for malware before delivery.
 - ☐ To **scan outbound** content, e.g., for data loss prevention.

Outline



■ Unit Overview

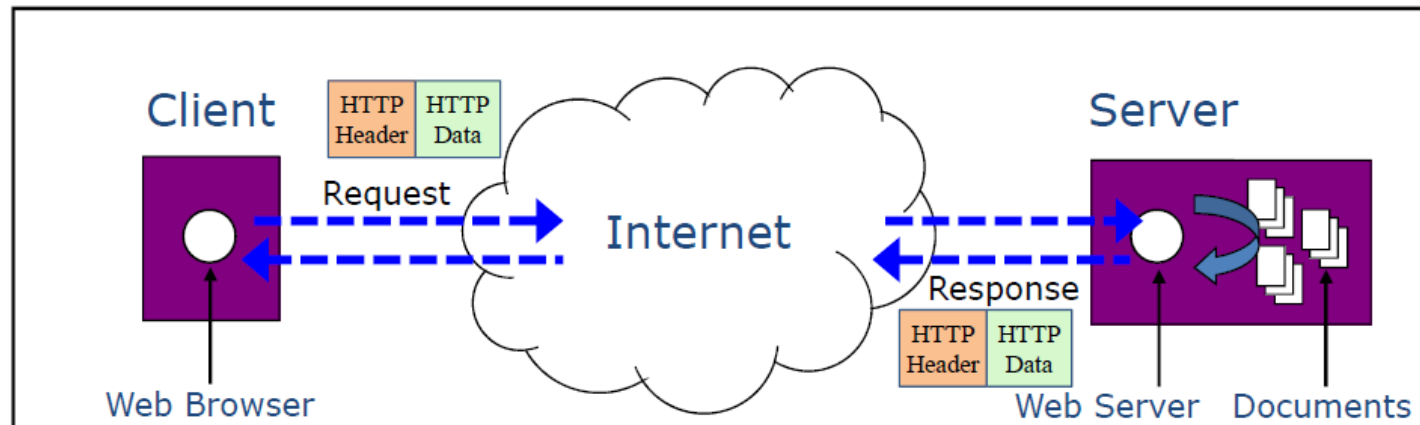
■ Introduction to the Web

- ☐ The Internet and the Web
- ☐ IP addresses and Domain names
- ☐ The Internet Stack
- ☐ Client requests and Server responses
- ☐ HTTP
- ☐ HTTPS

HTTP is a Request and Response Protocol



- Clients **request** a document from the server
- The **server** generates a **response**



- HTTP request and response messages consist of:
 - **Header** a document from the server
 - **Data** (e.g. HTML document).

HTTP data

HTTP
Header

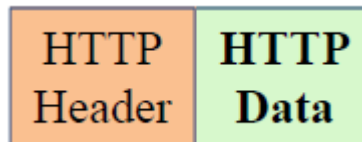
HTTP
Data



- HTTP data is often HTML text but a Web page may consist of files in many data formats.
- **Multipurpose Internet Mail Extensions** (MIME) originally developed to help Internet “mail” by providing a standard way of dealing with files of different types.
- Browsers use the associated **MIME type** information to determine how to handle a document they send or receive:
- **.htm** or **.html** files are associated with the “**text/html**” **MIME type**, and this tells the browser to show them internally.
- Other files such as sound or video files have different **MIME types** and may need to be passed to a “helper application” by the browser.
- *Examples:*

<code>text/html</code>	HTML text
<code>application/zip</code>	file compression format (open with WinZip etc)
<code>audio/x-wav</code>	Microsoft “wave” format (audio)
<code>image/tiff</code>	TIFF image format
<code>video/mpeg</code>	MPEG video format
- Not all HTTP messages contain data – e.g. a GET or HEAD request only have a header

