ThreadGroup in Java

Java provides a convenient way to group multiple threads in a single object. In such a way, we can suspend, resume or interrupt a group of threads by a single method call.

Note: Now suspend(), resume() and stop() methods are deprecated.

Java thread group is implemented by *java.lang.ThreadGroup* class.

A ThreadGroup represents a set of threads. A thread group can also include the other thread group. The thread group creates a tree in which every thread group except the initial thread group has a parent.

A thread is allowed to access information about its own thread group, but it cannot access the information about its thread group's parent thread group or any other thread groups.

Constructors of ThreadGroup class

There are only two constructors of ThreadGroup class.

No.	Constructor	Description
1)	ThreadGroup(String name)	creates a thread group with given name.
2)	ThreadGroup(ThreadGroup parent, String name)	creates a thread group with a given parent group and name.

Methods of ThreadGroup class

There are many methods in ThreadGroup class. A list of ThreadGroup methods is given below.

S.N.	Modifier and Type	Method	Description
1)	void	checkAccess()	This method determines if the currently running thread has permission to modify the thread group.
2)	int	activeCount()	This method returns an estimate of the number of active threads in the thread group and its subgroups.
3)	int	activeGroupCount()	This method returns an estimate of the number of active groups in the thread group and its subgroups.
4)	void	destroy()	This method destroys the thread group and all of its subgroups.
5)	int	enumerate(Thread[] list)	This method copies into the specified array every active thread in the thread group and its subgroups.
6)	int	getMaxPriority()	This method returns the maximum priority of the thread group.
7)	String	getName()	This method returns the name of the thread group.
8)	ThreadGroup	getParent()	This method returns the parent of the thread group.
9)	void	interrupt()	This method interrupts all threads in the thread group.
10)	boolean	isDaemon()	This method tests if the thread group is a daemon thread group.
11)	void	setDaemon(boolean daemon)	This method changes the daemon status of the thread group.
12)	boolean	isDestroyed()	This method tests if this thread group has been destroyed.

13)	void	list()	This method prints information about the thread group to the standard output.
14)	boolean	parentOf(ThreadGroup g	This method tests if the thread group is either the thread group argument or one of its ancestor thread groups.
15)	void	suspend()	This method is used to suspend all threads in the thread group.
16)	void	resume()	This method is used to resume all threads in the thread group which was suspended using suspend() method.
17)	void	setMaxPriority(int pri)	This method sets the maximum priority of the group.
18)	void	stop()	This method is used to stop all threads in the thread group.
19)	String	toString()	This method returns a string representation of the Thread group.

Let's see a code to group multiple threads.

```
ThreadGroup tg1 = new ThreadGroup("Group A");
Thread t1 = new Thread(tg1,new MyRunnable(),"one");
Thread t2 = new Thread(tg1,new MyRunnable(),"two");
Thread t3 = new Thread(tg1,new MyRunnable(),"three");
```

Now all 3 threads belong to one group. Here, tg1 is the thread group name, MyRunnable is the class that implements Runnable interface and "one", "two" and "three" are the thread names.

Now we can interrupt all threads by a single line of code only.

```
Thread.currentThread().getThreadGroup().interrupt();
```

ThreadGroup Example

File: ThreadGroupDemo.java

```
public class ThreadGroupDemo implements Runnable{
    public void run() {
          System.out.println(Thread.currentThread().getName());
    }
   public static void main(String[] args) {
      ThreadGroupDemo runnable = new ThreadGroupDemo();
          ThreadGroup tg1 = new ThreadGroup("Parent ThreadGroup");

          Thread t1 = new Thread(tg1, runnable,"one");
          t1.start();
          Thread t2 = new Thread(tg1, runnable,"two");
          t2.start();
          Thread t3 = new Thread(tg1, runnable,"three");
          t3.start();

          System.out.println("Thread Group Name: "+tg1.getName());
         tg1.list();

    }
   }
```

Output:

```
one
two
three
Thread Group Name: Parent ThreadGroup
java.lang.ThreadGroup[name=Parent ThreadGroup,maxpri=10]
```

Thread Pool Methods Example: int activeCount()

Let's see how one can use the method activeCount().

