链滴

Java 基础 1 多线程的两种实现方式 继承 Thread 、实现 Runnable 接口

作者：syyQ

原文链接：https://ld246.com/article/1507400889042

来源网站：链滴

一个类只要继承了 Thread 类同时覆写了本类中的 run()方法就可以实现多线程操作了，但是一类只能继承一个父类，而实现 Runnable 不受这个的约束 。
Thread
例子 1:
<pre><code class="highlight-chroma">public class demo {
  public static void main(String[] args){
    ThreadDemo d1 = new ThreadDemo("syy");
    ThreadDemo d2 = new ThreadDemo("qb");
    d1.start();
    d2.start();
    }
}

 class ThreadDem extends Thread {
  private String name;
    ThreadDemo(String name){
      this.name = name;
    }
    public void run(){
      for(int x=1;x&t;10;x++)
      System.out.println(Thread.currentThread().getName()+" "+name +"===="+x);
    }
}
</code></pre>
此时输出的结果可能为线程 0、1 的来回切换
如果调用 d1.run()来"启动"线程，发现并没有来回切换。启用多线程时我们使用 start()方法，在 DK 的文档中可以发现，一旦调用 start()方法，则会通过 JVM 找到 run()方法。
public class Thread extends Object implements Runnable
发现 Thread 类也是 Runnable 接口的子类。
*是重写 run 方法
Runnable
例 2:
<pre><code class="highlight-chroma">public class demo
{
  public static void main(String[] args){
    RunnableDemo r1 =new RunnableDemo("syy");
    RunnableDemo r2 = new RunnableDemo("q");
    new Thread(r).start();
</code></pre>

```
</span></span><span class="highlight-line"><span class="highlight-cl">       new Thread(r
).start();
</span></span><span class="highlight-line"><span class="highlight-cl">     }
</span></span><span class="highlight-line"><span class="highlight-cl"> }
</span></span><span class="highlight-line"><span class="highlight-cl"> class RunnableD
mo implements Runnable {
</span></span><span class="highlight-line"><span class="highlight-cl">   private String n
me;
</span></span><span class="highlight-line"><span class="highlight-cl">     RunnableDemo
String name){
</span></span><span class="highlight-line"><span class="highlight-cl">       this.name = n
me;
</span></span><span class="highlight-line"><span class="highlight-cl">     }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">   public void run()
{
</span></span><span class="highlight-line"><span class="highlight-cl"> // TODO Auto-g
nerated method stub
</span></span><span class="highlight-line"><span class="highlight-cl">   for(int x=1;x&lt;
0;x++)
</span></span><span class="highlight-line"><span class="highlight-cl">     System.out.pri
tln(Thread.currentThread().getName()+" "+name +"===="+x);
</span></span><span class="highlight-line"><span class="highlight-cl">   }
</span></span><span class="highlight-line"><span class="highlight-cl"> }
</span></span></code></pre>
<p>实现效果与例 1 相同。<br>
使用 Runnable 定义的子类中没有 start()方法通过 Thread 类来启动 Runnable 实现的多线程.</p>
<p>Runnabe 的好处:</p>
<ul>
<li>避免点继承的局限，一个类可以继承多个接口。</li>
<li>适合于资源的共享</li>
</ul>
```