第六章 任务执行

#### 一．在线程中执行任务

1. 选择清晰的任务边界以及明确的任务执行策略。

串行的执行任务

1. 概念

在单个线程中串行的执行各项任务。

2. 缺点

（1）无法提供高吞吐率或快速响应性。

一次只能处理一个请求，如果处理方法很快可以立即返回，那么这种方法可行。但是实际情况并非如此，在web请求中包含I/O操作，如果因为网络问题导致操作阻塞等其它问题，会导致线程的运行时间过长，而其它用户还必须等待。

（2）服务器利用资源低

显示的为任务创建线程

1. 概念

为每个请求创建一个线程（不要这么做）。

2. 优点

（1）使得任务处理过程从主线程分离出来，使得主线程可以接收下一个请求，提高响应性。

（2）任务可以并行处理，可以同时服务多个请求，但是任务代码必须是线程安全的。

3. 缺点

（1）线程生命周期的开销非常高。线程的创建与销毁并不是没有代价的。

（2）资源消耗高。

（3）稳定性差。可能会抛出OutOfMemoryError异常。

#### 二. Executor框架

Executor基于生产者-消费者模式，提交任务的操作相当于生产者，执行任务的线程相当于消费者。

Executor将任务提交与执行解耦开来。

执行策略

尽量利用Executor来创建线程。

线程池

1. 优点

（1）线程池中的线程已经存在，不会因为创建线程而延迟任务的执行。

（2）减少了创建线程与销毁线程的开销。

Executor的生命周期

1. 运行、关闭、已终止。

（1）运行：初始创建的时候处于运行状态

（2）关闭：

shutdown方法将执行平缓的关闭过程：不再接收新的任务，同时等待提交的任务执行完成（包括还未执行的任务）。

shutdownNow方法将执行粗暴的关闭：它将尝试取消所有运行中的任务，并且不再启动队列中尚未开始执行的任务。

（3）终止：提交关闭请求，所有任务完成后，进入终止状态。

延迟任务与周期任务

1. 延迟任务

指定时间后执行的任务。

2. 周期任务

每隔指定时间就执行一次任务。

3. Timer与TimerTask

Timer负责管理延迟任务与周期任务。

（1）Timer的问题

（a）Timer在执行任务的时候只会创建一个线程，如果某个任务的执行时间过长，那么将破坏其他TimerTask的定时精确性。

（b）如果TimerTask抛出了一个未检查的异常，Timer将表现出糟糕的行为。Timer不会捕获异常，因此此时Timer将中止定时线程。而且Timer也不会恢复线程的执行。

4. 使用ScheduledThreadPoolExecutor代替Timer

ScheduledThreadPoolExecutor scheduled = new ScheduledThreadPoolExecutor(2);

scheduled.scheduleAtFixedRate(new Runnable() {

@Override

public void run() {

loge("time:");

}

}, 0, 40, TimeUnit.MILLISECONDS);//0表示首次执行任务的延迟时间，40表示每次执行任务的间隔时间，TimeUnit.MILLISECONDS执行的时间间隔数值单位

5. 使用DelayQueue来调度自己的服务。

DelayQueue类的主要作用：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。这种队列是有序的，即队头对象的延迟到期时间最长。注意：不能将null元素放置到这种队列中。

#### 三. 找出可利用的并行性

携带结果的任务Callable与Future

1. Future和Executor结合使用

书中的例子：可以异步下载图像

在一开始的时候Executor就开始下载图像，然后渲染文本，最后调用future的get方法。

2. 在异构任务并行化中存在的局限

只有当大量相互独立的任务可以并发进行处理的时候，才能体现出将程序的工作负载分配到多个任务中带来的真正性能提升。

3. CompletionService：Executor与BlockingQueue

CompletionService将Executor与BlockingQueue功能融合在一起。

CompletionService获取多线程返回值，返回一个Future对象，然后通过take方法获取Future。

4. 为任务设置时限

5. ExecutorService的invokeAll方法：参数为一组任务，可以返回多个Future，支持限时，当时间到达后，将返回所有任务的执行结果，包括未执行完的任务。