FileName: ActiveCountExample.java

```
// code that illustrates the activeCount() method

// import statement
import java.lang.*;



class ThreadNew extends Thread
{
// constructor of the  class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run method
public void run()
{

for (int j = 0; j < 1000; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered " + e);
}
}
}
}

public class ActiveCountExample
```

```
{
// main method
public static void main(String argvs[])
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("The parent group of threads");

ThreadNew th1 = new ThreadNew("first", tg);
System.out.println("Starting the first");

ThreadNew th2 = new ThreadNew("second", tg);
System.out.println("Starting the second");

// checking the number of active thread by invoking the activeCount() method
System.out.println("The total number of active threads are: " + tg.activeCount());
}
}
```

Output:

```
Starting the first
Starting the second
The total number of active threads are: 2
```

Thread Pool Methods Example: int activeGroupCount()

Now, we will learn how one can use the activeGroupCount() method in the code.

FileName: ActiveGroupCountExample.java

```
// Java code illustrating the activeGroupCount() method

// import statement
import java.lang.*;


class ThreadNew extends Thread
{
// constructor of the  class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run() method
public void run()
{

for (int j = 0; j < 100; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered " + e);
}


}

System.out.println(Thread.currentThread().getName() + " thread has finished executing");
}
}

public class ActiveGroupCountExample
{
```

```
// main method
public static void main(String argvs[])
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("The parent group of threads");

ThreadGroup tg1 = new ThreadGroup(tg, "the child group");

ThreadNew th1 = new ThreadNew("the first", tg);
System.out.println("Starting the first");

ThreadNew th2 = new ThreadNew("the second", tg);
System.out.println("Starting the second");

// checking the number of active thread by invoking the activeGroupCount() method
System.out.println("The total number of active thread groups are: " + tg.activeGroupCount());
}
}
```

Output:

```
Starting the first
Starting the second
The total number of active thread groups are: 1
the second thread has finished executing
the first thread has finished executing
```

Thread Pool Methods Example: void destroy()

Now, we will learn how one can use the destroy() method in the code.

FileName: DestroyExample.java

```
// Code illustrating the destroy() method


// import statement
import java.lang.*;



class ThreadNew extends Thread
{
// constructor of the  class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run() method
public void run()
{

for (int j = 0; j < 100; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered " + e);
}


}

System.out.println(Thread.currentThread().getName() + " thread has finished executing");
}
}

public class DestroyExample
{
// main method
public static void main(String argvs[]) throws SecurityException, InterruptedException
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("the parent group");

ThreadGroup tg1 = new ThreadGroup(tg, "the child group");

ThreadNew th1 = new ThreadNew("the first", tg);
System.out.println("Starting the first");

ThreadNew th2 = new ThreadNew("the second", tg);
System.out.println("Starting the second");

// waiting until the other threads has been finished
th1.join();
th2.join();

// destroying the child thread group
tg1.destroy();
System.out.println(tg1.getName() + " is destroyed.");

// destroying the parent thread group
tg.destroy();
System.out.println(tg.getName() + " is destroyed.");
}
}
```

Output:

```
Starting the first
Starting the second
the first thread has finished executing
the second thread has finished executing
the child group is destroyed.
the parent group is destroyed.
```

Thread Pool Methods Example: int enumerate()

Now, we will learn how one can use the enumerate() method in the code.

FileName: EnumerateExample.java

```
// Code illustrating the enumerate() method

// import statement
import java.lang.*;



class ThreadNew extends Thread
{
// constructor of the class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run() method
public void run()
{

for (int j = 0; j < 100; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered " + e);
}


}


System.out.println(Thread.currentThread().getName() + " thread has finished executing");
}
}


public class EnumerateExample
{
// main method
public static void main(String argvs[]) throws SecurityException, InterruptedException
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("the parent group");

ThreadGroup tg1 = new ThreadGroup(tg, "the child group");

ThreadNew th1 = new ThreadNew("the first", tg);
System.out.println("Starting the first");

ThreadNew th2 = new ThreadNew("the second", tg);
System.out.println("Starting the second");

// returning the number of threads kept in this array
Thread[] grp = new Thread[tg.activeCount()];
int cnt = tg.enumerate(grp);
for (int j = 0; j < cnt; j++)
{
System.out.println("Thread " + grp[j].getName() + " is found.");
}
}
}
```

Output:

```
Starting the first
Starting the second
Thread the first is found.
Thread the second is found.
the first thread has finished executing
the second thread has finished executing
```

Thread Pool Methods Example: int getMaxPriority()

The following code shows the working of the getMaxPriority() method.

