# Web Development

Introduction

# Todays Lecture

We will look at:-

- Server Side technologies
- HTTP protocol (what is http in http://www.cct.ie )
- Web Servers (How do web pages get served to our browser)
- Intro to PHP programming language
- Installing development environment

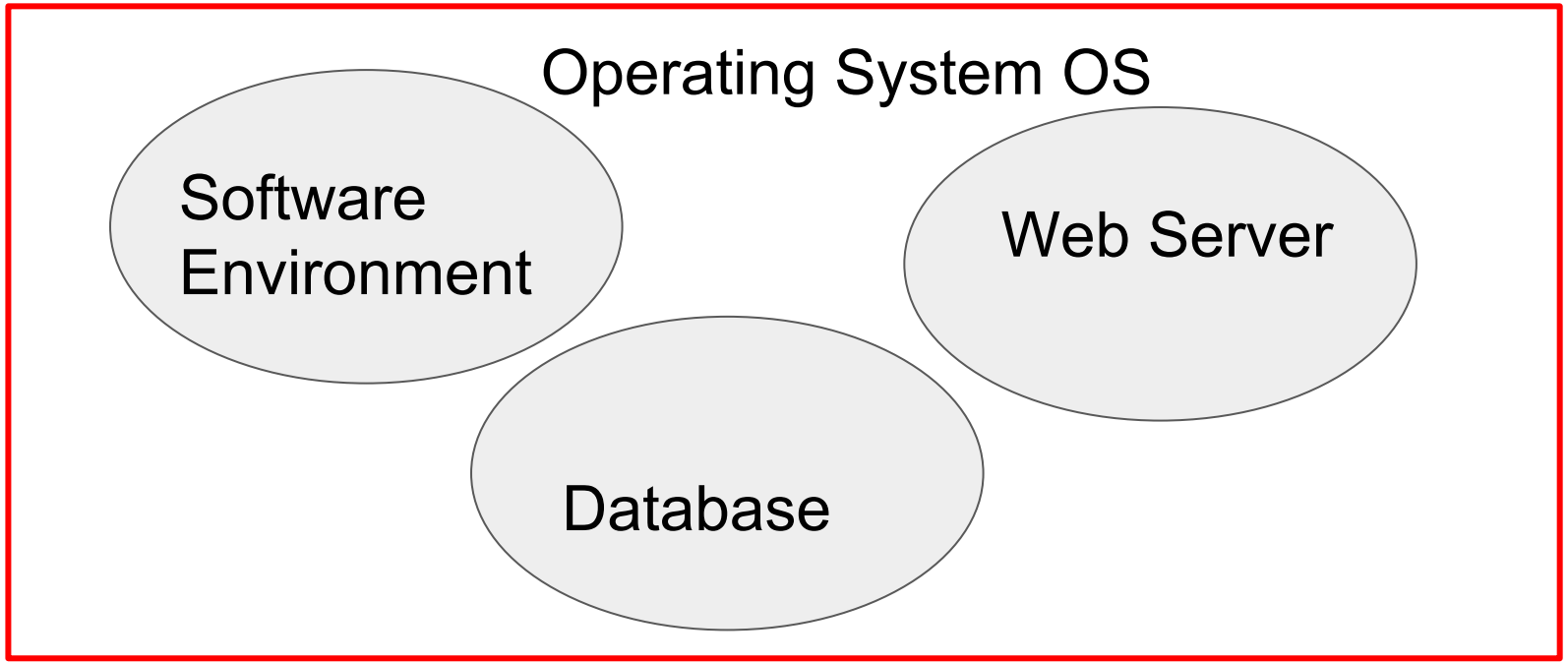
# What you have done before

- In previous modules you have worked with the basics of creating a website, using simple HTML and CSS i.e. creating static **client side** web pages
- This module is about stepping it up a notch and making use of the development technologies that currently exist to make your website more rich and dynamic i.e creating **server side** web pages

