Checklist review today:

JVM

* Class Loader (main memory)
  + Loading (parent delegation mechanism, recursive method, super class of lower level class))
    - Bootstrap Class Loader (java.util, java,lang, …)
    - Extension Class Loader (JDBC driver, ODBC driver)

* + - Application Class Loader (User define class)

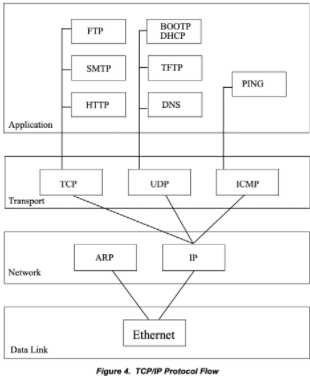
ClassLoader Sub System

* + Linking (3 steps)
    - Verify (correctness of .class file)
    - Prepare (allocated memory for static fields for class or interface)
    - Resolve (the process of dynamically determining concrete values from symbolic references in the run-time constant pool.)
  + Initialization
    - initialize (initialize static block, static class, constant pool…)
* Runtime Memory/Data Area
  + Method Area (class level data, static field)
  + Heap Area (object, new keyword)
  + Stack Area (private by single thread, call method))
    - Threads
  + PC Register (execution address of current thread)
    - Threads
  + Native Method Stack (implement by other language like c/c++)
* Execution Engine
  + Interpreter (execute byte code line by line)
  + JIT Compiler (find hotspot to improve performance of Interpreter)
  + Garbage Collector
* Native Method Interface (JNI) (bridge between execution engine and Native Method Library)
  + Native Method Library (implement by other language like c/c++)

New things learned today:

OSI & TCP/IP Model

* open system interconnection model (OSI)
  + 7.application layer
    - human computer interaction layer, where application can access the network services
    - HTTP (Hypertext Transfer Protocol)
    - FTP (File Transfer Protocol)
    - POP (Post Office Protocol)
    - SMTP (Simple Mail Transfer Protocol)
    - DNS (Domain Name System)
  + 6. Presentation Layer,
    - ensure that data is in usable format and is where data encryption occurs
  + 5, Session
    - maintain connection and is for controlling ports and sessions
  + 4, transport
    - transmits data using transmission protocols: TCP, UDP
  + 3, network
    - decides which physical path the data will take
  + 2, data link layer
    - define the format of data on the network
  + 1, physical layer,
    - transmit raw bit stream over physical medium
* TCP/IP
  + application layer
    - correspond to application, presentation, session layer in OSI
  + Transport layer
    - correspond to transport layer in OSI
  + Internet layer
    - correspond to network layer in OSI
  + Network Access Layer
    - correspond to data link and physical layer in OSI



* HTTP
  + Hypertext Transfer Protocol
  + HTTP Requests
    - HTTP Version type
    - a URL
    - HTTP method
    - HTTP request headers
    - HTTP body (optional)
  + HTTP response
    - http status code
    - http response header
    - http body (optional)
  + Status Code
    - 1xx information
    - 2xx success
      * 200 - OK, (get/put/post)
      * 201 created, success and new resources has been created, (post)
      * 202 - accepted, request has been received, the process has not been completed
      * 204 - no content, (put), usually update the resources without changing the current page displayer to the user
    - 3xx redirection
      * 307 - temporary redirect
      * 308 - permanent redirect
    - 4xx client error
      * 400 - bad request, the server could not understand the request due to invalid syntax
      * 401 - unauthorized, -> unauthenticated, the client is not authenticated
      * 403 - forbidden -> no permission
      * 404 - not found, the server can not find the requested resources
    - 5xx server error
      * 500 - internal server error
      * 501 - not implemented, method not supported by the server
      * 502 - bad gateway. The error response means that the server, while working as a gateway to get a response needed to handle the request, got an invalid response
  + HTTP request Method

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| method | safe | idempotent | cacheable | Des |
| get | yes | yes | yes | read/retrieve data from server |
| post | no | no | no | create |
| put | no | yes | no | whole update |
| delete | no | yes | no | remove |
| patch | no | no | no | partial update |

* + - safe:
      * a http method is safe if it doesn’t alter the state of the server
    - idempotent
      * if an identical request is made once or several times, the server will be the same state
    - cacheable
      * private browser cache
      * shared proxy cache (ISP)

Learned Plan tomorrow:

* Review Garbage Collator
  + types od GC
  + GC use in different generation (heap)
* Review OOP
  + Abstraction
  + Encapsulation
  + Inheritance
  + Polymorphism
* Review OSI, TCP/IP, HTTP
  + layers of OSI, TCP/IP
  + HTTP methods
  + HTTP code