逻辑教育
Logic education

大师班第19天

和谐学习，不急不躁

LG_Cooci

逻辑教育
Logic education

线程和进程的定义

* 线程是进程的基本执行单元，一个进程的所有任务都在线程中执行
* 进程要想执行任务，必须得有线程，进程至少要有一条线程
* 程序启动会默认开启一条线程，这条线程被称为主线程或 UI 线程

* 进程是指在系统中正在运行的一个应用程序
* 每个进程之间是独立的，每个进程均运行在其专用的且受保护的内存空间内
* 通过“活动监视器”可以查看 Mac 系统中所开启的进程

逻辑教育
Logic education

进程与线程的关系

地址空间：同一进程的线程共享本进程的地址空间，而进程之间则是独立的地址空间。

资源拥有：同一进程内的线程共享本进程的资源如内存、I/O、cpu等，但是进程之间的资源是独立的。

1: 一个进程崩溃后，在保护模式下不会对其他进程产生影响，但是一个线程崩溃整个进程都死掉。所以多进程要比多线程健壮。

2: 进程切换时，消耗的资源大，效率高。所以涉及到频繁的切换时，使用线程要好于进程。同样如果要求同时进行并且又要共享某些变量的并发操作，只能用线程不能用进程

3: 执行过程：每个独立的进程有一个程序运行的入口、顺序执行序列和程序入口。但是线程不能独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制。

4: 线程是处理器调度的基本单位，但是进程不是。

5: 线程没有地址空间,线程包含在进程地址空间中

逻辑教育
Logic education

多线程的意义

* 优点
  * 能适当提高程序的执行效率
  * 能适当提高资源的利用率（CPU，内存）
  * 线程上的任务执行完成后，线程会自动销毁
* 缺点
  * 开启线程需要占用一定的内存空间（默认情况下，每一个线程都占 512 KB）
  * 如果开启大量的线程，会占用大量的内存空间，降低程序的性能
  * 线程越多，CPU 在调用线程上的开销就越大
  * 程序设计更加复杂，比如线程间的通信、多线程的数据共享

多线程的原理

- 线程1
- 线程2
- 线程3
- 线程4
- 线程5

逻辑教育
Logic education

线程和进程的定义

时间片的概念：CPU在多个任务直接进行快速的切换，这个时间间隔就是时间片

* （单核CPU）同一时间，CPU 只能处理 1 个线程
  * 换言之，同一时间只有 1 个线程在执行
* 多线程同时执行：
  * 是 CPU 快速的在多个线程之间的切换
  * CPU 调度线程的时间足够快，就造成了多线程的“同时”执行的效果
* 如果线程数非常多
  * CPU 会在 N 个线程之间切换，消耗大量的 CPU 资源
  * 每个线程被调度的次数会降低，线程的执行效率降低

逻辑教育
Logic education

压栈 - 栈帧 - 堆栈溢出

栈

局部变量
函数的参数
(Id self SEL _cmd)

指针

堆

x x x x

开辟空间 alloc

Table 2-1 Thread creation costs

Item	Approximate cost	Notes
Kernel data structures	Approximately 1 KB	This memory is used to store the thread data structures and attributes, much of which is allocated as wired memory and therefore cannot be paged to disk.
Stack space	512 KB (secondary threads) 8 MB (OS X main thread) 1 MB (iOS main thread)	The minimum allowed stack size for secondary threads is 16 KB and the stack size must be a multiple of 4 KB. The space for this memory is set aside in your process space at thread creation time, but the actual pages associated with that memory are not created until they are needed.
Creation time	Approximately 90 microseconds	This value reflects the time between the initial call to create the thread and the time at which the thread's entry point routine began executing. The figures were determined by analyzing the mean and median values generated during thread creation on an Intel-based iMac with a 2 GHz Core Duo processor and 1 GB of RAM running OS X v10.5.

