The Client Server Model

ASP.NET web forms are one of a number of different technologies for writing applications for the web.

To summarise, ASP.NET web forms allow you as a programmer to write applications whilst at the same time hiding much of the complexity of what is going on under the bonnet. This is what you are currently doing on Visual Web Development.

In this module we aim to take the lid off the technology and see not only how it works in more detail, but also understand some of the implications related to security on a global stage.

Let’s start by unpacking some of the ideas we have covered so far.

Firstly ASP.NET is a server side technology. So what exactly does that mean?

The Projector Screen “Model”

Imagine that you are at the cinema watching a film.

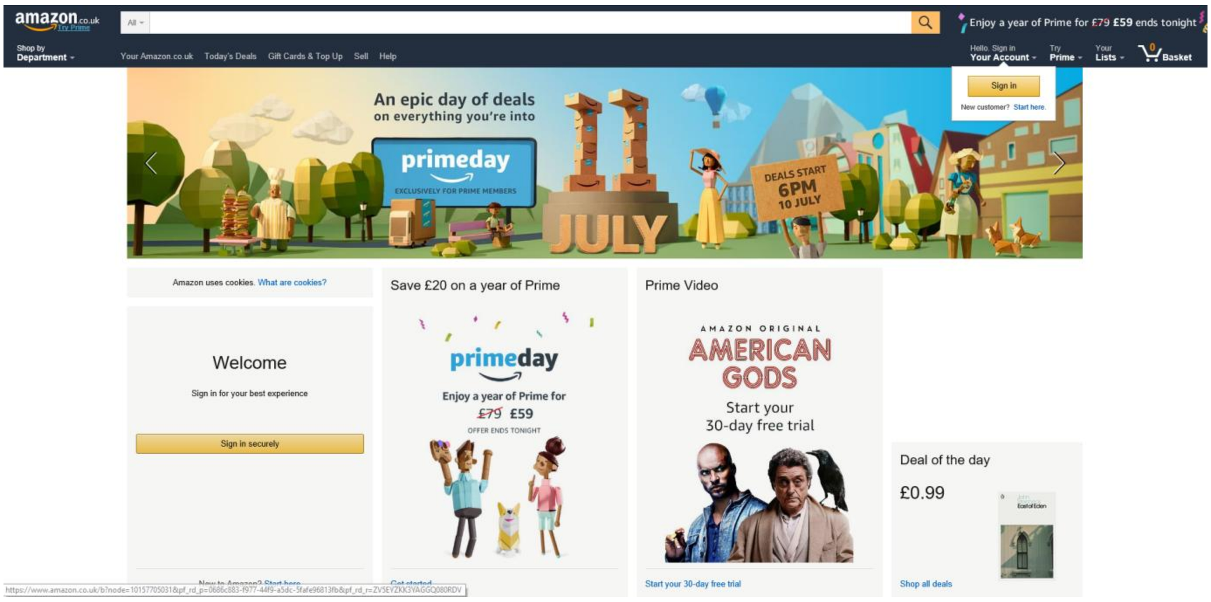
So, here is the question, where is the film you are watching on the screen coming from?

The film isn’t being generated by the screen; it is being generated by the projector at the back of the cinema.

But what does this have to do with server side languages?

The Client Server Model

OK, when you look at an application on screen such as Amazon, you see the web application in your browser of choice.