HTTP headers

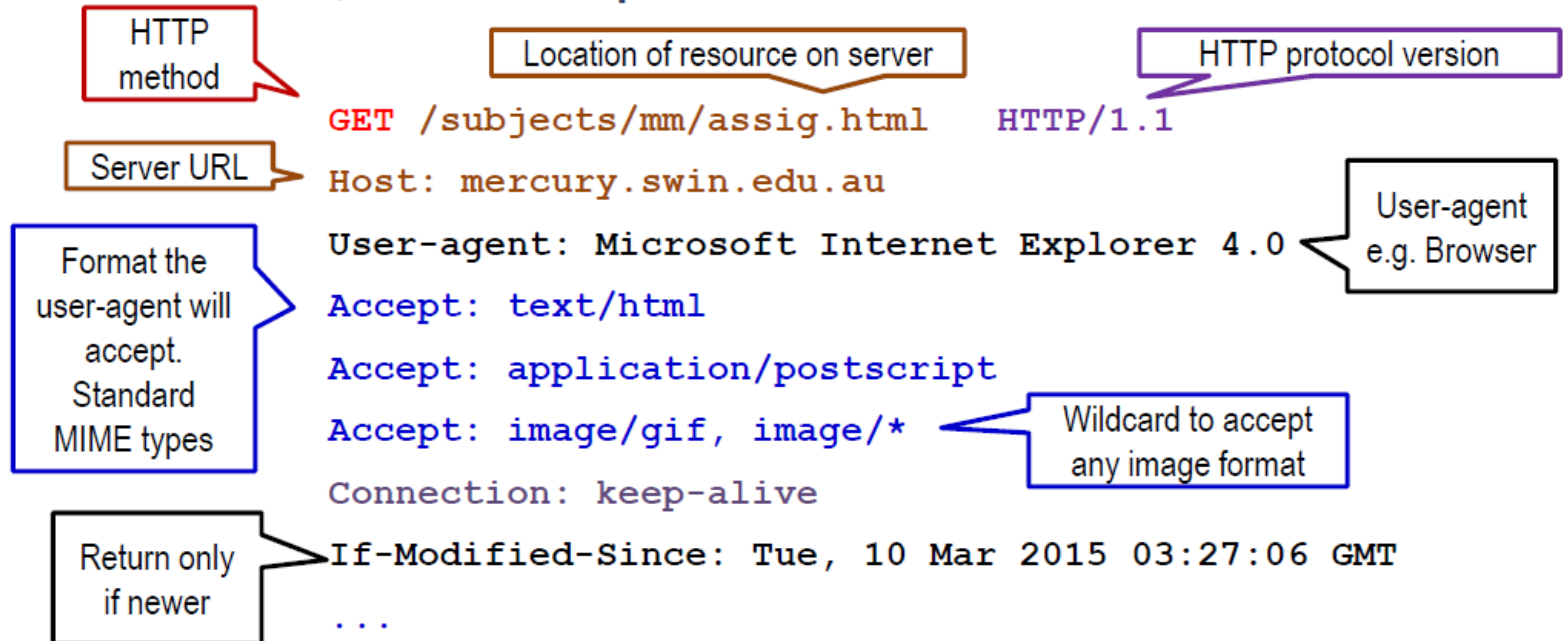


- Request header
- Response header
- Viewing headers
 - ☐ Firefox plugin <http://livehttpheaders.mozdev.org/> allows you to view request and response headers as you browse
 - ☐ Also web site <http://www.rexswain.com/httpview.html>



HTTP Request Header

■ Sample HTTP Request (text sent to the server)



- After the request header and an extra blank line, the client can send extra data (for POST or PUT requests) that is described using the **Content-Length** (bytes long) header.

... Also see later when we discuss GET and POST and using forms.

<http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html>



HTTP methods

- **GET** send the document requested by the URL
- **POST** send some data after the header
- **HEAD** send only the header information of the requested document
- **PUT** replace the contents of the document with attached data
- **DELETE** delete the indicated document

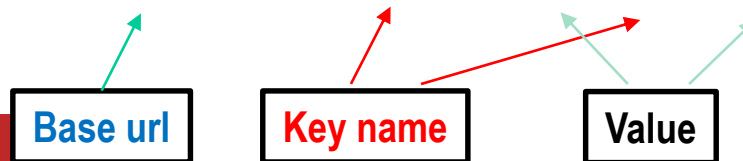
Browsers use **GET** and **POST**



HTTP Get method

- The **GET** request method is designed to retrieve information from the server.
- As part of a GET request, some data can be passed within the URI's **query string**, specifying for example search terms, date ranges, or other information that defines the query.
- Query strings (name / value pairs) is sent to the URL after a ? symbol.

