Naming a thread and fetching name of current thread in Java

Difficulty Level : Easy Last Updated : 18 Dec, 2021

Thread can be referred to as a lightweight process. Thread uses fewer resources to create and exist in the process; thread shares process resources. The main thread of Java is the thread that is started when the program starts. now let us discuss the eccentric concept of with what ways we can name a thread.

Methods: There are two ways by which we can set the name either be it directly or indirectly which we will be peeking through.

1. Creating the thread and Passing the thread's name (Direct method)
2. Using setName() method of Thread class (Indirect Method)

Method 1: Creating the thread and passing the thread's name

It is a direct method of naming threads in java, each thread has a name that is: Thread-0, Thread-1, Thread-2,….so on. Java provides some methods to change the thread name. There are basically two methods to set the thread name. Both methods are defined in java.lang.Thread class.

Geek, now you must be wondering how to set the thread's name directly? In java we can set the thread name at the time of creating the thread and bypassing the thread's name as an argument as shown in the below example as follows:

Example:

```
// Java Program Illustrating How to Set the name
// of Thread at time of Creation

// Importing I/O classes from java.io package
import java.io.*;

// Class 1
// Helper class
class ThreadNaming extends Thread {

    // Parametrized constructor
    ThreadNaming(String name)
```

```
    {
        // Call to constructor of
        // the Thread class as super keyword
        // refers to parent class
        super(name);
    }

    // run() method for thread
    @Override public void run()
    {
        // Print statement when thread is called inside
        // main()
        System.out.println("Thread is running.....");
    }
}

// Class 2
// Main class
class GFG {

    // main driver method
    public static void main(String[] args)
    {

        // Creating two threads
        ThreadNaming t1 = new ThreadNaming("geek1");
        ThreadNaming t2 = new ThreadNaming("geek2");

        // Getting the above created threads names.
        System.out.println("Thread 1: " + t1.getName());
        System.out.println("Thread 2: " + t2.getName());

        // Starting threads using start() method
        t1.start();
        t2.start();
    }
}
```

Output

```
Thread 1: geek1
Thread 2: geek2
Thread is running.....
Thread is running.....
```

Way 2: Using setName() method of Thread class

We can set(change) the thread's name by calling the setName method on that thread object. It will change the name of a thread.

Syntax:

```
public final void setName(String name)
```

Parameter: A string that specifies the thread name

Example:

```
// Java Program Illustrating How to Get and Change the
// Name of a Thread

// Importing input output classes
import java.io.*;

// Class 1
// Helper class extending Thread class
class ThreadNaming extends Thread {

    // run() method for thread which is called
    // as soon as start() is called over threads
    @Override public void run()
    {

        // Print statement when run() is called over thread
        System.out.println("Thread is running.....");
    }
}

// Class 2
// Main class
class GFG {

    // Main driver method
    public static void main(String[] args)
    {

        // Creating two threads via above class
        // as it is extending Thread class
        ThreadNaming t1 = new ThreadNaming();
        ThreadNaming t2 = new ThreadNaming();

        // Fetching the above created threads names
        // using getName() method
        System.out.println("Thread 1: " + t1.getName());
```

```
            System.out.println("Thread 2: " + t2.getName());

            // Starting threads using start() method
            t1.start();
            t2.start();

            // Now changing the name of threads
            t1.setName("geeksforgeeks");
            t2.setName("geeksquiz");

            // Again getting the new names of threads
            System.out.println(
                "Thread names after changing the "
                + "thread names");

            // Printing the above names
            System.out.println("New Thread 1 name:  "
                               + t1.getName());
            System.out.println("New Thread 2 name: "
                               + t2.getName());
        }
    }
```

Output

```
Thread 1: Thread-0
Thread 2: Thread-1
Thread is running.....
Thread names after changing the thread names
New Thread 1 name:  geeksforgeeks
New Thread 2 name: geeksquiz
Thread is running.....
```

How to fetch the name of the current thread?

Now let us dwell on fetching the name of the current thread. We can fetch the current thread name at the time of creating the thread and bypassing the thread's name as an argument.

Method: currentThread()

It is defined in java.langThread class.

Return Type: It returns a reference to the currently executing thread

Syntax:

```
public static Thread currentThread()
```

Example:

```
// Java program to Illustrate How to Get Name of
// Current Executing thread
// Using getName() Method

// Importing reqiored I/O classes
import java.io.*;

// Class 1
// Helper class extending to Thread class
class ThreadNaming extends Thread {

    // run() method for this thread
    @Override public void run()
    {
        // Display message
        System.out.println(
            "Fetching current thread name..");

        // Getting the current thread name
        // using getname() method
        System.out.println(
            Thread.currentThread().getName());
    }
}

// Class 2
// Main class
class GFG {

    // Main method driver
    public static void main(String[] args)
    {

        // Creating two threads inside main() method
        ThreadNaming t1 = new ThreadNaming();
        ThreadNaming t2 = new ThreadNaming();

        // Starting threads using start() method which
        // automatically calls run() method
        t1.start();
        t2.start();
    }
}
```

Output

```
Fetching current thread name..
Thread-0
Fetching current thread name..
Thread-1
```

This article is contributed by **Nitsdheerendra**. If you like GeeksforGeeks and would like to contribute, you can also write an article using write.geeksforgeeks.org or mail your article to review-team@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.