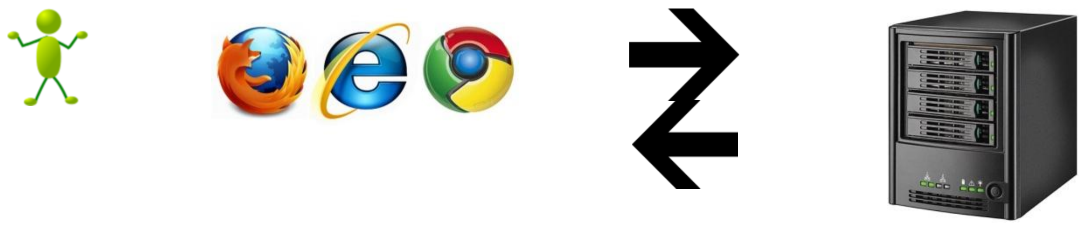

amazon.co.uk
Try Prime
Shop by Department
Your Amazon.co.uk
Today's Deals
Gift Cards & Top Up
Sell
Help
Enjoy a year of Prime for £79 £59 ends tonight
Hello. Sign in Your Account
Try Prime
Your Lists
Basket
Sign in
New customer? Start here.
An epic day of deals on everything you're into
primeday
EXCLUSIVELY FOR PRIME MEMBERS
11 JULY
DEALS START 6PM 10 JULY
Amazon uses cookies. What are cookies?
Welcome
Sign in for your best experience
Sign in securely
Save £20 on a year of Prime
primeday
Enjoy a year of Prime for £79 £59
OFFER ENDS TONIGHT
Prime Video
AMAZON ORIGINAL
AMERICAN GODS
Start your 30-day free trial
Start your 30-day free trial
Deal of the day
£0.99
Shop all deals

You interact with the page, browse, add things to your cart and buy stuff.

The thing to keep in mind is that the pages you are interacting with are not generated by the browser on your local machine; they are being generated by the server. The server is a second computer in some other location, which takes input from your browser, generates pages and then sends them to your browser.

This is called the client server model.

Consider the following diagram…



You

Browser

Network for data transfer

Web Server

As a user of the site,

You open your browser and type in a URL.

- The browser sends data over the network to the server asking for the home page of the site
- The server looks for the page
- The page might be a static web page; a page generated by code or in many cases a combination of the two
- Once the page has been generated the server sends the page to the requesting browser over the network
- Once received the browser renders the page to show you the page you asked for

Some important things to note:

- As indicated above the page is generated by the server.

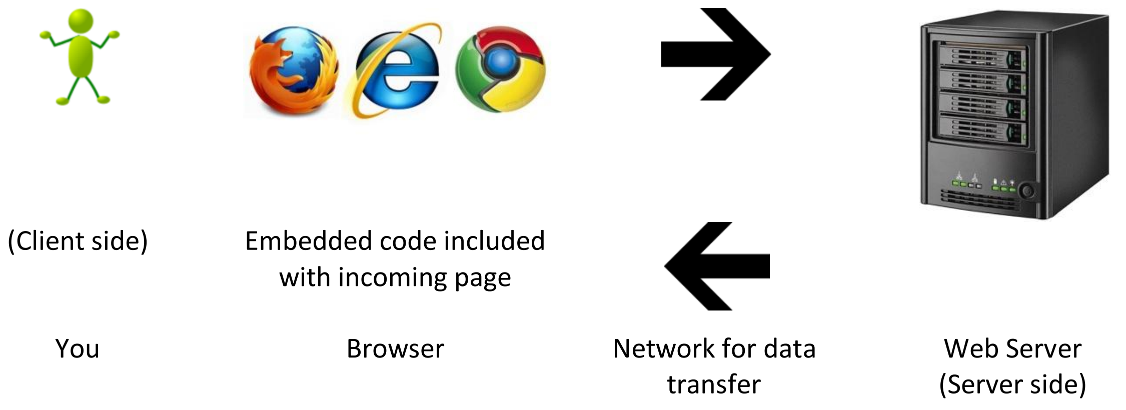
- There are a variety of technologies that may be used, PHP, ASP etc.

Extending the Model

As with most things in life the majority of explanations tend to be a simplification, as is true of the above.

Although the page is mostly generated at the server, the browser may also play a part in the generation of the page. Each browser has its own built in programming language, typically Java Script.

In the case of languages such as Java Script the browser generates the web page but this time includes code in the page that will be run on the browser when it renders the page.

