Android : More on Multi-Threading

Éric Languénou

2012-2013

UNIVERSITÉ DE NANTES

Threads more

UNIVERSITÉ DE NANTES

What is the difference between implementing Runnable and extending Thread?

link (visited march 24 2013)

`http://www.xyzws.com/Javafaq/`

`what-is-the-difference-between-implementing-runnable-and-extending-thread/29`

Appendix A.

UNIVERSITÉ DE NANTES

```
class Runner implements Runnable {
  private int counter;
  public void run() {
    try {
      for (int i = 0; i != 2; i++) {
        System.out.println(Thread.currentThread().
            getName() + ": "
              + counter++);
        Thread.sleep(1000);
      }
    }
    catch(InterruptedException e) {
      e.printStackTrace();
    }
  }
} //end runner
```

Threads

UNIVERSITÉ DE NANTES

```
class Strider extends Thread {
private int counter;
Strider(String name)    {
  super(name);
}
public void run()   {
  try {
    for (int i = 0; i != 2; i++) {
      System.out.println(Thread.currentThread().
          getName() + ": "
          + counter++);
      Thread.sleep(1000);
    }
  }
  catch(InterruptedException e)     {
     e.printStackTrace();
  }
 }
} //end strider
```

Threads

UNIVERSITÉ DE NANTES

```
public class Program {
public static void main (String[] args) {
  Runner r = new Runner();
  Thread t1 = new Thread(r, "Thread A");
  Thread t2 = new Thread(r, "Thread B");
  Thread s1 = new Strider("Thread C");
  Thread s2 = new Strider("Thread D");
  t1.start();
  t2.start();
  s1.start();
  s2.start();
}
} //end main
```

Threads

UNIVERSITÉ DE NANTES

The output result is

```
Thread A: 0
Thread B: 1
Thread C: 0
Thread D: 0
Thread A: 2
Thread B: 3
Thread C: 1
Thread D: 1
```

- A class that implements Runnable is not a thread and just a class. For a Runnable to become a Thread, You need to create an instance of Thread and passing itself in as the target.

Threads

UNIVERSITÉ DE NANTES

- In most cases, the Runnable interface should be used if you are only planning to override the run() method and no other Thread methods. This is important because classes should not be subclassed unless the programmer intends on modifying or enhancing the fundamental behavior of the class.
- When there is a need to extend a superclass, implementing the Runnable interface is more appropriate than using the Thread class. Because we can extend another class while implementing Runnable interface to make a thread. But if we just extend the Thread class we can't inherit from any other class.

Appendix B.

UNIVERSITÉ DE NANTES

What is the difference between synchronized and volatile ?

- The value of a volatile variable is not locally cached by a thread (all reads and writes will go to "main memory" (may produce "lost-update" problem).
- Access to the volatile variables is similar to code enclosed in a synchronized block, but **without locking states**.
- Therefore, volatile is not suitable for complex operations where you need to prevent simultaneous access to a variable for the duration of the operation: in such cases, you should use object synchronization (sync methods or statements).