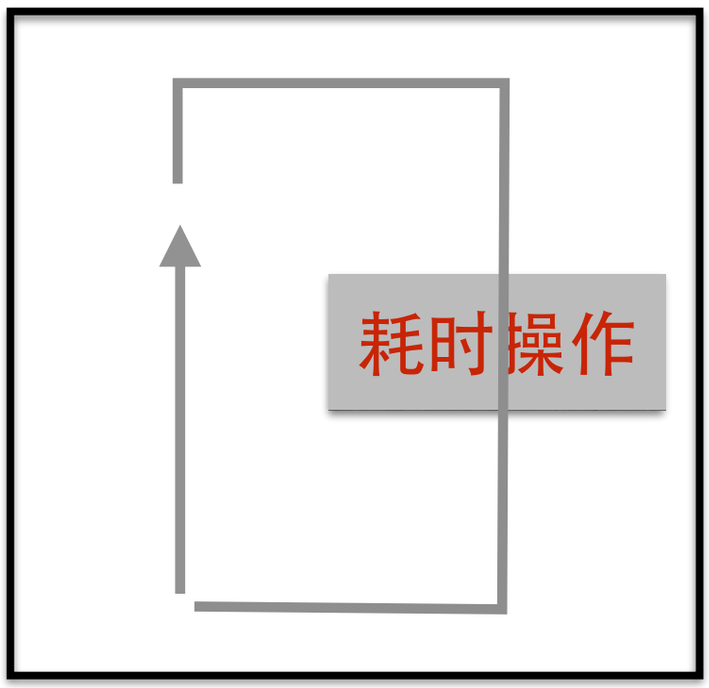
主线程

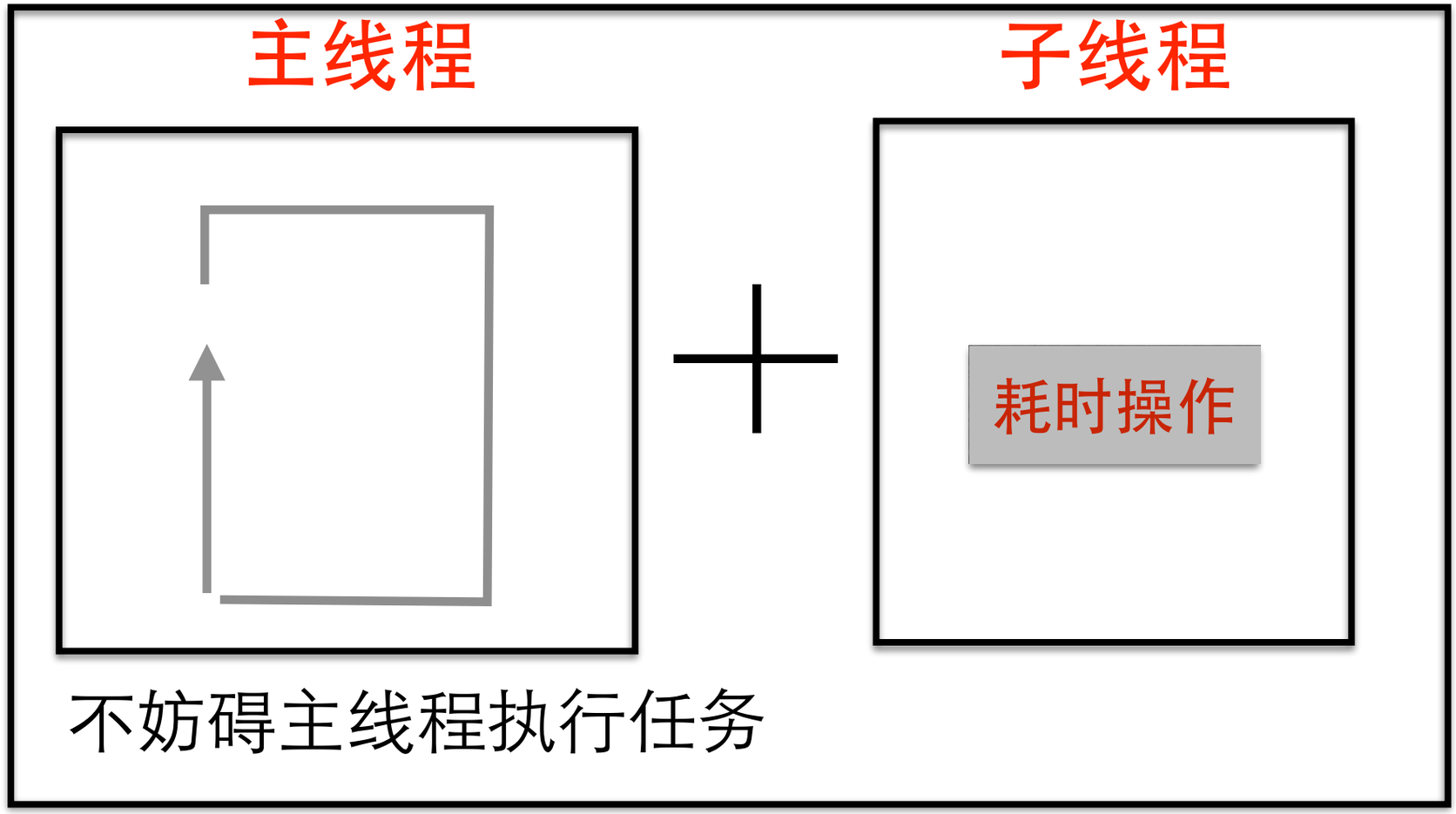
不使用子线程



阻塞主线程的执行

使用子线程

1. PThread
2. NSThread



