Multithreading

Processes

- An application consists of one of more processes
- A process is an executing program

Threads

- One or more threads run in the context of a process
- A thread is a basic unit to which the operating system allocates processor time
- A thread can execute any part of the process code, including parts currently being executed by another thread

```
public class MyThread extends Thread {
    @Override
    public void run() {
        long j = Thread.currentThread().getId();
        System.out.println("child thread" + j + " in run()");
    }

    public static void main(String[] args) {
        MyThread t1 = new MyThread();
        t1.start();
        t1.run();
        long j = Thread.currentThread().getId();
        System.out.println("parent thread" + j + " in main()");
    }
}
```

Thread Scheduler

- Determines execution order of threads in the JVM
- We cannot predict the exact execution order
    - It is JVM and vendor dependent

Creating and Running a Thread

- `t.start()`
    - A new thread is created that invokes the `run()` method
- `t.run()`
    - No new thread created

- Not overriding `run()` method
  - Thread class `run()` method will be executed
- Overloading `run()` method
  - Thread class `start()` always invokes the no argument `run()` method
- Overriding `start()` is never recommended

Life Cycle of a Thread

- **New Born** -> `t.start()` -> **Ready/Runnable** -> if thread scheduler allocates CPU -> **Running** -> If run method is completed -> **Dead**

Defining a Thread by Implementing a Runnable Interface

join()

sleep()

- Pause execution of current thread for specified amount of time

Multitasking

- Concurrent execution of multiple processes
- Improves performance by reducing response time

Process Based

- Each task is a process

Thread Based

- Each task is a separate thread of the same program

Concurrency Vs Parallelism

Synchronization

- Pro
  - Resolve data inconsistency
- Con
  - Increase waiting time of the thread and affects performance of the system

Double-Checked Locking

- Improves performance

- Volatile keyword ensures that multiple threads handle the unique instance variable correctly when it is being initialized to the singleton instance