# Server-side Technologies

- This module is focused on the backend of websites and the technologies that are currently available to us to use.
- We are not as focused on making things look pretty, we are focused on using server-side code to make more our applications more functional and to consume resources and services over the internet..
- We need server side technologies as there are tasks that purely client side technologies cannot perform e.g. accessing databases , accessing services from external applications (APIs - Application Programming Interface)
- For security reasons client side technologies cannot perform certain tasks

# Development Environment




We will look at each of these components how to configure them and how they interact with each other

# The HTTP Request

- We are going to begin by looking at the processes that happen when we request a web page through a web browser
- When we are looking for a website, we typically enter the address into the address bar of our favorite web browser, e.g. Chrome, Firefox, Internet Explorer.
- What is actually happening when we press the enter button?
- We can break this into two different parts, what happens when the user presses enter and the server side magic.
- You'll notice when we type in www.cct.ie to a browser https: is prepended to the address. **What is this ?**

# HTTP Protocol

- **HTTP** is a *stateless asynchronous request-response application-layer protocol*. A client sends a request message to the server. The server then returns a response message to the client. In other words, HTTP is a *pull* protocol, a client pulls a page from the server

- Think of *protocols* as a standard way of communicating between a client and a server in an agreed way.

  Other protocols

  **FTP** - File Transfer Protocol , transferring files from client to server

  **SMPT** - Simple Mail Transfer Protocol, Internet standard for email transmission

  **HTTPS** - similar to HTTP but is more secure allows for the transfer of encrypted data over the internet to avoid “packet sniffing” . Becoming standard.

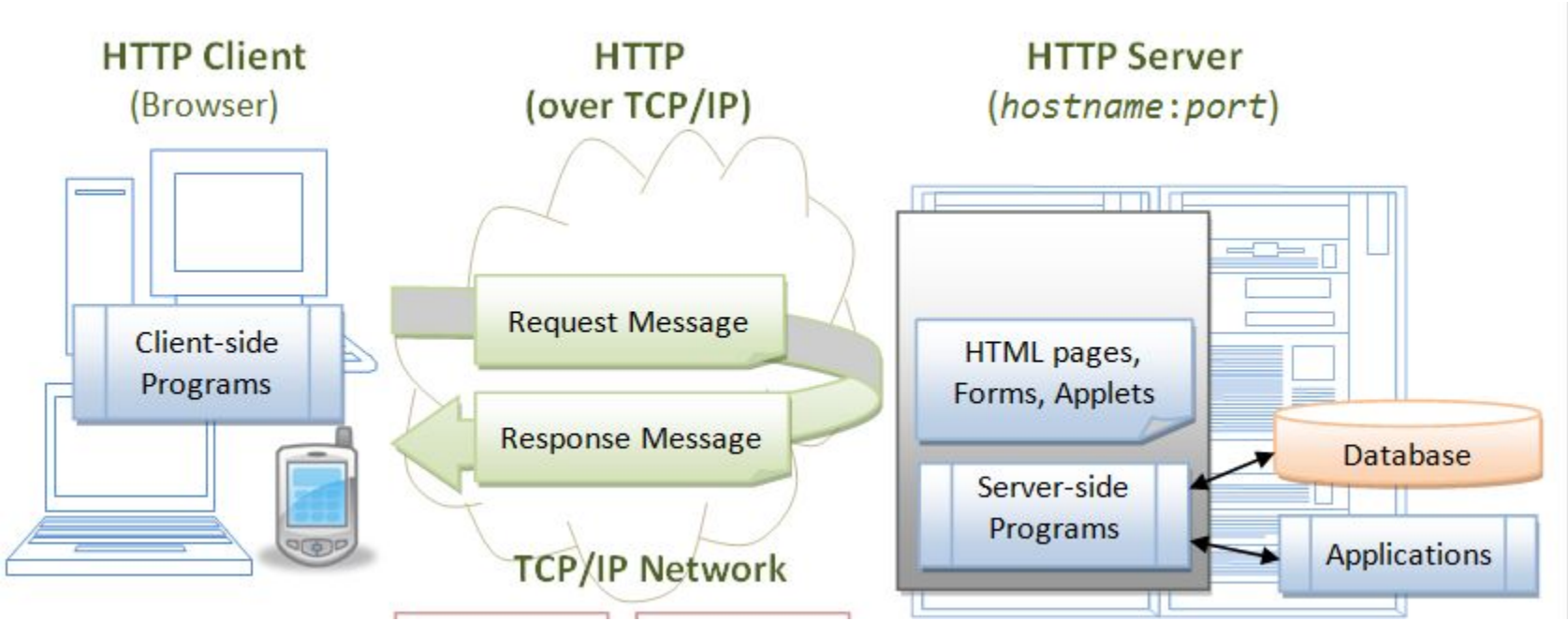
A formal definition of the Hypertext Transfer Protocol (HTTP/1.1): can be seen in the RFC document https://tools.ietf.org/html/rfc7231

