EXPERIMNET NO. 8

1. What is process and what is thread?

Ans: A process is a program that is running on your computer. This can be anything from a small background task, such as a spell-checker or system events handler to a full-blown application like Internet Explorer or Microsoft Word. All processes are composed of one or more threads. Most operating systems have many background tasks running, your computer is likely to have many more processes running than actual programs. The term "process" can also be used as a verb, which means to perform a series of operations on a set of data. For example, your computer's CPU processes information sent to it by various programs.

A thread is an independent path of execution within a program. Many threads can run concurrently within a program. Every thread in java is created and controlled by the java.lang.Thread class.

2.What is multiprocessing, multitasking, multiprogramming and multithreading?

Ans: Multiprocessing is the use of two or more central processing units (CPUs) within a single computer system. The term also refers to the ability of a system to support more than one processor or the ability to allocate tasks between them.

Multitasking is the ability to perform more than one activity concurrently on a computer. We can further break multitasking into process based and thread based.

Multiprogramming is also the ability of an operating system to execute more than one program on a single processor machine. More than one task/program/job/process can reside into the main memory at one point of time. A computer running excel and firefox browser simultaneously is an example of multiprogramming.

Multithreading in java is a process of executing two or more threads simultaneously to maximum utilization of CPU. Multithreaded applications are where two or more threads run concurrently; hence it is also known as Concurrency in Java. This multitasking is done, when multiple processes share common resources like CPU, memory, etc.

3. What are the benefits of multithreading?

Ans:

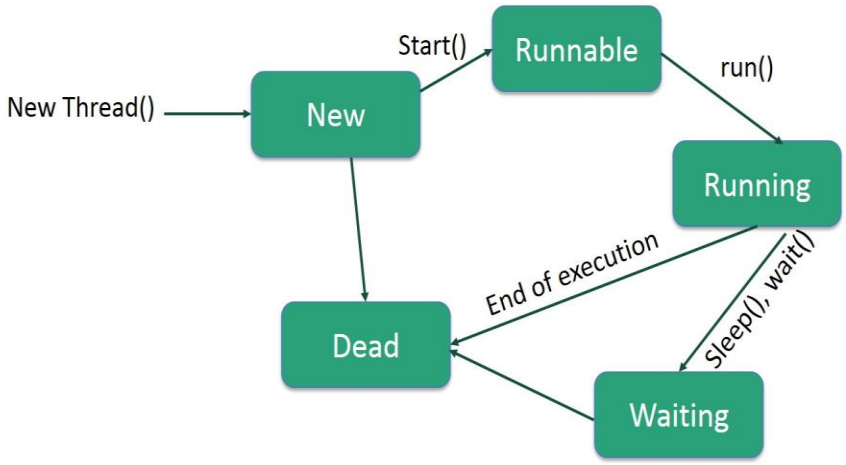
1) Improved throughput. ...
2) Simultaneous and fully symmetric use of multiple processors for computation and I/O.
3) Superior application responsiveness. ..
4) Improved server responsiveness. ...
5) Minimized system resource usage. ...
6) Program structure simplification. ...
7) Better communication.

4. What is life cycle of Thread in java?

Ans: 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.

A thread can be in one of the five states. According to sun, there is only 4 states in thread life cycle in java, new, runnable, non-runnable and terminated. There is no running state. The life cycle of the thread in java is controlled by JVM. The java thread states are as follows:

i. New
ii. Runnable
iii. Running
iv. Non-Runnale (Blocked)
v. Terminated



1) New
The thread is in new state if you create an instance of thread class but before the invocation of start() method.

2) Runnable
The thread is in runnable state after invocation of start() method, but the thread scheduler has not selected it to be the running thread.

3) Running
The thread is in running state if the thread scheduler has selected it.

4) Non-Runnable(Blocked)
This is the state when the thread is still alive, but is currently not eligible to run.

5) Terminated
A thread is in terminated or dead state when its run() method exists.

5. How to create thread in java?

Ans: There are two ways to create a thread:

1.By extending Thread class

2.By implementing Runnable interface

Thread class:

Thread class provide constructors and methods to create and perform operations on a thread. Thread class extends Object class and implements Runnable interface.

