DAEMON THREAD

Daemon thread is a low priority thread that runs in background to perform tasks such as garbage collection. Daemon thread in java is a service provider thread that provides services to the user thread. Its life depend on the mercy of user threads i.e. when all the user threads dies, JVM terminates this thread automatically.

There are many java daemon threads running automatically e.g. gc, finalizer etc.

• It provides services to user threads for background supporting tasks. It has no role in life than to serve user threads.

• Its life depends on user threads.

• It is a low priority thread.

The command jconsole typed in the command prompt provides information about the loaded classes, memory usage, running threads etc.

The purpose of the daemon thread is that it provides services to user thread for background supporting task. If there is no user thread, why should JVM keep running this thread.

That is why JVM terminates the daemon thread if there is no user thread.

Properties:

• They cannot 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 for Java Daemon thread by Thread class

The java.lang.Thread class provides two methods for java daemon thread.

Method Description

public void setDaemon(boolean status) used to mark the current thread as daemon thread or user thread.

public boolean isDaemon() used to check that current is daemon.

// 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);
}
}
```

Output:

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

// 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();
```

```
}

}
```

Output:

```
Thread name: Thread-0

Check if its DaemonThread: false
```

Daemon vs User Threads

• **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.

• **Usage:** Daemon thread is to provide services to user thread for background supporting task.

The following program is an example for daemon thread.

```
public class DaemonThread_example extends Thread{

public void run(){

if(Thread.currentThread().isDaemon()){//checking for daemon thread

System.out.println("daemon thread work");

}

else{

System.out.println("user thread work");

}

}

public static void main(String[] args){

TestDaemonThread1 t1=new TestDaemonThread1();//creating thread

TestDaemonThread1 t2=new TestDaemonThread1();

TestDaemonThread1 t3=new TestDaemonThread1();

t1.setDaemon(true);//now t1 is daemon thread

t1.start();//starting threads
```

```
t2.start();
t3.start();
}
}
```

The following program is another example for daemon thread.

```
class DaemonThread1_example extends Thread{
public void run(){
System.out.println("Name: "+Thread.currentThread().getName());
System.out.println("Daemon: "+Thread.currentThread().isDaemon());
}
public static void main(String[] args){
TestDaemonThread2 t1=new TestDaemonThread2();
TestDaemonThread2 t2=new TestDaemonThread2();
t1.start();
t1.setDaemon(true);//will throw exception here
t2.start();
}
        }
```