多线程技术方案

方案	简介	语言	线程生命周期	使用频率
pthread	• 一套通用的多线程API • 适用于 Unix / Linux / Windows 等系统 • 跨平台\可移植 • 使用难度大	C	程序员管理	几乎不用
NSThread	• 使用更加面向对象 • 简单易用，可直接操作线程对象	OC	程序员管理	偶尔使用
GCD	• 旨在替代 NSThread 等线程技术 • 充分利用设备的多核	C	自动管理	经常使用
NSOperation	• 基于GCD（底层是GCD） • 比 GCD 多了一些更简单实用的功能 • 使用更加面向对象	OC	自动管理	经常使用

逻辑教育
Logic education

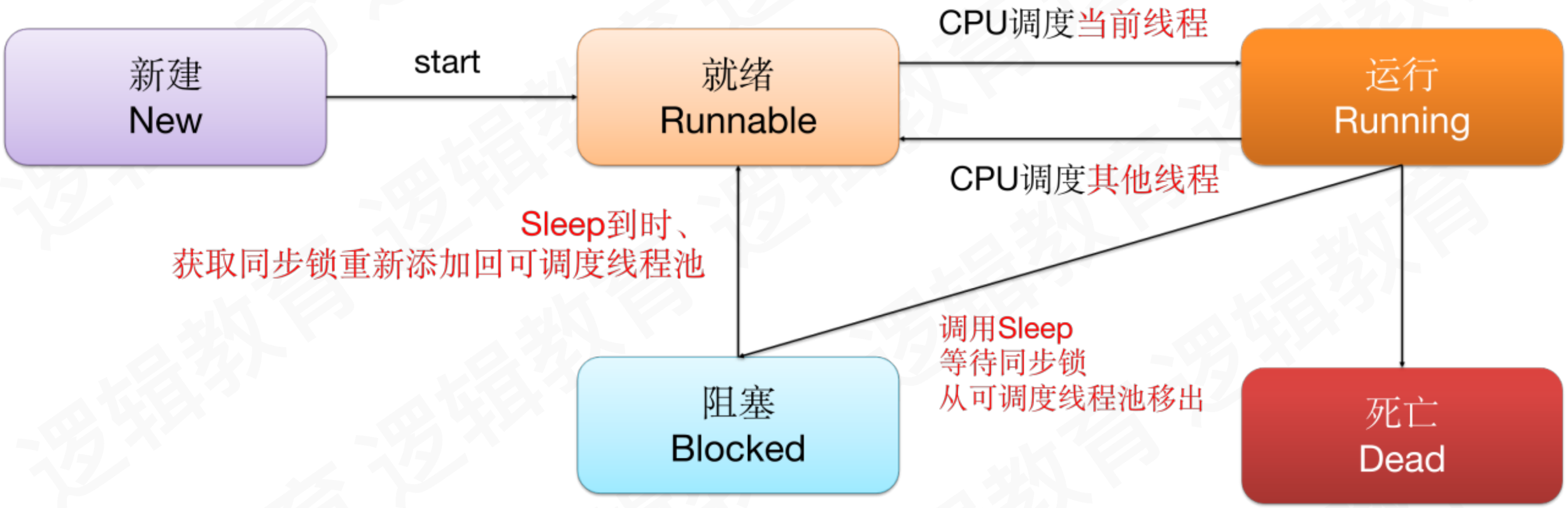
C与OC的桥接

* `__bridge`只做类型转换，但是不修改对象（内存）管理权；

* `__bridge_retained`（也可以使用`CFBridgingRetain`）将Objective-C的对象转换为Core Foundation的对象，同时将对象（内存）的管理权交给我们，后续需要使用CFRelease或者相关方法来释放对象；

