Notes - Daemon Threads

Daemon threads are the ones which does not prevent the JVM from exiting the application once it finishes. Daemon threads are handy for performing background tasks such as garbage collection or collecting application statistics etc. Note that the Java Virtual Machine exits when the only threads running are all daemon threads.

Example -

In the below example at the end of the while loop grp will point to system thread group. And enumerate method lists the threads in that group and copies the references to the given array. It also returns the number of threads copied. And later I am printing the thread name along with the boolean value checking if it is a daemon thread or not, using the method isDaemon().

```
public class Main1 {

   public static void main(String[] args) {

       System.out.println("System threads..........");

       Thread thr = Thread.currentThread();
       ThreadGroup grp = thr.getThreadGroup();
       while (grp.getParent() != null) {
           grp = grp.getParent();
       }

       Thread [] threads = new Thread[10];
       int n = grp.enumerate(threads);

       for (int i=0; i < n; i++) {
           System.out.println(
               "Thread Name: " + threads[i].getName() +
               " ; isDaemon: " + threads[i].isDaemon());
       }
   }
}
```

Output -

```
System threads..........
Thread Name: Reference Handler ; isDaemon: true
Thread Name: Finalizer ; isDaemon: true
Thread Name: Signal Dispatcher ; isDaemon: true
Thread Name: main ; isDaemon: false
```

You can see that Reference Handler, Finalizer, Single Dispatcher all these are daemon threads because they run the background and they doesn't stop the application from exiting.

setDaemon(boolean on) -

The method of the Thread class makes a enables us to set whether a thread is a daemon thread or user thread.

Example -

```
class MyTask implements Runnable {

   @Override
   public void run() {
       for (;;) {
           System.out.print("T");
       }
   }
}
public class Main {
   public static void main(String[] args) {
       Thread thr = new Thread(new MyTask());
       thr.setDaemon(true);
       thr.start();

       for (int i=1; i <= 200; i++) {
           System.out.print(" M ");
       }
   }
}
```

Output -

MMTTMMT

A combination of M and T but the application ends once the main ends.