



Hypertext Transfer Protocol (HTTP)



Contents

- HTTP and Client-Server model
- DNS and URL
- HTTP Request and response
- Character encoding
- Media type

HTTP

■ HTTP = Hypertext Transfer Protocol

- Application-level protocol for distributed, collaborative, hypermedia information systems.
- Used for retrieving inter-linked resources led to the establishment of the World Wide Web.
- HTTP is a request/response standard of a client and a server.
- Client is the end-user using web browser
- Server is the web site.
- Between client and server there may be several intermediaries: proxies, gateways, and tunnels.
- Typically, an HTTP client initiates a request to server over TCP
- An HTTP server listens at a particular port (80 by default) waits for the request messages from clients.
- Standardization: RFC2616 (HTTP 1.1):

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

Examples of Web server and client

- Web server

- IIS: windows
 - Apache HTTP server (Apache): windows, linux

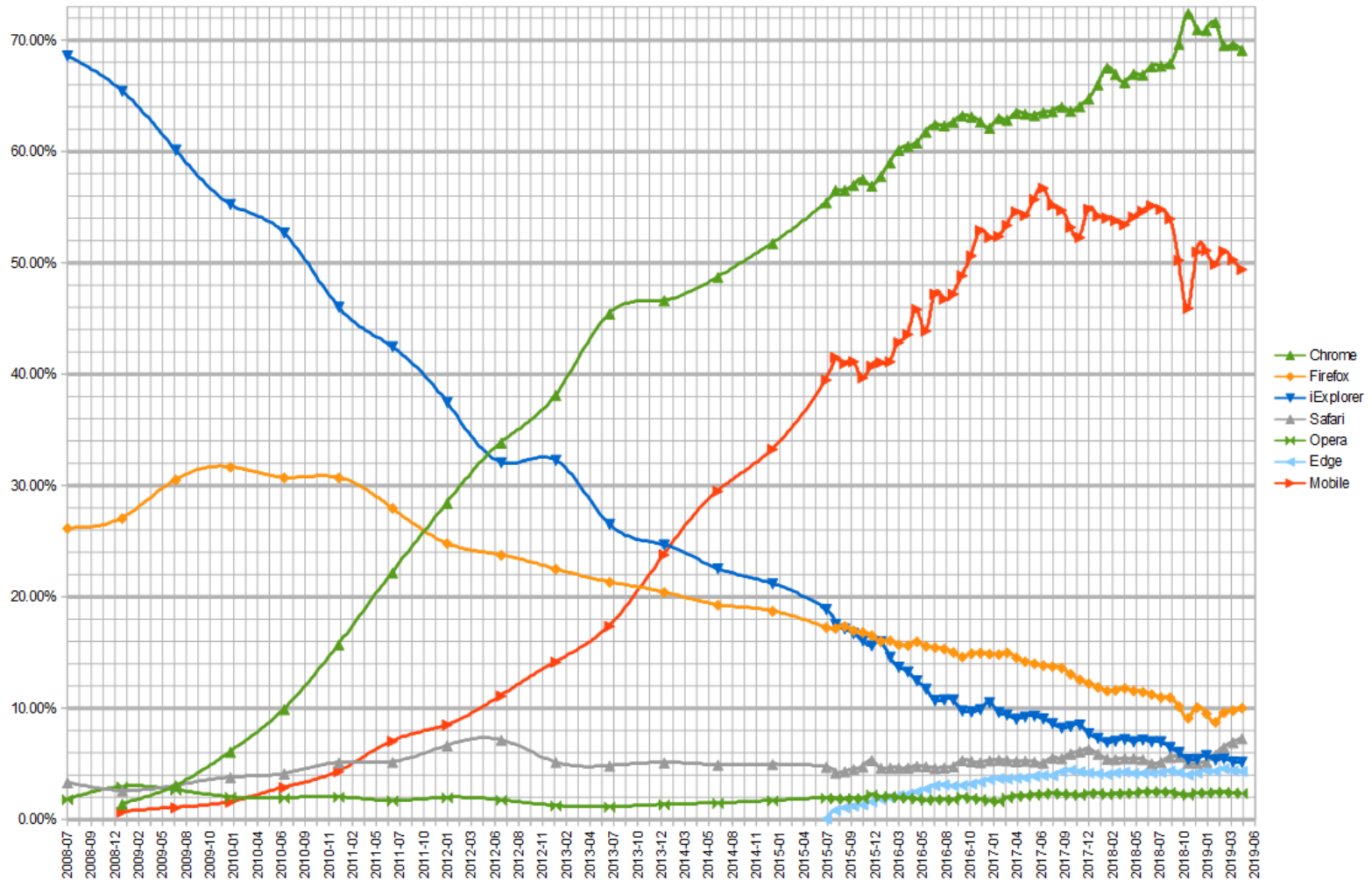
- Web client (Browser)

- Internet Explorer:
 - Free with Windows license, started in 1995
 - Support Windows
 - Mozilla Firefox from Mozilla Corporation
 - free browser,
 - Support: Linux, Mac OS X, Microsoft Windows, and many other Unix-like operating system
 - **Safari:** developed by [Apple Inc.](#)
 - Support Mac OS, Windows, iPhone OS
 - Chrome: from Google
 - Free
 - Support