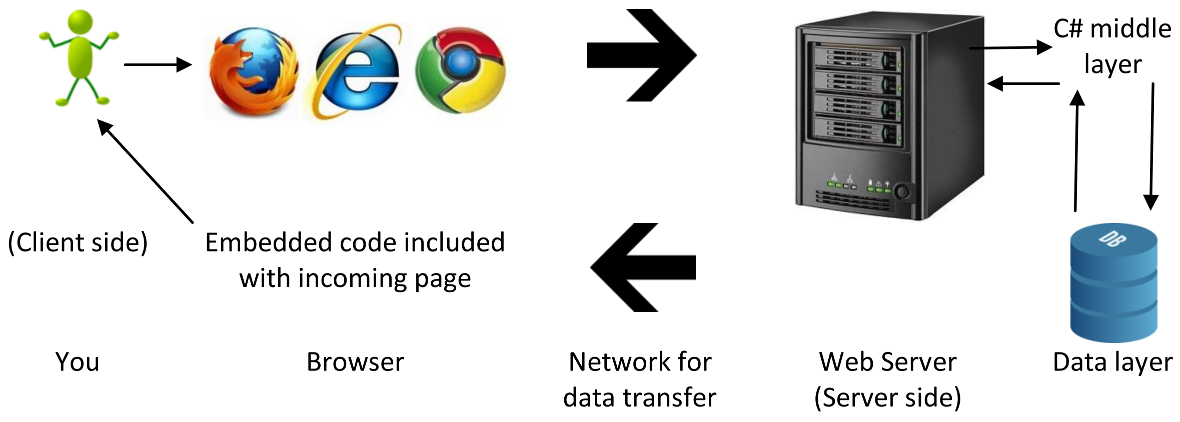

(Client side)
Embedded code included with incoming page
You
Browser
Network for data transfer
Web Server (Server side)

So, languages such as ASP & PHP are referred to as server side languages, in that they are run at the server. Languages such as Java Script are referred to as client side languages as they are run at your end of the process, the browser/client side.

Database Processing

Again, we may extend the above model to see how the database might fit in. The processing at the server could be quite simple, i.e. serving up a single static page. The processing could be more sophisticated, i.e. using classes and database connectivity.