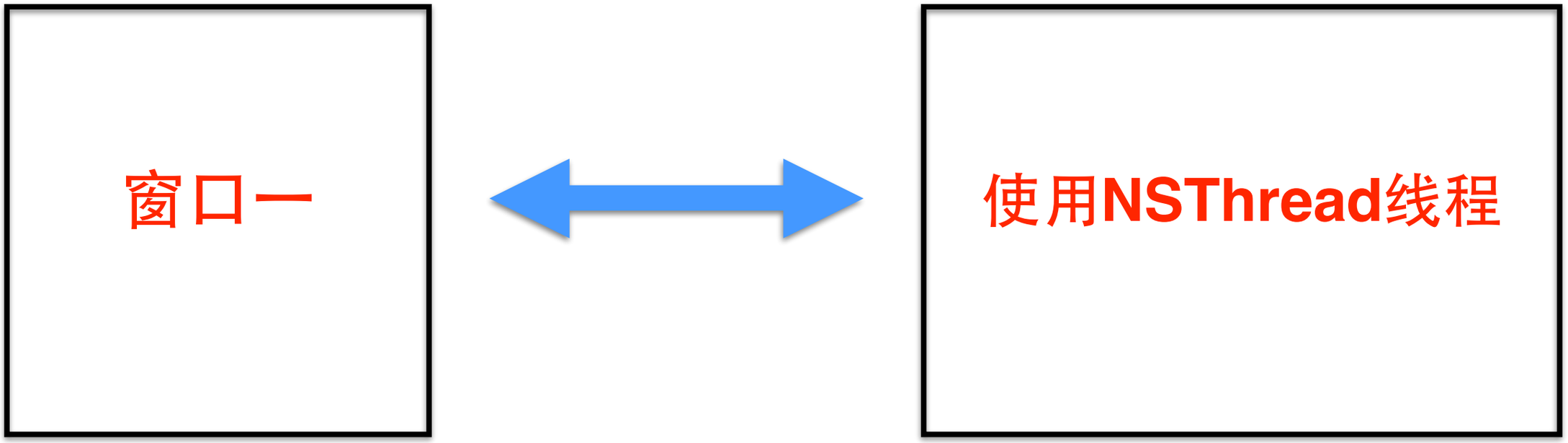
PThread

```
1.  pthread_t pthread;
2.  pthread_create(&pthread, NULL, task, NULL);
3.  void *task(void *data) {
     //执行耗时操作
    }
```