# Web Application

- A *web application* (or webapp), unlike standalone (or desktop) application, runs over the Internet. Examples of web apps are google, amazon, ebay, facebook and twitter.
- A webapp is typically a *3-tier* (or *multi-tier*) *client-server database application* run over the Internet .
- It generally comprises five components:

# Web Application

1. **HTTP Server**: E.g., Apache HTTP Server, Apache Tomcat Server, Microsoft Internet Information Server (IIS), nginx, Google Web Server (GWS), and others.
2. **HTTP Client (or Web Browser)**: E.g., Internet Explorer (MSIE), FireFox, Chrome, Safari, and others.
3. **Database**: E.g., Open-source MySQL, Apache Derby, mSQL, SQLite, PostgreSQL, OpenOffice's Base; Commercial Oracle, IBM DB2, SAP SyBase, MS SQL Server, MS Access; and others.
4. **Client-Side Programs**: could be written in HTML Form, JavaScript, VBScript, Flash, and others.
5. **Server-Side Programs**: could be written in Java Servlet/JSP, ASP.NET, PHP, Perl, Python, CGI, and others.

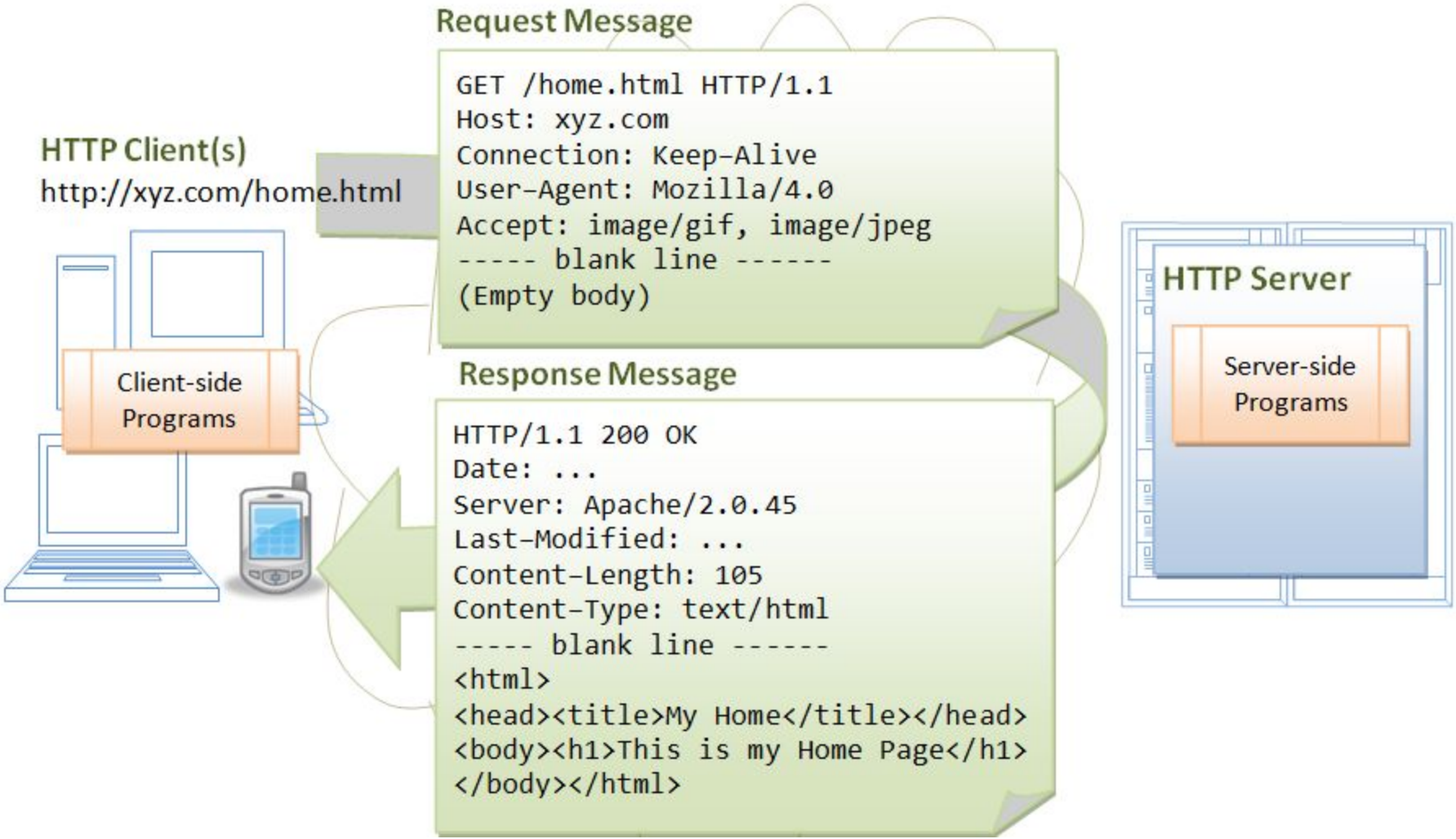

HTTP Client
(Browser)
HTTP
(over TCP/IP)
HTTP Server
(hostname:port)
Client-side
Programs
Request Message
Response Message
TCP/IP Network
HTML pages,
Forms, Applets
Server-side
Programs
Database
Applications

# HTTP Request

**Step 1** Web address entered by a user is translated into an I.P address using a Domain Name Server (DNS) who has the job of translating the text the user has entered into the machine readable I.P address

**Step 2** The I.P address of the website we are looking for is given to us and then the web browser makes a connection to the web server at the location specified by the address and asks for the page we are requesting.

**Step 3** The web server looks for the page we are requesting and then sends it back to the user (response).


Request Message
GET /home.html HTTP/1.1
Host: xyz.com
Connection: Keep-Alive
User-Agent: Mozilla/4.0
Accept: image/gif, image/jpeg
----- blank line ------
