UNIT 8: THREADS

UNIT STRUCTURE

8.1 Learning Objectives

8.2 Introduction

8.3 Introduction to New Thread

8.4 Creating New Threads by Extending the Thread Class

8.5 Creating a Thread by Implementing Runnable Interface Static Members

8.6 Threads in the Running State

8.7 Sleeping and Interruptions

8.8 Signaling with Wait and Notify

8.9 Let Us Sum Up

8.10 Further Reading

8.11 Answer to Check Your Progress

8.12 Model Questions

8.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn about thread in Java
- describe process of creation of thread
- learn about thread State

8.2 INTRODUCTION

In the previous unit, we have discussed the concept of packages and interface in Java programming language. This unit will introduce you to the concept of threads. We will discuss how to create a new thread at the beginning and various concept associated with threads.

8.3 INTRODUCTION TO NEW THREAD

Thread is the smallest independent unit of a process, it is a lightweight subprocess. While executing a Java program, it follows a path

and that path or the direction followed by a Java program during execution is called a thread. Every Java program has at least one thread.

A thread when created, by default, is a new thread. The lifecycle of a thread starts with a new thread state. The codes are not executed or initiated in this state.

A thread in Java can be created in the following two ways:

- Extending java.lang.Thread class
- Implementing Runnable interface

The number of services assigned to a given thread is referred to as its priority. Any thread generated in the JVM is given a priority. The priority scale runs from 1 to 10.

- 1 is known as the lowest priority.
- 5 is known as standard priority.
- represents the highest level of priority.

8.4 CREATING NEW THREADS BY EXTENDING THE THREAD CLASS

Threads can be created by extending the thread class. The thread class is extended by creating a new class.

Program 8.1:

```
import java.io.*;
import java.util.*;
public class ABC extends Thread
{
      // initiated run method for Thread
public void run()
{
System.out.println("Thread Started Running...");
}
public static void main(String[] args)
{
      ABC g1 = new ABC();
```

```
            // invoking Thread
            g1.run();
        }
    }
```



CHECK YOUR PROGRESS

Q 1: What is valid constructors for Thread?

.....

Q 2: What is the name of the method used to start a thread execution?

.....

Q 3: What is run() method in thread?

.....

8.5 CREATING A THREAD BY IMPLEMENTING RUNNABLE INTERFACE STATIC MEMBERS

Threads can also be created by implementing interfaces.

Program 8.2:

```
import java.io.*;
import java.util.*;
public class ABC implements Runnable
 {
      // method to start Thread
      public void run()
      {
            System.out.println(
                  "Thread is Running Successfully");
      }
      public static void main(String[] args)
      {
            ABC g1 = new ABC();
            // initializing Thread Object
```

```
                Thread t1
                      = new Thread(g1);
                t1.run();
            }
        }
```

8.6 THREADS IN THE RUNNING STATE

When the processor assigns a time slot to the thread, it enters the running state. A thread from the runnable state is chosen for execution by the thread scheduler and the process to run the thread starts in this state. Only a thread from a runnable state could enter the running state. The CPU allots the time for execution of the thread in the running state.

The runnable state of a thread is a state in which the thread is ready to run is said to be in a Runnable state It is the responsibility of the thread scheduler to give the thread, time to run.

It is the second phase of a new-born thread is the execution phase. When the start() method is called on a the new instance of a thread, it enters into a runnable state. In the runnable state, thread is ready for execution and is waiting for availability of the processor (CPU time). There are many threads that are ready for execution, they all are waiting in a queue (line). If all threads have equal priority, a time slot is assigned for each thread execution on the basis of first-come, first-serve manner by CPU. The process of allocating time to threads is known as time slicing. A thread can come into runnable state from running, waiting, or new states.

8.7 SLEEPING AND INTERUPT

Sleep is an important method to help the other threads to get executed, that is to make processing time available for other threads in the process. The sleep() method makes the current thread stop and suspend its execution for a specific period of time.

Program 8.3:

```
// important import statements
```

```
import java.lang.Thread;
import java.io.*;
public class TestSleepMethod2
{
   // main method
   public static void main(String argvs[])
   {
    try
    {
      for (int j = 0; j < 5; j++)
      {
       // The main thread sleeps for the 100
       //milliseconds, which is 1 sec
       // whenever the loop runs
        Thread.sleep(100);
       // displaying the value of the variable
       System.out.println(j);
     }
   }
   catch (Exception expn)
   {
    // catching the exception
    System.out.println(expn);
   }
  }
}
```

Interrupt is also an important method in thread execution, as it helps to interrupt the thread in execution. If a thread is sleeping or waiting, the interrupt() method helps to call back the thread.

