cphbusiness

COPENHAGEN BUSINESS ACADEMY

HTTP

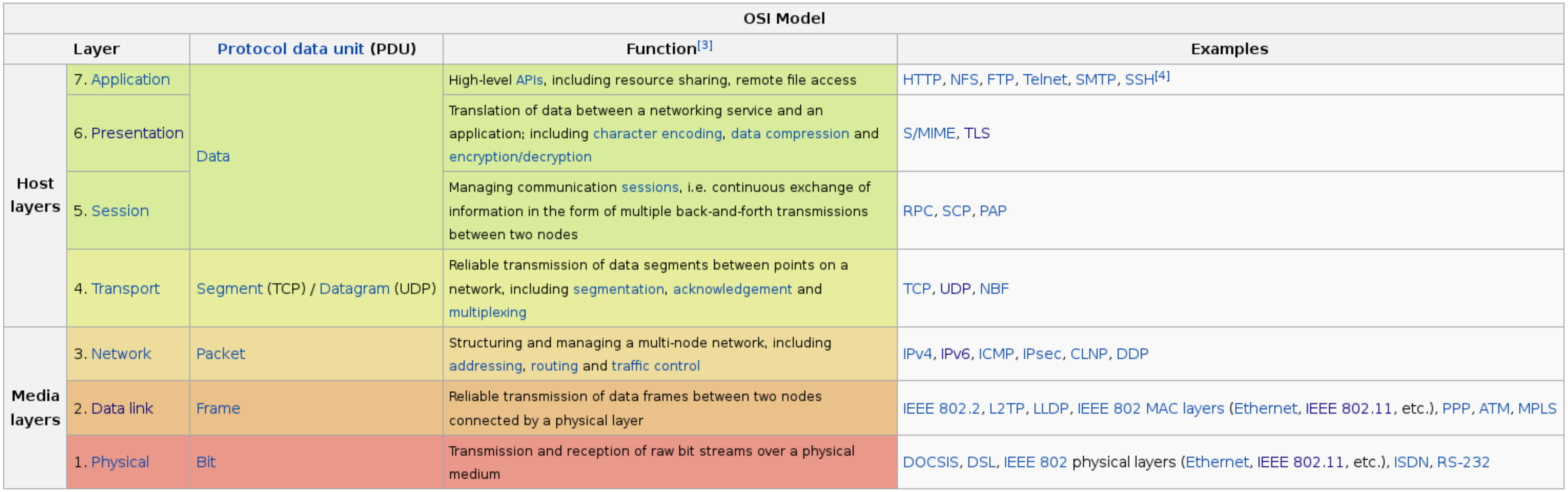
Jens Egholm Pedersen <jeep@cphbusiness.dk>

Literature:
http://www.tutorialspoint.com/http/http_tutorial.pdf

http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
http://en.wikipedia.org/wiki/HTTP_cookie
http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html

