COMP3016 Web Technologies

- Introduction and Discussion

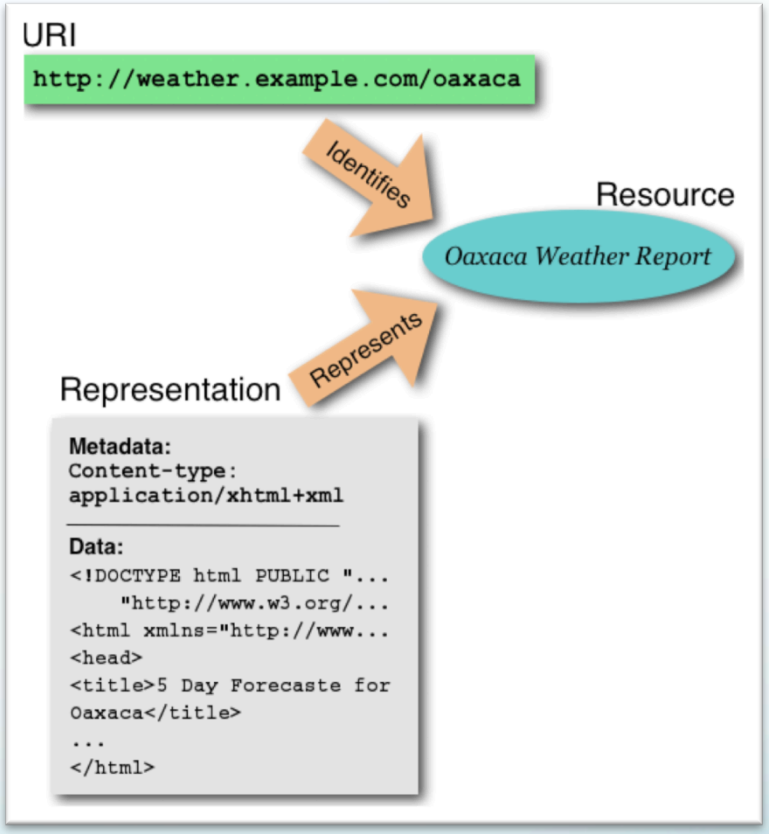
  - What is the Web?
  - What makes it so Webby?
  - What was new about it that we didn't have before?
  - What is the USP of the Web?

How Does the Web Work?

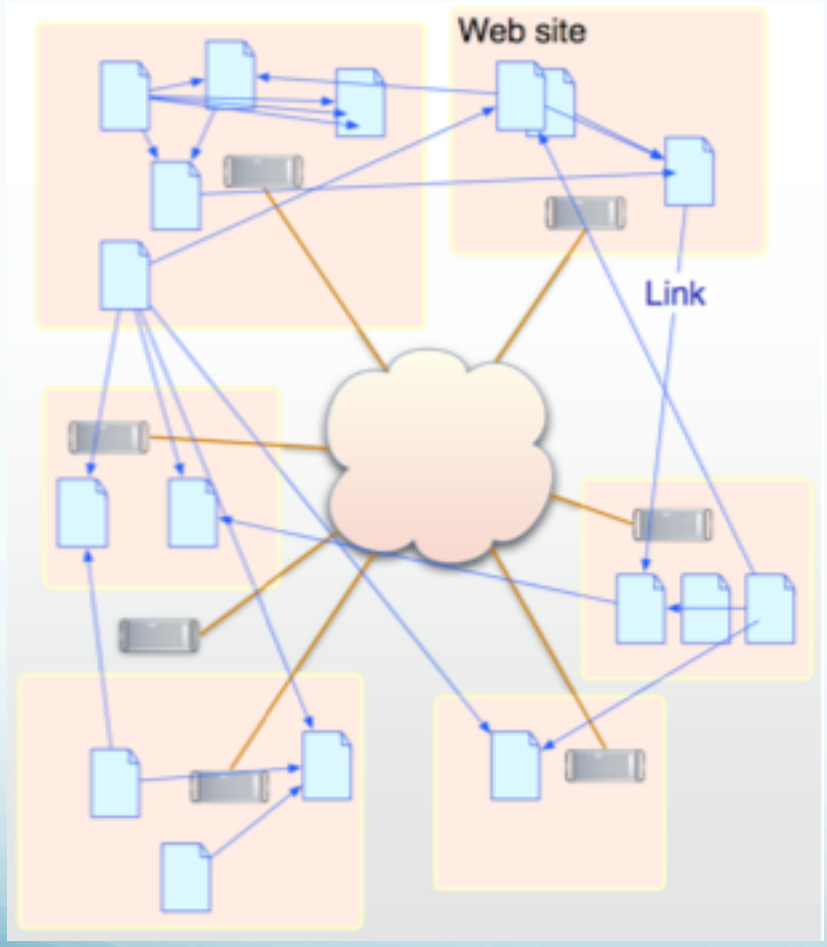
- This man is reading the New York Times on the Web.
- What technology underpins his activity?
- EXERCISE: Brainstorm all the programs, protocols, standards, data formats and TLAs you can think of that contribute to the Web as you use it.

