Chapter 13 - Multi threading

Multiprocessing and multithreading both are used to acheive multitasking

| Process1 | Process2 | Process3 |

Thread 1
§ Thread 2
§
§
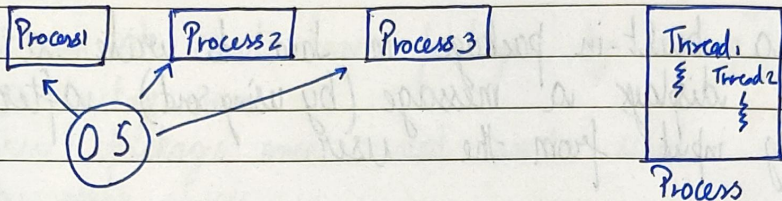
Process

In a nut Shell...
→ Threads use shared memory area.
→ Threads ⇒ faster Content switching
→ A Thread is light-weight whereas a process is heavyweight
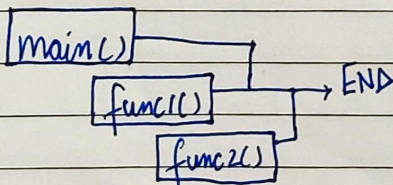
For Example → A word processor can have one thread running in foreground as an editor and another in the background auto saving the document!

Flow of control in Java

1₌ Without threading :

| main() | → | func1() | → | func 2 | → END

2₌ With threading:

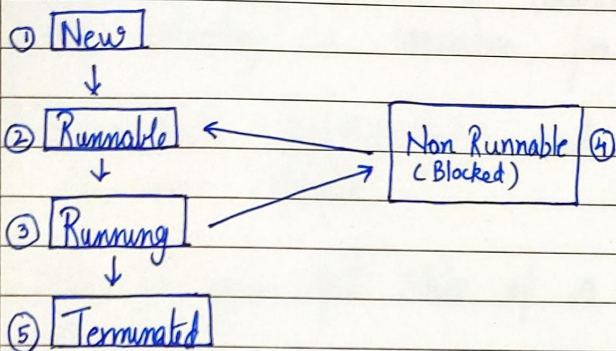| main() |
| func1() | → END
| func2() |
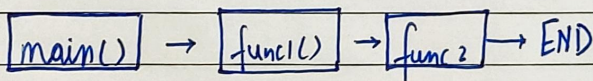
Creating a Thread

There are two ways to create a thread in Java.

1. By extending Thread class
2. By implementing Runnable interface

Life cycle of a Thread

① New
↓
② Runnable ← Non Runnable ④ (Blocked)
↓
③ Running →
↓
⑤ Terminated

① New → Instance of thread created which is not yet started by invoking start()

② Runnable → After invocation of start() & before it is selected to be run by the scheduler.

③ Running → After thread scheduler has selected it.

④ Non Runnable → Thread alive, not eligible to run.

⑤ Terminated → run() method has exitted

The Thread class

Below are the commonly used Constructors of Thread class:

① Thread()
② Thread (String name)   ④ Thread (Runnable r, String name)
③ Thread (Runnable r)

Methods of Thread class

Thread class offers a lot of methods such as run(), Start(), join(), get Priority(), Set Priority () etc. More can be found on visiting Java docs