`/test/demo_form.asp?unitname=Internet&code=HIT1307`



HTTP POST method



- It is often used when uploading a file or submitting a completed web form.
- Request that a web server accept the data enclosed in the request message's body for storage

```
POST /test/demo_form.asp HTTP/1.1
Host: swinburne.edu.au
unitname=Internet&code=HIT1307
```

*Default form media type MIME is:
application/x-www-form-urlencoded*

Comparing Get and Post methods



	GET	POST
BACK button/Reload	Harmless	Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)
Bookmarked	Can be bookmarked	Cannot be bookmarked
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data
History	Parameters remain in browser history	Parameters are not saved in browser history
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL Never use GET when sending passwords or other sensitive information!	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
Visibility	Data is visible to everyone in the URL	Data is not displayed in the URL ₂₂



HTTP Response

■ Sample Response:

HTTP method → HTTP/1.1 Status 200 Document follows → Status line
Status code
Server: Apache/2.2.15 (CentOS)
Date: Tue, 02 Aug 2015 12:08:10 GMT
Content-Type: text/html → Content type is required
Content-Length: 3434
Last-modified: Mon, 1 Jan 2002 08:12:54 GMT
(... extra line break, then the document data ...)

■ The document is sent as a stream of bytes

- ☐ It doesn't matter if the document is a simple ASCII file or a complicated multimedia presentation – it gets encoded and sent!
- ☐ The browser reads the “type” of data from the Content-Type header (a MIME type value) and decode the document as it arrives.

■ <http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>

HTTP Response - Status Codes



■ Response **Status code** examples:

Code	Reason	Explanation
<i>2xx codes Success</i>		
200	Document follows	<i>The request succeeded</i>
204	No Response	<i>Request successful but no data to send</i>
<i>3xx codes Redirection</i>		
301	Moved Permanently	<i>The document has moved to a new URL</i>
302	Moved Temporarily	<i>The document has moved temporarily</i>
304	Not modified	<i>Cacheable Doc unchanged if GET conditional (e.g "If-Modified-Since")</i>
<i>4xx codes Client errors</i>		
401	Unauthorized	<i>The document is restricted</i>
403	Forbidden	<i>Access is forbidden</i>
404	Not Found	<i>The document could not be found</i>
<i>5xx codes Server Errors</i>		
500	Server Error	<i>Server has an error</i>

Web pages and HTTP



A web page is built up by the browser with a series of HTTP requests:

1. The browser sends an HTTP request to the server. The server responds with the HTML text.
2. The browser parses the HTML for linked page elements e.g. images, then sends a separate request for each element specifying the mime type.

The server responds with the resource



Example - The data source files

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="description" content="cat"/>
    <meta name="keywords" content="cat, cute"/>
    <meta name="author" content="C. DeVille" />
    <title> My Cat </title>
  </head>
  <body>
    <h1>
      My Cat
    </h1>
    
