线程的创建

一、实现java.lang.Runnable接口(优先使用)：

1.创建一个实现Runnable接口的类

2.实现类去实现Runnable中的抽象方法：run()

3.创建实现类的对象

4.将此对象作为参数传递到Thread构造器中，创建Thread对象

5.通过Thread类的对象调用start()

```
public class MyRunnable implements Runnable {
    @Override
    public void run() {
        for(;;){
            System.out.println("==================");
        }
    }
}
public class ThreadText {
    public static void main(String[] args) {
        Runnable  runnable=new MyRunnable ();
        Thread thread=new Thread(runnable);
        thread.start();
    }
}
```

二、实现 Callable 接口获取线程的返回值

```
/*
 * 一、创建执行线程的方式三：实现 Callable 接口。 相较于实现 Runnable 接口的方式，方法可以有返回值，并且可以抛出异常。
 *
 * 二、执行 Callable 方式，需要 FutureTask 实现类的支持，用于接收运算结果。 异步运算 FutureTask 是  Future 接口的实现类
 */
public class CallableDemo implements Callable<Integer> {
        private int sum;
        @Override
        public Integer call() throws Exception {
```

```
        System.out.println("Callable子线程开始计算啦！");
        Thread.sleep(2000);

        for(int i=0 ;i<5000;i++){
            sum=sum+i;
        }
        System.out.println("Callable子线程计算结束！");
        return sum;
    }
    public static void main(String[] args) {
        //创建线程池
        ExecutorService es = Executors.newSingleThreadExecutor();
        //创建Callable对象任务
        CallableDemo calTask=new CallableDemo();
        //提交任务并获取执行结果
        Future<Integer> future =es.submit(calTask);
        //关闭线程池
        es.shutdown();
        try {
            Thread.sleep(2000);
        System.out.println("主线程在执行其他任务");

        if(future.get()!=null){
            //输出获取到的结果
            System.out.println("future.get()-->"+future.get());
        }else{
            //输出获取到的结果
            System.out.println("future.get()未获取到结果");
        }

        } catch (Exception e) {
            e.printStackTrace();
        }
        System.out.println("主线程在执行完成");
    }
```

三、java.lang.Thread类

1、创建定义线程并继承r=thread

2、重写thread的run方法(run不能手动调用)

3、创建子类对象

4、调用start()：

1)启动线程

2)调用当前线程的run()

```
public class ThreadCreate01 extends Thread{
    @Override
    public void run() {
        for(;;){
            System.out.println("==================");
        }
    }
}
public class ThreadText {
    public static void main(String[] args) {
    ThreadCreate01 t1=new ThreadCreate01();
    t1.start();
    }
}
```

四、线程池创建