Commonly used Constructors of Thread class:

- Thread()
- Thread(String name)
- Thread(Runnable r)
- Thread(Runnable r,String name)

Runnable interface:

The Runnable interface should be implemented by any class whose instances are intended to be executed by a thread. Runnable interface have only one method named run().

- public void run(): is used to perform action for a thread.

6. How run method executes for thread in java?

Ans: public void run() :

If this Thread object was instantiated using a separate Runnable target, the run() method is invoked on that Runnable object.

Start delegates to a native method that launches a thread through operating system routines and run() is invoked from within this newly spawned thread. When a standalone application is started a main thread is automatically created to execute the main().

For example Code :

```
public class Test extends Thread {
   public static void main(String[] args) throws Exception
{
    new Thread(new Test()).start();
  throw new RuntimeException("Exception from main thread");
    }
   public void run() {
throw new RuntimeException("Exception from child thread");
   }
}
```

Output :

```
 java.lang.RuntimeException: Exception from child thread
at com.Test.run(Test.java:11)
at java.lang.Thread.run(Thread.java:662)
 java.lang.RuntimeException: Exception from main thread
at com.Test.main(Test.java:8)
```

7.What is difference between sleep and wait method?

Ans:

i. A wait can be woken up by another thread calling notify on the monitor which is being waited on whereas a sleep cannot.
ii. A wait must happen in a block synchronized on the monitor object whereas sleep does not.
iii. Another point is that you call wait on Object itself whereas you call sleep on Thread.
iv. While sleeping a Thread does not release the locks it holds, while waiting releases the lock on the object that wait() is called on.
v. wait() method is used for multi-thread-synchronization and sleep() method is used for time-synchronization.
vi. wait() is the method of Object class, while sleep() is the method of java.lang.Thread class.
vii. wait() is the non-static method, while sleep() is the static method.
viii. wait() method needs to be called from a loop in order to deal with false alarm, while sleep() better not to call from loop.
ix. wait() should be notified by notify() or notifyAll(), while after the specified amount of time sleep() is completed.

8. What is use of notify and notifyAll method?

Ans:

notify() :

When a thread calls notify() method on a particular object, only one thread will be notified which is waiting for the lock or monitor of that object. The thread chosen to notify is random i.e randomly one thread will be selected for notification. Notified thread doesn't get the lock of the object immediately. It gets once the calling thread releases the lock of that object. Until that it will be in blocked state. It will move from blocked state to running state ince it gets the lock.

notifyAll():

When a thred calls notifyAll() method on a particular object, all threads which are waiting for the lock of that objects are notified. All notified threads will move from waiting state to blocked state. All these threads will get the lock of object on a priority basis. The thread which gets the lock of the object moves to running state. The remaining threads will remain in blocked state until they get the object lock.

9. What is use of join and yield method?

Ans:

Use of yield method :

1. Whenever a thread calls java.lang.Thread.yield method, it gives hint to the thread scheduler that it is ready to pause its execution. Thread scheduler is free to ignore this hint.
2. If any thread executes yield method, thread scheduler checks if there is any thread with same or high priority than this thread. If processor finds any thread with higher or same priority then it will move the current thread to Ready/Runnable state and give processor to other thread and if not current thread will keep executing.

Use of join method :

1. java.lang.Thread class provides the join() method which allows one thread to wait until another thread completes its execution. If t is a thread object whose thread is currently executing, then t.join(); it causes the current thread to pause its execution until thread it join completes its execution.
2. If thread is interrupted then it will throw interruptedexception.
3. If multiple threads calling join() methods it means overloading on join allows to specify a waiting period. As with sleep, join is dependent on the os for timing.

10. What is use of synchronized keyword?

Ans: The java synchronized keyword is an essential tool in concurrent programming in java. Its overall purpose is to only allow one thread at a time into particular section of code thus allowing us to protect, for example, variables or data from being corrupted by simultaneous modifications from being corrupted by simultaneous modifications from different threads.