

实现多线程的方法?

Oracle 官网指出实现多线程的方法为 2 种

1. 方法1：实现Runnable接口
2. 方法2：继承 Thread类

两种实现方法对比

在方法1中，在Thread 类中 通过 判断 target 是否为 空，不为空则调用 target的 run方法，target为 Runnable

Thread.java中：

```
/* what will be run. */
private Runnable target; // target 为 Runnable

/**
 * If this thread was constructed using a separate
 * <code>Runnable</code> run object, then that
 * <code>Runnable</code> object's <code>run</code> method is called;
 * otherwise, this method does nothing and returns.
 * <p>
 * Subclasses of <code>Thread</code> should override this method.
 */
@Override
public void run() {
    if (target != null) {
        target.run();
    }
}
```

在方法2 中，继承了 Thread 重写了run方法，所以不会调用当前的run

两种方法的对比

◆ 方法1（实现Runnable接口）更好

◆ 两种方法的本质对比

■ 方法一：最终调用target.run();

■ 方法二：run()整个都被重写

思考题：同时使用这两种方法？

```
/**
 * 同时使用 Runnable 和 run 方法
 */
public class BothRunnableThread {
    /**
     * 使用 匿名内部类 实现
     * @param args
     */
    public static void main(String[] args) {

        new Thread(new Runnable() {
            @Override
            public void run() {
                System.out.println("Runnable方法实现");
            }
        }) {
            // run方法被重写，Runnable 对象不会被执行
            @Override
            public void run() {
                System.out.println("Thread方法实现");
            }
        }.start();
    }
}
```

// 输出结果？
// Thread方法实现

总结:

创建线程只有一种方式就是构造 Thread 类，而实现线程的执行单元有两种方式：

方法1：实现 Runnable 接口的 run方法，并把 Runnable 实例传给Thread类

方法2：重写 Thread 的 run方法（继承Thread类）

错误观点:

典型错误观点分析

◆ “无返回值是实现runnable接口，有返回值是实现callable接口，所以callable是新的实现线程的方式”

◆ 定时器

◆ 匿名内部类

◆ Lambda表达式

```
/**
 * lambda 表达式实现
 */
public class Lambda {
    public static void main(String[] args) {
        new Thread(() ->
System.out.println(Thread.currentThread().getName())).start();
    }
}

/** 错误观点
 * 定时器创建线程
 */
public class DemoTimerTask {

    public static void main(String[] args) {
        Timer timer = new Timer();

        timer.schedule(new TimerTask() {
            @Override
            public void run() {
                System.out.println(Thread.currentThread().getName());
            }
        });
    }
}
```

```

        }, 1000, 1000);
    }
}

/**
 * 匿名内部类 实现
 */
public class AnonymousInnerClassDemo {

    public static void main(String[] args) {

        new Thread(){
            @Override
            public void run() {
                System.out.println(Thread.currentThread().getName());
            }
        }.start();

        new Thread(new Runnable() {
            @Override
            public void run() {
                System.out.println(Thread.currentThread().getName());
            }
        }).start();
    }
}

```

常见面试问题？Runnable方法好

◆ 实现Runnable接口和继承Thread类哪种方式更好？

1. 从代码架构角度
2. 新建线程的损耗
3. Java不支持双继承