This might extend the picture like so…

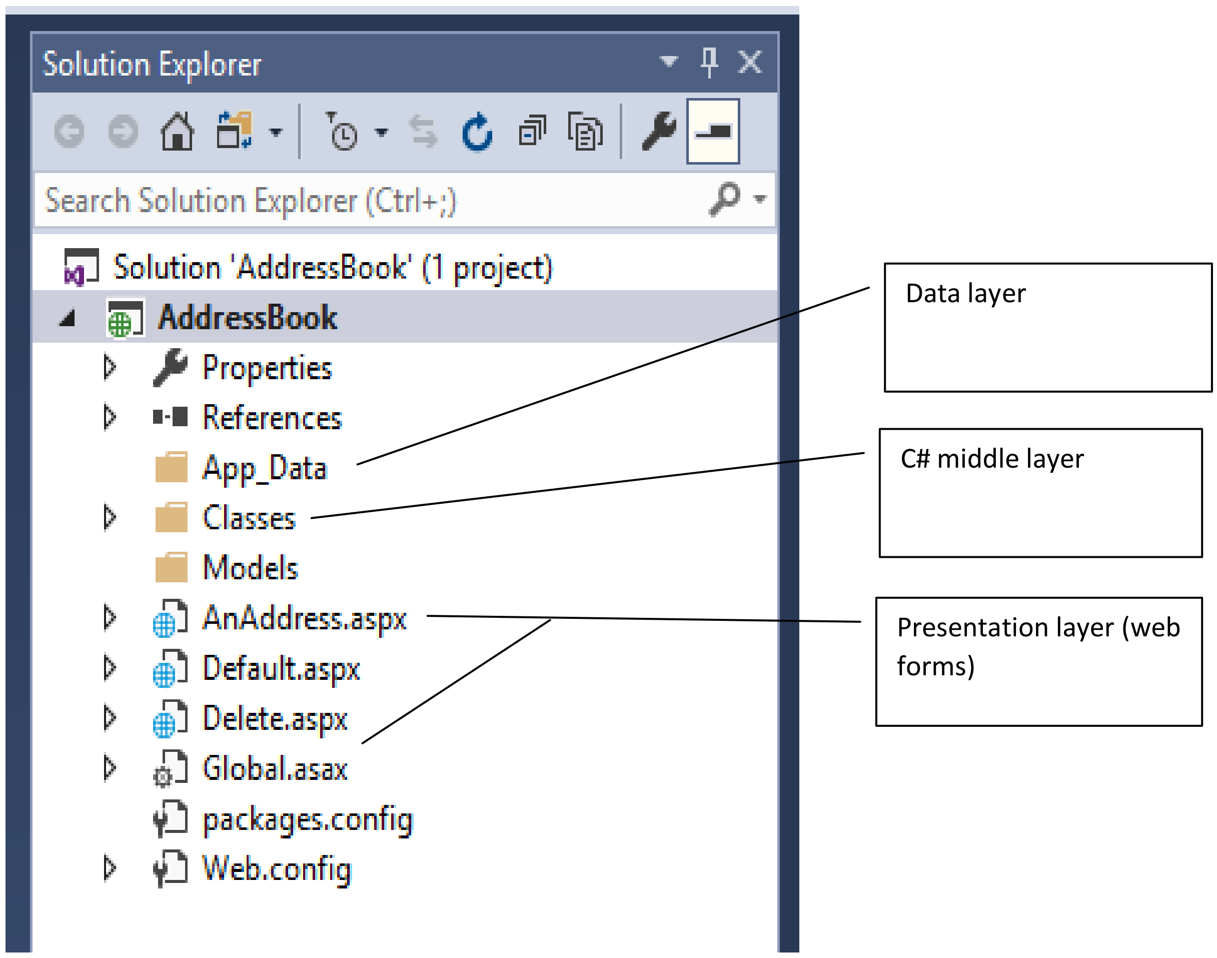


Not only may code be present now at the client and server ends of the system, we have now added code that may exist in the middle layer, language of your choice, e.g. C# along with code at the data layer, SQL and a data store to consider.

Relating this to Visual Web Development

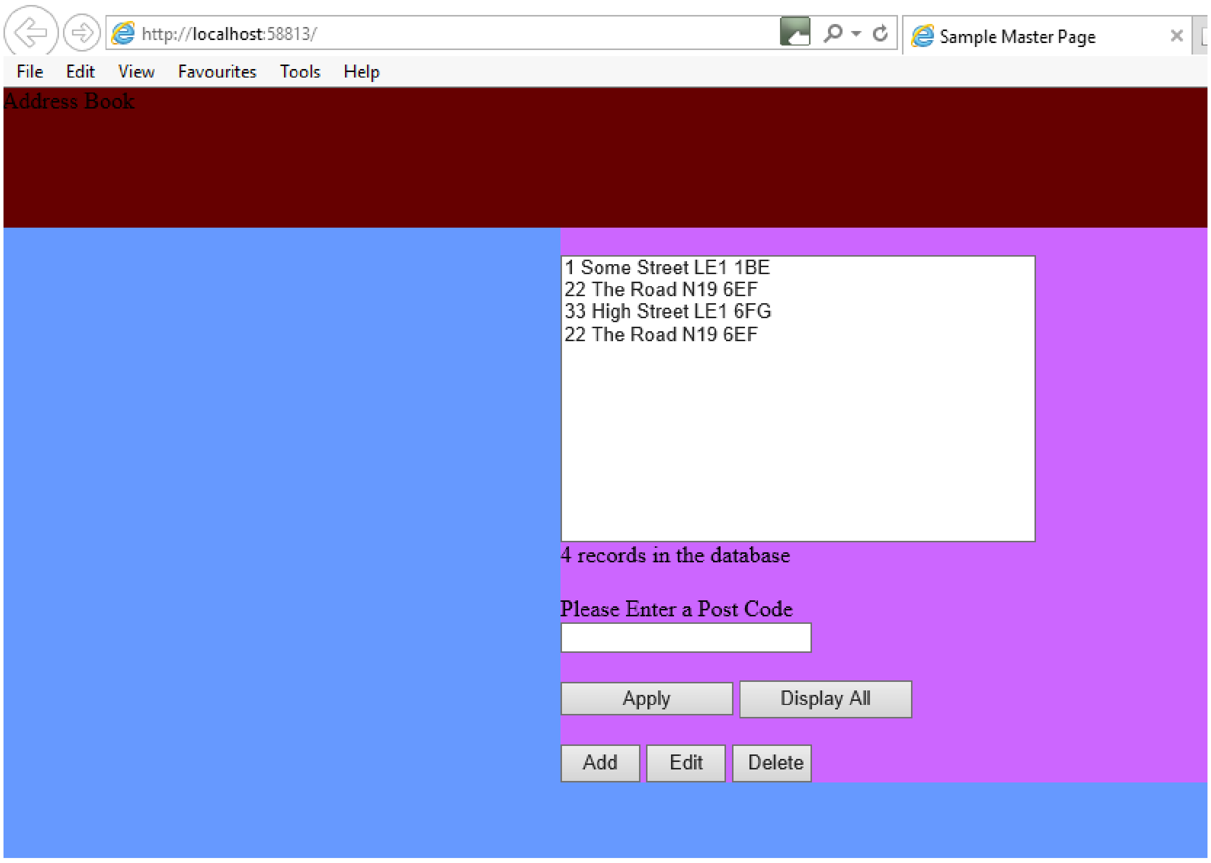
If we look at Visual Studio, we can see all of the above components.

