NETWORK

LEARNING OBJECTIVES

- Understand and use properties in the HTTP protocol, like Headers, Caching, HTTP Sessions and Cookies
- Understand the Same Origin Policy related to REST and how to solve problems using CORS
- Understand, at a conceptual level, the concepts in the underlying structure of the internet
- Understand the Concepts Ports and IP and the Protocols TCP and UDP.
- Know how to implement simple, multi-threaded, TCP (UDP) servers in Java
- Understand the concept Virtualization
- Understand and Implement modern Server Architectures
- Understand and use concepts required to setup a Server with a domain name, Reverse Proxy and SSL

BUSINESS COMPETENCES

- The necessary background to implement and code user defined network protocols on top of TCP or UDP
- The minimal network knowledge, required to setup a Web Server Infrastructure used by a professional web-application

PLAN

Day1 - HTTP
Day2 - Internet
Day3 - Socket programming
Day4 - Server configuration

SUBJECTS

OSI MODEL (OPEN SYSTEMS INTERCONNECTION MODEL)

Conceptual model that characterizes and standardizes communication
Describes how computers communicate with one another over a network
Describes how different software and hardware components involved in a network communication should divide labor and interact with one another
Goal is interoperability of diverse communication systems with standard protocols
Defines networking framework to implement protocols in terms of a vertical stack of seven layers
A layer serves the layer above it and is served by the layer below it
Protocols are used between two endpoints and are fundamental for communication
All data that goes over a network connection passes through each of the seven layers, going from the upper level software oriented services to the lower level more hardware oriented functions

LAYERS

- Layer 7: Application Layer (Protocols: HTTP/FTP/TELNET/SSH/SMTP/POP3/DNS)
- Layer 6: Presentation Layer
- Layer 5: Session Layer
- Layer 4: Transport Layer (Protocols: TCP/UDP)
- Layer 3: Network Layer (Protocols: IP)
- Layer 2: Data Link Layer
- Layer 1: Physical Layer

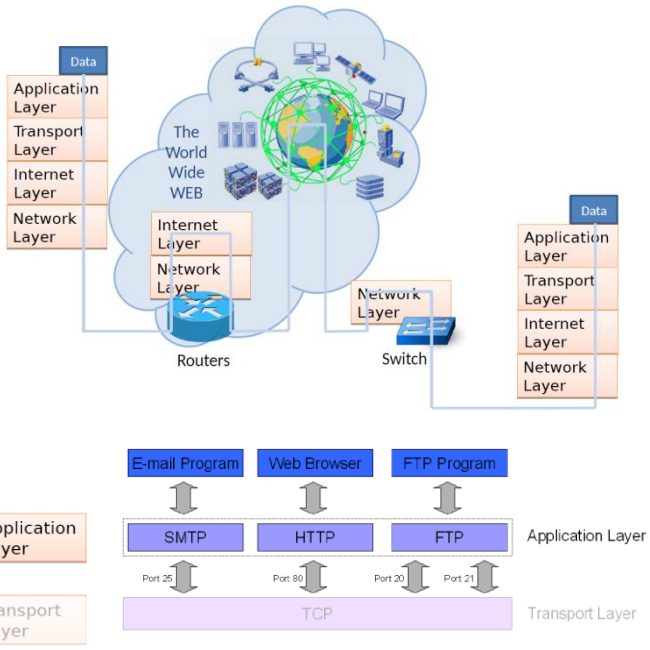
TRANSMISSION CONTROL PROTOCOL / INTERNET PROTOCOL (TCP/IP)

TCP/IP model is in a way an implementation of the OSI model

TCP defines how applications can create reliable channels of communication across a network and IP defines addressing and routing
Protocol by which data is sent from one computer to another on the Internet, organized into four abstraction layers which specifies how data should be packetized, addressed, transmitted, routed, and received
Each computer/host on the Internet has one IP address that uniquely identifies it from all other computers on the Internet
When data is sent or received, it gets divided into packets, containing both the sender's and the receiver's IP address
During transmission, each layer adds a header to the data that directs and identifies the packet

LAYERS

- Layer 4: Application layer
- Layer 3: Transport layer
- Layer 2: Internet layer
- Layer 1: Link layer



HTTP (HYPERTEXT TRANSFER PROTOCOL)

Basic protocol of the web
Links documents on different servers via hyperlinks
Stateless application-level protocol
Request / Response
Initiated by a request / Replied with a response

HTTP package

- Request line / Status line
  - Request line: Method / URI / Version
    Request: GET /logo.gif HTTP/1.1
  - Status line: Version / Status code / Status text
    Response: HTTP/1.1 200 OK
- Headers
  Simple key-value pairs
- Empty line (<CR><LF>)
  The request / status line and headers must end with <CR><LF>
- Message
  Optional message body

Methods

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 This can be used to check the functionality of a web server

Session

The core HTTP protocol itself is stateless, but web applications built on top of HTTP do not have to be stateless

Different ways to introduce state

- Query strings /index.html?user=jimmy&password=1234
- Hidden form variables
- Cookies
- Local storage
- Session storage

Cookies

Users sessions for a website exists in temporary memory only while users are navigating the website
Web browsers normally delete session cookies when users close their browsers

Session / Persistent

Caching

Set expiration

Authentication

POSTMAN

Browser tool

Inspector (F12 -> Network & Storage)
Request / Response / Header / Body / Cookies

CORS

Same Origin Policy

A web browser permits scripts contained in a web page to access data in another web page, but only if both web pages have the same origin

Set in response header with Access-Control-Allow-Origin

Problem

Fetching data from external rest api's might not be allowed

Solution

Extract data via server and make it available locally

REFERENCES

https://en.wikipedia.org/wiki/OSI_model
https://en.wikipedia.org/wiki/Internet_protocol_suite
https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
https://en.wikipedia.org/wiki/List_of_HTTP_status_codes
https://en.wikipedia.org/wiki/List_of_HTTP_header_fields
https://en.wikipedia.org/wiki/HTTP_cookie
http://www.tutorialspoint.com/http/http_tutorial.pdf
https://en.wikipedia.org/wiki/Same-origin_policy
https://en.wikipedia.org/wiki/Cross-origin_resource_sharing