  </body>
</html>
```





HTTP GET Request - Part 1

<https://mercury.ict.swin.edu.au/cat/mycat.html>

Request Header

GET /cat/mycat.html HTTP/1.1

Host: mercury.ict.swin.edu.au

User-Agent: Mozilla/5.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;

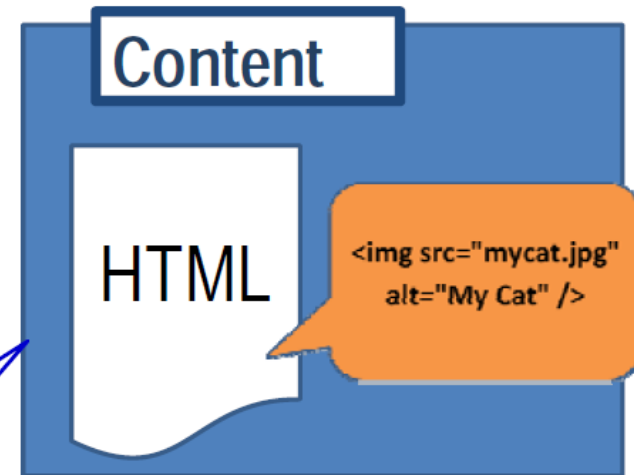
Response Header

HTTP/1.1 200 OK

Date: Tue, 10 Mar 2015 03:30:56 GMT

Content-Length: 424

... Then body ... In the **body** of the
http message





HTTP GET Request - Part 2

Request Header

GET /cat/**mycat.jpg** HTTP/1.1

Host: mercury.ict.swin.edu.au

User-Agent: Mozilla/5.0

Accept: image/png,**image/***;q=0.8,*/*;q=0.5

Response Header

HTTP/1.1 200 OK

Date: Tue, 10 Mar 2015 03:30:56 GMT

Content-Length: 424

In the **body** of the
http message

Content

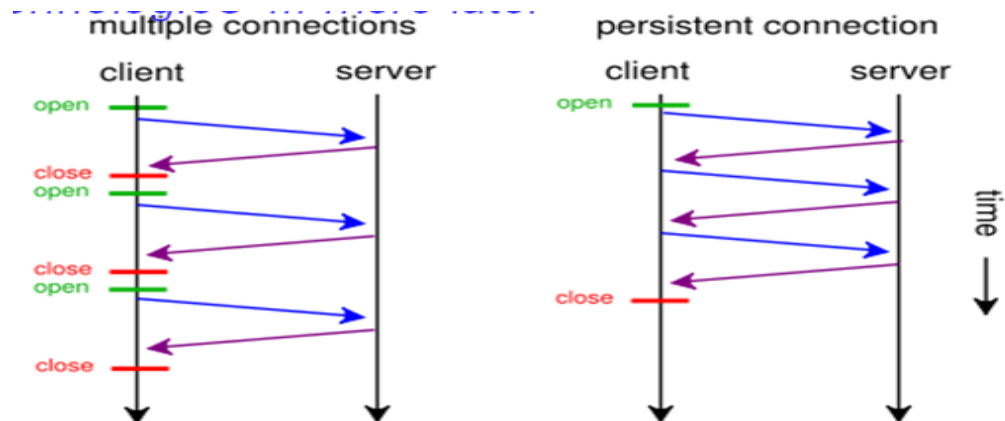


HTTP is “Stateless”



■ HTTP – has *Stateless* Connections

- This may seem like a limitation but in fact allows for a highly scalable, distributed software architecture
- The connection between the client and server is active only for single requests (default in HTTP 1.0) or for a limited time (default in HTTP 1.1 - depends on server e.g. Apache 2.2 ~ 5 secs).
- No memory of past connections
- *When we need **stateful** connections, we need to also use other client and/or server side technologies ... more later*



HTTP/1.1



- Most Web servers and browsers (from IE4+ and NS4.5+) are HTTP 1.1 compliant and can handle compressed file transfers.
- **HTTP 1.1 is now considered “stable”** - there is no current activity to change it or create a new version.
- The current version HTTP 1.1 includes:
 - **Virtual hosting** - allows multiple domain names to be assigned to a single IP address. eg. Both **swin.edu.au** & **swinburne.edu.au** can be one IP address.
