


MULTITHREADING

SUBJECT: JAVA PROGRAMMING

SAVIYA VARGHESE
DEPT OF BCA
2020-21

1 .Multithreaded Programming

1.1 Multi-threaded and Multitasking

- ▶ Java is a *multi-threaded programming language* which means we can develop multi-threaded program using Java.
 - ▶ A multi-threaded program contains two or more parts that can run concurrently and each part can handle a different task at the same time making optimal use of the available resources specially when your computer has multiple CPUs.
 - ▶ **Multithreading in Java** is a process of executing multiple threads simultaneously.
 - ▶ A thread is a lightweight sub-process, the smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.
- 

Multitasking

- ▶ By definition, multitasking is when multiple processes share common processing resources such as a CPU.
- ▶ Multi-threading extends the idea of multitasking into applications where you can subdivide specific operations within a single application into individual threads.
- ▶ Each of the threads can run in parallel.
- ▶ The OS divides processing time not only among different applications, but also among each thread within an application.



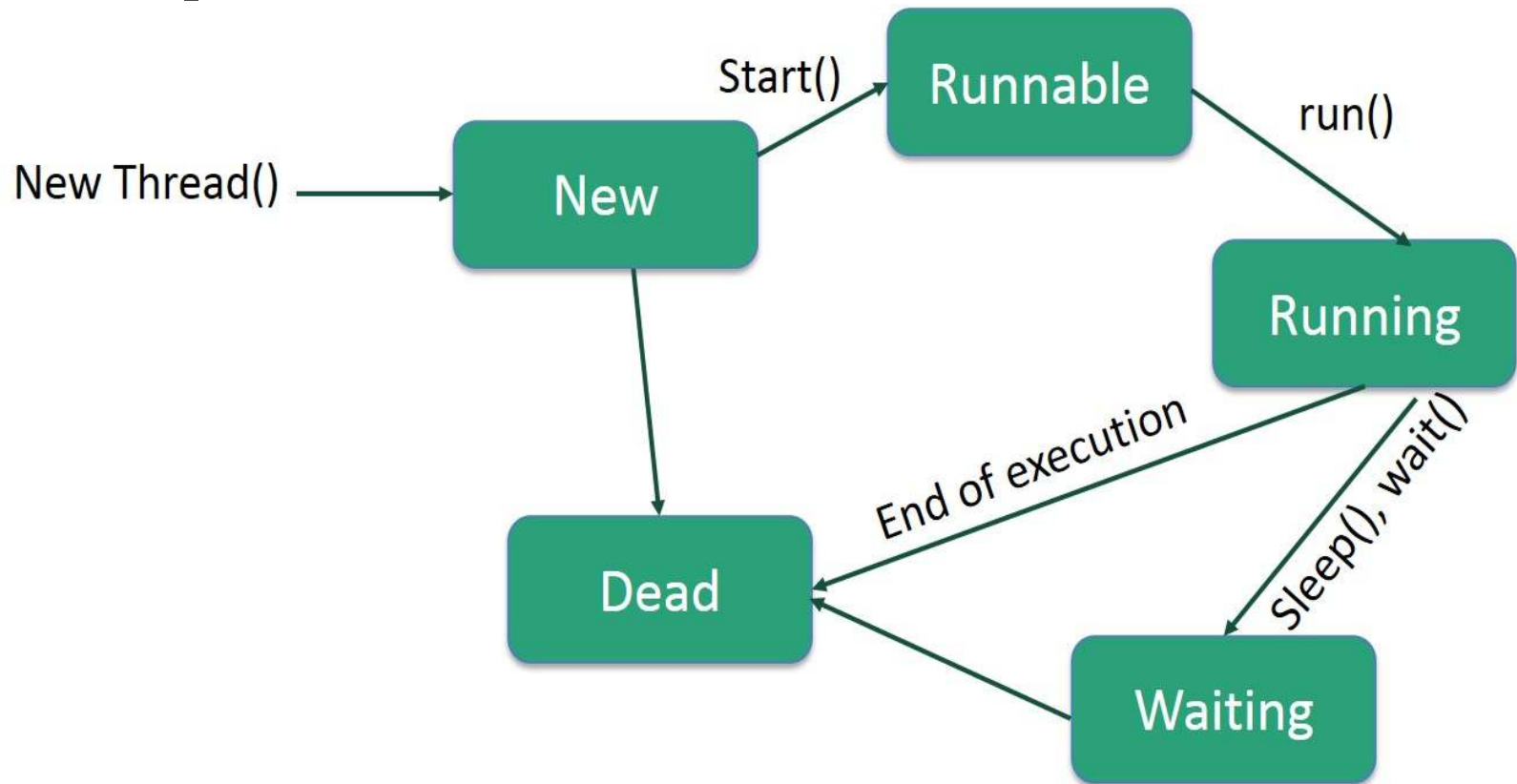
Advantages of Java Multithreading

- 1) It **doesn't block the user** because threads are independent and you can perform multiple operations at the same time.
- 2) You **can perform many operations together, so it saves time.**
- 3) Threads are **independent**, so it doesn't affect other threads if an exception occurs in a single thread.



Life Cycle of a Thread

- ▶ A thread goes through various stages in its life cycle.
- ▶ For example, a thread is born, started, runs, and then dies.



stages of the life cycle

- ▶ **New** – A new thread begins its life cycle in the new state. It remains in this state until the program starts the thread. It is also referred to as a **born thread**.
- ▶ **Runnable** – After a newly born thread is started, the thread becomes runnable. A thread in this state is considered to be executing its task.
- ▶ **Waiting** – Sometimes, a thread transitions to the waiting state while the thread waits for another thread to perform a task. A thread transitions back to the runnable state only when another thread signals the waiting thread to continue executing.



Timed Waiting – A runnable thread can enter the timed waiting state for a specified interval of time. A thread in this state transitions back to the runnable state when that time interval expires or when the event it is waiting for occurs.

Terminated (Dead) – A runnable thread enters the terminated state when it completes its task or otherwise terminates.