Protocols

- Used between any two endpoints
- Fundamental for communication

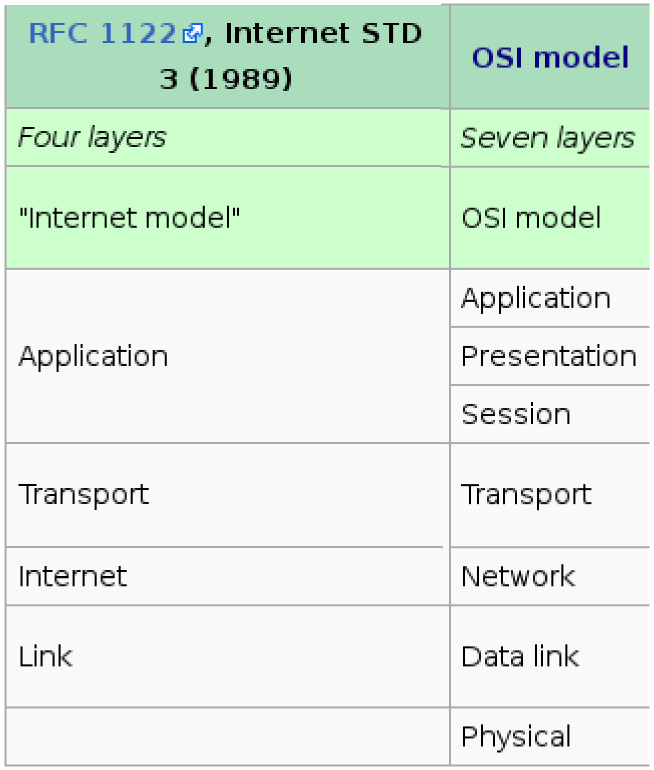


OSI Model				
Layer		Protocol data unit (PDU)	Function[3]	Examples
Host layers	7. Application	Data	High-level APIs, including resource sharing, remote file access	HTTP, NFS, FTP, Telnet, SMTP, SSH[4]
	6. Presentation		Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption	S/MIME, TLS
	5. Session		Managing communication sessions, i.e. continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes	RPC, SCP, PAP
	4. Transport	Segment (TCP) / Datagram (UDP)	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing	TCP, UDP, NBF
Media layers	3. Network	Packet	Structuring and managing a multi-node network, including addressing, routing and traffic control	IPv4, IPv6, ICMP, IPsec, CLNP, DDP
	2. Data link	Frame	Reliable transmission of data frames between two nodes connected by a physical layer	IEEE 802.2, L2TP, LLDP, IEEE 802 MAC layers (Ethernet, IEEE 802.11, etc.), PPP, ATM, MPLS
	1. Physical	Bit	Transmission and reception of raw bit streams over a physical medium	DOCSIS, DSL, IEEE 802 physical layers (Ethernet, IEEE 802.11, etc.), ISDN, RS-232