 - **Efficient request handling** - allows the reuse the same TCP/IP connection when using the same server rather than establishing a new connection every time, e.g. images, scripts, etc. can all be downloaded using the same TCP/IP connection that was used to download the html page. Hence fewer TCP connections are required - less traffic and faster delivery.
 - **Efficient caching** - the HTTP/1.1 caching model is better defined allowing both servers and clients to control the level of “cachability” and the conditions under which the cache should update its contents.

HTTP/2



- Proposed standard – published **RFP** May 2015
- Similar high level syntax to HTTP 1.1
- Aim: Decrease latency to improve page load speed in web browsers by considering:
 - ☐ Data compression of HTTP headers
 - ☐ Server push technologies
 - ☐ Loading page elements in parallel over a single TCP connection
 - ☐ ...
- Support
 - ☐ Browser Support
 - ☐ progressively being implemented – Chrome support by default
 - ☐ Server Support
 - ☐ IIS Windows 10 and Windows Server 2016; Apache 2.4.12

Outline



■ Unit Overview

■ Introduction to the Web

- ☐ The Internet and the Web
- ☐ IP addresses and Domain names
- ☐ The Internet Stack
- ☐ Client requests and Server responses
- ☐ HTTP
- ☐ HTTPS

HTTPS



■ HTTPS:

- ☐ HyperText Transfer Protocol over a **Secure Socket Layer** (or just “**HTTP over SSL**”).

■ **SSL**: An open non-proprietary protocol for encrypted data transfer.

- ☐ SSL “Strengths”: 40 bit and 128 bit (length of session key)

- ☐ SSL Certificates (“digital certificates”) can be used by servers to verify their identity to visitors through a “trusted third party” method.

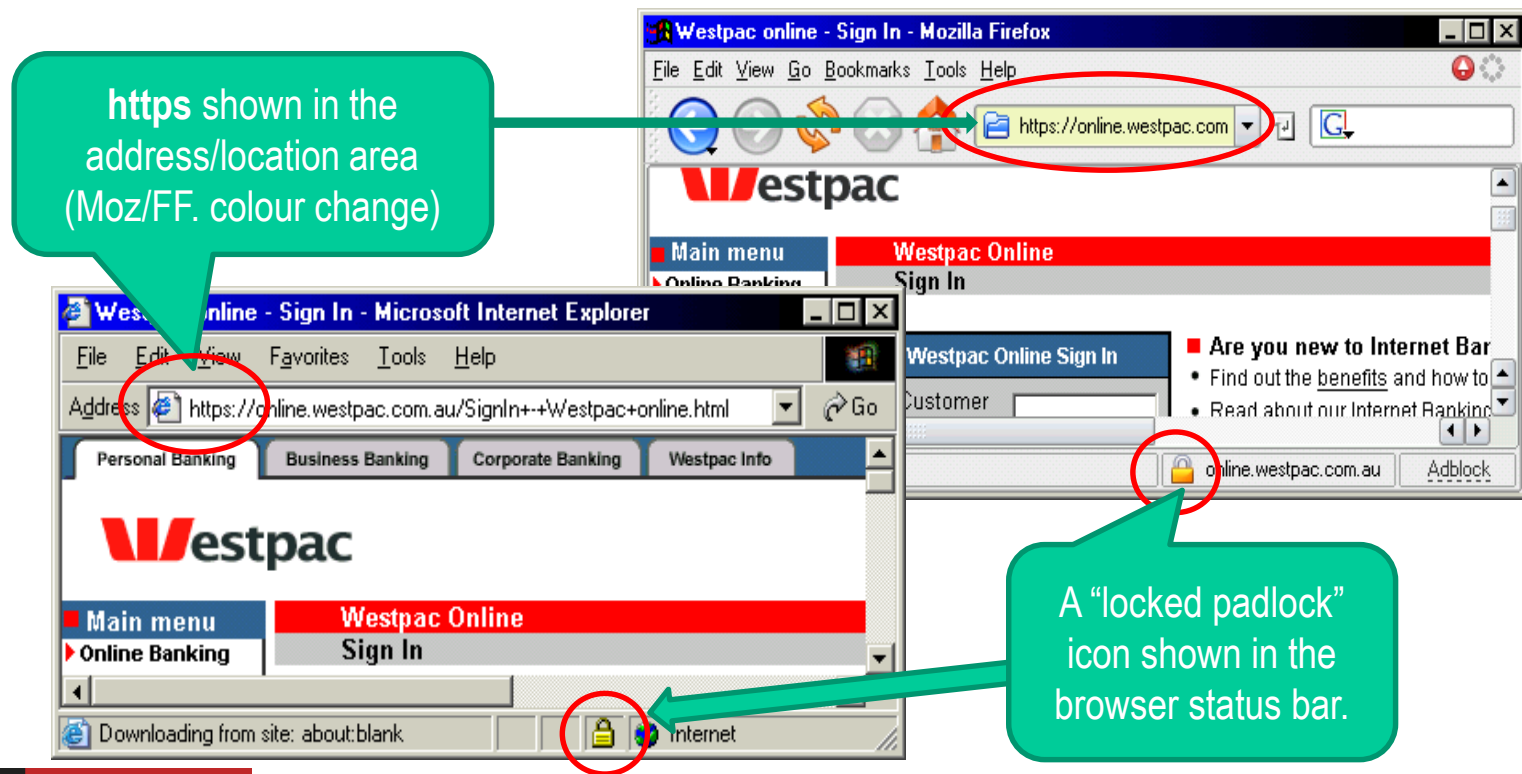
■ Encrypts and decrypts all information transferred between client and server.

■ HTTPS uses a standard port **443** instead of the HTTP port 80.

HTTPS



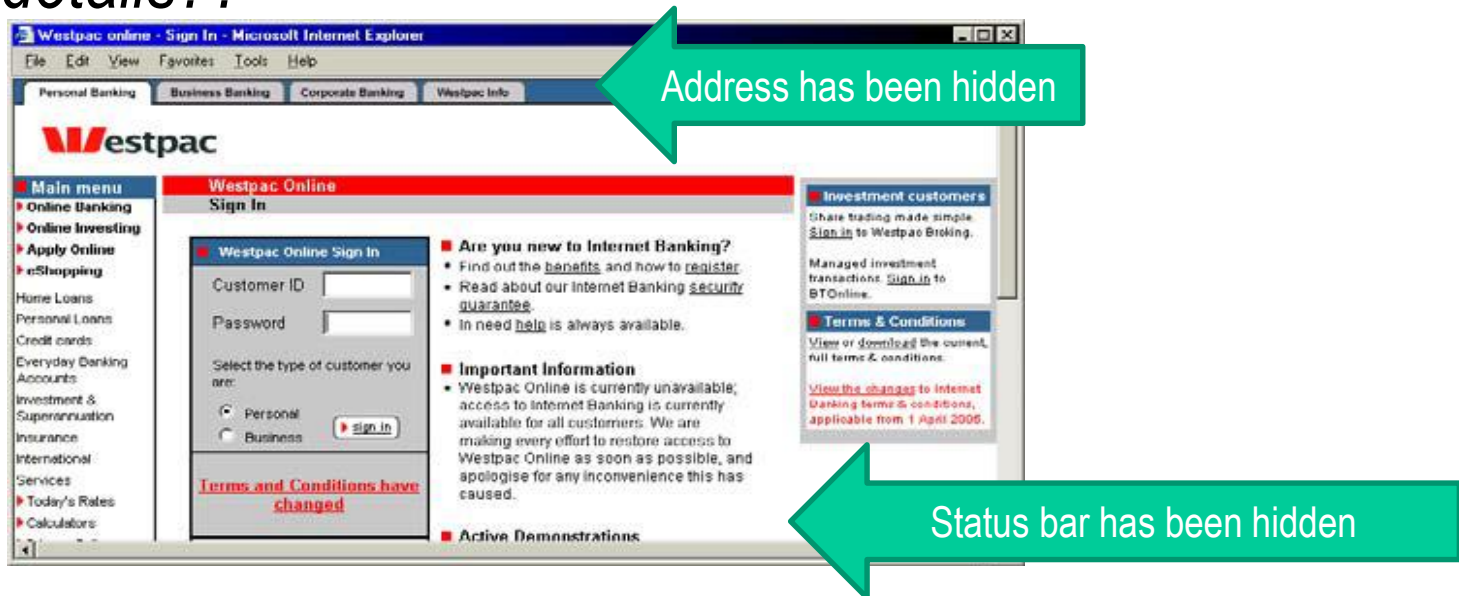
- Web browsers indicate with messages, icons, colours, whether a connection to a website is HTTPS or not.



HTTPS



- *Would you sign in to this site with your banking details??*



*JavaScript has been used to open a browser window
