GeeksforGeeks
A computer science portal for geeks

Custom Search

Practice GATE CS Placements Videos Contribute

Login/Register

# Daemon thread in Java

Daemon thread is a low priority thread that runs in background to perform tasks such as garbage collection.

3.2

**Properties:**

- They can not prevent the JVM from exiting when all the user threads finish their execution.
- JVM terminates itself when all user threads finish their execution
- If JVM finds running daemon thread, it terminates the thread and after that shutdown itself. JVM does not care whether Daemon thread is running or not.
- It is an utmost low priority thread.

**Methods:**

1. **void setDaemon(boolean status):** This method is used to mark the current thread as daemon thread or user thread. For example if I have a user thread tU then tU.setDaemon(true) would make it Daemon thread. On the other hand if I have a Daemon thread tD then by calling tD.setDaemon(false) would make it user thread.
   **Syntax:**

   ```
   public final void setDaemon(boolean on)
   parameters:
   on : if true, marks this thread as a daemon thread.
   exceptions:
   IllegalThreadStateException: if only this thread is active.
   SecurityException: if the current thread cannot modify this thread.
   ```

2. **boolean isDaemon():**
   This method is used to check that current is daemon. It returns true if the thread is Daemon else it returns false.
   **Syntax:**

```
public final boolean isDaemon()
returns:
This method returns true if this thread is a daemon thread;
false otherwise
```

```java
// Java program to demonstrate the usage of
// setDaemon() and isDaemon() method.
public class DaemonThread extends Thread
{
    public void run()
    {
        // Checking whether the thread is Daemon or not
        if(Thread.currentThread().isDaemon())
        {
            System.out.println("This is Daemon thread");
        }

        else
        {
            System.out.println("This is User thread");
        }
    }

    public static void main(String[] args)
    {

        DaemonThread t1 = new DaemonThread();
        DaemonThread t2 = new DaemonThread();
        DaemonThread t3 = new DaemonThread();

        // Setting user thread t1 to Daemon
        t1.setDaemon(true);

        // starting all the threads
        t1.start();
        t2.start();
        t3.start();

        // Setting user thread t3 to Daemon
        t3.setDaemon(true);

    }
}
```

Run on IDE

Output:

```
This is Daemon thread
This is User thread
This is Daemon thread
```

### Exceptions in Daemon thread

If you call the setDaemon() method after starting the thread, it would throw **IllegalThreadStateException**.

```
// Java program to demonstrate the usage of
// exception in Daemon() Thread
public class DaemonThread extends Thread
{
    public void run()
    {
        System.out.println("Thread name: " + Thread.currentThread().getName());
        System.out.println("Check if its DaemonThread: "
                        + Thread.currentThread().isDaemon());
    }

    public static void main(String[] args)
    {
        DaemonThread t1 = new DaemonThread();
        DaemonThread t2 = new DaemonThread();
        t1.start();

        // Exception as the thread is already started
        t1.setDaemon(true);

        t2.start();
    }
}
```

Run on IDE

Runtime exception:

```
Exception in thread "main" java.lang.IllegalThreadStateException
        at java.lang.Thread.setDaemon(Thread.java:1352)
        at DaemonThread.main(DaemonThread.java:19)
```

Output:

```
Thread name: Thread-0
Check if its DaemonThread: false
```

This clearly shows that we cannot call the setDaemon() method after starting the thread.

### Daemon vs User Threads

1. **Priority:** When the only remaining threads in a process are daemon threads, the interpreter exits. This makes sense because when only daemon threads remain, there is no other thread for which a daemon thread can provide a service.
2. **Usage:** Daemon thread is to provide services to user thread for background supporting task.

This article is contributed by **Saket Kumar**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.