Usage share of web browsers

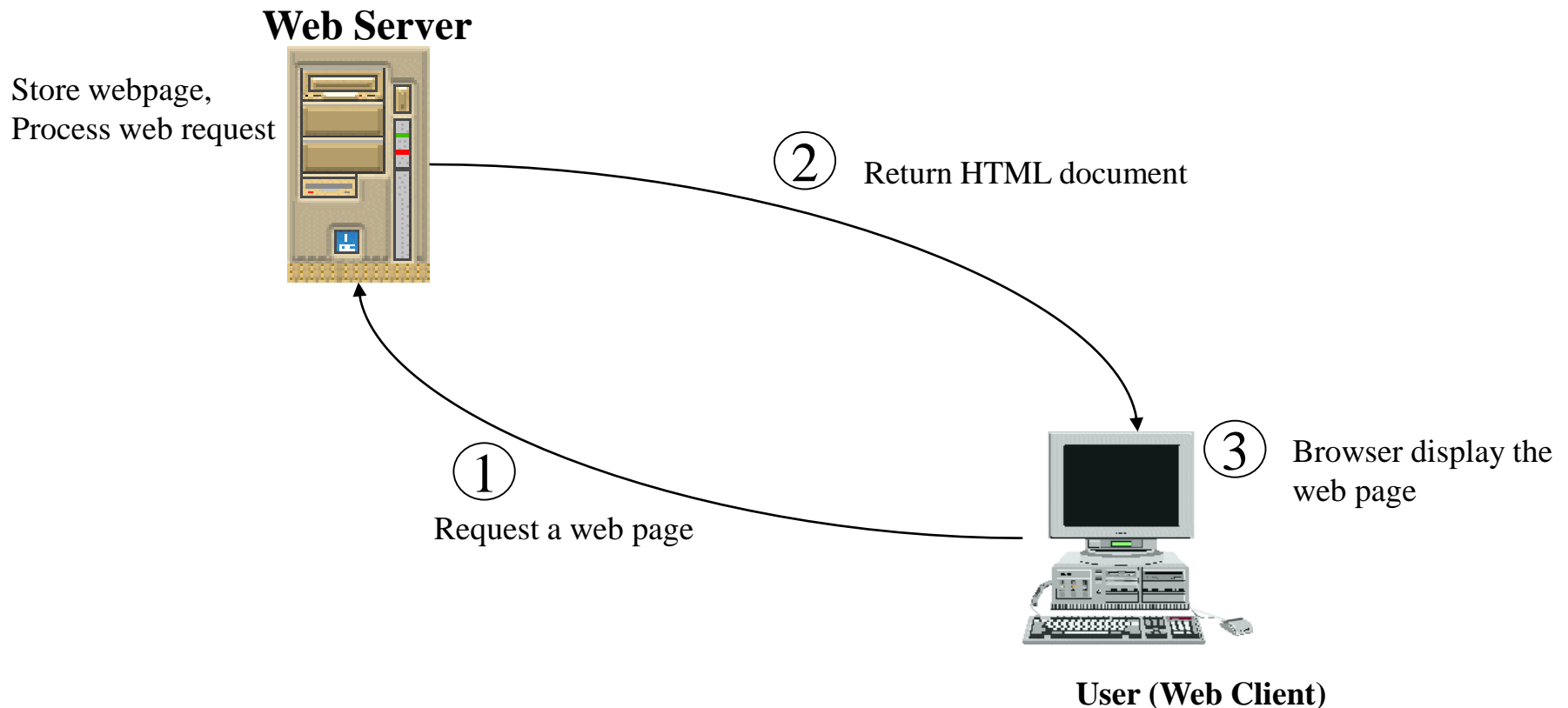
■ Wikipedia: http://en.wikipedia.org/wiki/Usage_share_of_web_browsers

Usage share of browsers (source StatCounter)

