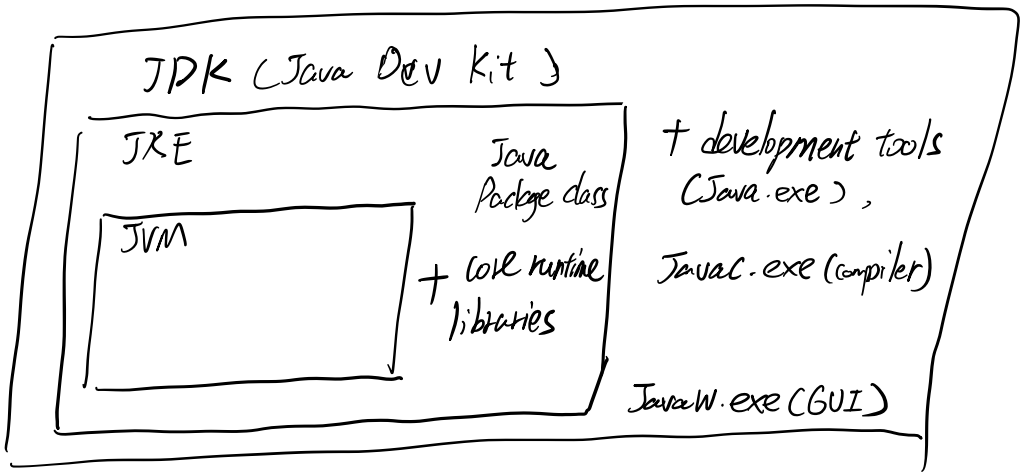
Java

Description: OOP language with two important features, platform independent and GC.

platform-independent: Using ByteCode and JVM, we can make the code understandable by any platform.

GC: Java uses GC (daemon thread) to free up the memory, under most situations, developers do not worry about manually allocate and free up memory.

JDK vs JRE vs JVM



JRE: an environment to execute Java program locally

JVM: convert byte code to machine-specific code with the libraries support for that code

Java Datatypes

- Primitive
  - integer type [byte, int, short, long]
  - floating point type [double, float]
  - character — char
  - Bool — boolean
- reference type
  - class
  - Interface
  - array type ([])

JVM

- class loader ↕
  - loading
    - Bootstrap (Root) – JVM internal classes
    - Extension – extension of standard core Java classes
    - Application – found in classpath environment vars.
  - linking – Verify, prepare, resolve
  - Initializing

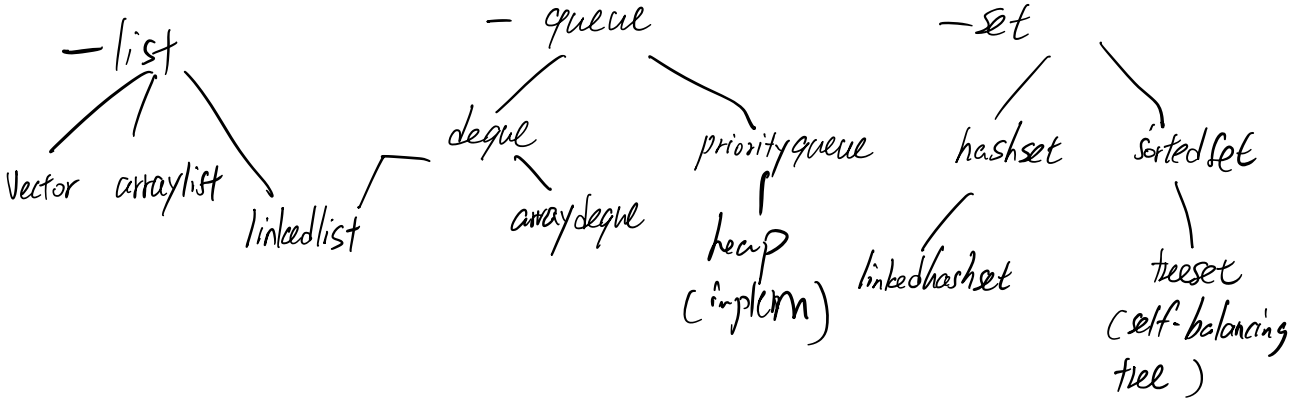
- Runtime Data Area ↕
  - heap, stack, PC register, native method stack

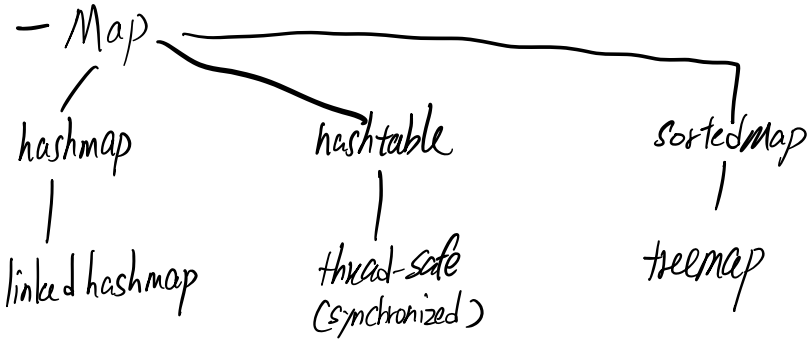
- Execution Engine
  - GC
  - JIT compiler
  - Interpreter

When JVM requests a class, class loader tries to locate the class and look for class definition into runtime. If the class isn't loaded, it will recursively delegate the request to the parent class loader.

If the last child class loader can't find the class, it will throw NoClassDefFoundError or ClassNotFoundException.

Collections : a way to organize data





list is an ordered sequence of elements that can be accessed by index of element

Set is an unordered, distinct list of elements that only can be accessed by element itself

Queue is a sequence of elements that allow to be inserted at the end of queue and removed at the beginning of the queue
- Deque is designed to be inserted and removed from both ends.

Map is a data structure that stores key-value pair, we can access value based on key.

List vs Set vs Map

list – access element based on index, ordered, allow duplicated elements, and null values

set – does not allow duplicate elements, unordered, and only allows one null elem.

map – stores key-value mapping

ArrayList

- internal: dynamically resizable array
  - ensureCapacity() to resize the array.
- inherits AbstractList, implements the List interface

ArrayList vs Vector.

- different grow capacity: arraylist by default would grow 1.5 times the size of array, vector would grow 2 times the size of array
- vector is thread-safe, but will performance overhead, arraylist is not thread safe.

Array Blocking Que
Priority Blocking
Linked Blocking
Synchronous Blocking

Blocking Queue

Atomic Op

thread Communication

lock mechanism (read/write lock)
wait/notify
volatile

dirty read:

read uncommitted data from another transaction

non-repeatable read

read committed data from an update query from another transaction

phantom read:

read committed data from a query that has been updated (either delete or insert) from another transaction

Isolation level

how a transaction is isolated from another transaction. It controls how the locks are held, how read operation for a row affect by another transaction.

read uncommitted

- allow the transaction to read uncommitted data, which will result in phantom, dirty read, non-repeatable read.

read committed.

will result in non-repectable read, phantom read

repectable read

does everything that read committed does, also will wait for other transaction's update operations

But insert query doesn't wait, this will still have phantom read problem

What is a thread pool?

A thread pool is a collection of pre-instantiated, idle threads which are ready to be given work.

The threads don't terminate right away, when one thread completes the work, the thread becomes idle, ready to be dispatched to another job.

- overhead of creating and destroying threads is limited to creating and destroying the number of active worker threads.