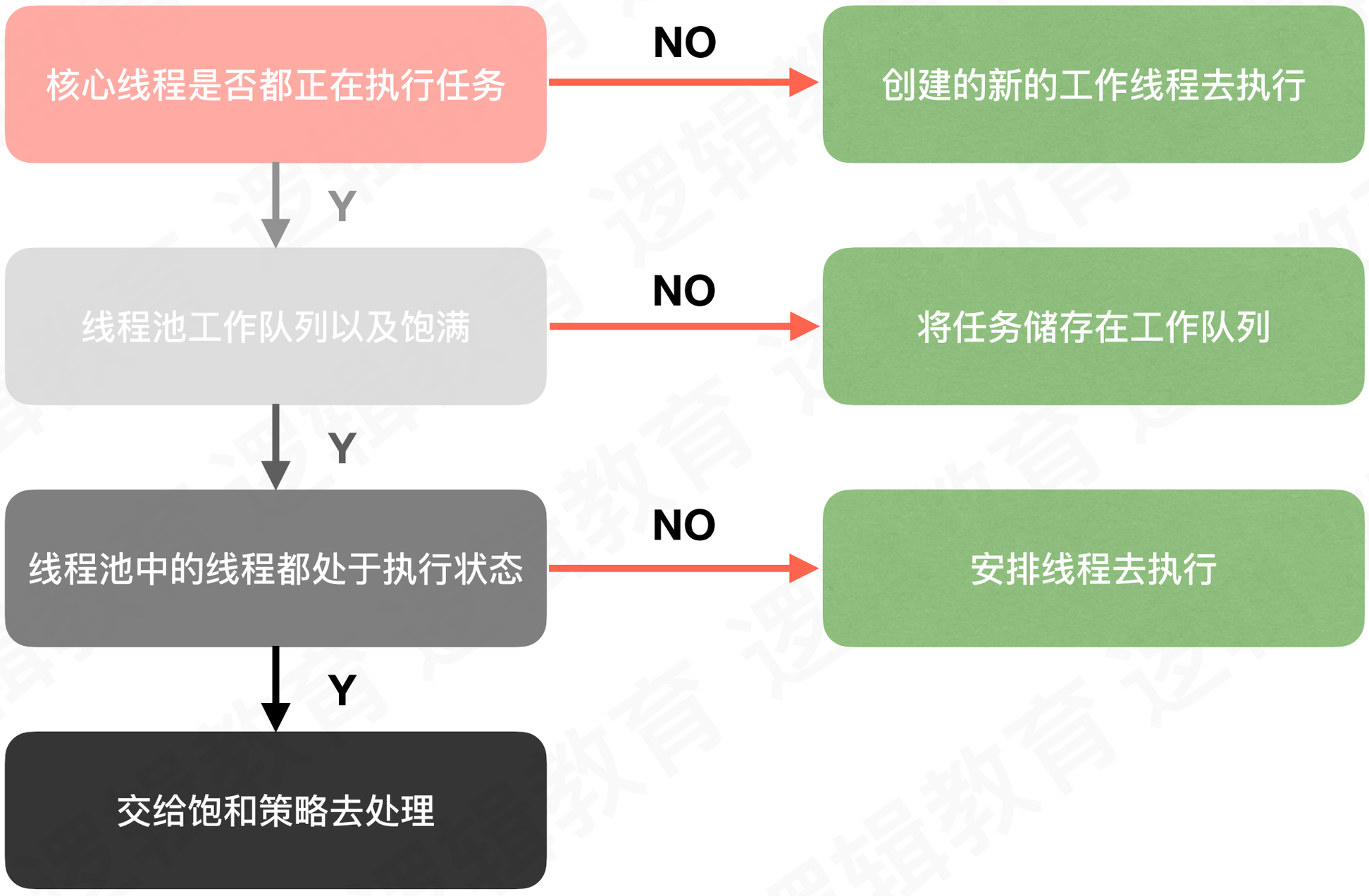
* `__bridge_transfer`（也可以使用`CFBridgingRelease`）将`Core Foundation`的对象转换为Objective-C的对象，同时将对象（内存）的管理权交给ARC。