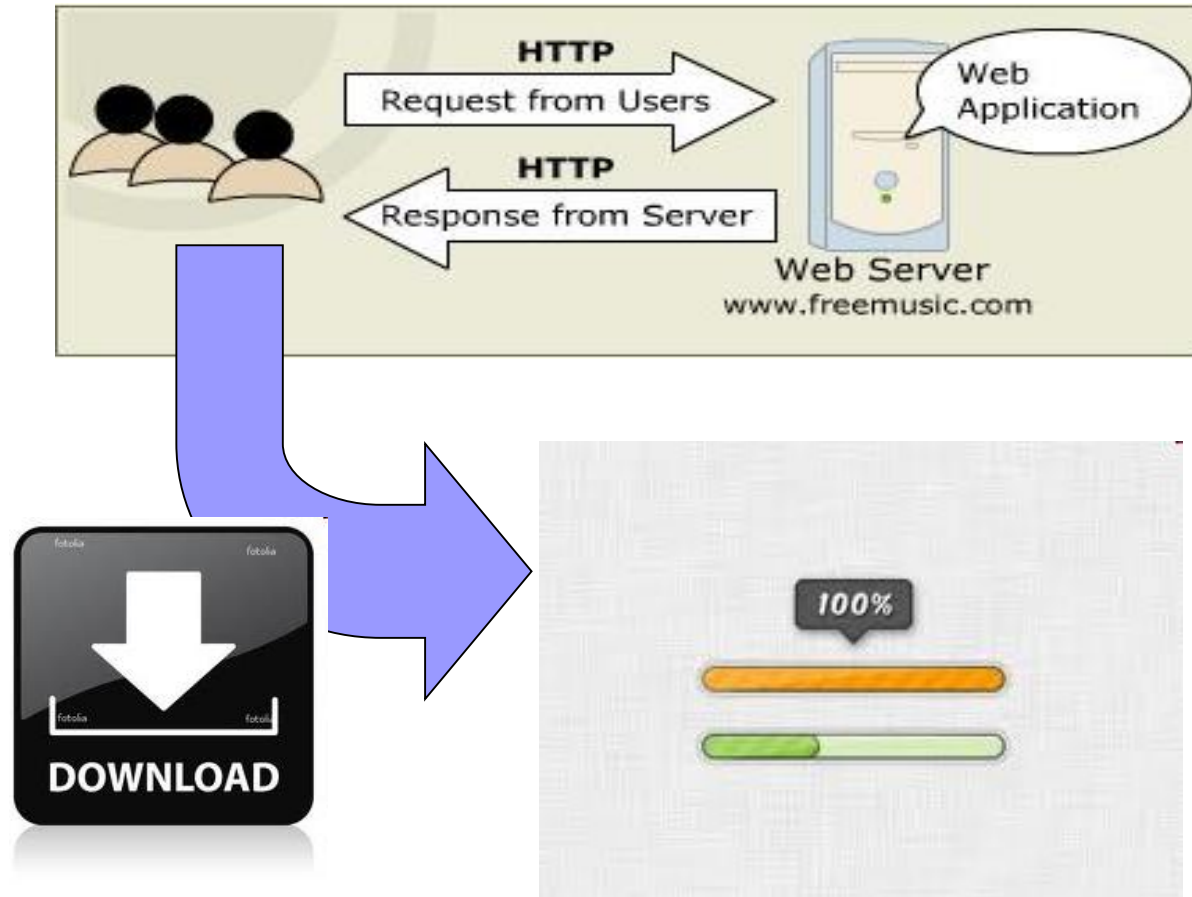


Server Client model

- Server client model of Web system
- HTTP protocol between Client, Server

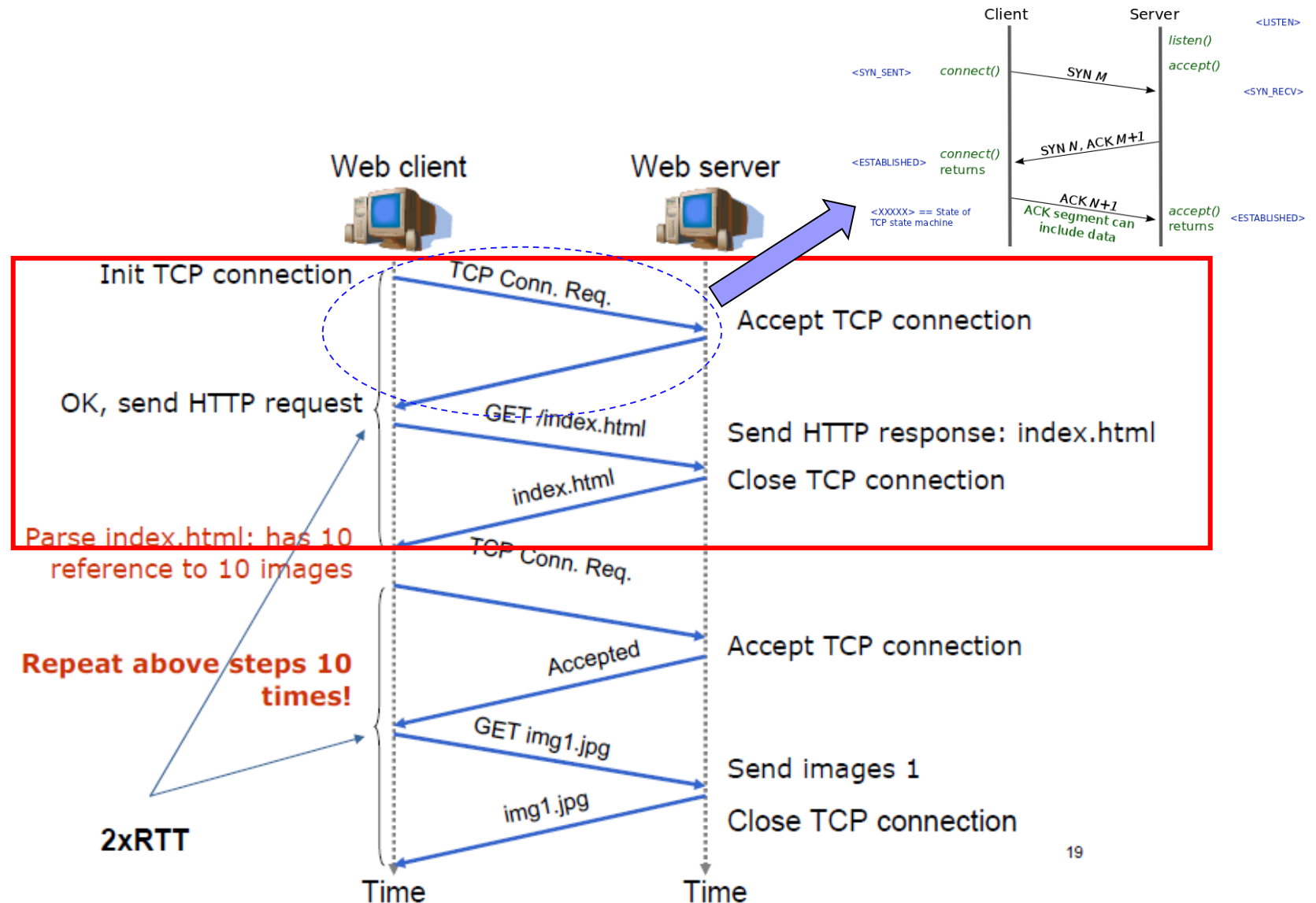


HTTP request and response



How browser know to display download progress bar for 1 HTTP request/response?

HTTP request and response v.1.0