hiding address and **status bars** – not HTTPS no
security!*

HTTPS



Maybank2u.com Online Financial Services - Mozilla Firefox 4.0 Beta 7

File Edit View History Bookmarks Tools Help

maybank2u.com.my https://www.maybank2u.com.my/mbb/m2u/common/M2ULogin.do?action=Login

Most Visited Y! Mail Gmail Go Analytics Go Trends Go Insights AdW: TrafEstimator OnI9 Intention Blogger in draft Fb Twitter YouTube ClickBank

Spam (11) - Yahoo! Mail Maybank2u.com Online Financial Services

maybank2u.com
Friday, 10 December 2010 11:20:07

Welcome

Log in to Maybank2u.com online banking

Please make sure you login at the correct
Maybank2u.com URL/Address

New users: First time log-in please click here

Online stocks: Click here to login

Username:

Password:

Forgot Password?
[Click here to reset password](#)

Don't have an online banking account?
[Click here for information on opening an account](#)


Stay safe online!

- Never login via email links
- Never reveal your PIN / Password to anyone
- [Click here to notify us of any Maybank2u.com "phishing" website](#)

Forgot your Online Banking password? [Click here](#)

Forgot your PayBills password? [Click here](#)

For other online banking enquiries, call our Customer Care hotline at 1-300-88-6688 or 603-7844 3696 if you are overseas (24 hours daily, including holidays).






[Help](#) | [Terms & Conditions](#) | [Security, Privacy & Client Charter](#) | [FAQ](#)




HTTPS



Maybank2u.com Online Financial Services - Mozilla Firefox 4.0 Beta 7

File Edit View History Bookmarks Tools Help

1    http://67.19.188.10/~globusa/mbb/m2u/common/Access.php

2    Feedback

Most Visited Y! Mail Gmail Go Analytics Go Trends Go Insights AdW: TrafEstimator Onl9 Intention Blogger in draft Fb Twitter YouTube ClickBank

Spam (11) - Yahoo! Mail Maybank2u.com Online Financial Services Maybank2u.com Online Financial Services

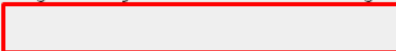
maybank2u.com