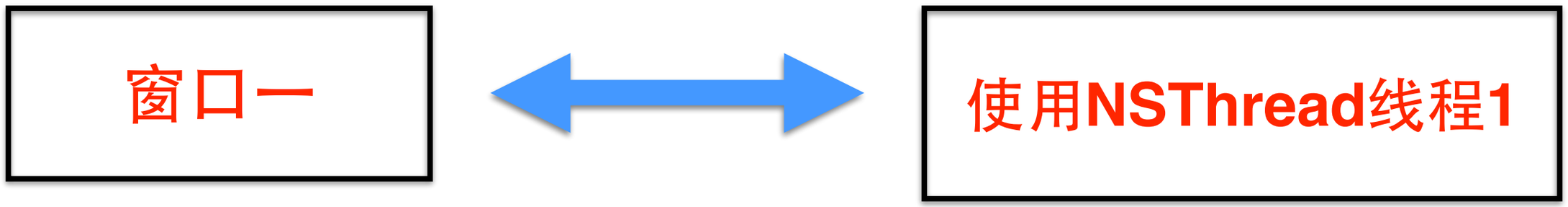
NSThread

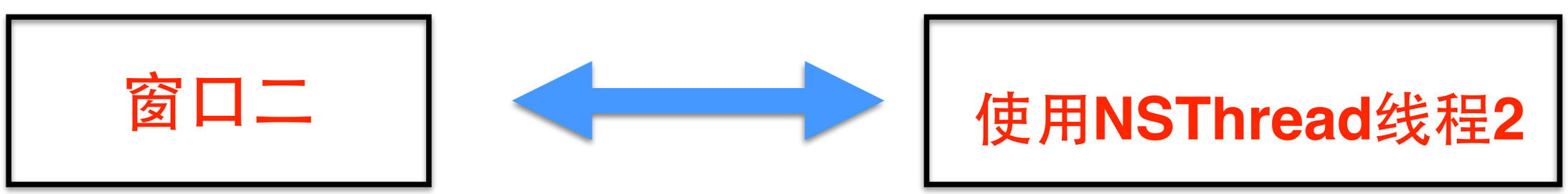
```
1. NSThread *thread = [[NSThread alloc] initWithTarget:self
   selector:(task) object:nil];
2. thread.name = @“线程名字”;
3. [thread start];
4. - (void)task {
     //耗时操作
   }
```