Web Architecture

- Resources are identified by URIs
- Resources have different representations (e.g. HTML, text, PDF)
- Key components of the Web Architecture:
  - Identification
  - Interaction
  - Formats

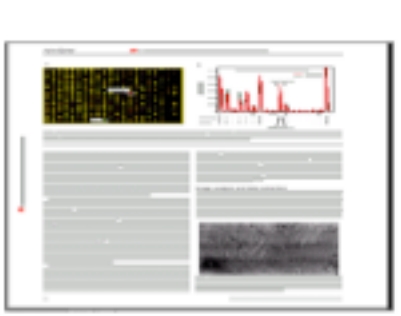


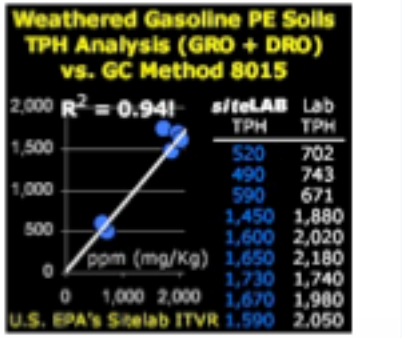
Web Principles: Web of Documents and Data



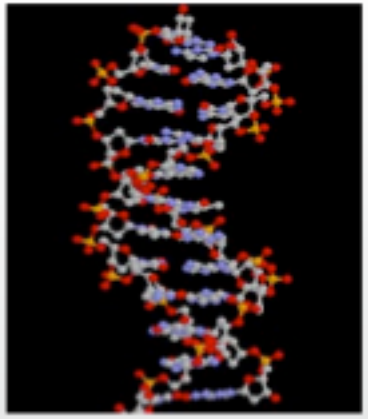
URIs identify *any* resource

- Publications
- Multimedia
- Web data set (XHTML)
- Databases
- Scientific structures
- Workflows
- People





Web data set (XHTML)



URIs and URLs

- network resources are identified by Universal Resource Indicators (URIs)
- The most familiar is the absolute URI known as the HTTP URL:
  - `http-url = “http:” “//” host [“:” port] [abs_path]`
  - `port` defaults to “80”
- examples:
  - `http://users.ecs.soton.ac.uk:80/index.html`
  - `http://users.ecs.soton.ac.uk/index.html`
  - `http://users.ecs.soton.ac.uk`

Web Principles

- All entities of interest, such as information resources, real-world objects, and vocabulary terms should be identified by URI references
- URI references should be *dereferenceable*, meaning that an application can look up a URI over the HTTP protocol and retrieve data about the identified resource (a representation).
- Data should be provided using a standard format (HTML, XML, RDF etc)
- Data should be interlinked with other data

Rules of the Web (2006)

- Use URIs as names for things
- Use HTTP URIs so that people can look up those names.
- When someone looks up a URI, provide useful information, using the standards (HTML, XML, RDF)
- Include links to other URIs, so that they can discover more things.