FileName: GetMaxPriorityExample.java

```
// Code illustrating the getMaxPriority() method

// import statement
import java.lang.*;
```

```
class ThreadNew extends Thread
{
// constructor of the class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run() method
public void run()
{

for (int j = 0; j < 100; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered " + e);
}


}

System.out.println(Thread.currentThread().getName() + " thread has finished executing");
}
}

public class GetMaxPriorityExample
{
// main method
public static void main(String argvs[]) throws SecurityException, InterruptedException
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("the parent group");
ThreadGroup tg1 = new ThreadGroup(tg, "the child group");

ThreadNew th1 = new ThreadNew("the first", tg);
System.out.println("Starting the first");

ThreadNew th2 = new ThreadNew("the second", tg);
System.out.println("Starting the second");

int priority = tg.getMaxPriority();

System.out.println("The maximum priority of the parent ThreadGroup: " + priority);



}
}
```

Output:

```
Starting the first
Starting the second
The maximum priority of the parent ThreadGroup: 10
the first thread has finished executing
the second thread has finished executing
```

Thread Pool Methods Example: ThreadGroup getParent()

Now, we will learn how one can use the getParent() method in the code.

FileName: GetParentExample.java

```
// Code illustrating the getParent() method

// import statement
import java.lang.*;


class ThreadNew extends Thread
{
// constructor of the class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run() method
public void run()
{

for (int j = 0; j < 100; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered" + e);
}

}

System.out.println(Thread.currentThread().getName() + " thread has finished executing");
}
}

public class GetMaxPriorityExample
{
```

```
// main method
public static void main(String argvs[]) throws SecurityException, InterruptedException
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("the parent group");

ThreadGroup tg1 = new ThreadGroup(tg, "the child group");

ThreadNew th1 = new ThreadNew("the first", tg);
System.out.println("Starting the first");

ThreadNew th2 = new ThreadNew("the second", tg);
System.out.println("Starting the second");

// printing the parent ThreadGroup
// of both child and parent threads
System.out.println("The ParentThreadGroup for " + tg.getName() + " is " + tg.getParent().getName());
System.out.println("The ParentThreadGroup for " + tg1.getName() + " is " + tg1.getParent().getName());



}
}
```

Output:

```
Starting the first
Starting the second
The ParentThreadGroup for the parent group is main
The ParentThreadGroup for the child group is the parent group
```

```
the first thread has finished executing
the second thread has finished executing
```

Thread Pool Methods Example: void interrupt()

The following program illustrates how one can use the interrupt() method.

FileName: InterruptExample.java

```
// Code illustrating the interrupt() method

// import statement
import java.lang.*;


class ThreadNew extends Thread
{
// constructor of the class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run() method
public void run()
{

for (int j = 0; j < 100; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered " + e);
}
}

System.out.println(Thread.currentThread().getName() + " thread has finished executing");
}
}

public class InterruptExample
{
// main method
public static void main(String argvs[]) throws SecurityException, InterruptedException
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("the parent group");

ThreadGroup tg1 = new ThreadGroup(tg, "the child group");

ThreadNew th1 = new ThreadNew("the first", tg);
System.out.println("Starting the first");

ThreadNew th2 = new ThreadNew("the second", tg);
System.out.println("Starting the second");

// invoking the interrupt method
tg.interrupt();

}
}
```

Output:

```
Starting the first
Starting the second
The exception has been encountered java.lang.InterruptedException: sleep interrupted
The exception has been encountered java.lang.InterruptedException: sleep interrupted
the second thread has finished executing
the first thread has finished executing
```

Thread Pool Methods Example: boolean isDaemon()

The following program illustrates how one can use the isDaemon() method.

