

THREADGROUP.ACTIVEGROUPCOUNT METHOD

http://www.tutorialspoint.com/java/lang/threadgroup_activegroupcount.htm

Copyright © tutorialspoint.com

Description

The **java.lang.ThreadGroup.activeGroupCount** method returns an estimate of the number of active groups in this thread group.

Declaration

Following is the declaration for **java.lang.ThreadGroup.activeGroupCount** method

```
public int activeGroupCount()
```

Parameters

- NA

Return Value

This method returns the number of active thread groups with this thread group as an ancestor.

Exception

- NA

Example

The following example shows the usage of java.lang.ThreadGroup.activeGroupCount method.

```
package com.tutorialspoint;

import java.lang.*;

public class ThreadGroupDemo implements Runnable
{
    public static void main(String[] args) {
        ThreadGroupDemo tg = new ThreadGroupDemo();
        tg.func();
    }

    public void func() {
        try {
            // create a parent ThreadGroup
            ThreadGroup pGroup = new ThreadGroup("Parent ThreadGroup");

            // create a child ThreadGroup for parent ThreadGroup
            ThreadGroup cGroup = new ThreadGroup(pGroup, "Child ThreadGroup");

            // create a thread
            Thread t1 = new Thread(pGroup, this);
            System.out.println("Starting " + t1.getName() + "...");
            t1.start();

            // create another thread
            Thread t2 = new Thread(cGroup, this);
            System.out.println("Starting " + t2.getName() + "...");
            t2.start();

            /* returns the number of active thread groups with this
            thread group as an ancestor */
            System.out.println("Active group(child) threads in \"" +
                pGroup.getName() + "\" = " + pGroup.activeGroupCount());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```

        // block until the other threads finish
        t1.join();
        t2.join();
    }
    catch (InterruptedException ex) {
        System.out.println(ex.toString());
    }
}

// implements run()
public void run() {

    for(int i=0;i<1000;i++) {
        i++;
    }
    System.out.println(Thread.currentThread().getName() +
        " finished executing.");
}
}

```

Let us compile and run the above program, this will produce the following result:

```

Starting Thread-0...
Starting Thread-1...
Active group(child) threads in "Parent ThreadGroup" = 1
Thread-0 finished executing
Thread-1 finished executing

```

Loading [MathJax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js