5 Stars of Linked Data (2010)

★ Available on the web (whatever format), but with an open licence

★★ Available as machine-readable structured data (e.g. excel instead of image scan of a table)

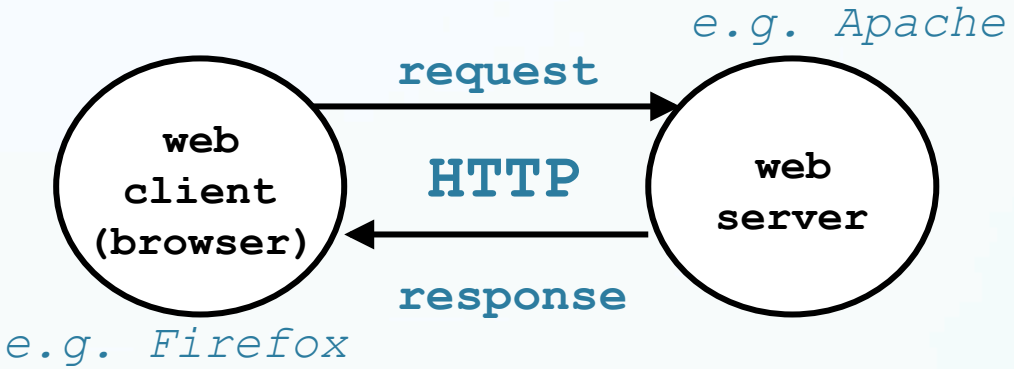
★★★ as (2) plus non-proprietary format (e.g. CSV instead of excel)

★★★★ All the above plus, Use open standards from W3C (RDF and SPARQL) to identify things, so that people can point at your stuff

★★★★★ All the above, plus: Link your data to other people's data to provide context

The Web Experience

- A user clicks on a link in a browser.
- The browser communicates with a web server using HTTP
- The server sends an HTML document back
- The browser displays the document
- The user clicks on another link and activates another URL



Pre Web: File Transfer

- A user typed a host address into a client.
- The client communicated with a file server using File Transfer Protocol (FTP)
- The user typed commands into the client
  - to navigate to the right directory
  - to GET the right file from a DIR listing
  - to specify BINARY or ASCII transfers to make sure that line endings were treated correctly.
- The server sent a PostScript or text document back
- The client **stored the document on the hard disk**
- The user **printed** the document

Pre Web: FTP

FTP commands

```
Remote system type is Windows_NT.
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
11-22-05  03:40PM       <DIR>          MSC
10-20-06  11:55AM       <DIR>          Rob
226 Transfer complete.
ftp> cd MSC
250 CWD command successful.
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
11-22-05  03:40PM       <DIR>          Aaa
11-22-05  03:40PM       <DIR>          Alice
09-06-00  02:31PM             13478006 arc explorer.exe
01-31-00  12:00AM              1441792 archdata.mdb
11-22-05  03:40PM       <DIR>          blaster
11-22-05  03:40PM       <DIR>          courses
07-04-02  04:17PM                12612 Curtin-Aleph500.enz
11-22-05  03:40PM       <DIR>          Distance_education
02-27-02  12:52PM                19456 Doc1.doc
11-22-05  03:40PM       <DIR>          documents
07-30-02  10:50AM                95744 docuxplorer.doc
07-31-03  10:24AM                 1502 error.txt
11-22-05  03:40PM       <DIR>          Examples
02-27-02  10:18AM               105984 ftp-http-protocols.doc
08-04-03  08:01AM             15646208 infotrove_nt_2_15.exe
07-30-02  08:26AM              1018090 InfoTrove_user_manual.pdf
11-22-05  03:40PM       <DIR>          journalism
03-16-01  09:58AM                12835 Library Catalog (CDL).enz
11-22-05  03:40PM       <DIR>          mc1111
11-23-99  11:56AM              3062925 n16e30.exe
04-13-02  12:10PM                81408 PhDProposal.doc
07-05-01  11:04AM                 1956 readme.txt
03-11-02  09:02AM                 5656 sample data.sav
11-22-05  03:40PM       <DIR>          SONY
11-22-05  03:40PM       <DIR>          TMAGIC
05-14-99  12:08PM              3743302 tmagic.zip
02-18-02  10:00AM                13680 topic8-supplement.html
11-22-05  03:40PM       <DIR>          wethu
226 Transfer complete.
ftp> get readme.txt
local: readme.txt remote: readme.txt
200 PORT command successful.
150 Opening ASCII mode data connection for readme.txt(1956 bytes).
WARNING! 60 bare linefeeds received in ASCII mode
File may not have transferred correctly.
226 Transfer complete.
1956 bytes received in 0.01 secs (137.1 kB/s)
ftp>
```