See also: OSI model, application layer

Internet protocol suite

- Yet another model
- Known as TCP/IP

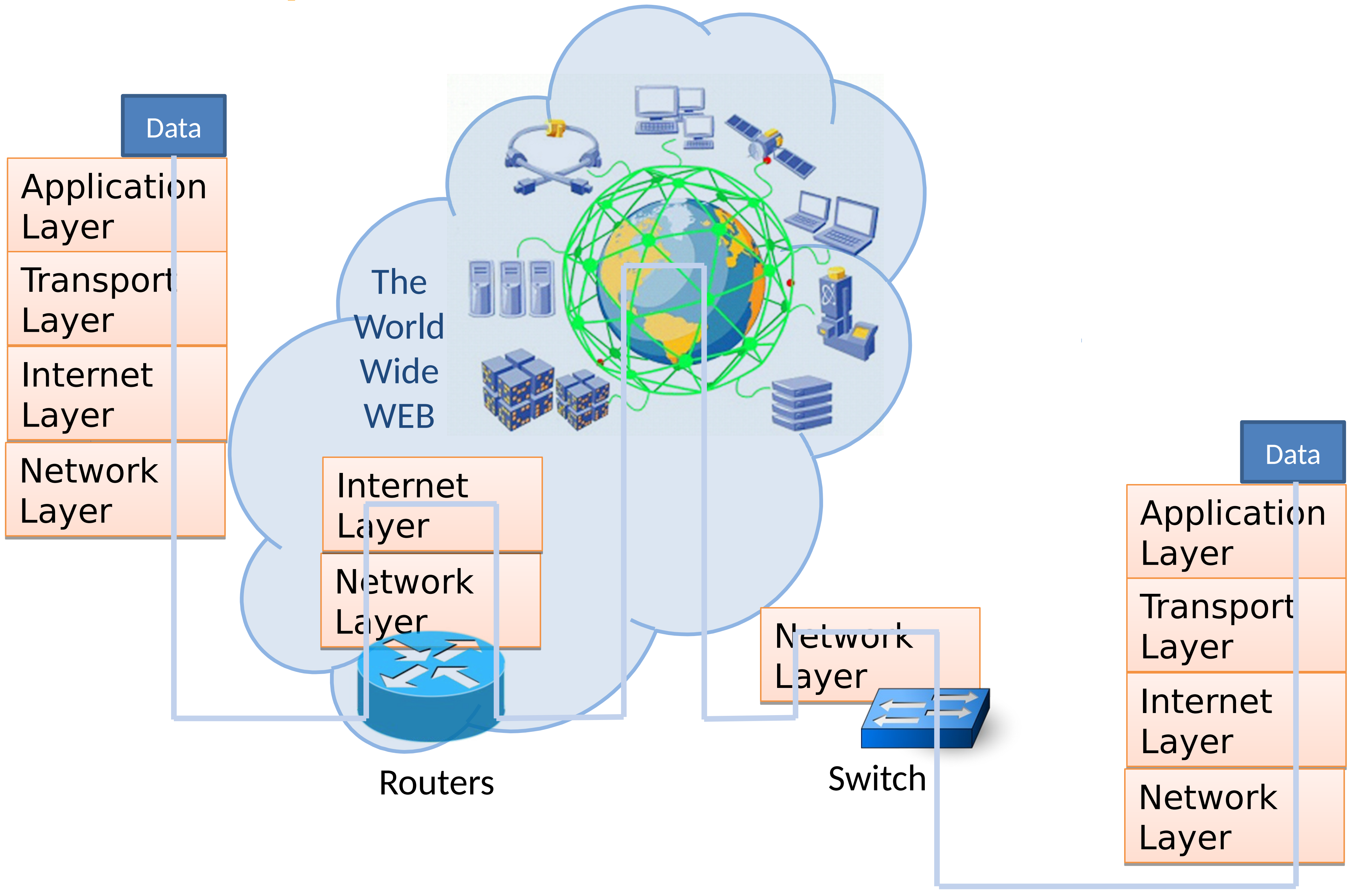


RFC 1122, Internet STD 3 (1989)	OSI model
Four layers	*Seven layers*
"Internet model"	OSI model
Application	Application
	Presentation
	Session
Transport	Transport
Internet	Network
Link	Data link
	Physical