Looking in solution explorer we see…



Now here is the bit you may have missed to this point. Where is the client and where is the server? For this example we will look at address book finished available from the module web site.

If you run the program by pressing F5 the browser starts and the main page for the site is displayed…



Address Book

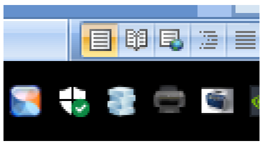
```
1 Some Street LE1 1BE
22 The Road N19 6EF
33 High Street LE1 6FG
22 The Road N19 6EF
```

4 records in the database

Please Enter a Post Code

Clearly the program is being displayed in the browser, but where is the server?

The server is actually running too, it may be seem in the Windows task bar…



(The server is the middle blue icon. You may need to expand the task bar to see all icons!)

What is happening is that the machine you are writing your program on is acting as both client and server.

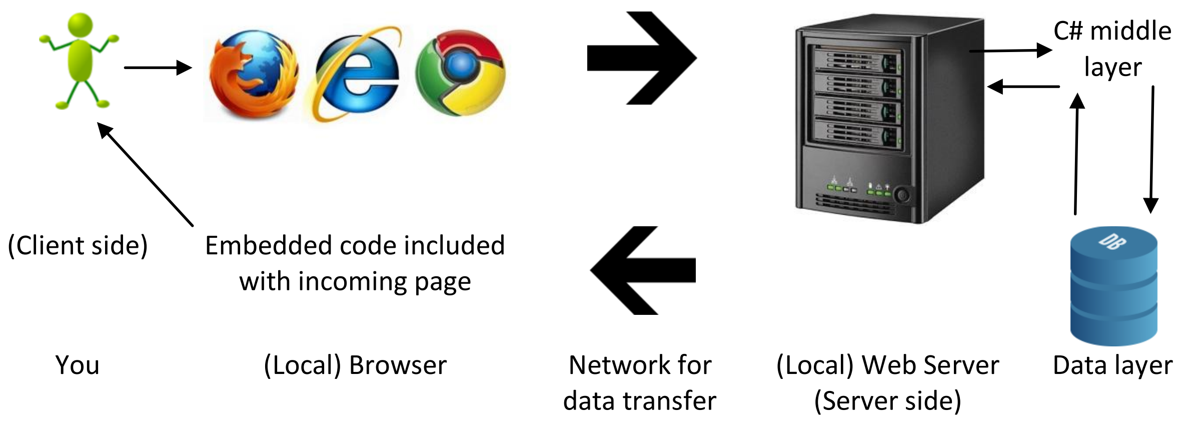
The computer is essentially "talking to itself" as it carries out both roles.

One clue is in the URL of the application…

"localhost" tells us that the computer is using itself both as client and server.

So the sequence of events is…

- You press F5 to run your program
- Visual Studio launches both local browser (client) and local development server
- Your browser then communicates with the local server in exactly the same manner as our previous diagram


C# middle layer
(Client side)
Embedded code included with incoming page
You
(Local) Browser
Network for data transfer
(Local) Web Server (Server side)
Data layer
DB

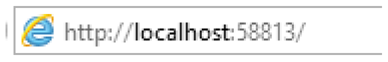
This is one of the reasons why Visual Studio is such a nice environment to develop in, as it has its own built in development server. If you use other environments you will typically need to set up your own web server (e.g. Apache) to allow the browser to access the server's facilities.

So let's re-wind a bit. We have looked at how the client browser and remote server communicate with each other. We have also considered the idea of client and server based code. We have also hinted that ASP.NET is not the only way we could develop systems like this.