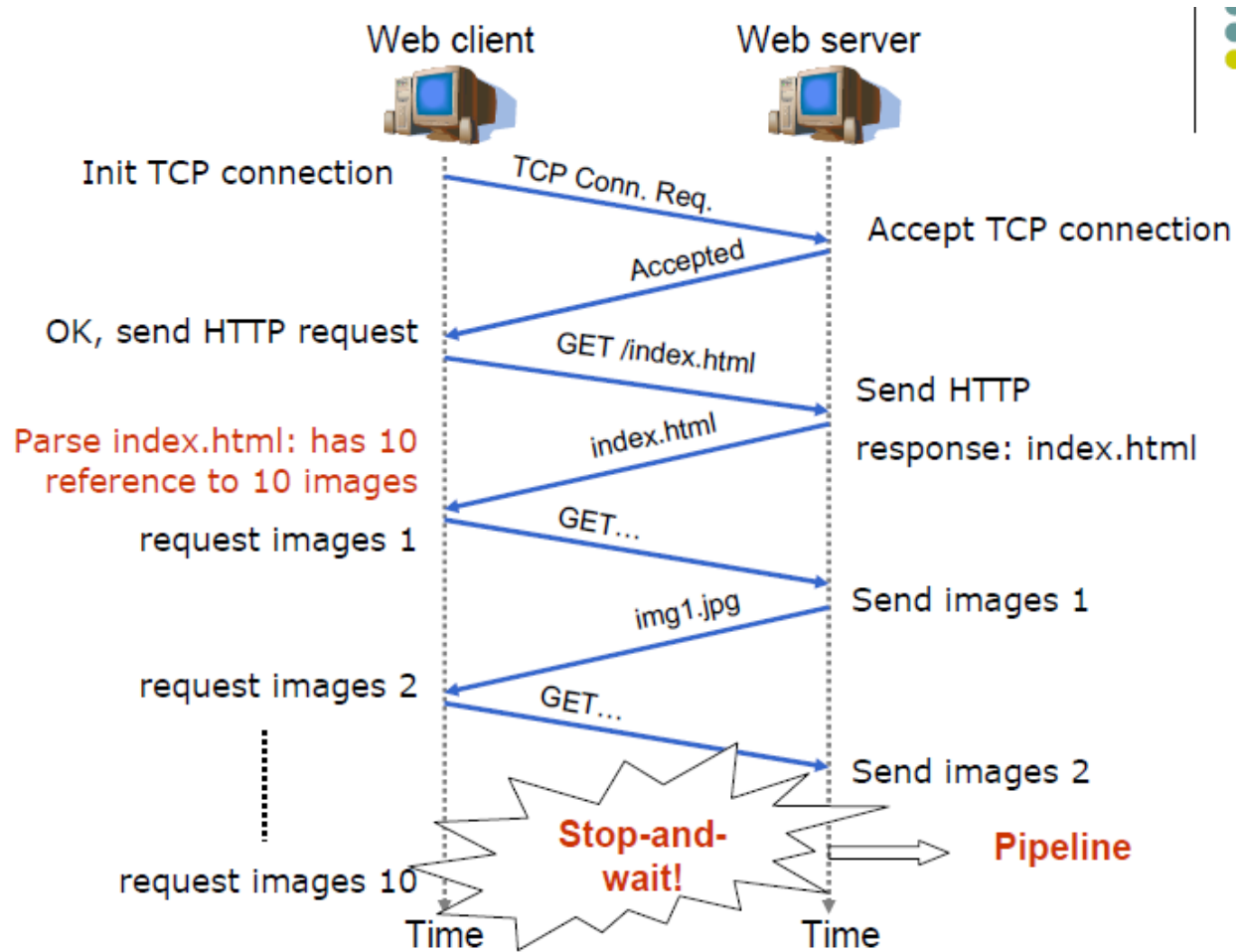
HTTP request and response

■ Procedure:

- HTTP Client (Web Browsers) opens the connection
- HTTP Client sends the request message to an HTTP server asking for resource.
- The server returns the response message with the request resource.
- Once the resource is delivered, Server closes the connection.

