java中多线程的创建和使用

①继承thread类和实现runnable接口

②thread类的主要方法

③线程的调度和设置优先级

线程的生命周期

线程的同步

线程通信

程序（program）：是为完成特定的任务、用某种语言编写的一组指令的集合，即指一段静态的代

码，静态对象

进程（process）：是程序的一次执行过程，或是正在运行的一个程序，动态过程，有它自身的产

生、存在和消亡的过程

程序是静态的，进程是动态的

线程（thread）：进程可进一步细化为线程，是一个程序内部的一条执行路径

若一个程序可同一时间执行多个线程，就是支持多线程的

创建多线程的第一种方式：继承java.lang.Thread类

```
//创建一个继承于thread的子类
class sub extends Thread{
        //重写thread类的run（）方法，方法内实现此子线程要完成的功能
        public void run(){
                int sum =0;
                for (int i = 0; i <= 100; i++) {
                        sum+=i;
                        System.out.print(Thread.currentThread().getName()+"::");
                        System.out.println(i);
                }
        }
}

//创建一个子类的对象
```

```
sub sub = new sub();
//调用线程的start（），启动此线程，调用相应的run方法
    //一个线程只能执行一次start
//不能通过thread实现类对象的run去启动一个线程
sub.start();
int sum = 0;
for (int i = 0; i <= 100; i++) {
        sum+=i;
        System.out.print(Thread.currentThread().getName()+"::");
        System.out.println(i);
}
```

thread的常用方法

①start（）；启动线程，并执行相应的run方法

②run（）；子线程要执行的代码存入run方法中

③currentThread()；静态的，调取当前的线程

④getName（）；获取此线程的名字

⑤setname（）；设置线程的名字

⑥yield（）；调用此方法的线程释放当前的CPU的执行器

⑦join（）；在a线程中调用b线程的join（）方法，表示当执行到此方法时，a线程停止运行，直到b线

程执行完毕，a线程在接着join（）之后的代码执行

⑧isAlive（）；判断当前线程是否安全

⑨sleep（long l1）；显示的让当前线程睡眠l1毫秒

①线程通信：wait（）notify（）notifyAll（）

设置线程的优先级

调度策略：①时间片

②抢占式：高优先级的线程抢占cpu

java的调度方法：

①同优先级线程组成先进先出队列（先到先服务）使用时间片策略

②对高优先级，使用优先级调度的抢占式策略

线程的优先级控制：MAX_PRIORITY(10)

MIN_PRIORITY(1)

NORM_PRIORITY(5)

涉及的方法：getPriority（）；返回线程优先级

getPriority（int newPriority）；改变线程的优先级

线程创建时继承父线程的优先级