**Lab Sheet3**. Create a single thread by implementing Runnable interface.

**Aim:** The aim of this Java program is to demonstrate the creation and execution of a new thread using the **Runnable** interface. It shows how to create a thread, assign a task to it, and run the task in parallel with the main thread.

**Algorithm:**

1. Create a class named **Task** that implements the **Runnable** interface.
2. Inside the **run()** method of the **Task** class, print a message using **System.out.println()**. This message will be printed by a separate thread.
3. Create a **TestThread** class with a **main** method.
4. Inside the **main** method, create an instance of the **Task** class named **task**.
5. Create a new **Thread** instance named **t1**, passing the **task** object as a parameter to its constructor.
6. Set the name of the **t1** thread to "first" using the **setName** method.
7. Start the **t1** thread using the **start()** method. This will cause the **run()** method of the **Task** class to be executed in a separate thread.
8. Print a message using **System.out.println()** in the main thread.

**Program Explanation:**

In this program, we create a new thread using the **Runnable** interface. The **Task** class implements the **Runnable** interface, which requires the implementation of the **run()** method. The **run()** method contains the code that will run in a separate thread.

In the **main** method of the **TestThread** class, we perform the following steps:

1. Create an instance of the **Task** class named **task**.
2. Create a new thread named **t1** and pass the **task** object to its constructor. This associates the **run()** method in the **task** object with the **t1** thread.
3. Set the name of the **t1** thread to "first" using the **setName** method.
4. Start the **t1** thread using the **start()** method. This starts the execution of the **run()** method in the **Task** class in a separate thread.
5. In the main thread, we print "printing" and "to Java" using **System.out.println()**.

The key point is that the messages "printing" and "to Java" will be printed by the main thread, while the message "Welcome" in the **run()** method of the **Task** class will be printed by the **t1** thread. This demonstrates the concurrent execution of code in multiple threads.

Top of Form

import java.lang.\*; //optional

class Task implements Runnable {

public void run() {

System.out.println(Thread.currentThread()+" printing ");

System.out.println("Welcome");

}

}

public class TestThread {

public static void main(String[] args) {

Task task = new Task();

Thread t1= new Thread(task);

t1.setName("first");

t1.start();

System.out.println(Thread.currentThread()+" printing ");

System.out.println("to Java");

}

}

**Output:**

Thread[main,5,main] printing

to Java

Thread[first,5,main] printing

Welcome

**Q2**. Create a single thread by extending Thread class

**Aim:** The aim of this Java program is to demonstrate the creation and execution of a new thread by extending the **Thread** class. It illustrates how to create a custom thread class by extending **Thread** and overriding its **run()** method to execute a specific task in a separate thread.

**Algorithm:**

1. Create a custom class **MyThread** that extends the **Thread** class.
2. Inside the **run()** method of the **MyThread** class, define the actions you want the thread to perform. In this case, calculate the sum of two numbers, print a message indicating the thread has started, display the result, and indicate that the thread has completed its work.
3. Create a **TestThread** class with a **main** method.
4. Inside the **main** method:
   * Print a message indicating that the main thread has started.
   * Create an instance of the **MyThread** class and name it **t**.
   * Set the name of the **t** thread to "first" using the **setName** method.
   * Start the **t** thread using the **start()** method, which will invoke the **run()** method of the **MyThread** class.
   * Print a message indicating that the main thread has completed.

**Program Explanation:**

In this program, we create a custom thread class **MyThread** by extending the **Thread** class. The **MyThread** class overrides the **run()** method to specify the tasks that the thread should perform. These tasks include:

1. Calculating the sum of two numbers, **a** and **b**.
2. Printing a message indicating that the thread has started running.
3. Displaying the result of the addition.
4. Printing a message indicating that the thread has completed its work.

In the **main** method of the **TestThread** class, we perform the following actions:

1. Print a message indicating that the main thread has started.
2. Create an instance of the **MyThread** class named **t**.
3. Set the name of the **t** thread to "first" using the **setName** method.
4. Start the **t** thread by calling the **start()** method. This initiates the execution of the **run()** method of the **MyThread** class in a separate thread.
5. Print a message indicating that the main thread has completed.

The key point to understand is that the tasks specified in the **run()** method of the **MyThread** class are executed in parallel with the main thread. This demonstrates how to create and run custom threads in Java by extending the **Thread** class.

Top of Form

Program:

class MyThread extends Thread {

// run() method to perform action for thread.

public void run()

{

int a= 10;

int b=12;

int result = a+b;

System.out.println(Thread.currentThread()+" started running..");

System.out.println("Sum of two numbers is: "+ result);

System.out.println(Thread.currentThread()+" completed..");

}

}

public class TestThread {

public static void main( String args[] )

{

System.out.println(Thread.currentThread()+" started");

// Creating instance of the class extend Thread class

MyThread t = new MyThread();

t.setName("first");

//calling start method to execute the run() method of the Thread class

t.start();

System.out.println(Thread.currentThread()+" completed");

}

}

**Output:**

Thread[main,5,main] started

Thread[main,5,main] completed

Thread[first,5,main] started running..

Sum of two numbers is: 22

Thread[first,5,main] completed..