- HTTP doesn't store any connection information and ➔ stateless protocol.
- In HTTP Connection last for only one transaction. A transaction consists a of several request-response pairs.
- The default port is 80.

HTTP request and response v.1.1

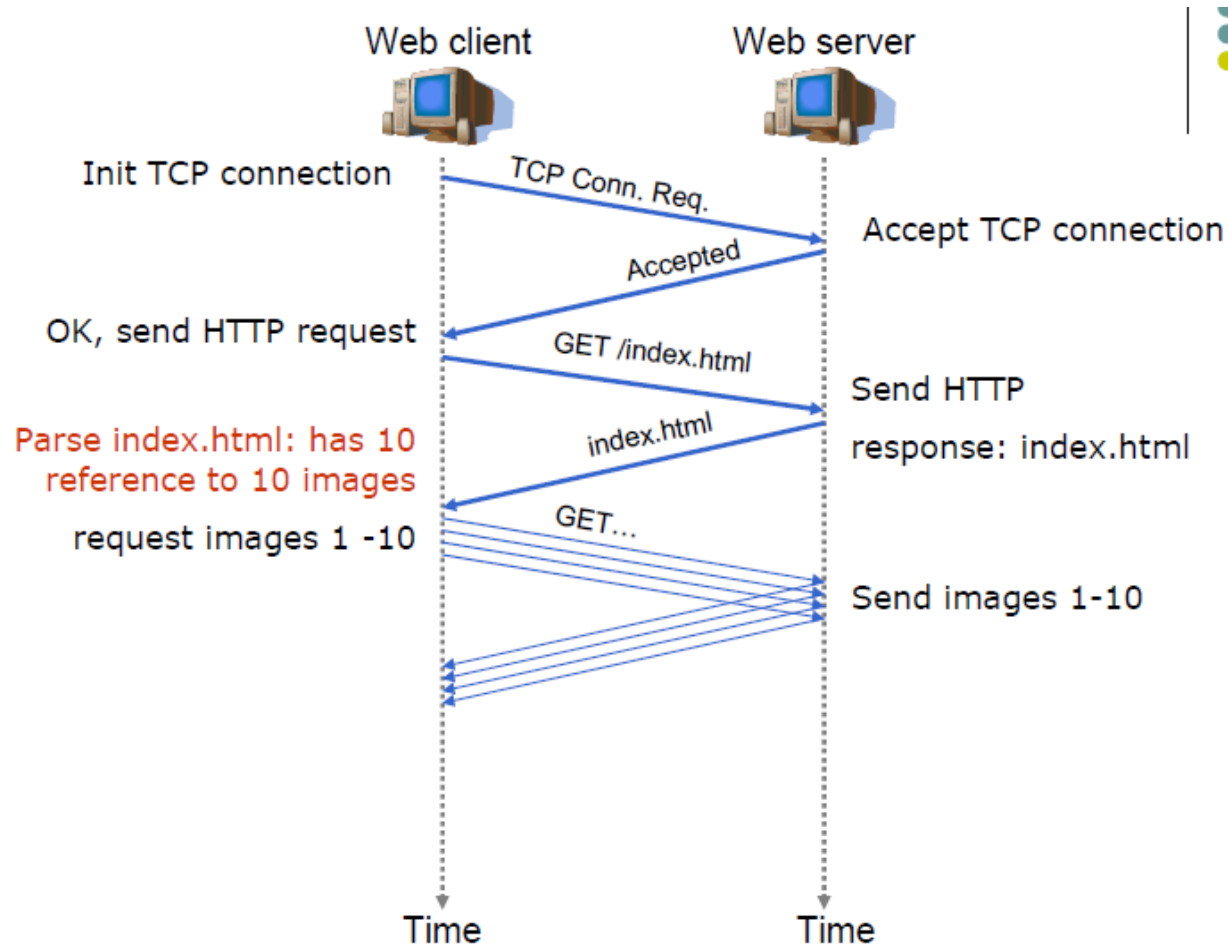


RFC2616: HTTP/1.1 servers SHOULD maintain persistent connections and use TCP's flow control mechanisms to resolve temporary overloads, rather than terminating connections with the expectation that clients will retry

HTTP 1.1

- This is the HTTP version currently in common use.
- Introduced critical performance optimizations and feature enhancements
 - persistent and pipelined connections
 - compression/decompression
 - content negotiations
 - virtual hosting (a server with a single IP Address hosting multiple domains)
 - faster response and great bandwidth savings by adding cache support.
- Many performance optimization in “black art”
 - Connection management - particularly knowing when and how to close connections - is one of the practical point of HTTP

HTTP request and response v.1.1 pipeline



RFC2616: A client that supports persistent connections MAY "pipeline" its requests (i.e., send multiple requests without waiting for each response). A server MUST send its responses to those requests in the same order that the requests were received.

