Java Thread Priority in Multithreading

In a Multi threading environment, thread scheduler assigns processor to a thread based on priority of thread. Whenever we create a thread in Java, it always has some priority assigned to it. Priority can either be given by JVM while creating the thread or it can be given by programmer explicitly.
Accepted value of priority for a thread is in range of 1 to 10.

There are 3 static variables defined in Thread class for priority.

public static int MIN_PRIORITY: This is minimum priority that a thread can have. Value for this is 1.
public static int NORM_PRIORITY: This is default priority of a thread if do not explicitly define it. Value for this is 5.
public static int MAX_PRIORITY: This is maximum priority of a thread. Value for this is 10.

Get and Set Thread Priority:

1. **public final int getPriority():** java.lang.Thread.getPriority() method returns priority of given thread.
2. **public final void setPriority(int newPriority):** java.lang.Thread.setPriority() method changes the priority of thread to the value newPriority. This method throws IllegalArgumentException if value of parameter newPriority goes beyond minimum(1) and maximum(10) limit.

Examples of getPriority() and setPriority()

```
import java.lang.*;

class ThreadDemo extends Thread
{
    public void run()
    {
        System.out.println("Inside run method");
    }

    public static void main(String[]args)
    {
        ThreadDemo t1 = new ThreadDemo();
        ThreadDemo t2 = new ThreadDemo();
        ThreadDemo t3 = new ThreadDemo();

        System.out.println("t1 thread priority : " +
                              t1.getPriority()); // Default 5
        System.out.println("t2 thread priority : " +
                              t2.getPriority()); // Default 5
        System.out.println("t3 thread priority : " +
                              t3.getPriority()); // Default 5

        t1.setPriority(2);
        t2.setPriority(5);
        t3.setPriority(8);

        // t3.setPriority(21); will throw IllegalArgumentException
        System.out.println("t1 thread priority : " +
                              t1.getPriority());  //2
        System.out.println("t2 thread priority : " +
                              t2.getPriority()); //5
        System.out.println("t3 thread priority : " +
                              t3.getPriority());//8

        // Main thread
        System.out.print(Thread.currentThread().getName());
        System.out.println("Main thread priority : "
                       + Thread.currentThread().getPriority());

        // Main thread priority is set to 10
        Thread.currentThread().setPriority(10);
        System.out.println("Main thread priority : "
                       + Thread.currentThread().getPriority());
    }
}
```

```
Output:
t1 thread priority : 5
t2 thread priority : 5
t3 thread priority : 5
t1 thread priority : 2
t2 thread priority : 5
t3 thread priority : 8
Main thread priority : 5
Main thread priority : 10
```

Note:

- Thread with highest priority will get execution chance prior to other threads. Suppose there are 3 threads t1, t2 and t3 with priorities 4, 6 and 1. So, thread t2 will execute first based on maximum priority 6 after that t1 will execute and then t3.

- Default priority for main thread is always 5, it can be changed later. Default priority for all other threads depends on the priority of parent thread.

Example:

```
// Java program to demonstrat ethat a child thread
// gets same priority as parent
import java.lang.*;

class ThreadDemo extends Thread
{
    public void run()
    {
        System.out.println("Inside run method");
    }

    public static void main(String[]args)
    {
        // main thread priority is 6 now
        Thread.currentThread().setPriority(6);

        System.out.println("main thread priority : " +
                   Thread.currentThread().getPriority());

        ThreadDemo t1 = new ThreadDemo();

        // t1 thread is child of main thread
        // so t1 thread will also have priority 6.
        System.out.println("t1 thread priority : "
                                  + t1.getPriority());
    }
}
```

Output:

```
Main thread priority : 6
t1 thread priority : 6
```

- If two threads have same priority then we can’t expect which thread will execute first. It depends on thread scheduler’s algorithm(Round-Robin, First Come First Serve, etc)
- If we are using thread priority for thread scheduling then we should always keep in mind that underlying platform should provide support for scheduling based on thread priority.
- Some OS won’t provide proper support for thread priority.