Client and Server (cont.)

A server must always be on or else no-one can access the particular webpages. It needs to always be listening even if it's not doing anything.

The picture is not always this simple. When there's huge traffic to a website (e.g. facebook or google), there's a piece of software somewhere on the internet that decides which (e.g.) facebook server or server farm to send your request to.

HTTP is a connectionless protocol, because it differs from a phone call.
A phone call relies on a connection. Both parties can make and receive calls.
Everything your web browser sees is a response to some request you have made. Servers cannot call you.

The server typically can't tell if a second communication has come from the same client or from a different one.

Client and server setup typically seen as a limitation these days. For example, facebook notifications:

- Sometimes the client program (facebook mobile for example) must keep sending requests many times per second to the server to see if there are any messages. This is a waste of bandwidth.

Over this system, a site can't say "20 users online", because the server has no idea how many people are talking to it.

To fix this, a *session* is set up (using cookies), where there's information retained between requests.

In this (client-server) system, there's no info retained between requests.

DNS is an example of a client-server system that is not http, it's used to translate web addresses (e.g. www.ucc.ie) into IP addresses (e.g. 60.9.156.1).
Impossible to have a single registry of addresses because it would be too big and would need to change too regularly.

> Your request goes out (do you know "www.ucc.ie"?). The first server checks its cache (short-term memory). If it doesn't have it it forwards your request to other servers. When the answer is eventually found, it is communicated back and the servers must tell other servers to stop looking.

Each communication in this setup follows the client-server system.

Another example might be email. It is internet but not WWW.

A typical http request:

- client sends "GET www.ucc.ie"
  - GET is a http command type
  - www.ucc.ie is a URL
- server responds "200 content"
  - content is html
  - 200 is status code ok

HTTP is not a programming language or markup language. There are very few commands in http. The most common ones are GET, POST, DELETE, and HEAD. A really obscure one would be PUT.

POST is for sending data. See below about PUT.

HEAD is like GET but the content isn't sent, just the response code. The payload is empty. It's used for testing.

PUT is like POST but different:

If you go through the ryanair website, when you eventually click make purchase and get the message "do not refresh the page", if you were to refresh you would make your purchase twice (in theory). This a side-effect of using POST twice.

PUT is meant not to have this side-effect, by resisting duplication of the request.

Some common status codes are 200(OK), 404(Not Found) and 403(Forbidden).

URL

Parts of a URL:

- scheme
    - required (e.g. http://)
- userinfo (user:password)
    - optional (rarely used nowadays because it's insecure)

    - used in phishing to get your details (see below)
    - preceeded by "://"
- hostname (stuff.com)
    - required
    - preceeded by @
- port number (8080)
    - optional
    - assumed to be 80 (web server) if it's left out
    - preceeded by ":"
- path (In this case /dir/subdir/file.html)
    - optional
    - separator is always "/" in URLs, regardless of computer
- query (page=3&lan=en)
    - optional
    - communicates data with the request (like POST)
    - preceeded by "?"
    - "&" separates query commands from one another
    - always name=value for each query
- fragment (position)
    - optional
    - preceeded by "#"

http://user:password@stuff.com:8080/dir/subdir/file.html?page=3&lan=en#position

In phishing, the user:password field is filled by a fake address like boi.ie/...@(fake website here)

So you think it's the real link, but it's actually just submitted as a username and then ignored by the website. The @ sign marks the separation.