HTTP Message Structure

- By free text as many application level protocols
- Fields by keyword, separated by line – CRLF (Carriage Return & Line Feed)
- Zero or more header lines CRLF
- A blank line ie. a CRLF
- An optional message body like file, query data or query output.

➔ Chrome development tool to debug HTTP messages sent & received within browser

Chrome development tool

← → ↻ w3.org/Protocols/rfc2616/rfc2616-sec8.html ☆ New

part of [Hypertext Transfer Protocol -- HTTP/1.1](#)
RFC 2616 Fielding, et al.

8 Connections

8.1 Persistent Connections

8.1.1 Purpose

Prior to persistent connections, a separate TCP connection was established to fetch each URL, increasing the load on HTTP servers and causing congestion on the Internet. The use of inline images and other associated data often require a client to make multiple requests of the same server in a short amount of time. Analysis of these performance problems and results from a prototype implementation are available [\[26\]](#) [\[30\]](#).

Implementation experience and measurements of actual HTTP/1.1 (RFC 2068) implementations show good results [\[39\]](#). Alternatives have also been explored, for example, T/TCP [\[27\]](#).

Persistent HTTP connections have a number of advantages:

- By opening and closing fewer TCP connections in routers and hosts (clients, servers, proxies, tunnels, or caches), and memory used for blocks can be saved in hosts.

The screenshot shows the Chrome DevTools interface with the Network tab selected. The top toolbar includes buttons for 'Elements', 'Console', 'Sources', 'Network', 'Performance', 'Memory', 'Application', and 'Security'. The Network tab has a filter set to 'All' and a checkbox for 'Only show requests with SameSite issues'. A timeline at the top shows a request starting at 100 ms and ending at 800 ms. The selected request is 'rfc2616-sec8.html'. The 'Headers' sub-tab is active, showing the following details:

- General**
 - Request URL: <https://www.w3.org/Protocols/rfc2616/rfc2616-sec8.html>
 - Request Method: GET
 - Status Code: 200
 - Remote Address: 128.30.52.100:443
 - Referrer Policy: no-referrer-when-downgrade
- Response Headers**
 - accept-ranges: bytes
 - cache-control: max-age=21600
 - content-encoding: gzip
 - content-length: 5911
 - content-security-policy: upgrade-insecure-requests
 - content-type: text/html; charset=iso-8859-1
 - date: Wed, 18 Mar 2020 02:59:58 GMT
 - etag: "47fc-3e3073913b100-gzip"
 - expires: Wed, 18 Mar 2020 08:59:58 GMT

HTTP Request Message

The diagram illustrates the structure of an HTTP Request Message. It consists of a request line, header lines, and a body (indicated by CR, LF). The request line is annotated with a red arrow pointing to the text 'GET /dccn/index.html HTTP/1.1'. The header lines are annotated with a red bracket and the text 'header lines'. The body is annotated with a red arrow pointing to the text '(extra carriage return, line feed)' and the text 'indicates end of message'.

request line
(GET, POST,
HEAD commands)

GET /dccn/index.html HTTP/1.1

header
lines

Host: www.it-hut.edu.vn
User-agent: Mozilla/4.0
Connection: close
Accept-language: en-us

CR, LF

(extra carriage return, line feed)

indicates end
of message

HTTP Request Message: initial line

- Initial line has three parts, separated by spaces:
 - An HTTP Method Name (HTTP keyword)
 - The local path of the requested resource.
 - The version of HTTP being used.
- Example of initial line:

GET	/path/to/file/index.html	HTTP/1.1
Method	URI	HTTP version

- Methods: GET, POST
- The HTTP version always takes the form "**HTTP/x.x**", uppercase.

GET and POST methods

- GET method used for getting information:
 - document,
 - a simple database query
- Parameter of GET is seen in the URL
 - Ex: <http://www.google.co.uk/search?hl=en&q=java&meta=>
- POST method is used when submitting information
 - credit card number,
 - information to be saved in the database.
 - Data is included in the body of the request
- Data send using POST is not visible to the client and there is not limit on amount of data being sent.

HTTP Response Message: initial line

- The initial response line, called the status line, has three parts separated by spaces:
 - The version of HTTP being used.
 - A response status code that gives the result of the request.
 - An English reason phrase describing the status code.
- Examples:
 - HTTP/1.0 200 OK
 - HTTP/1.0 404 Not Found

HTTP Response Message

The diagram illustrates the structure of an HTTP response message. It consists of three main parts: a status line, header lines, and data. Red arrows point from descriptive labels to each part of the message.

status line
(protocol
status code
status phrase)

header lines

**data, e.g.,
requested
HTML file**

```
HTTP/1.1 200 OK
Connection close
Date: Tue, 16 Mar 2008 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 15 Mar 2008 .....
Content-Length: 8990
Content-Type: text/html

data data data data data ...
```