(Empty body)
HTTP Client(s)
http://xyz.com/home.html
Client-side Programs
Response Message
HTTP/1.1 200 OK
Date: ...
Server: Apache/2.0.45
Last-Modified: ...
Content-Length: 105
Content-Type: text/html
----- blank line ------
<html>
<head><title>My Home</title></head>
<body><h1>This is my Home Page</h1>
</body></html>
HTTP Server
Server-side Programs

# What is a Web Server?

- A web server delivers the files (or resource)  that the user is looking for.
- The hardware that the web server is running on does not really matter too much.
- We are interested in the software that is deployed on this hardware and how it creates and delivers the content to whoever asked for the content.

# What is a Web Server?

- For our purposes a Web Server is a process that runs on a computer permanently connected to the internet.
- A Web Server runs on a computer which requires an operating system. So when running a web server the choice of operating system can be important. Different web servers will run on one or many operating systems
- Depending on your development environment it is important to chose a Web Server that is compatible with your environment

# Web Server Software

- Lots of web server packages currently exist on the web, but Apache and IIS are the two big players in the game.
- Generally we choose a web server that is compatible with the development environment we choose to develop in
- In a development environment we can install a web server suitable for our development environment (programming languages, databases, OS etc.)
- In production, we generally choose a web hosting provider that is compatible with our technology which runs a web server that supports our applications needs

# Server Operating Systems

- Lots of different operating systems exist on the market today. Principally

  Linux

  Windows (and all its versions)

  MacOS

- Just like everything in this world, you can pick between software that is free, or software that costs money.

