Thread

What is a process

- A process is an instance of a program running in a computer. It is close in meaning to task , a term used in some operating systems. In UNIX and some other operating systems, a process is started when a program is initiated (either by a user entering a shell command or by another program). Like a task, a process is a running program with which a particular set of data is associated so that the process can be kept track of. An application that is being shared by multiple users will generally have one process at some stage of execution for each user.

What is a Thread

- Within a program, a Thread is a separate execution path. It is a lightweight process that the operating system can schedule and run concurrently with other threads. The operating system creates and manages threads, and they share the same memory and resources as the program that created them. This enables multiple threads to collaborate and work efficiently within a single program.
- A thread is a single sequence stream within a process. Threads are also called lightweight processes as they possess some of the properties of processes. Each thread belongs to exactly one process. In an operating system that supports multithreading, the process can consist of many threads.

Creating Thread in Java

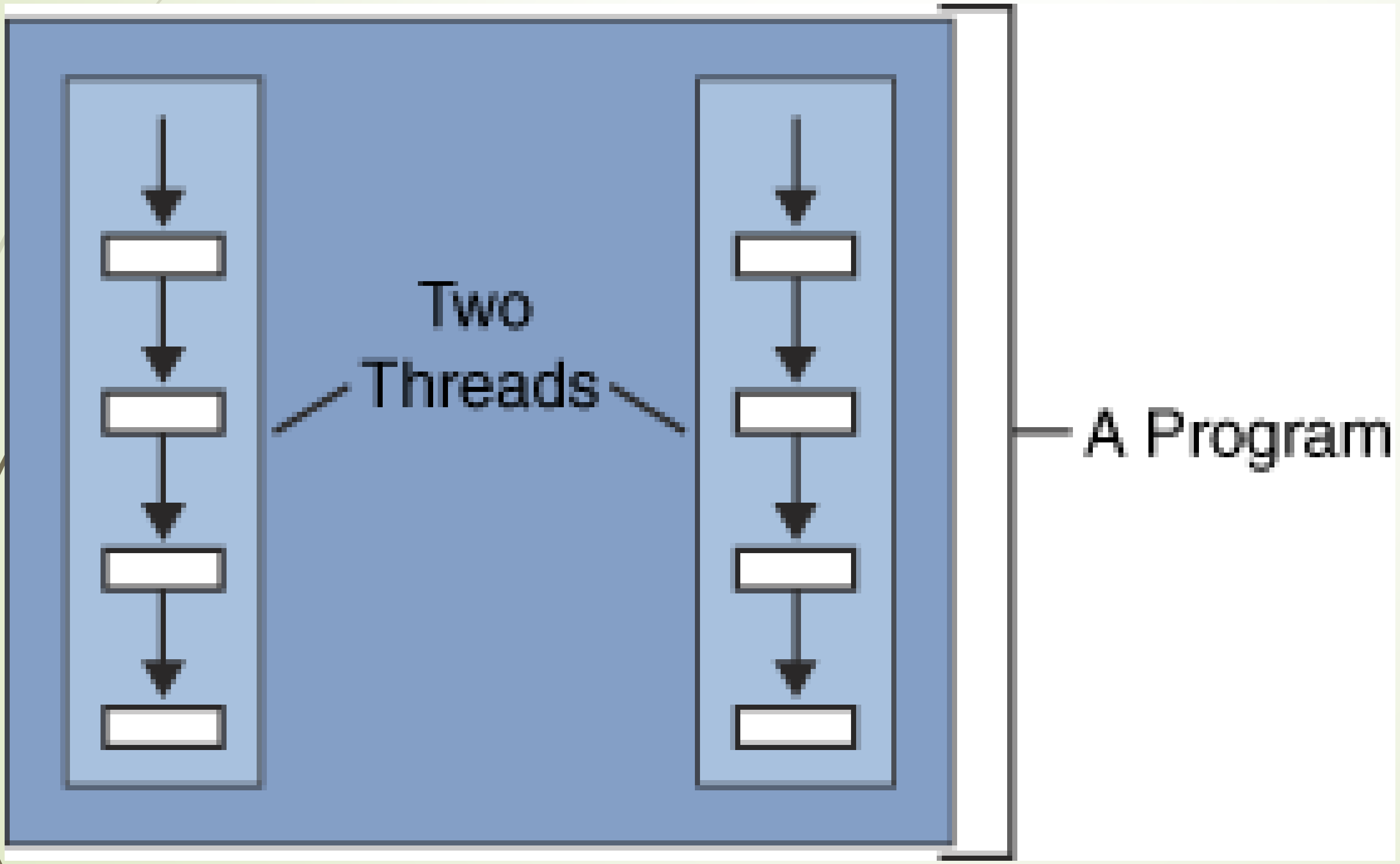
- Thread in java can be created using Thread class
- There are two ways to obtain a Thread instance

