Java Thread Priorities

Priority of a thread describes how early it gets executed and selected by the thread scheduler. In Java, when we create a thread, always a priority is assigned to it. In a Multithreading environment, the processor assigns a priority to a thread scheduler. The priority is given by the JVM or by the programmer itself explicitly. The range of the priority is between 1 to 10 and there are three constant variables which are static and used to fetch priority of a Thread. They are as following:

1. public static int MIN_PRIORITY

It holds the minimum priority that can be given to a thread. The value for it is 1.

2. public static int NORM_PRIORITY

It is the default priority that is given to a thread if it is not defined. The value for this is 5.

3. public static int MAX_PRIORITY

It is the maximum priority that can be given to a thread. The value for it is 10.

Get and Set methods in Thread priority

1. public final int getPriority()

In Java, getPriority() method is in java.lang.Thread package. it is used to get the priority of a thread.

2. public final void setPriority(int newPriority)

In Java setPriority(int newPriority) method is in java.lang.Thread package. It is used to set the priority of a thread. The setPriority() method throws IllegalArgumentException if the value of new priority is below minimum and above maximum limit.

Example 1 : Fetch Thread Priority

If we don’t set thread priority of a thread then by default it is set by the JVM. In this example, we are getting thread’s default priority by using the getPriority() method.

```
class MyThread extends Thread
{
	public void run()
	{
		System.out.println("Thread Running...");
	}
	public static void main(String[]args)
	{
		MyThread p1 = new MyThread();
		MyThread p2 = new MyThread();
		MyThread p3 = new MyThread();
		p1.start();
		System.out.println("P1 thread priority : " + p1.getPriority());
		System.out.println("P2 thread priority : " + p2.getPriority());
		System.out.println("P3 thread priority : " + p3.getPriority());
	}
}
```

Output:

```
P1 thread priority : 5
Thread Running...
P2 thread priority : 5
P3 thread priority : 5
```

Example 2 : Thread Constants

We can fetch priority of a thread by using some predefined constants provided by the Thread class. these constants return the max, min and normal priority of a thread.

```
class MyThread extends Thread
{
	public void run()
	{
		System.out.println("Thread Running...");
	}
	public static void main(String[]args)
	{
		MyThread p1 = new MyThread();
		p1.start();
		System.out.println("max thread priority : " + p1.MAX_PRIORITY);
		System.out.println("min thread priority : " + p1.MIN_PRIORITY);
	    System.out.println("normal thread priority : " +p1.NORM_PRIORITY);
	}
}
```

Output:

```
Thread Running...
max thread priority : 10
min thread priority : 1
normal thread priority : 5
```

Example : Set Priority

To set priority of a thread, setPriority() method of thread class is used. It takes an integer argument that must be between 1 and 10.

```
class MyThread extends Thread
{
	public void run()
	{
		System.out.println("Thread Running...");
	}
	public static void main(String[]args)
	{
		MyThread p1 = new MyThread();
		// Starting thread
		p1.start();
		// Setting priority
		p1.setPriority(2);
		// Getting priority
		int p = p1.getPriority();
		System.out.println("thread priority : " + p);
	}
}
```

Output:

```
thread priority : 2
Thread Running...
```

Example:

In this example, we are setting priority of two threads and running them to see the effect of thread priority. Does setting higher priority thread get CPU first? See the below example.

```
class MyThread extends Thread
{
	public void run()
	{
System.out.println("Thread Running... "+Thread.currentThread().getName());
	}
	public static void main(String[]args)
	{
		MyThread p1 = new MyThread();
		MyThread p2 = new MyThread();
		// Starting thread
		p1.start();
		p2.start();
		// Setting priority
		p1.setPriority(2);
		p2.setPriority(1);
		// Getting -priority
		System.out.println("first thread priority : " + p1.getPriority());
		System.out.println("second thread priority : " + p2.getPriority());
	}
}
```

Output:

```
Thread Running... Thread-0
```

first thread priority : 5

second thread priority : 1

Thread Running... Thread-1

Note: Thread priorities cannot guarantee that a higher priority thread will always be executed first than the lower priority thread. The selection of the threads for execution depends upon the thread scheduler which is platform dependent.