# Operating Systems

Choosing a Production environment.

- Generally we wouldn’t run our own web servers in a production environment unless you want to have control over your server environment.
- We will choose a hosting provider that meets the needs of our application and the software technologies it uses.
- Lots of hosting options out there
- It's possible that the operating system we develop in may be different to the production OS. Dependent on environment you’ve chosen
- We’ll discuss hosting options in more detail in later lectures

# How To Choose...

Extract from Hostingireland.ie

If you have designed a basic site with HTML you will find Linux hosting offers the best features for you, even if you use Windows in your home or office.

If you want to create a dynamic site using a server side scripting language and perhaps a database then you will have to choose more carefully:

If you prefer to use PHP or PERL with a MySQL database then Linux hosting is for you. - or maybe Java hosting if you need Java Applications.

If you prefer to use ASP, ASP.NET, Microsoft SQL Server or MySQL database then Windows hosting is for you.

# Web Server

Popular Web Servers

| | | |
|---|---|---|
| Apache HTTP | - | PHP |
| Apache Tomcat | - | JSP (JavaServer Pages) |
| IIS | - | ASP.Net |

- We need to pick which web server we are going to run given the technology we are developing in which may limit our choices

- We'll be looking at Apache HTTP Server, a popular *industrial-strength* web server, is provided free (with source codes) by the Apache Software Foundation (@ http://www.apache.org).
- The mother site for Apache HTTP Server is http://httpd.apache.org.

# How To Choose...

- In general we begin building an application, choosing  the software we want to develop in first e.g. PHP/MySQL and develop locally on our development machines.

- When the time comes to move to production then we need to make the decisions about where and how our application will run based on our needs as to the operating system and web server requirements.

- For example PHP is relatively Operating System independent while ASP.NET (a Microsoft technology) will only run in a Window OS environment.

- So depending on our choice of technologies the options for choosing a production environment may be wide or limited depending on the technologies we have chosen to develop in.

# Web Server Software

- Web servers generally work on port 80, if you remember from your networking classes that different services work on different ports.
- A port number is used to identify a specific process to which an internet or other network message is to be forwarded when it arrives at a server.
- Think of a webserver like a piece of software that is never stops.
- The web server sits and listens on port 80 for any requests coming in. If someone requests a particular file, it serves them that file (or resource).
- A web server should always be running i.e. high availability. This is important when choosing a web hosting provider. Look for their availability statistics before entering into an agreement.

# Web Server Software

The availability of a website is measured by the percentage of a year in which the website is publicly accessible and reachable via the Internet.

A **service-level agreement** (**SLA**) is defined as an official commitment that prevails between a service provider and a client . Particular aspects of the service – quality, availability, responsibilities – are agreed between the service provider and the service user.

# URLs

- URL stands for **U**niform **R**esource **L**ocator. It is generally the address we use to locate files/resources on the web.

An example of a URL is: http://www.cct.ie/index.php

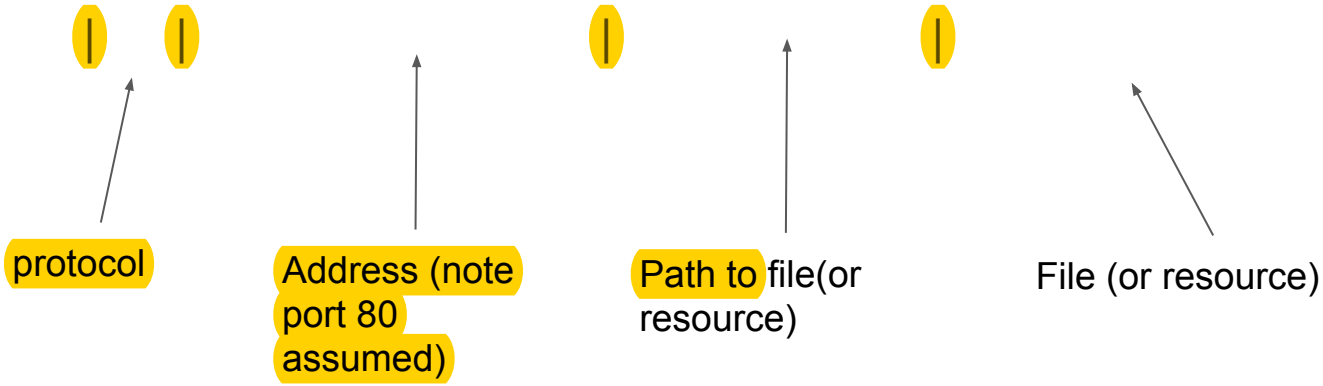
**http** – this is the protocol we are using