HTTP Response codes

200 OK

- request succeeded, requested object later in this message

301 Moved Permanently

- requested object moved, new location specified later in this message (Location:)

400 Bad Request

- request message not understood by server

404 Not Found

- requested document not found on this server

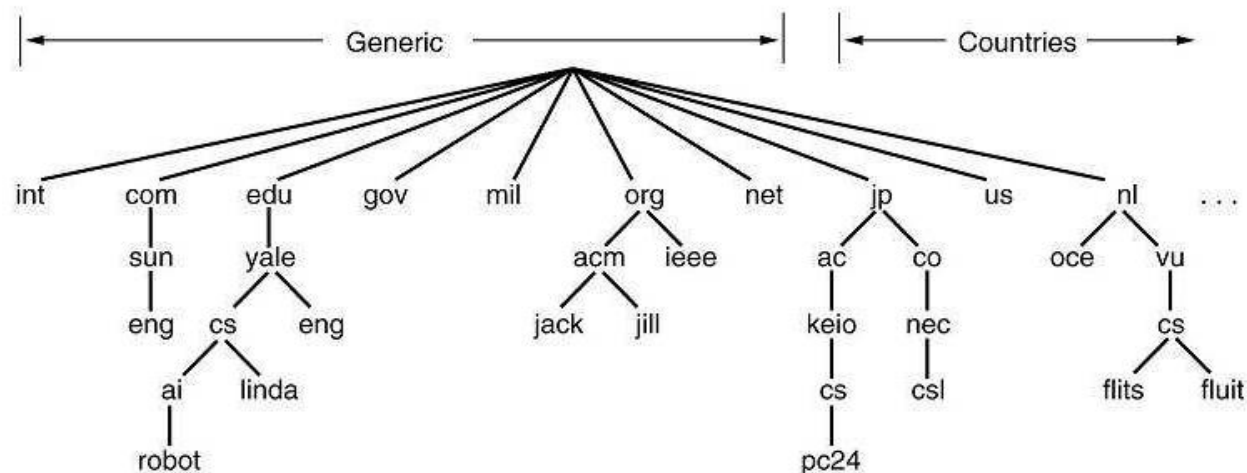
505 HTTP Version Not Supported

Other methods

- <https://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9> - 9 Method Definitions
 - GET
 - POST
 - HEAD: identical to GET except that the server MUST NOT return a message-body in the response
 - PUT: requests that the enclosed entity be stored under the supplied Request-URI
 - DELETE: requests that the origin server delete the resource identified by the Request-URI
 - TRACE: (mostly for debugging) to invoke a remote, application-layer loop-back of the request message
 - CONNECT: use with a proxy that can dynamically switch to being a tunnel (e.g. SSL tunneling)

Domain name

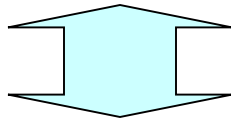
- The Domain Name System is a hierarchical naming system for computers, services, or any resource participating in the Internet
- Example of domain name
 - www.keio.ac.jp
 - www.hedspi.hut.edu.vn
 - .hut.edu.vn



Domain name

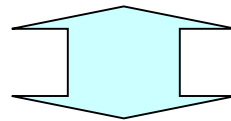
- Domain Name Service (DNS) map a domain name with an IP address
- Domain name
 - Variable length
 - Easy to memory by human being
 - Independent of geographical location of a machine
- IP address
 - Fixed length
 - Easy to be processed by computer
 - Related to routing matter