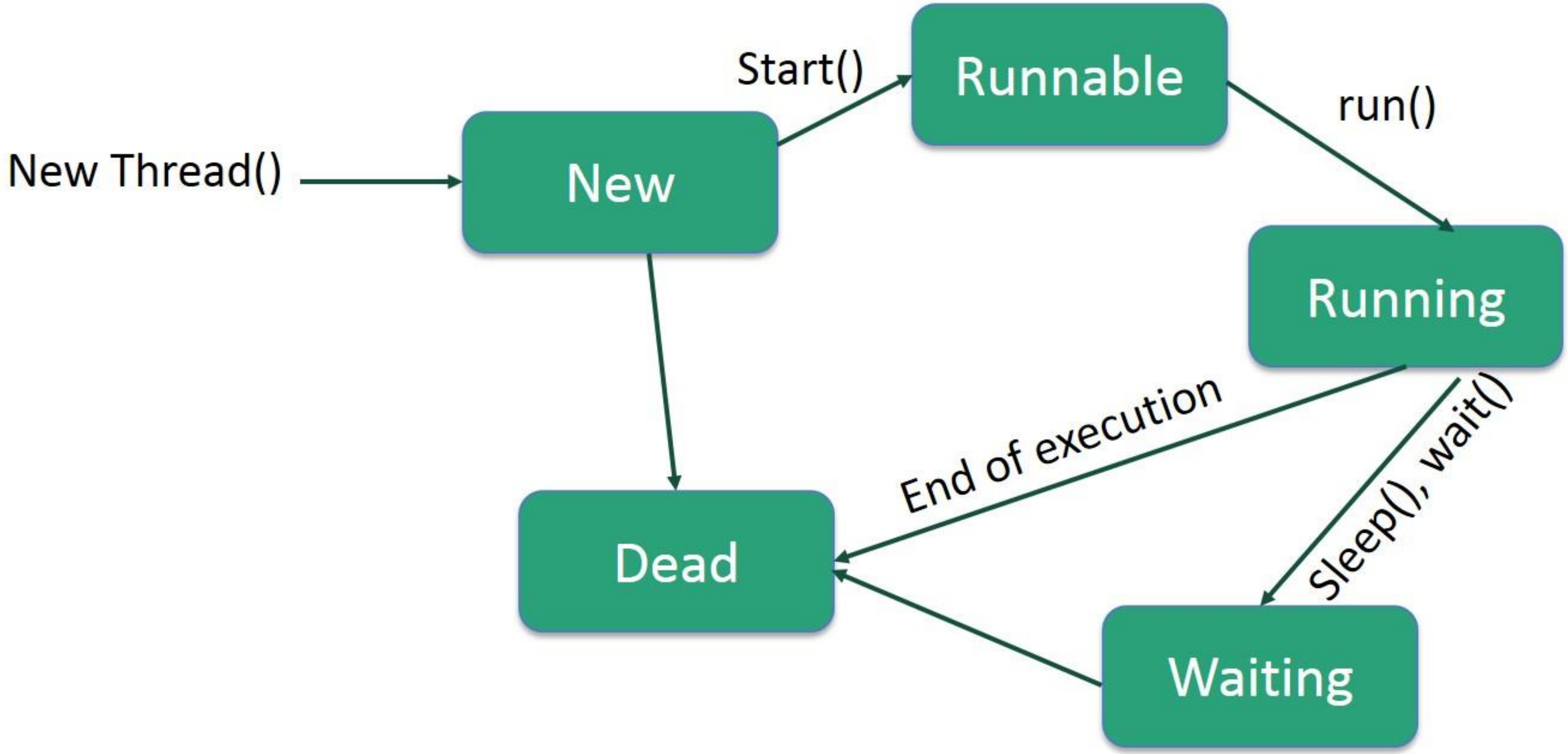
  By extending Thread class

  By implementing Runnable interface

Thread class methods

- start()
- run()
- getName()
- setName()
- currentThread()
- getPriority()
- setPriority()
- isAlive()
- join()


Two
Threads
A Program


New Thread()
New
Start()
Runnable
run()
Running
End of execution
Sleep(), wait()
Waiting
Dead

Creating a Thread using Thread class

```
public class MyThread extends Thread
{
  public void run()
{
System.out.println("Thread started");
}
  public static void main(String args[])
{
 MyThread th=new MyThread();
th.start();
}
}
```

Creating a Thread using Runnable interface

```
public class MyRunnable implements Runnable
 {
 public void run()
  {
  System.out.println("Thread with runnable instance started");
 }
   public static void main(String args[])
 {
  Thread th=new Thread(new MyRunnable());

th.start();
 }
 }
```

Thread methods

join method

- The join method allows one thread to wait for the completion of another. If t is a Thread object whose thread is currently executing,

t.join();

causes the current thread to pause execution until t's thread terminates.

sleep() method ,pauses a thread for a duration specified in milliseconds

Synchronization in Thread

Synchronization can be applied in thread using synchronized keyword with function or using synchronized blocks

Using Synchronization allows a thread to acquire lock over a resource .

The lock is released when the Thread is terminated or run method is over

Other threads will have to wait for the resource to be released by the Thread that has acquired lock