**www.cct.ie** - This is the domain address

*index.php* this is the file we are requesting

# URLs

http://www.example.com/directory-name/file.php(.html)




User generally wouldn't type this address in but may be requested via web application e.g. through a link on a web page

e.g. <a href="http://……>Clickable Link</a>

# Web Server Software

How does a web server know where to find the files (or resources ) we require

- The web server has access to a  directory where all of the files that can be served to a user via a request are located.
- Think of it as a folder on a computer.
- The folder is where the files we want served over the internet are typically kept in a folder called **htdocs** or **www**
- Anything inside of this folder is live for a client to request. Anything outside of this folder can not be accessed over the web.

# Web Server Software

If you type in the name of a website e.g. www.cct.ie there is a folder on its web server called ***/htdocs/*** where the files for the cct website live.

- If we want the index page we ask for www.cct.ie/index.php
- Inside the **htdocs** folder there is a file called *index.php*

The web server running on the hosting provider for cct manages this process and is configured to know where to look for files requested.

The web server has a **Document Root** set (in this case *htdocs* ) which is where the files accessible by the web are placed

# Web Server - cont.

- If you do not tell the web server that you want a particular file, and just enter www.cct.ie the web server has a default file that it will send to you called *index.php*
- This again is configured by the web server to determine what is sent back by default.
- In time we may want to change this to *index.html,* to do so we would have to reconfigure the web server configuration settings

  Check that www.cct.ie and www.cct.ie/index.php return the same page

# Software and Ports

- Web servers run on port 80 by default and this is assumed when we type in a URL into a browser
- This is why we type www.cct.ie rather than www.cct.ie:80 as the port 80 is assumed. (check that both return the same page)
- Sometimes we need to run multiple web servers on our development machines so sometime we may start a web server on a port other than :80
- If so we need to specify the port of the web server process we require
- e.g. www.localhost:8080 where **localhost** is our local web server named process and 8080 is the port we started the web server process on

# Server HTTP Status Codes

- When we are working with a web server, the HTTP protocol has a number of standard codes which the server returns as part of the response to the client who has issued a request.
- Code we want to see is code 200 which indicates everything is ok
- One codes we want to avoid is the 404 code, which means that the requested page cannot be found.
- Although these are the most common codes, there are many more
- We can interrogate the response code and take appropriate action e.g. error messages of dreaded 404 web page

