**How do you define a thread by extending a Thread?**

**class** MyThread **extends** Thread {

}

Refer to ThreadDemo.java for more details

**What is Thread Scheduler?**

This is part of JVM, it is responsible to schedule threads i.e if multiple threads are waiting to get the chance of execution then in which order threads will be executed is decided by thread-scheduler.

We can’t expect exact algorithm followed by thread-scheduler it is varied from JVM to JVM. Hence we can’t expect thread-execution order and exact output.

Hence whenever situation comes to multi-threading there is no gurantee for exact output but we can provide several possible outputs.

**What is the difference between myThread.start() and myThread.run()?**

myThread.start() - a new thread will be created which is responible for the execution of run method.

MyThread.run() no new thread will be created and main thread is responsible by execute just as a normal method call.

**What is importance of Thread class start method?**

It is responsible to

1. Register the thread with thread-scheduler and perform all other mandatory activities.

2. Invoke the run() method.

**Is overloading of run method?**

Yes possible, but we have invoke it as normal method

**What is illegalThreadStateException?**

After starting a thread if we are trying to re-start the same thread, then we will get run-time exception saying illegalThreadStateException.

**public class** IllegalThreadStateTest {  
 **public static void** main(String[] args) {  
 MyThread2 myThread2 = **new** MyThread2();  
 myThread2.start();  
 myThread2.start();  
 }  
 }  
  
 **class** MyThread2 **extends** Thread {  
 }

Simple Lifecycle

New -> Ready -> Running -> Dead

How do you define a thread using Runnable interface?

**class** MyThread11 **implements** Runnable {

**public void** run() {

}

}

Refer to RunnableThread.java for more information

What are different constructor of Thread class?

Their are 8 ways you can create thread in Java. Listed below

Thread thread1 = **new** Thread();

Thread thread2 = **new** Thread(**"name"**);

Thread thread3 = **new** Thread(myThread11);

Thread thread4 = **new** Thread(myThread11, **"name"**);

Thread thread5 = **new** Thread(**new** ThreadGroup(**"A"**), **"name"**);

Thread thread6 = **new** Thread(**new** ThreadGroup(**"A"**), myThread11);

Thread thread7 = **new** Thread(**new** ThreadGroup(**"A"**), myThread11, **"name"**);

Thread thread8 = **new** Thread(**new** ThreadGroup(**"A"**), myThread11, **"name"**, 5);

What is thread name?

Every thread in java has some name, it may default name generated by JVM or customized name provided by programmer.

We can get and set names of a thread.

Refer ThreadNames.java

What is Thread priorities?

Every thread in java has some proiority, it may be default priority genarated by JVM or customized priority provided by programmer.

What is the valid Thread priority range?

The valid thread priority is 1 to 10

**public static final int** MIN\_PRIORITY = 1;

**public static final int** NORM\_PRIORITY = 5;

**public static final int** MAX\_PRIORITY = 10;

Valid priorities

Thread.LOW\_PRIORITY **//invalid**

Thread.HIGH\_PRIORITY **//invalid**

Thread.MIN\_PRIORITY  **//**valid

Thread.NORM\_PRIORITY //valid

**Who is going to use this Thread Priorities?**

Thread scheduler will use priorities while allocating processor. The thread which is having highest priority will get the chance first.

If two threads having same priority, then we can’t expect exact execution order, it depends on thread-scheduler.

**How you get and set priority of a thread?**

Thread class defines the following methods to get and set priority of a thread.

public final int getPriority();

public final void setPriority(int b);

Allowed values range 1 to 10. Otherwise runtime Exception, IllegalArgumentException

Example:

t.setPriority(7); //valid

t.setPriority(17); //illegal

**What is default thread-priorties for threads?**

The default priority only for main thread is five. But for remaining threads default priority will be inherited from parent to child. i.e Whatever priority parent thread has, the same priority will be their for the child thread.

Refer PriorityTest.java

We can change the priorities and change the execution order of our program

Refer PriorityTest1.java

How you can able to prevent the thread execution?

Using

1. yield()

2. join()

3. sleep()

Define yield method?

Yield method causes to pause current executing thread, to give the chance for waiting thread of same-priority. If there is no waiting thread or all waiting threads have low priority then same thread can continue it execution.

Define life-cycle after yield method?

Normal life-cycle:

**MT mT = new MT()[NEW/BORN] ---> mt.start()[RUNNABLE] --> if TS allocates processor --> [RUNNING] --> if run() is complete --> [DEAD]**

If the thread calls yield method the thread will goes from **RUNNING** state to **RUNNABLE** state.

Refer ThreadYieldDemo.java for more details

Some platform would not give support for yield.

**Define join method with example and lifecycle?**

If a thread want to wait until the completion of some other thread, then we should go for **join()** method.

For example: If a thread **t1**  want to wait until completing **t2.** Then **t1** has to call **t2.join().**

If t1 executes t2.join(), then immediately **t1** will be entered into waiting state until **t2.completes. Once t2 completes t1 can continue its execution.**

**public final void join() throws IE**

**public final void join(long milliSeconds) throws IE //join using time**

**public final void join(long milliSeconds, int nano) throws IE// using nano**

**Lifecycle:**

If a running method calls join(), the thread will enter into waiting state.

The waiting thread will come to **Runnable** state,

if t2 is complete

if time expires

if waiting thread got interrupted

Refer ThreadJoinDemo.java and ThreadJoinDemo1.java

**What happen if I call join on the current thread?**

**DeadLock: Refer ThreadJoinDemo2.java for more details**

**Define sleep() method?**

**If a Thread don’t want to perform any operation, for a particular amount of time then we can go for sleep() method.**

**Complete signature of sleep**

**public static native void sleep(long ms) throws InterruptedException**

**public static void sleep(long ms, int nano) throws InterruptedException**

**Every sleep method throws InterruptedException, which is checked exception.**

**Lifecycle:**

If a running method calls sleep(ms), the thread will enter into sleeping state.

The waiting thread will come to **Runnable** state,

if time expires

if waiting thread got interrupted

Refer SlideRotator.java for more details

**How a thread can interupt another thread?**

A thread can interrupt a sleeping thread or waiting thread by using **interrupt()** of thread class.

public void interrupt()

Refer InterruptedExceptionTest.java for more details

**If other thread is not sleeping and u r interrupt, what will happen?**

Interrupt call will wait...:)

Compare yield, join, sleep

|  |  |  |  |
| --- | --- | --- | --- |
| Property | Yield | Join | Sleep |
| Purpose | To pause it execution, and give oppurtunity for other thread of same priroty | Wait for another thread | When a thread don’t want to perform any operation for some amount of time. |
| Overloaded | No(only one) | Yes | Yes |
| Final | No | Yes | No |
| Throw IE | No | Yes | Yes |
| Native | Yes | No | 1 method is native |
|  |  |  |  |