使用NSThread模拟一个窗口的卖票样例一

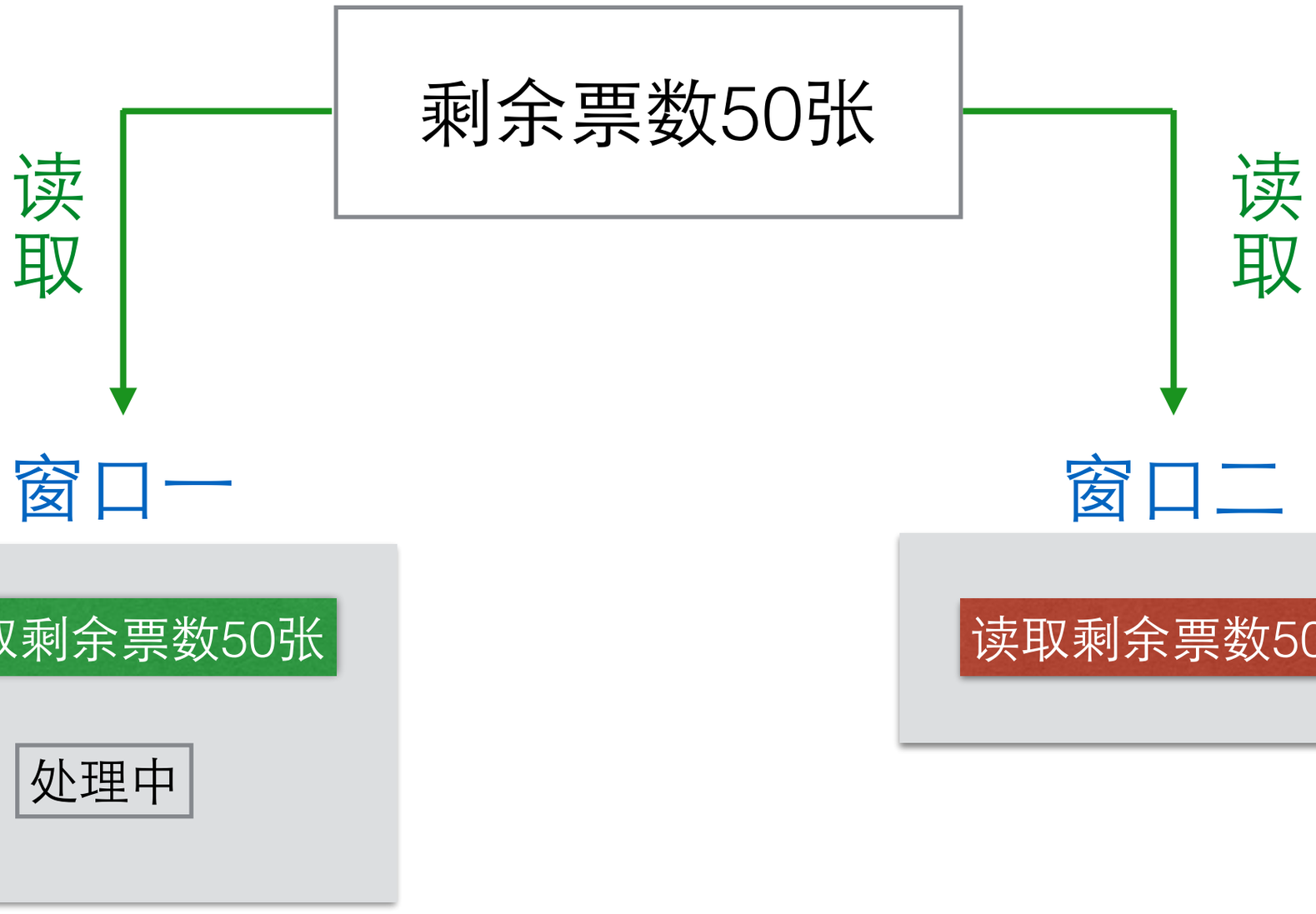


使用NSThread模拟两个窗口的卖票样例二



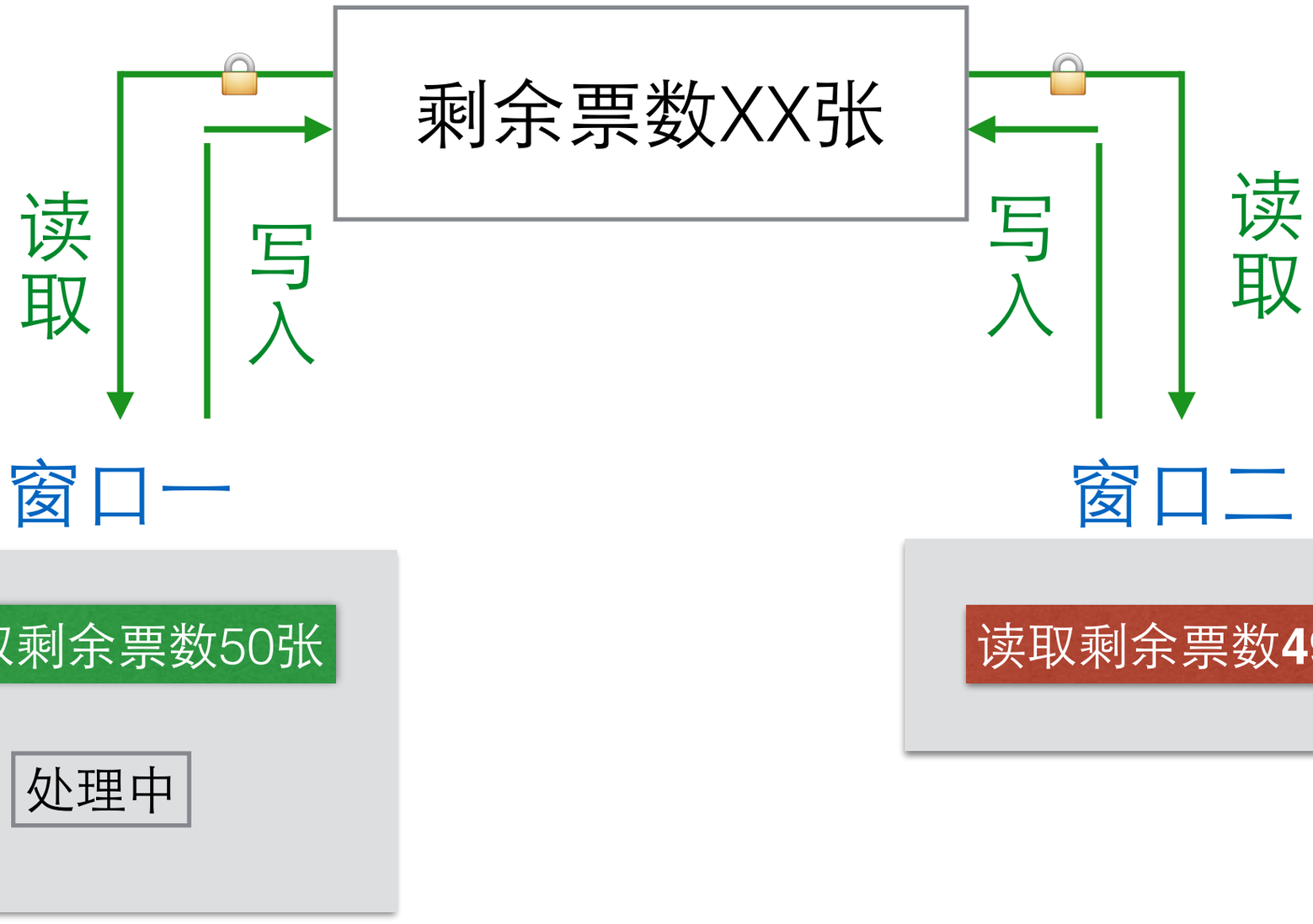


两个窗口同时卖票的流程