# Server HTTP Status Codes

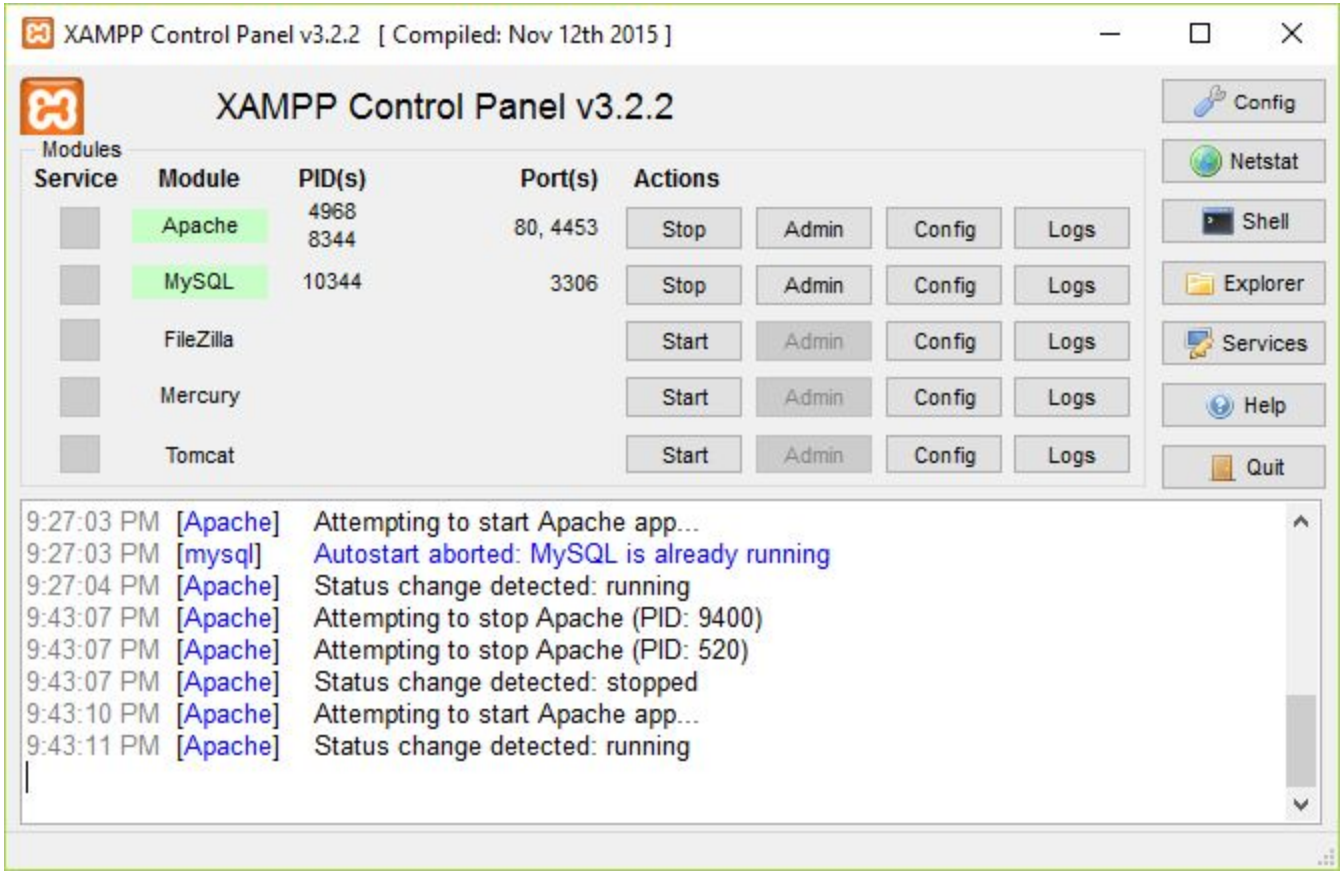
For a full list of the codes take a look at the Wikipedia article:
https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

In general response codes can be classified as

- **1xx** (Informational): The request was received, continuing process
- **2xx** (Successful): The request was successfully received,  understood, and accepted
- **3xx** (Redirection): Further action needs to be taken in order to complete the request
- **4xx** (Client Error): The request contains bad syntax or cannot be fulfilled
- **5xx** (Server Error): The server failed to fulfill an apparently valid request

# XAMPP

- When we are developing a website, we may need a Webserver and a Database running on our local development environment
- XAMPP is a package that was developed to install an Apache web server and a MySQL database locally on your machine.
- Once installed it provides a  simple interface to allow you to start and stop these services