Friday, 10 December 2010 04:19:43 3

4

Welcome

Log in to Maybank2u.com online banking



New users: [First time log-in please click here](#)

Online stocks: [Click here to login](#)

Username:

Password:

Login

Forgot Password?
[Click here to reset password](#)

Don't have an online banking account?
[Click here for information on opening an account](#)



Stay safe online!

- Never login via email links
- Never reveal your PIN / Password to anyone
- [Click here to notify us of any Maybank2u.com "phishing" website](#)

Forgot your Online Banking password? [Click here](#)

Forgot your PayBills password? [Click here](#)

For other online banking enquiries, call our Customer Care hotline at 1-300-88-6688 or 603-7844 3696 if you are overseas (24 hours daily, including holidays).

 5 

[Help](#) | [Terms & Conditions](#) | [Security, Privacy & Client Charter](#) | [FAQ](#)

HTTPS



- ***When is a web site “secure”?***

- ☐ Using HTTP and having a “login” screen (user name and password) to establish a session with a server, does NOT mean the connection is “secure”. (It can offer some “usage” protection).
- ☐ An un-encrypted connection means that other machines may be able to “listen” to your data (traffic) as it travels on the internet.

- Even if we have a secure HTTPS connection there is still the problem of “stolen identity” – of users and servers!

- ☐ If a server can pretend to be the bank and intercept your request, you will still have a “secure” connection, but with a thief!
- ☐ Digital certificates are a way of verifying server (and client) identity, and that is why certificates are part of SSL and similar systems.



Learning Outcomes

- What is the Internet?
- What is the Internet stack and what are the functions of the TCP and IP layers?
- Functions of browsers and servers
- What is HTTP?
- HTTP messages and Web pages
- HTTP is 'stateless'. What does this mean ?
- What is HTTPS ?

The end of the lecture



Lab 01 starts next week.