- Pre web interaction was characterised by DOWNLOADING instead of BROWSING.

User types commands directly to server. User prints the file to read its contents.

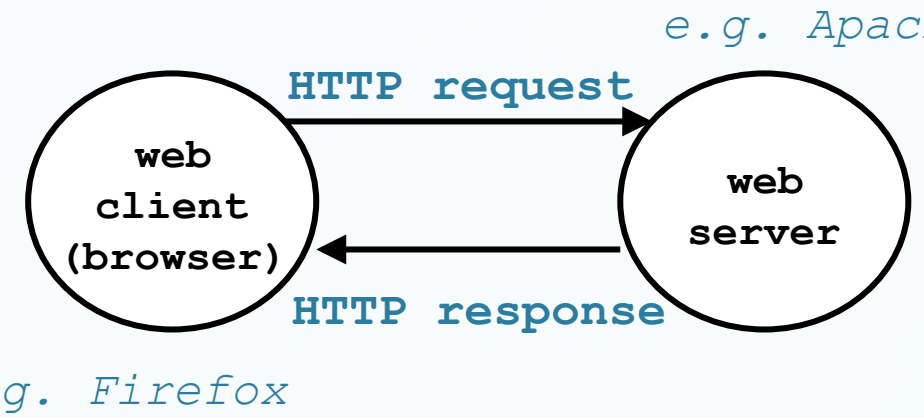
PostScript data

```
%%BeginSetup
%%Feature: *Resolution 600dpi
TeXDict begin

%%EndSetup
%%Page: 1 1^M
/Times-Roman findfont
12 scalefont
setfont
newpath
131 562 moveto
(Hello, world!) show
131 542 moveto
(Strickly speaking, \
wavelets are topic of pure mathematics!!!!) show
/Helvetica findfont
16 scalefont
setfont
131 522 moveto
(Does the line above look better than the one below?) show
1 0 bop 1339 271 a FH(W)-15 b(A)g(VELETS)44 b(F)l(OR)g(KIDS)1470
391 y FG(A)37 b(T)-9 b(utorial)36 b(In)m(tro)s(duction)1986
812 y FF(By)992 932 y(Brani)i(Vid)n(ak)n(o)n(vic)201
b FE(and)162 b FF(Peter)37 b(Mueller)1711 1052 y FD(Duke)e(University)
"kidsE.ps" [unix] 3414L, 231145C written                    3366,1        99%
```

HTTP Protocol

- An HTTP message is
  - Request *or*
  - Response



HTTP message = Request or Status line
Message-header lines
blank line
Message body

message-header = field-name : field value

message-body = *any sequence of bytes e.g.* HTML file

HTTP/1.1 requests

```
Request = Method SP Request-URI SP HTTP-VERSION CRLF
          *(general-header | request-header | entity header)
          CRLF
          [ message-body ]
```