线程生命周期



逻辑教育
Logic education

线程池



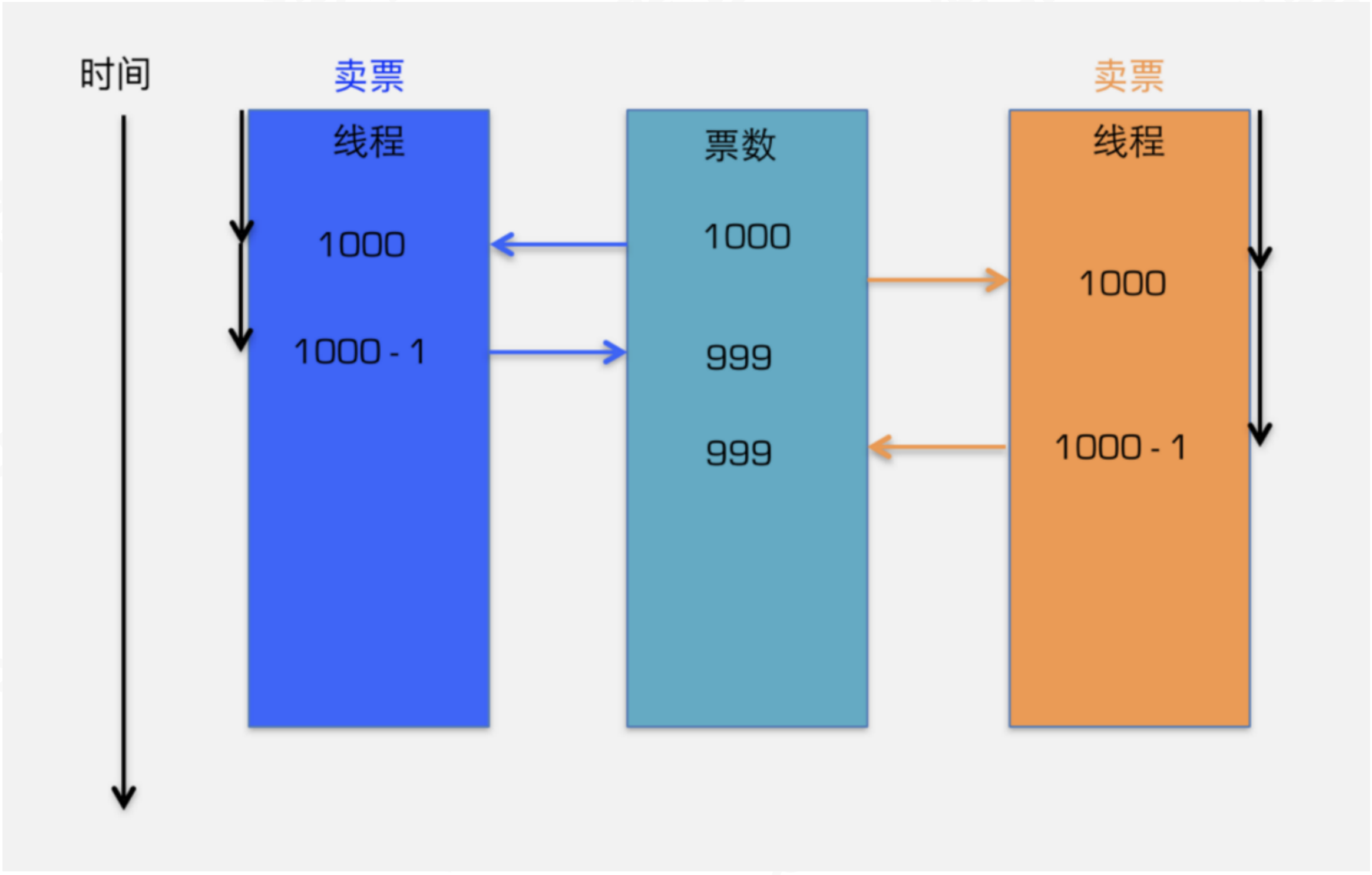
逻辑教育
Logic education

饱和策略

- AbortPolicy 直接抛出RejectedExecutionExeception异常来阻止系统正常运行
- CallerRunsPolicy 将任务回退到调用者
- DisOldestPolicy 丢掉等待最久的任务
- DisCardPolicy 直接丢弃任务

这四种拒绝策略均实现的RejectedExecutionHandler接口

线程经典案例



逻辑教育
Logic education

互斥锁

* 互斥锁小结
  * 保证锁内的代码，同一时间，只有一条线程能够执行！
  * 互斥锁的锁定范围，应该尽量小，锁定范围越大，效率越差！

* 互斥锁参数
  * 能够加锁的任意 NSObject 对象
  * 注意：锁对象一定要保证所有的线程都能够访问
  * 如果代码中只有一个地方需要加锁，大多都使用 self，这样可以避免单独再创建一个锁对象

逻辑教育
Logic education

atomic与nonatomic 的区别

nonatomic 非原子属性

atomic 原子属性(线程安全)，针对多线程设计的，默认值

保证同一时间只有一个线程能够写入(但是同一个时间多个线程都可以取值)

atomic 本身就有一把锁(自旋锁)

单写多读：单个线程写入，多个线程可以读取

atomic：线程安全，需要消耗大量的资源

nonatomic：非线程安全，适合内存小的移动设备

iOS 开发的建议

所有属性都声明为 nonatomic

尽量避免多线程抢夺同一块资源

尽量将加锁、资源抢夺的业务逻辑交给服务器端处理，减小移动客户端的压力

逻辑教育
Logic education

线程和Runloop的关系

1：runloop与线程是一一对应的，一个runloop对应一个核心的线程，为什么说是核心的，是因为runloop是可以嵌套的，但是核心的只能有一个，他们的关系保存在一个全局的字典里。

2：runloop是来管理线程的，当线程的runloop被开启后，线程会在执行完任务后进入休眠状态，有了任务就会被唤醒去执行任务。

3：runloop在第一次获取时被创建，在线程结束时被销毁。

4：对于主线程来说，runloop在程序一启动就默认创建好了。

5：对于子线程来说，runloop是懒加载的，只有当我们使用的时候才会创建，所以在子线程用定时器要注意：确保子线程的runloop被创建，不然定时器不会回调。

逻辑教育
Logic education

Hello Cooci

我就是我，颜色不一样的烟火