FileName: IsDaemonExample.java

```
// Code illustrating the isDaemon() method

// import statement
import java.lang.*;



class ThreadNew extends Thread
{
// constructor of the class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run() method
public void run()
{

for (int j = 0; j < 100; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered" + e);
}


}

System.out.println(Thread.currentThread().getName() + " thread has finished executing");
}
}

public class IsDaemonExample
{
// main method
public static void main(String argvs[]) throws SecurityException, InterruptedException
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("the parent group");

ThreadGroup tg1 = new ThreadGroup(tg, "the child group");

ThreadNew th1 = new ThreadNew("the first", tg);
System.out.println("Starting the first");

ThreadNew th2 = new ThreadNew("the second", tg);
System.out.println("Starting the second");

if (tg.isDaemon() == true)
{
System.out.println("The group is a daemon group.");
}
else
{
System.out.println("The group is not a daemon group.");
}


}
}
```

Output:

```
Starting the first
Starting the second
The group is not a daemon group.
the second thread has finished executing
the first thread has finished executing
```

Thread Pool Methods Example: boolean isDestroyed()

The following program illustrates how one can use the isDestroyed() method.

FileName: IsDestroyedExample.java

```
// Code illustrating the isDestroyed() method

// import statement
import java.lang.*;



class ThreadNew extends Thread
{
// constructor of the class
ThreadNew(String tName, ThreadGroup tgrp)
{
super(tgrp, tName);
start();
}

// overriding the run() method
public void run()
{

for (int j = 0; j < 100; j++)
{
try
{
Thread.sleep(5);
}
catch (InterruptedException e)
{
System.out.println("The exception has been encountered" + e);
}


}


System.out.println(Thread.currentThread().getName() + " thread has finished executing");
}
}


public class IsDestroyedExample
{
// main method
public static void main(String argvs[]) throws SecurityException, InterruptedException
{
// creating the thread group
ThreadGroup tg = new ThreadGroup("the parent group");


ThreadGroup tg1 = new ThreadGroup(tg, "the child group");


ThreadNew th1 = new ThreadNew("the first", tg);
System.out.println("Starting the first");


ThreadNew th2 = new ThreadNew("the second", tg);
System.out.println("Starting the second");


if (tg.isDestroyed() == true)
{
System.out.println("The group has been destroyed.");
}
else
{
System.out.println("The group has not been destroyed.");
}


}
}
```

Output:

```
Starting the first
Starting the second
The group has not been destroyed.
the first thread has finished executing
the second thread has finished executing
```

← Prev

Next →

Youtube For Videos Join Our Youtube Channel: Join Now

Feedback

- Send your Feedback to feedback@javatpoint.com

Help Others, Please Share

Learn Latest Tutorials

Splunk tutorial SPSS tutorial

Splunk

SPSS

Swagger tutorial

Swagger

T-SQL tutorial

Transact-SQL

Tumblr tutorial

Tumblr

React tutorial

ReactJS

Regex tutorial

Regex

Reinforcement learning tutorial

Reinforcement Learning

R Programming tutorial

R Programming

RxJS tutorial

RxJS

React Native tutorial

React Native

Python Design Patterns

Python Design Patterns

Python Pillow tutorial

Python Pillow

Python Turtle tutorial

Python Turtle

Keras tutorial

Keras

Preparation

Aptitude

Aptitude

Logical Reasoning

Reasoning

Verbal Ability

Verbal Ability

Interview Questions

Interview Questions

Company Interview Questions

Company Questions

Trending Technologies

Artificial Intelligence

AWS Tutorial

AWS

Selenium tutorial

Cloud Computing

Artificial Intelligence

Selenium

Cloud Computing

Hadoop tutorial

Hadoop

ReactJS Tutorial

ReactJS

Data Science Tutorial

Data Science

Angular 7 Tutorial

Angular 7

Blockchain Tutorial

Blockchain

Git Tutorial

Git

Machine Learning Tutorial

Machine Learning

DevOps Tutorial

DevOps

B.Tech / MCA

DBMS tutorial

DBMS

Data Structures tutorial

Data Structures

DAA tutorial

DAA

Operating System

Operating System

Computer Network tutorial

Computer Network

Compiler Design tutorial

Compiler Design

Computer Organization and Architecture

Computer Organization

Discrete Mathematics Tutorial

Discrete Mathematics

Ethical Hacking

Ethical Hacking

Computer Graphics Tutorial

Software Engineering

html tutorial

Web Technology

Computer Graphics

Software Engineering

Cyber Security tutorial

Cyber Security

Automata Tutorial

Automata

C Language tutorial

C Programming

C++ tutorial

C++

Java tutorial

Java

.Net Framework tutorial

.Net

Python tutorial

Python

List of Programs

Programs

Control Systems tutorial

Control System

Data Mining Tutorial

Data Mining

Data Warehouse Tutorial

Data Warehouse