JUC

笔记本:	工作笔记		
创建时间:	2020/10/30 10:37	**更新时间:**	2020/10/31 19:51
作者:	438842220@qq.com		
URL:	about:blank		

JAVA.UTIL.CONCURRENT

注：以下绿色为接口，蓝色为类

Runnable/Callable

前一个没有返回值，不能抛异常，后一个相反

Executor

excute一个Runnable对象

ExecutorService

它比executor更广泛，提供了一系列生命周期管理方法。接受Runnable和Callable对象，返回Futrure。因此一般用这个接口管理和实现多线程。

AbstractExecutorService

它是ExecutorService的默认实现，同时它是一个抽象类。

ScheduledExecutorService

它是一个提供定时调度的接口。

ForkJoinPool

java7引入的ForkJoin框架，同时引入了一种新的线程池ForkJoinPool。

展开讲讲fork/join。见示例代码：jucDemo.calculator

ThreadPoolExecutor

继承并实现了AbstractExecutorService的功能。

入参包括：

corePoolSize/maximumPoolSize/keepAliveTime/unit/workQueue

Executors的四种线程池都是从这里来的。

ScheduledThreadPoolExecutor

正常情况下，定时器我们都是用Timer和TimerTask这两个类就能完成定时任务，并且设置延长时间和循环时间间隔。

ScheduledThreadPoolExecutor也能完成Timer一样的定时任务，并且时间间隔更加准确。

Executors

包含五种实现：

newFixedThreadPool/newWorkStealingPool/newSingleThreadExecutor/newCachedThreadPool/newScheduledThreadPool

分别用于：

定长线程池，可控制线程最大并发数，超出的线程会在队列中等待。

（1.8新增）这个线程池不会保证任务的顺序执行，也就是 WorkStealing 的意思，抢占式的工作。

单线程化的线程池，它只会用唯一的工作线程来执行任务，保证所有任务按照指定顺序(FIFO/LIFO/优先级)执行。

可缓存线程池，如果线程池长度超过处理需要，可灵活回收空闲线程，若无可回收，则新建线程。

定长线程池，支持定时及周期性任务执行。

BlockingQueue

继承了Queue接口

阻塞队列一共有四套方法分别用来进行insert、remove和examine，当每套方法对应的操作不能马上执行时会有不同的反应，下面这个表格就分类列出了这些方法：

-	Throws Exception	Special Value	Blocks	Times Out
Insert	add(o)	offer(o)	put(o)	offer(o, timeout, timeunit)
Remove	remove(o)	poll()	take()	poll(timeout, timeunit)
Examine	element()	peek()		

ThrowsException：如果操作不能马上进行，则抛出异常

SpecialValue：如果操作不能马上进行，将会返回一个特殊的值，一般是true或者false

Blocks:如果操作不能马上进行，操作会被阻塞

TimesOut:如果操作不能马上进行，操作会被阻塞指定的时间，如果指定时间没执行，则返回一个特殊值，一般是true或者false

需要注意的是，我们不能向BlockingQueue中插入null，否则会报NullPointerException。

SynchronousQueue

```
            队列内部仅允许容纳一个元素。当一个线程插入一个元素后会被阻塞，除非这个元素被另一个线程消费。
        PriorityBlockingQueue
            底层数据结构是最小二叉堆。支持优先级。同PriorityQueue一样，可以自定义实现compareTo方法来指定元素排序规则。
        LinkedBlockingQueue/ArrayBlockingQueue
            类似LinkedList和ArrayList。一个基于链表，一个基于数组。
            链表的数据结构：单向指针，记录头结点尾节点。
        DelayQueue
            是一个无界的BlockingQueue，用于放置实现了Delayed接口的对象，其中的对象只能在其到期时才能从队列中取走。
            队列有序，即队头对象的延迟到期时间最长。
            实现例子：下单之后如果三十分钟之内没有付款就自动取消订单。
        BlockingDeque
            双端队列。继承了BlockingQueue。
            LinkedBlockingDeque
                BlockingDeque的实现。和LinkedBlockingQueue不同的是，双端均可出入。
        TransferQueue
            java7中加入。暂空。
Delayed
Future
CompletionStage
ConcurrentMap（已完成待加入）
CompletionService
其他
Lock
Atomic
```