203.162.7.194



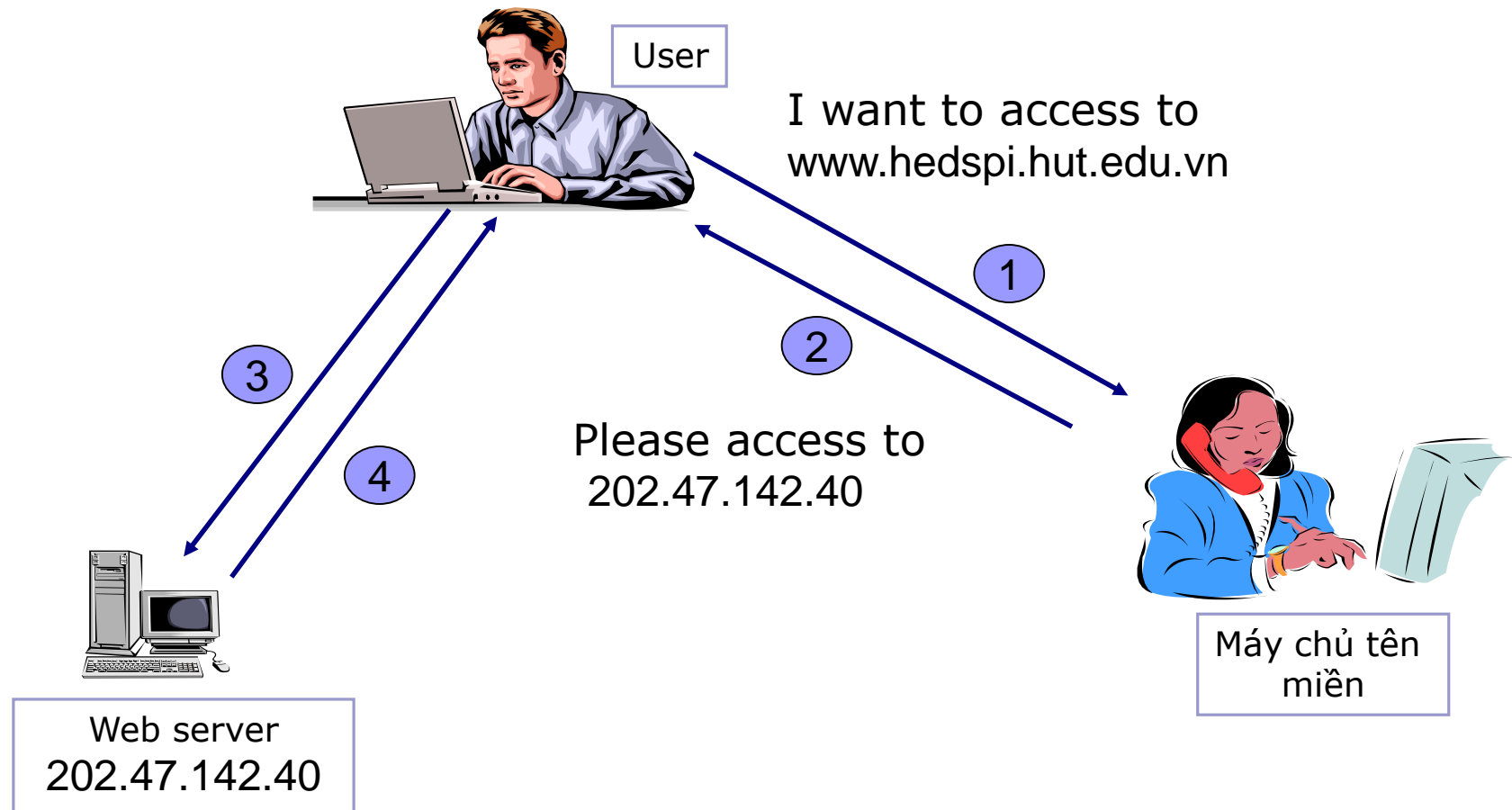
www.hut.edu.vn

www.hedsapi.hut.edu.vn



202.47.142.40

Example of DNS in web system



URL

- Uniform Resource Locator (URL) is a type of Uniform Resource Identifier (URI) that specifies where an identified resource is available and the mechanism for retrieving it
- Format:
 - protocol: name of site/main document#fragment identifier
 - Ex: <http://www.fit.hut.edu.vn/nhansu.htm>
- Two types of URLs:
 - Absolute URL: full Internet address including the protocol, network location, and optional path and file name.
 - Ex: <http://www.microsoft.com>
 - Relative URL: URL with one or more of its parts missing. Browsers take the missing information from the page containing the URL.
 - Ex: index.htm.

Character encoding

■ Character encoding

- Mapping of sequence of characters and a with something else for facilitating the transmission of data or storage in computers
 - sequence of natural numbers, octets or electrical pulses

■ Unicode

- a computing industry standard allowing computers to consistently represent and manipulate text expressed in most of the world's writing systems.
- More than 100,000 characters

■ In Japan, 4 different encodings (Unicode, ISO-2022-JP, EUC-JP, Shift-JIS).

- Some web pages do not have the encoding specification → web browsers must guess the right encoding for such pages.

Character encoding

- Unicode defines two mapping methods:
 - *Unicode Transformation Format (UTF)* encodings
 - UTF-8: 8 bits in one code value
 - UTF-16: 16 bit in one code value
 - UTF-32: 32 bit in one code value
 - *Universal Character Set (UCS)* encodings
 - UCS-2 is an obsolete subset of UTF-16;
 - UCS-4 and UTF-32 are functionally equivalent.

Character Encoding

- Many character encoding standards, such as ISO 8859 series, the encoding is straightforwardly related to the scalar position of the characters in the coded character set.
 - Ex: letter A in the ISO 8859-1 is 65th character in coded set → is encoded by 65.
- For Unicode,
 - there isn't a trivial, one-to-one mapping coded character set value \leftrightarrow encoded value.
 - There are a number of ways of encoding the same character.
 - For example, the letter à can be represented by two bytes in one encoding and four bytes in another.
- UTF-8 uses
 - 1 byte for characters in the ASCII set,
 - 2 bytes for characters in several more alphabetic blocks,
 - 3 bytes for the rest of the BMP.
 - 4 bytes for supplementary characters.
- UTF-16 uses
 - 2 bytes for any character in the BMP,
 - 4 bytes for supplementary characters.
- UTF-32 uses 4 bytes for all characters.

Media type

- Multimedia Internet MEdia : MIME
 - Text, Image, Audio, Video
- Transmission of media type in HTTP message
 - HTTP requires that data be transmitted in the context of **e-mail-like** messages, even though the data may not actually be e-mail.