See also: Internet Protocol Suite on Wikipedia

The TCP/IP Protocol Stack-2

cphbusiness



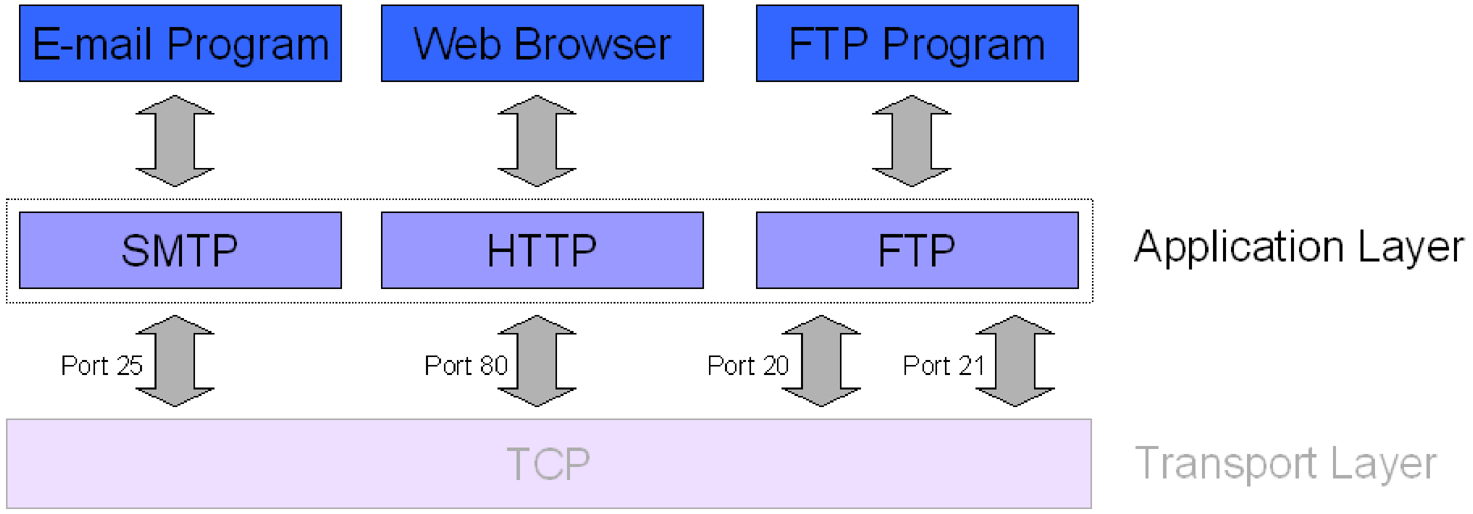
Example Application Layer Protocols

Application Layer

Transport Layer

Internet Layer

Network Layer



In the early days of the internet

- Imagine you have a lot of servers
- All servers have data
- Define a protocol to share the data

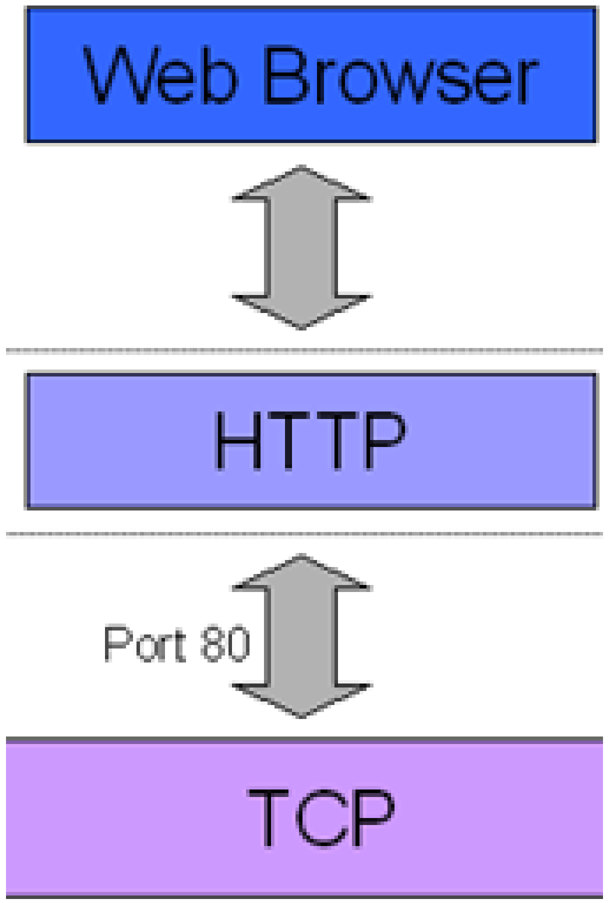
See also: Internet Protocol Suite on Wikipedia

CERN

- Tim Berners-Lee 1989
- Tons of documents needs to be shared

See also: RFC 2068

The Hyper Text Transfer Protocol



HTTP v. 1

- Hypertext Transfer Protocol
  - Links document on different servers (hyperlinks)
- Stateless
  - Initiated by a request
  - Replied with a response
- A HTTP package contains four things
  - Request line / response status
  - Headers
  - Empty line (`<CR><LF>`)
  - Message

See also: HTTP on Wikipedia, RFC 2616

cphbusiness

HTTP v. 1 Requests

- Request line
  - Method + URI + version
  - `GET /index.html HTTP/1.1`
- Request headers
- An empty line
- Optional message body

See also: RFC 2616 – Section 5 : Requests

HTTP v. 1 Methods

- Request line
  - Method + URI + version
    - `GET /index.html HTTP/1.1`
- Methods also known as verbs
  - `GET`
  - `POST`
  - `PUT`
  - `DELETE`
  - `HEAD`
  - `OPTIONS`
  - `TRACE`
  - `CONNECT`