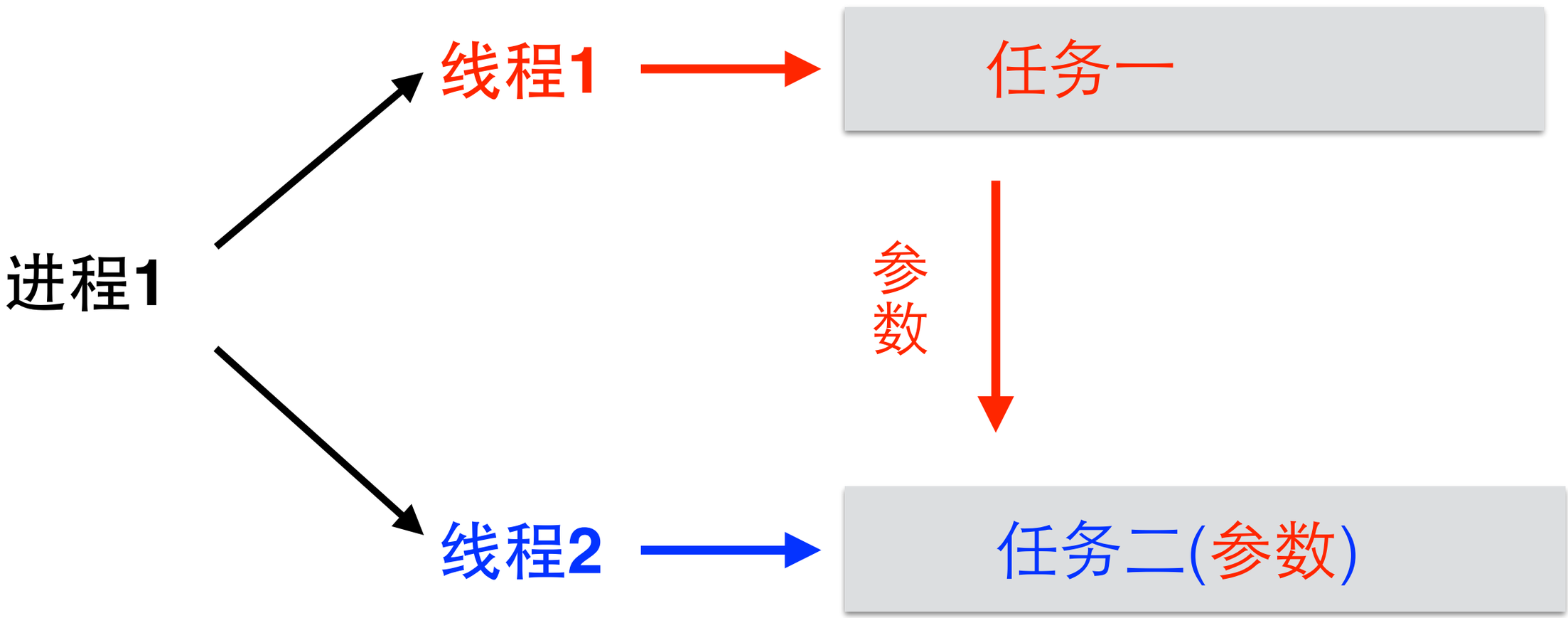
结论：多个线程同时访问同一个数据，会造成数据不一致问题

两个窗口使用加锁/解锁方式卖票的流程

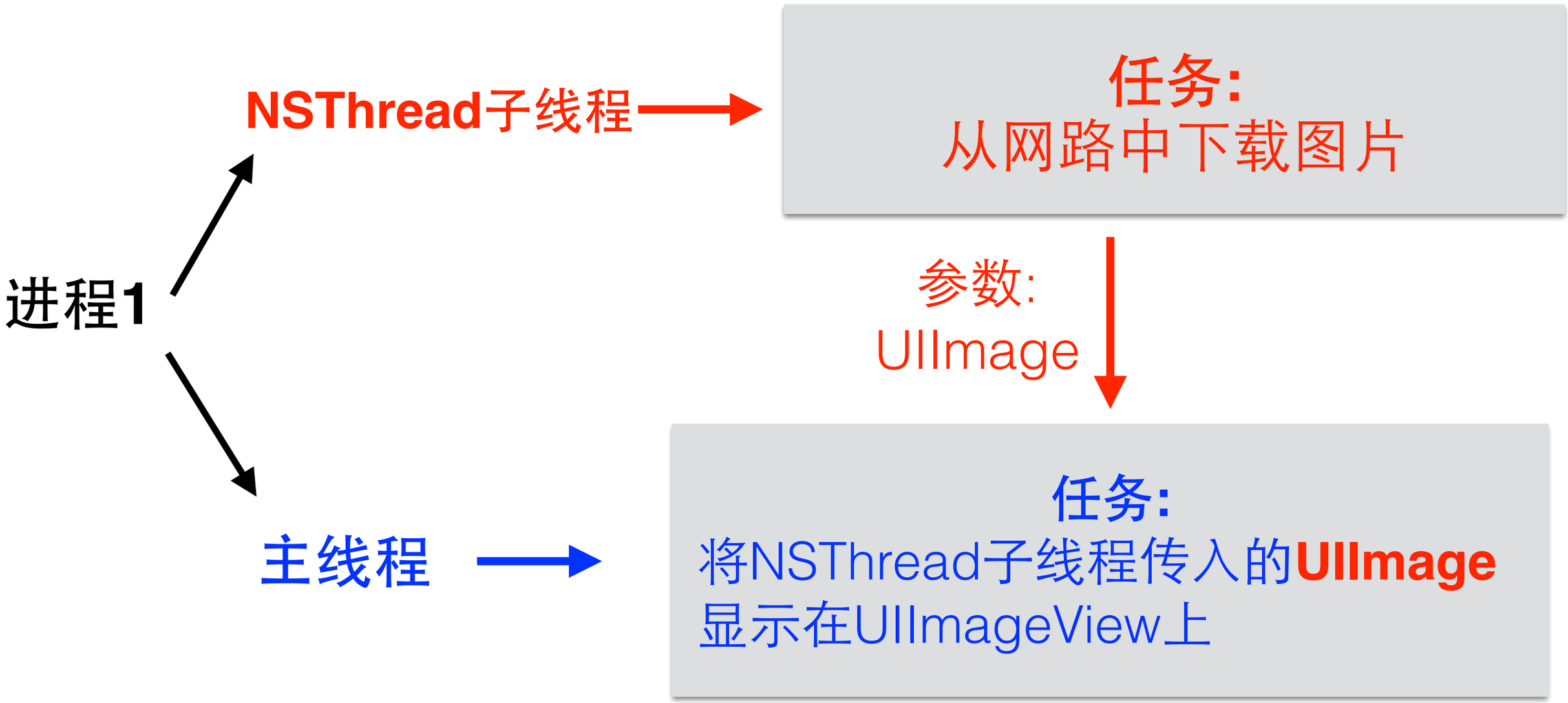


结论：只用使用了加锁/解锁的机制，才可以保证数据一致。

线程间通讯



子线程和主线程间通讯



从子线程回到主线程方式:

1. 使用NSObject的方法: performSelectorOnMainThread
2. 使用NSObject的方法:
   performSelector:onThread:[NSThread mainThread]

