JAVA.LANG.RUNTIME.ADDSHUTDOWNHOOK*THREADHOOK* METHOD

Description

The **java.lang.Runtime.addShutdownHook*Threadhook*** method registers a new virtual-machine shutdown hook.The Java virtual machine shuts down in response to two kinds of events:

- The program exits normally, when the last non-daemon thread exits or when the exit *equivalently*, *System. exit* method is invoked, or
- The virtual machine is terminated in response to a user interrupt, such as typing ^C, or a system-wide event, such as user logoff or system shutdown.

A shutdown hook is simply an initialized but unstarted thread. When the virtual machine begins its shutdown sequence it will start all registered shutdown hooks in some unspecified order and let them run concurrently. When all the hooks have finished it will then run all uninvoked finalizers if finalization-on-exit has been enabled. Finally, the virtual machine will halt. Note that daemon threads will continue to run during the shutdown sequence, as will non-daemon threads if shutdown was initiated by invoking the exit method.

Declaration

Following is the declaration for **java.lang.Runtime.addShutdownHook** method

```
public void addShutdownHook(Thread hook)
```

Parameters

- **hook** -- An initialized but unstarted Thread object

Return Value

This method does not return a value.

Exception

- **IllegalArgumentException** -- If the specified hook has already been registered, or if it can be determined that the hook is already running or has already been run
- **IllegalStateException** -- If the virtual machine is already in the process of shutting down
- **SecurityException** -- If a security manager is present and it denies RuntimePermission " *shutdownHooks* "

Example

The following example shows the usage of lang.Runtime.addShutdownHook method.

```
package com.tutorialspoint;

public class RuntimeDemo {

   // a class that extends thread that is to be called when program is exiting
   static class Message extends Thread {

   public void run() {
   System.out.println("Bye.");
   }
   }

   public static void main(String[] args) {
   try {
```

```
   // register Message as shutdown hook
   Runtime.getRuntime().addShutdownHook(new Message());

   // print the state of the program
   System.out.println("Program is starting...");

   // cause thread to sleep for 3 seconds
   System.out.println("Waiting for 3 seconds...");
   Thread.sleep(3000);

   // print that the program is closing
   System.out.println("Program is closing...");


   } catch (Exception e) {
   e.printStackTrace();
   }
   }
}
```

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

```
Program is starting...
Waiting for 3 seconds...
Program is closing...
Bye.
```