See also: RFC 2616 – Section 5 : Requests

HTTP Request Methods

cphbusiness

GET	Requests a representation of the specified resource. Requests using GET should only retrieve data and should have no other effect.
POST	Requests that the server accept the entity enclosed in the request as a new subordinate of the resource identified by the URI.
HEAD	Asks for the response identical to the one that would correspond to a GET request, but without the response body.
PUT	Requests that the enclosed entity be stored under the supplied URI. If the URI refers to an already existing resource, it is modified; if the URI does not point to an existing resource, then the server can create the resource with that URI
DELETE	Deletes the specified resource.
TRACE	Echoes back the received request so that a client can see what (if any) changes or additions have been made by intermediate servers.
OPTIONS	Returns the HTTP methods that the server supports for specified URL.

cphbusiness

HTTP v. 1 Response

- Status line
  - Version + Status code + Status text
  - `HTTP/1.1 200 OK`
- Response headers
- An empty line
- Optional message body

See also: RFC 2616 – Section 6 : Response

HTTP v. 1 Status codes

- 1xx Informational
  - `100 Continue`
- 2xx Success
  - `200 OK`
- 3xx Redirection
  - `301 Moved Permanently`
- 4xx Client error
  - `404 Not Found`
- 5xx Server error
  - `500 Internal Server Error`

See also: HTTP Status codes on Wikipedia

HTTP v. 1 Headers

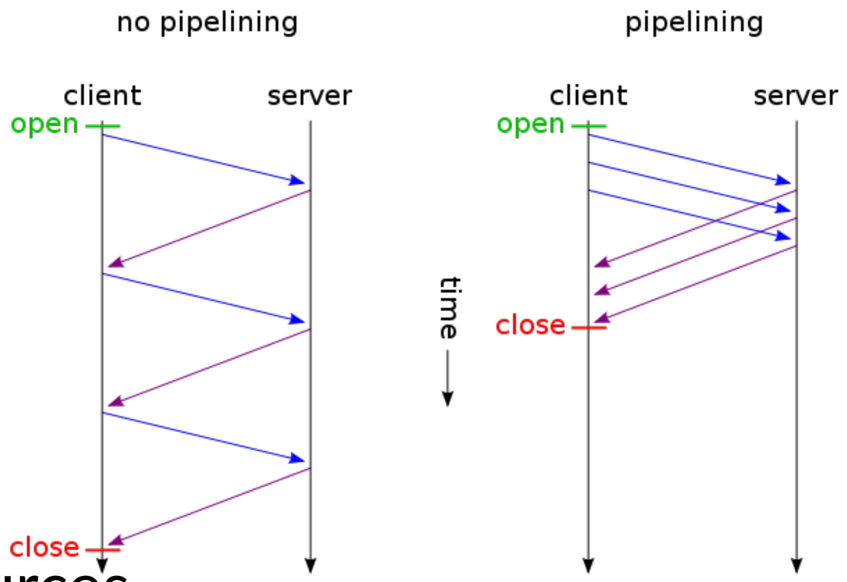
- Headers are simply key-value pairs
- Why have headers?
  - Because they can give useful information
- Request examples
  - `Accept: text/json`
- Response examples
  - `Date: Mon, 23 May 2005 22:38:34 GMT`
  - `Content-Type: text/html`

See also: List of HTTP headers on Wikipedia

cphbusiness

HTTP versions

- HTTP 1.0
  - Sequential requests
- HTTP 1.1
  - HTTP pipelining
- HTTP 2.0
  - Server can push multiple resources
  - Bundle HTML with CSS, JavaScript etc.