The Internet and TCP/IP

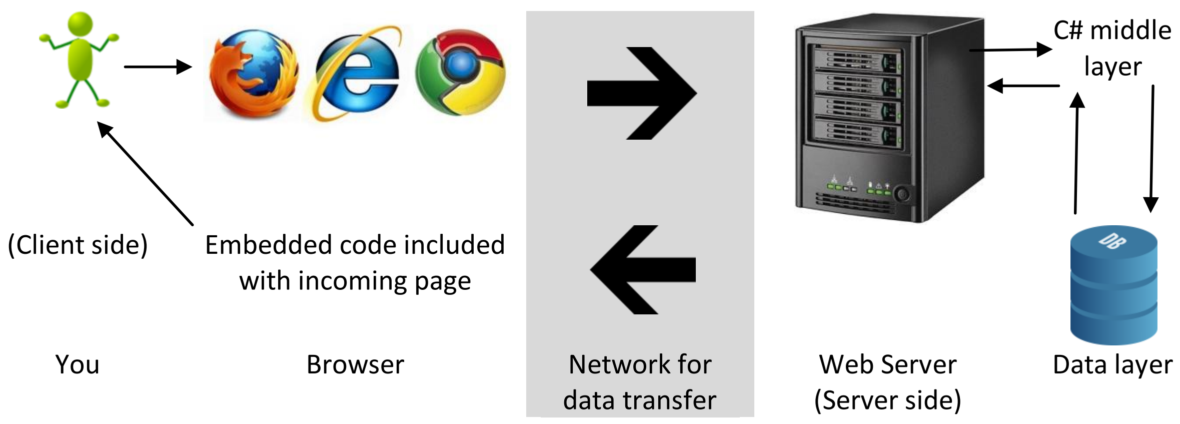
To fully understand what is going on we are also going to have to unpack some of the things going on when the browser makes a request to the server.

As we saw above the URL for our site running on local-host reads as follows.



http://localhost:58813/

We have established that local-host tells us that the client browser is talking to the local rather than remote computer. There are some additional details that are worth considering in terms of how the network for data transfer operates.



The internet is a huge network of interconnected computers that communicate with each other using a protocol called TCP/IP (Transmission Control Protocol / Internet Protocol) and was developed as a result of defence research in the 1960s. Since then TCP/IP has became the standard for the internet in 1982 and allows a program on one computer to communicate with a program on another.

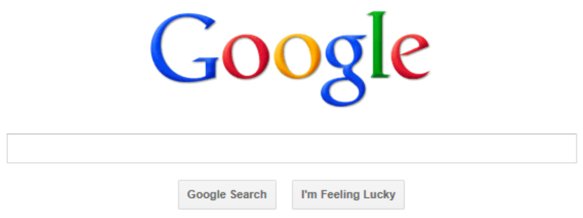
It is worth highlighting this point – it allows programs to talk to each other. We shall explore this idea as it is important to understanding exactly how the client server model works.

Rather than connecting every computer to every other computer on the network individual computers are connected to a local network and the local network has a single access point connecting it to the internet. In this sense the internet is a network of networks.

In order for the network protocol to work, each machine on the network is allocated an IP address. An IP address is a unique 32 bit number made up of four 8 bit numbers separated by a full stop.

For example if you visit http://209.85.227.105/

You will find yourself at the following page...



Since that is (one of) the IP addresses for Google.