Program 8.4:

```
class TestInterruptingThread2 extends Thread
{
        public void run()
```

```
{
Try
{
   Thread.sleep(100);
   System.out.println("task");
}
catch(InterruptedException e)
{
    throw new RuntimeException("Thread
     interrupted..."+e);
}

}
public static void main(String args[])
{
    TestInterruptingThread2 t2=new
    TestInterruptingThread2();
    t2.start();
    Try
    {
        t1.interrupt();
    }
    catch(Exception e)
    {
      System.out.println("Exception handled "+e);
   }
  }
}
```

8.8 SIGNALING WITH WAIT AND NOTIFY

In Java programming, when the wait() method is called on an object, the thread becomes inactive.it forces the current thread to wait, until it is

called.

notify() is the method which when called on an object activates the wait() in the thread. notify() wakes up the waiting threads.

Two threads having access to the same instance of a class can be used to pass signals. Here is an example where, one thread will call the doWait() method, and the other thread will call doNotify() method :

```
public class MachineObject
{
}
public class MyWaitNotify
{
MonitorObject myMachineObject = new MachineObject();
public void doWait()
{
    synchronized(myMachineObject)
{
      Try
{
        myMachineObject.wait();
}
catch(InterruptedException e){...}
}
}
public void doNotify()
{
    synchronized(myMachineObject)
{
      myMachineObject.notify();
   }
  }
 }
```



CHECK YOUR PROGRESS

Q 5: When the event for which a thread is blocked occurs?

.....

Q 6: What is the resume()?

.....

Q 7: Thread synchronization is required because all threads of a process share the same address space (True or False)

.....

Q 8: What is wait () method in thread?

.....

Q 9: What is notify() method?

.....



8.9 LET US SUM UP

- Thread is the smallest independent unit of a process, it is a lightweight subprocess. While executing a Java program, it follows a path and that path or the direction followed by a Java program during execution is called a thread. Every Java program has at least one thread.
- A thread in Java can be created in the following two ways:
  - Extending java.lang.Thread class
  - Implementing Runnable interface
- Threads can be created by extending the thread class. The thread class is extended by creating a new class.
- Threads can also be created by implementing interfaces.
- When the processor assigns a time slot to the thread, it enters the running state. A thread from the runnable state is chosen for execution by the thread scheduler and the process to run the thread starts in this state. Only a thread from a runnable state could enter the running state. The CPU allots the time for execution of the thread in the running state.
- The runnable state of a thread is a state in which the thread is ready to

run is said to be in a Runnable state It is the responsibility of the thread scheduler to give the thread, time to run.

- In java programming, when the wait() method is called on an object, the thread becomes inactive.it forces the current thread to wait, until it is called.
- notify() is the method which when called on an object activates the wait() in the thread. notify() wakes up the waiting threads.
- Sleep is an important method to help the other threads to get executed, that is to make processing time available for other threads in the process. The sleep() method makes the current thread stop and suspend its execution for a specific period of time.
- Interrupt is also an important method in thread execution, as it helps to interrupt the thread in execution. If a thread is sleeping or waiting, the interrupt() method helps to call back the thread.



8.10 FURTHER READINGS

1) Schildt, H. (2020). The complete reference Java.



8.11 ANSWER TO CHECK YOUR PROGRESS

Ans to Q No 1: Thread()

Ans to Q No 2: Start()

Ans to Q No 3: The run() method of a thread is like the main() method to an application. Starting the thread causes the object's run method to be called in that separately executing thread.

Ans to Q No 4: When thread moves to the ready queue.

Ans to Q No 5: The resume() method is deprecated. It resumes a suspended thread.

Ans to Q No 6: True

Ans to Q No 7: In java programming, when the wait() method is called on an object, the thread becomes inactive.it forces the current thread to wait, until it is called.

Ans to Q No 8: notify() is the method which when called on an object activates the wait() in the thread. notify() wakes up the waiting threads.



8.12 MODEL QUESTIONS

Q 1: What is thread in Java?

Q 2: How thread is intiate in Java?

Q 3: Explain the wait() and notify() in Java thread.

Q 4: Explain about the Java Thread Priorities.

*** ***** ***