GCD 一般工作原理

1. 创建空队列

空dispatch queue →

2. 把任务添加队列中

dispatch queue → 任务

3. 执行队列中的任务

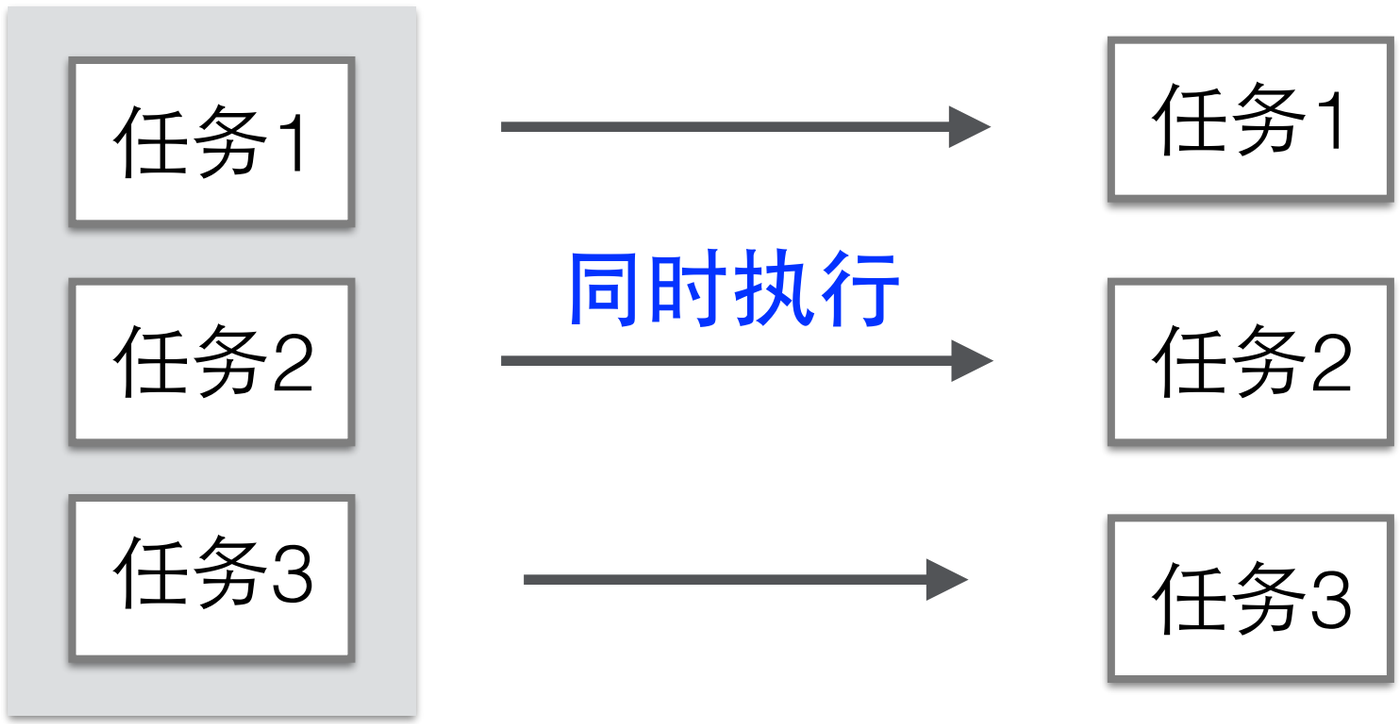
执行 → 处理 → 任务

GCD 常用术语一

GCD队列类型：

1. 串行队列(Serial Dispatch Queue): 一个一个顺序地执行

2. 并行队列(Concurrent Dispatch Queue): 同时执行多个任务



GCD 常用术语二

执行任务方式：

1. 同步执行(Synchronous Dispatch): 队列中的任务在主线程中执行

2. 异步执行(Asynchronous Dispatch): 队列中的任务在子线程中执行

GCD 队列类型和执行方式

队列类型 / 执行方式	串行队列	并行队列
同步执行	串行同步	并行同步
异步执行	串行异步	并行异步

GCD 四种组合

结论：

1.串行同步：队列中的任务顺序执行；在主线程中执行；

2.串行异步：队列中的任务顺序执行；在子线程中执行；主线程继续执行，不会等待子线程执行完毕

3.并行同步(一般不用)：队列中的任务顺序执行；在主线程中执行；

4.并行异步：队列中的任务同时执行；在子线程中执行；主线程继续执行，不会等待子线程执行完毕

GCD 中两种系统提供的队列

全局队列(**Global Queue**)：是全局的并行队列
主队列(**Main Queue**)：是在主线程中执行的队列(串行)

队列类型 / 执行方式	串行队列	并行队列/全局队列	主队列
同步执行	串行同步	~~并行同步~~	~~主队列同步~~
异步执行	串行异步	并行异步	主队列异步