- Method: tells the server what operation to perform
  - GET: retrieve contents of resource
  - PUT: store contents in resource
- Request-URI: identifies the resource to manipulate
  - data file (HTML), executable file (CGI)
- headers: parameterize the method
  - Accept-Language: en-us
  - User-Agent: Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)
- message-body: text characters

HTTP/1.1 responses

```
Response = HTTP-Version SP Status-Code SP Reason-Phrase CRLF
           *(general-header | response-header | entity header)
           CRLF
           [ message-body ]
```

- Status code: 3-digit number
- Reason-Phrase: explanation of status code
- headers: parameterize the response
  - Date: Thu, 22 Jul 1999 23:42:18 GMT
  - Server: Apache/1.2.5 BSDI3.0-PHP/FI-2.0
  - Content-Type: text/html
- message-body:
  - file

Example HTTP/1.1 conversation

```
sparrow> telnet users.ecs.soton.ac.uk 80
Connected to users.ecs.soton.ac.uk.
Escape character is '^]'.
```

Request sent by client

```
GET /lac/test.html HTTP/1.1
Host: users.ecs.soton.ac.uk
```

Response sent by server

```
HTTP/1.1 200 OK
Date: Thu, 22 Jul 1999 03:37:04 GMT
Server: Apache/1.3.3 Ben-SSL/1.28 (Unix)
Last-Modified: Thu, 22 Jul 1999 03:33:21 GMT
ETag: "48bb2-4f-37969101"
Accept-Ranges: bytes
Content-Length: 79
Content-Type: text/html

<html>
<head><title>Test page</title></head>
<body><h1>Test page</h1>
</html>
```

Another HTTP/1.1 conversation

```
sparrow> telnet www.google.com 80
Connected to www.google.com.
Escape character is '^]'.
```

Request sent by client:

```
GET /search?q=doctor-who HTTP/1.0
Host: sparrow.ecs.soton.ac.uk
```

Response sent by server:

```
HTTP/1.0 200 OK^M
Cache-Control: private, max-age=0^M
Date: Sun, 05 Oct 2008 16:34:28 GMT^M
Expires: -1^M
Content-Type: text/html; charset=ISO-8859-1^M
domain=.google.com^M
Server: gws^M
Connection: Close^M

<!doctype html><head><meta http-equiv=content-type
content="text/html; charset=ISO-8859-1"><title>doctor-
who - Google Search</title><style>body
{background:#fff; color:#000;margin:3px 8px}
#gbar{height:22px;padding-left:2px}.gbh,
```

GET

- Retrieves the information identified by the request URI.
  - static content (HTML file)
  - dynamic content produced by CGI program
    - passes arguments to CGI program in URI
- Can also act as a conditional retrieve when certain request headers are present:
  - If-Modified-Since
  - If-Unmodified-Since
  - If-Match
  - If-None-Match
  - If-Range
- Conditional GETs useful for caching

HEAD

- Returns same response header as a GET request would have...
- But doesn't actually carry out the request.
  - Some servers don't implement this properly.
  - example: espn.com
- Useful for applications that
  - check for valid and broken links in Web pages.
  - check Web pages for modifications.

