**Task：任务**

对于多线程，我们经常使用的是Thread。在我们了解Task之前，如果我们要使用多核的功能可能就会自己来开线程，然而这种线程模型在.net 4.0之后被一种称为基于“任务的编程模型”所冲击，因为task会比thread具有更小的性能开销，不过大家肯定会有疑惑，任务和线程到底有什么区别呢？另外使用过程中也有一些陷阱，需要绕开它们才能避免出错。

　Task是微软在.net framework 4.0发布的新的异步编程的利器，当然4.5新增了async、await，这儿我们先说Task相关。

**Task和Thread的区别**

Task是架构在Thread之上的，也就是说任务最终还是要抛给线程去执行。

Task跟Thread不是一对一的关系，比如开10个任务并不是说会开10个线程，这一点任务有点类似线程池，但是任务相比线程池有很小的开销和精确的控制。

在实际编程中，我们用的较多的是**Task**、**Task.Factory.StarNew**、**Task.Run**

实例化的Task对象，需要调用Start来启动任务，如果使用Task.Factory.StarNew，则不用调用Start方法来启动任务。

Task.Run的跟Task.Factory.StarNew和new Task相差不多，不同的是前两种是放进线程池立即执行，而Task.Run则是等线程池空闲后在执行。

Run方法只接受无参的Action和Func委托，另外两个接受一个object类型的参数。

# C[#之task与thread区别及其使用](https://www.cnblogs.com/huandong963/p/12594064.html)

1.什么是thread  
当我们提及多线程的时候会想到thread和threadpool，这都是异步操作，threadpool其实就是thread的集合，具有很多优势，不过在任务多的时候全局队列会存在竞争而消耗资源。thread默认为前台线程，主程序必须等线程跑完才会关闭，而threadpool相反。  
总结：threadpool确实比thread性能优，但是两者都没有很好的api区控制，如果线程执行无响应就只能等待结束，从而诞生了task任务。  
2.什么是task  
 task简单地看就是任务，那和thread有什么区别呢？Task的背后的实现也是使用了线程池线程，但它的性能优于ThreadPoll,因为它使用的不是线程池的全局队列，而是使用的本地队列，使线程之间的资源竞争减少。同时Task提供了丰富的API来管理线程、控制。但是相对前面的两种耗内存，Task依赖于CPU对于多核的CPU性能远超前两者，单核的CPU三者的性能没什么差别。

3.创建一个task任务有两种模式：new Task 和 Task.Run 和Task.Factory.StartNew

Task.Run的跟Task.Factory.StarNew和new Task相差不多，不同的是前两种是放进线程池立即执行，而Task.Run则是等线程池空闲后在执行。  
**1.使用factory创建会直接执行，使用new创建不会执行，必须等到start启动之后才执行。**  
**2.我们来看看task的生命周期，可以看出task确实是异步执行，并且wait很好地控制了task。**

**3.控制task的方法**

   通过wait()对单个task进行等待，Task.waitall()对多个task进行等待，waitany()执行任意一个task就往下继续执行。

4.**task的回调执行ContinueWith**

**5.task的取消---------------------**

首先创建一个取消task的令牌的实例，在不启动task直接取消：

var tokenSource = new CancellationTokenSource();//创建取消task实例

new Task(() => { }, tokenSource.Token);

tokenSource.Token.Register(()=>{ });

tokenSource.Cancel();

**6.task的嵌套**

在父子节点下，默认子节点是独立于父节点执行的，可使用TaskCreationOptions.AttachedToParent来使子节点和父节点同步，若父节点使用TaskCreationOptions.DenyChildAttach的话那么子节点配置的AttachedToParent的父子同步将不起作用！

**7.task死锁的问题**

我们可以设置最大等待时间，如果超过了等待时间，就不再等待，下面我们来修改代码，设置最大等待时间为5秒(项目中可以根据实际情况设置)，如果超过5秒就输出哪个任务出错了

**8.对Spinlock的使用**

举例来说Parallel.for和Parallel.foreach是线程不安全的，有可能达不到你的预期，此时就需要加锁来解决此问题，我们可以加lock和spinlock(自旋锁)来解决

SpinLock slock = new SpinLock(false);