See also : HTTP pipelining

Recap

- Hypertext Transfer Protocol
- Stateless (!)
- Requests
  - Methods / verbs
- Responses
  - Status codes

HTTP Session

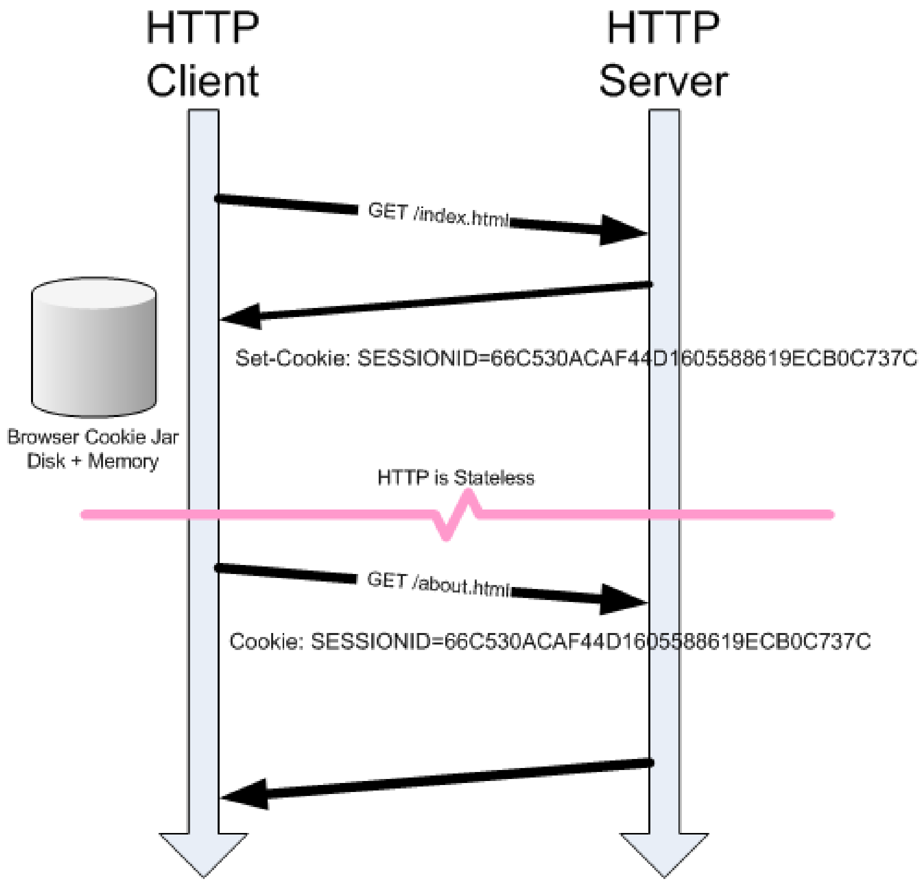
- Hack to introduce state
- A session is tied to a website
- Three ways of doing sessions
  - Query strings
    - /index.html?key=value&password=1234
  - Hidden variables in forms
  - Cookies
    - Time-limited
    - Cookie laws in Denmark
  - Local storage
  - Session storage

See also: HTTP Cookie on Wikipedia

HTTP Cookies

Session cookie

A user's session for a website exists in temporary memory only while the user is reading and navigating the website. Web browsers normally delete session cookies when the user closes the browser.


HTTP Client
HTTP Server
GET /index.html
Set-Cookie: SESSIONID=66C530ACAF44D1605588619ECB0C737C
Browser Cookie Jar
Disk + Memory
HTTP is Stateless
GET /about.html
Cookie: SESSIONID=66C530ACAF44D1605588619ECB0C737C

http://cscie12.dce.harvard.edu/lecture_notes/2011/20110504/handout.html

Browser tools

- Monitor requests and responses
  - Headers
  - Session cookies

REST

- Representational state transfer
- HTTP as state transitions
  - Before: /index.html?user=Jens
  - REST: /users/Jens
- Rules on HTTP verbs

See also: Wikipedia on REST

Safe methods

- No side-effects on data
  - Except logging etc.
- `GET, OPTIONS, HEAD, TRACE`

Idempotent methods

- Following requests have same result
- Safe methods
  - Since they have no side effects
- `PUT, DELETE`

See also: List of HTTP headers on Wikipedia

Non-Idempotent Methods

The POST method is not necessarily idempotent, and therefore sending an identical POST request multiple times may further affect state or cause further side effects (such as financial transactions). In some cases this may be desirable, but in other cases this could be due to an accident, such as when a user does not realize that their action will result in sending another request, or they did not receive adequate feedback that their first request was successful.