POST

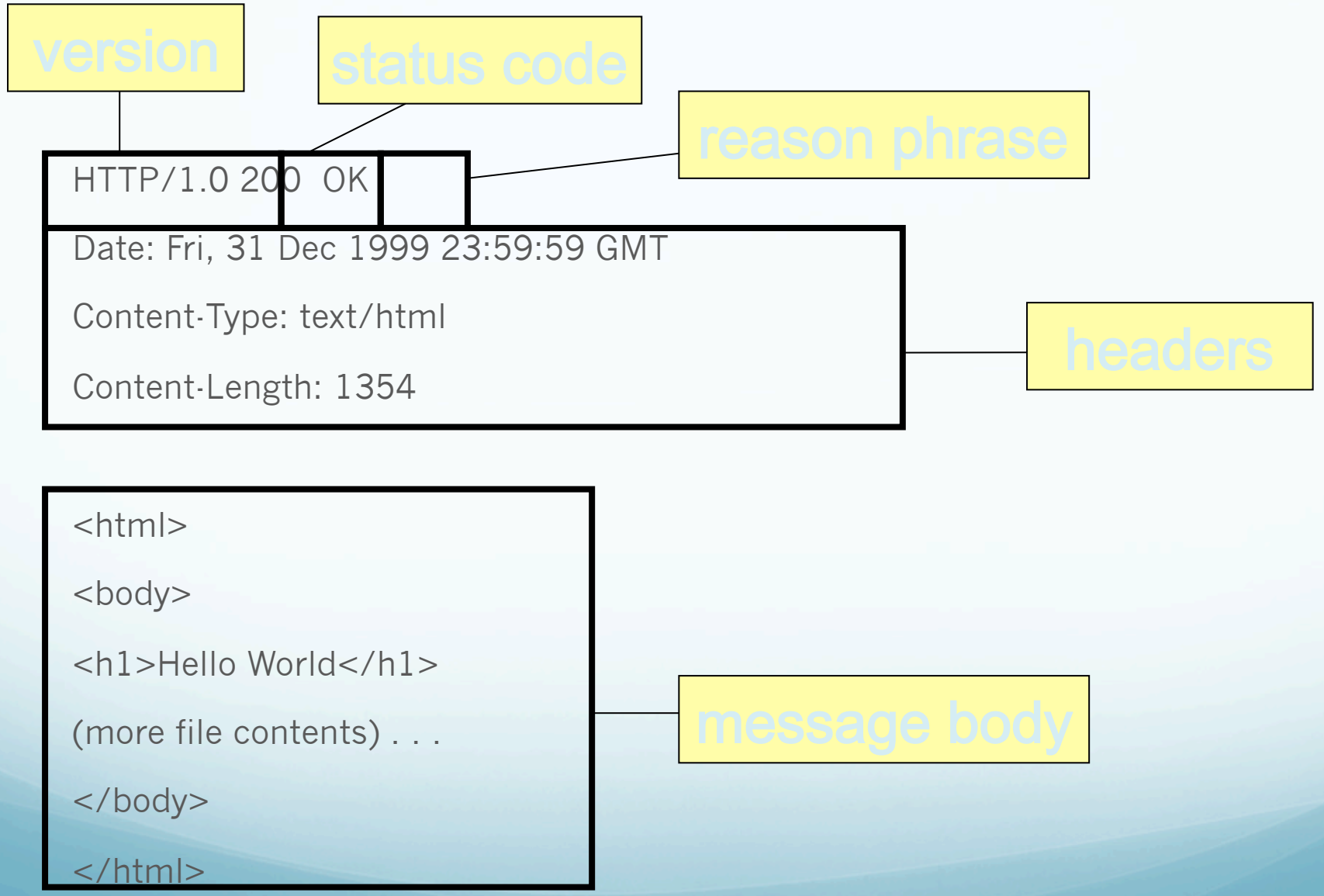
- Another technique for producing dynamic content.
- Executes program identified in request URI (the CGI program).
- Passes arguments to CGI program in the message body
  - unlike GET, which passes the arguments in the URI itself.
- Responds with output of the CGI program.