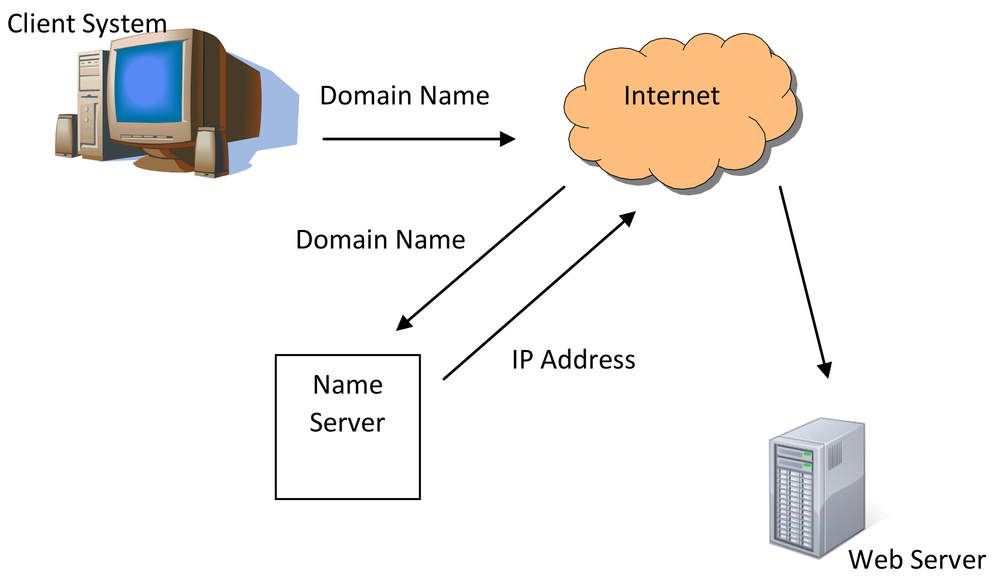
An organisation is often assigned blocks of IP addresses.

For example	www.dmu.ac.uk	has an IP address of	146.227.160.79
	www.cse.dmu.ac.uk	has an address of	146.227.57.2

Note the 146.227 which is common to the DMU machines.

Human beings tend to be very bad at remembering numbers. Since http://209.85.227.105/ is not obviously www.google.com there is also a system for machines to have a name.

The translation of names to IP addresses is handled by software systems called name servers using the Domain Name System (DNS).



Ports

Remember that TCP/IP allows programs on machines to communicate. If the machine has an IP address then how do we identify the specific program on the machine?

Ports are rather like telephone extension numbers. An organisation like DMU might have a single central number for people to call e.g. 0116 255 1551 but each person has an internal extension number for their personal phone e.g. extension 1234. The first number allows you to call the organisation; the second number puts you in touch with the individual.

Ports operate in exactly the same way. The IP address allows you to identify the machine, like the main telephone number 0116 255 1551, the port allows you to identify a specific program running on that machine.

When one machine wants to use a service on another it uses the IP address to identify the machine and makes a connection on a port number.

There is no law that states a specific port must be used for a service however there are certain ports that traditionally provide services.

- 80 HTTP (web pages)
- 21 FTP (File transfers)
- 119 NNTP (Network News Transfer Protocol)
- 443 HTTPS (secure web pages)

If we connect to port 80 then we are attempting to initiate a connection using the Hypertext Transfer Protocol (HTTP)

If we connect to the same machine specifying port 21 we are attempting to make a File Transfer Protocol (FTP) connection.

Each port number is a reference to a different program / service running on the machine.

The World Wide Web

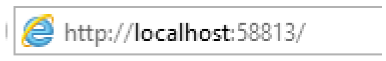
Up until about 1989 the Internet existed quite happily without the World Wide Web. People used a range of software and protocols to transfer data and communicate.

- File Transfer Protocol (FTP) was used if you wanted to share a file
- Telnet was used to log in remotely to control another machine
- Usenet was used to share information via news groups (one of the oldest parts of the internet!)

The World Wide Web added another set of programs and protocols and has become so successful that we tend to use internet and web interchangeably. (Incorrectly)

How a Computer Talks to Itself

When we looked at Visual Studio running the program we saw the following URL.


http://localhost:58813/

We can see that the browser is communicating with the local server, but how exactly is that working?

Remember above we defined TCP/IP as the standard that allows a program on one computer to communicate with a program on another.

But what if we have one computer acting as client and server?

If we are accessing "localhost" the IP address is always http://127.0.0.1

If we have a local client browser and we think in terms of the port it is using the browser will be using http://127.0.0.1:80 The 80 is typically left off the URL but you can put it on if you like!

http://209.85.227.105:80 is exactly the same as http://209.85.227.105/ due to 80 being the default port for HTTP.

If we have a local server we simply set it up on the same machine having the same IP address but using a different port, in the example above 58813, so it's full IP address is http://127.0.0.1:58813.

Implications of the Model

Economic implications

Social/Political implications

Security considerations

Technical considerations