- It is generally up to the web application to handle cases where a POST request should not be submitted more than once.

- Whether a method is idempotent is not enforced by the protocol or web server. It is perfectly possible to write a web application in which (for example) a database insert or other non-idempotent action is triggered by a GET or other request. Ignoring this recommendation, however, may result in undesirable consequences, if a user agent assumes that repeating the same request is safe when it isn't.

HTTP authentication

- Uses standard HTTP header fields
  - Does not require session or login pages
- Server
  - `WWW-Authenticate Basic realm="My Server"`
- Client
  - `Authorization: Basic QWxhZGRpbjpPcGVuU2VzYW1l`
- Shorthand
  - `https://Aladdin:OpenSesame@www.example.com/index.html`

See also: Basic HTTP Authentication on Wikipedia

Recap

- HTTP sessions
  - URL parameters
  - Cookies
- RESTful websites
  - Safe methods
  - Idempotent methods
- Basic HTTP authentication

Groups for CA

- Next week: project CA
- Start forming groups of 2-4

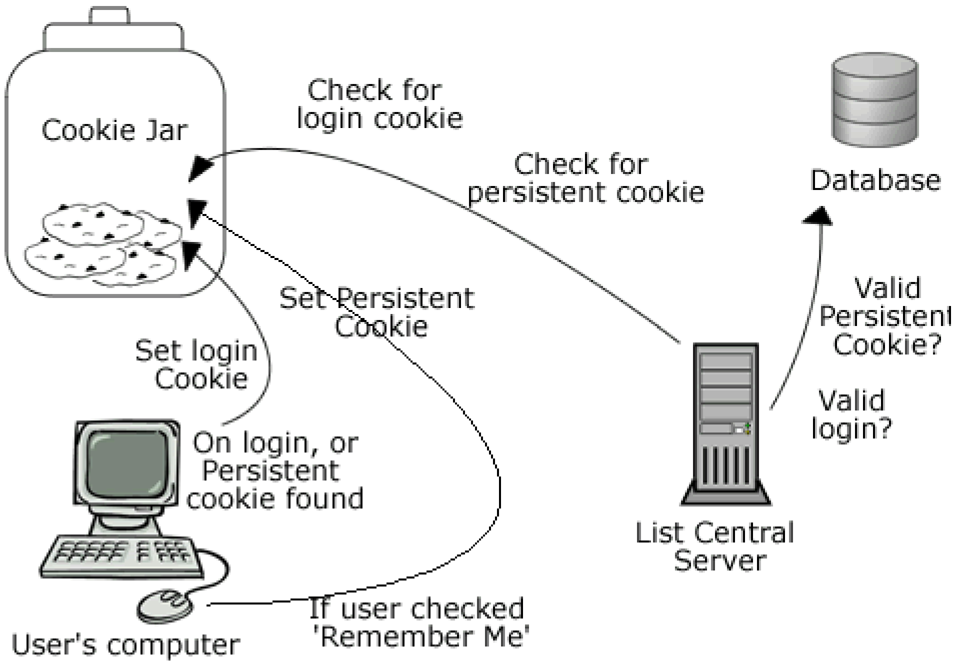
HTTP session recap

- State across requests
- Ways of maintaining data about users

HTTP Cookies

Persistent cookie

A persistent cookie will outlast user sessions. If a persistent cookie has its Max-Age set to 1 year, then, within the year, the initial value set in that cookie would be sent back to the server every time the user visited the server.



Plan for today

- Writing servers: Tomcat and Maven
- Implementing HTTP calls
- Handling session
  - URI
  - Cookies
- Doing basic HTTP authentication
- Recap on material so far

See also: Repository on GitHub

Topics so far

- Threads
  - Threads, thread-safety, producer-consumer
- Sockets
  - Ports, IP addresses, TCP, UDP
- HTTP
  - Stateless protocol, requests, responses, session