Example POST request

```
POST /search.cgi HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg,
 image/pjpeg, application/vnd.ms-excel, application/msword,
 application/vnd.ms-powerpoint, */*
Referer: http://www.ecs.soton.ac.uk/~lac/form.html
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)
Host: sparrow.ecs.soton.ac.uk
Content-Length: 19

first=les&last=carr
```

Response Example



Status Codes in Responses

- The status code is a three-digit integer, and the first digit identifies the general category of response:
  - 1xx indicates an informational message
  - 2xx indicates success of some kind
  - 3xx redirects the client to another URL
  - 4xx indicates an error on the client's part
    - Yes, the system blames it on the client if a resource is not found (i.e., 404)
  - 5xx indicates an error on the server's part

Status Codes 2xx

Status codes 2xx – Success

- The action was successfully received, understood, and accepted
- Usually upon success a status code 200 and a message OK are sent
- This is the default

More 2xx Codes

- 201 (Created)
  - Location header gives the URL
- 202 (Accepted)
  - Processing is not yet complete
- 204 (No Content)
  - Browser should keep displaying previous document

Status Codes 3xx

Status codes 3xx – Redirection

- Further action must be taken in order to complete the request
- The client is redirected to get the resource from another URL

More 3xx Codes

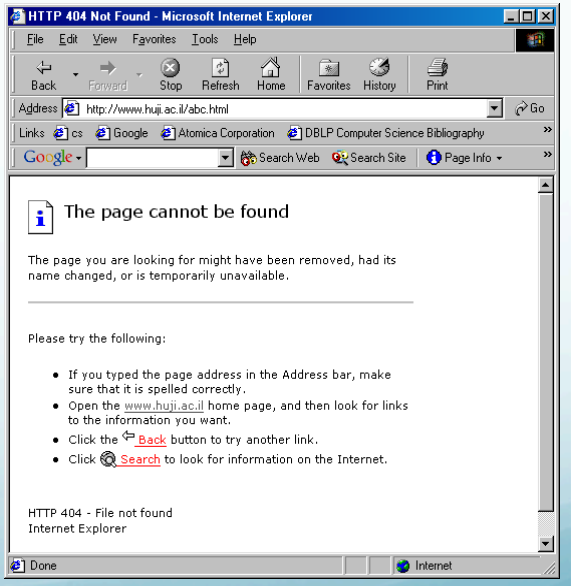
- 301 – Moved Permanently
  - The new URL is given in the Location header
  - Browsers should automatically follow the link to the new URL
- 302 – Moved Temporarily
  - Similar to 301, except that the URL given in the Location header is temporary
- 303 – See Other
  - Similar to 301 and 302, except that if the original request was POST, the new document (given in the Location header) should be retrieved with GET

Status Codes 4xx

Status codes 4xx – Client error

- The request contains bad syntax or cannot be fulfilled

404 File not found



4xx Codes

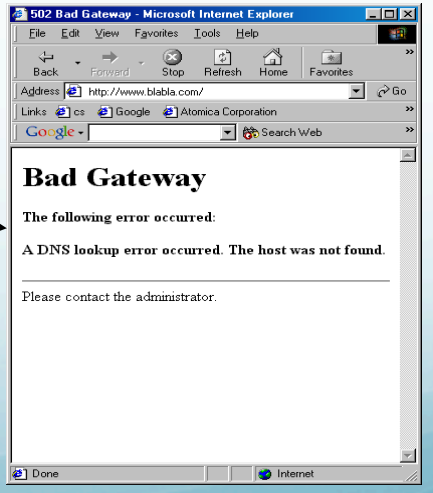
- 400 – Bad Request
  - Syntax error in the request
- 401 – Unauthorized
- 403 – Forbidden
  - “permission denied” to the server to access the page
- 404 – Not Found

Status Codes 5xx

Status codes 5xx – Server error

- The server failed to fulfill an apparently valid request

For example, 502 Bad gateway



5xx Codes

- 500 – Internal Server Error
- 501 – Not Implemented
- 502 – Bad Gateway
- 503 – Service Unavailable
  - The response may include a Retry-After header to indicate when the client might try again
- 505 – HTTP Version Not Supported
  - New in HTTP 1.1

Learning

Mary, Mary, quite contrary,

How does your garden grow?

With silver bells and cockle shells

And pretty maids all in a row

The Web Architecture For Children

TimBL, TimBL, very nimble,

How does your linked Web grow?

With URLs and HTMLs

And GET & POSTs all in a row