XAMPP Control Panel v3.2.2 [ Compiled: Nov 12th 2015 ]
XAMPP Control Panel v3.2.2
Modules
Service
Module
PID(s)
Port(s)
Actions
Apache
4968
8344
80, 4453
Stop
Admin
Config
Logs
MySQL
10344
3306
Stop
Admin
Config
Logs
FileZilla
Start
Admin
Config
Logs
Mercury
Start
Admin
Config
Logs
Tomcat
Start
Admin
Config
Logs
Config
Netstat
Shell
Explorer
Services
Help
Quit
9:27:03 PM [Apache] Attempting to start Apache app...
9:27:03 PM [mysql] Autostart aborted: MySQL is already running
9:27:04 PM [Apache] Status change detected: running
9:43:07 PM [Apache] Attempting to stop Apache (PID: 9400)
9:43:07 PM [Apache] Attempting to stop Apache (PID: 520)
9:43:07 PM [Apache] Status change detected: stopped
9:43:10 PM [Apache] Attempting to start Apache app...
9:43:11 PM [Apache] Status change detected: running

# XAMPP

- Once the web server is running we now need to place all our web pages in the folder location Specified by the Document Root. Generally this will be inside the *C:\xampp\htdocs* folder
- Once your web server and database are running, you may want to access your web site through a web browser.
- To do this, we will use **http://localhost/**... in the web browser to open files on the web server. Where localhost is our local web address configured in our web server configuration file.
- Everytime we want to view the changes we made, we must do it through the web browser and access the files served up by the  web server. No more viewing files directly.

# XAMPP

- XAMPP installs a MySql database server
- MySql is a RDBMS (Relational Database Management Systems)

- To log into your database, XAMPP has default login details such as *root* for the username and nothing (blank) as the password.
- These default details should be changed if you are running a real web server!
- We will see MySql in much more detail in next lectures

# XAMPP

- When we install XAMPP, the Apache web server comes packaged with an additional module called **PHP**
- **PHP** is a programming language that can take a simple HTML web page and make it perform additional tasks such as connecting to a database and generating dynamic web pages
- We introduce PHP next and will explore it in depth in the next few lectures

# PHP

- PHP is a Server side programming language.
- It is an interpretive language rather than a compiled language

- Instead of having the .html prefix we replace files with the .php prefix
- The contents of the file contains simple HTML, but we can have add sections of PHP code that will be executed (or parsed) on the server side and generate content added to our web page.
- PHP can perform complicated tasks such as form processing, database access and update etc before returning content to the web page

# PHP

- Whenever we want PHP code to run, we have to surround the PHP code with additional tags such as <?php as an opening tag and ?> tag to close the block

```
<html>
<body>
        <?php
                echo “<p>This is some php generated content</p>”;
        ?>
</body>
</html>
```

- The Web Server requires these tags as it indicates to the web server that what is contained within these tags is PHP code which the server needs to process (or parse)

# Summary - Development Process

Install required software development platform

| | | |
|---|---|---|
| PHP | - | Apache Http |
| JSP | - | Apache Tomcat |
| ASP.NET | - | IIS |

Then decide on a Web Server that is compatible with this platform

# Summary - Development Process

Your operating system may also determine the platform you use

| | | |
|---|---|---|
| PHP and JSP | - | are platform independent |
| ASP.Net | - | Windows only (as IIS proprietary to windows) |

We may require a Database . Will database run on OS and with Web Server ?

| | | |
|---|---|---|
| MySQL | - | Open source, free |
| OS | - | Windows, Mac OS, Linux OS |
| Web Server | - | Apache Http, IIS amongst others |
| Oracle | - | Commercial Licence required |
| | - | as above |
| MSSQL | - | Commercial Licence required |
| OS | - | windows, (recently linux version) |
| Web server | - | IIS |

# Conclusion

We looked today at:

- HTTP protocol
- Web Servers
- Architecture of client/server  http request - response
- Development Environment
- Introduction to PHP

# Finally - Homework !

Please use the instructions to install XAMPP on your computers to set up your development environment.

There is a troubleshoot doc. If you get stuck.

Please help each other to get this installed so we’re ready to go in the next session and get stuck into some programming.

# Questions ?

# Next Time...

- We will start developing in PHP next time. Looking at the syntax and how we can integrate with our client side html pages.
- We will also have a look around XAMPP and its components and configuration
- We’ll sort out any issues with installation of XAMPP