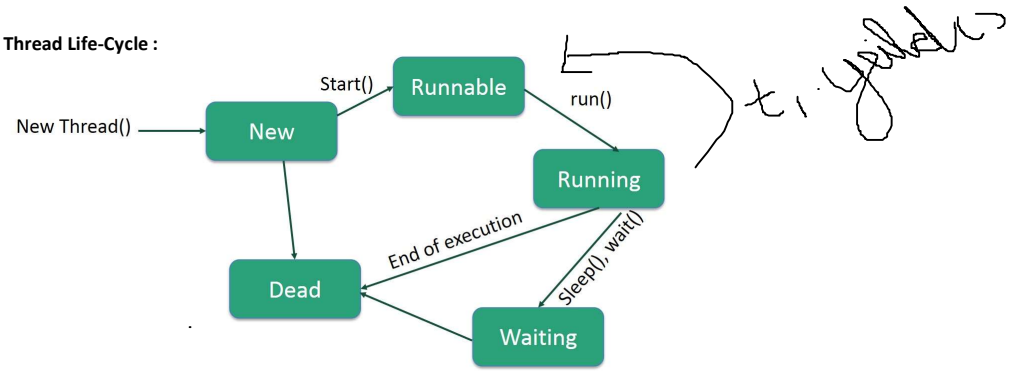
Multi Threading

Tuesday, August 15, 2023 1:28 PM

- Concurrancy -> Computer users take it for granted that their systems can do more than one thing at a time.
- To achieve Concurrancy only we are using **Multi-Thread**.
- Multi-processing :- OS.. Doing many tasks at same time using different process.
- Multi-Threading :- JVM run as a single process. Doing many task at same time using single process.
- Java program atleast run one thread called main thread .Ex: Sysout(10/0). Exxception in thread "main". Thread will create Stack . Stack -LIFO . e.printStackTrace().
- Multi threading is used in when ever the multiple users occur in same time. Ex:Gamming at a same time multiple users will exist.

Thread Life-Cycle :



- Developer is moving a new thread to Runnable.
- JVM is moving that thread from Runnable to Running.
- Thread Scheduler : Decides which Thread to be moved from Runnable to Running. Can set priority to threads. setPriority(int) ---> 1 to 10. Default priority will be 5.
- The Thread will move to JVM . JVM will wait for the ROM and Processor to take the thread. During the JVM wait its called as Runnable. After the execution of JVM it will move to the running state.
- When the created thread started then once more we make thread.start() it will get exception which is **IllegalThreadStateException**.
- For Example :
  - Lets take Traffic signal in center there is four roads going in the center the traffic signal the traffic police will make signal according to it like that Thread Scheduler . It was inside the JVM and its works. In traffic according to immediate sutivation like ambulance we make signal according to it like that we can set priority for the thread.
- OS : Multitasking when we open ,
  - Notepad
  - Firefox
  - VLC
  - Eclipse
- Processor: i5 Processor: Processor will decide the priority and threading for the OS.
- yeild() : currently executing thread giving its chances to other threads. One there threads which have same priority as of yielding thread. While we giving t1.yeild() it was in running state while we giving it move to the runnable state. Because while giving its OS and ROM to other threads its go to runnable state from running state.
- Join() : One thread wait for completion of other threads to join. In some sutivation one thread is depend on another thread, during that time join method is used. When we give join it will go to **wait state**. In join there will be given choice we can wait for other thread or we can wait for particular time for the other thread.
- Sleep() : sleep is used to make sleep the thread for the particular time. During sleep the thread will move to the wait state.
- **From wait state it will go to runnable after that only it go to running state. What ever come to wait state it will move to runnable again and go for running state.**
-