21 | 除了 Cocoa，iOS还可以用哪些 GUI 框架开发?

18 | 怎么减少 App 电量消耗?

27 | 如何用 Flexbox 思路开发？跟自动布局比，Flexbox 好在哪?

34 | iOS 黑魔法 Runtime Method Swizzling 背后的原理

43 | 剖析使 App 具有动态化和热更新能力的方案

28 | 怎么应对各种富文本表现需求?

22 | 细说 iOS 响应式框架变迁，哪些思想可以为我所用?

15 | 日志监控：怎样获取 App 中的全量日志?

29 | 如何在 iOS 中进行面向测试驱动开发和面向行为驱动开发?

25 | 怎样构建底层的发布和订阅事件总线?

02 | App 启动速度怎么做优化与监控?

16 | 性能监控：衡量 App 质量的那把尺

26 | 如何提高 JSON 解析的性能?

23 | 如何构造酷炫的物理效果和过场动画效果?

31 | iOS 开发学习资料和书单推荐

17 | 远超你想象的多线程的那些坑

多线程中的那些坑

三方库都谨慎使用多线程

AFNetworking

在 AFNetworking 2.0 中,把每个请求都封装成了单独的 NSOperationQueue,再由 NSOperationQueue 根据当前的 CPU 数量和系统负载来控制并发。那么,为什么AFNetworking 2.0 没有为每个请求创建一个线程,而只是创建了一个线程,用来接收 NSOperationQueue 的回调呢?

FMDB

FMDB 只通过 FMDatabaseQueue 开启了一个线程队列,来串行地操作数据库。这,又是为什么呢?

UIKit

UIKit 干脆就做成了线程不安全,只能在主线程上操作。

常驻线程

常驻线程定义

常驻线程,指的就是那些不会停止,一直存在于内存中的线程。我们在文章开始部分,说到的 AFNetworking 2.0 专门创建了一个线程来接收 NSOperationQueue 的回调,这个线程其实就是一个常驻线程。

如何创建常驻线程（AFNetworking2.0）

```
+ (void)networkRequestThreadEntryPoint:(id)__unused object {
    @autoreleasepool {
        [[NSThread currentThread] setName:@"AFNetworking"];

        NSRunLoop *runLoop = [NSRunLoop currentRunLoop];
        [runLoop addPort:[NSMachPort port] forMode:NSDefaultRunLoopMode];
        [runLoop run];
    }
}
```

创建常驻线程的固定写法，即创建一个线程，并为其添加runloop

常驻线程的隐患

如果你有 30 个或 每个库都常驻一个线程，那这样做,不但不能提高 CPU 的利用率,反而会降低程序的执行效率。也就是说,这样做的话,就不是充分利用而是浪费 CPU 资源了。

为什么使用常驻线程

NSURLConnection 发起请求后,所在的线程需要一直存活,以等待接收NSURLConnectionDelegate 回调方法。但是,网络返回的时间不确定,所以这个线程就需要一直常驻在内存中。

主线程还要处理大量的 UI 和交互工作,为了减少对主线程的影响,所以AFNetworking 2.0 就新建了一个常驻线程,用来处理所有的请求和回调。

如何优雅使用常驻线程

AFNetworking 在 3.0 版本时,使用苹果公司新推出的 NSURLSession 替换了NSURLConnection,从而避免了常驻线程这个坑。
NSURLSession 可以指定回调 NSOperationQueue,这样请求就不需要让线程一直常驻在内存里去等待回调了。

```
[[AFURLSessionManager manager]
[[AFURLSessionManager manager]
[[AFURLSessionManager manager]
```

并发（GCD）

GCD并发存在的性能问题

在进行数据读写操作时,总是需要一段时间来等待磁盘响应的,如果在这个时候通过GCD 发起了一个任务,那么 GCD 就会本着最大化利用 CPU 的原则,会在等待磁盘响应的这个空档,再创建一个新线程来保证能够充分利用 CPU。

而如果 GCD 发起的这些新任务,都是类似于就跟存储这样需要等待磁盘响应的任务的话, 那么随着任务数量的增加,GCD 创建的新线程就会越来越多,从而导致内存资源越来越紧张,等到磁盘开始响应后,再读取数据又会占用更多的内存。结果就是,本应的内存占用会引起更多的内存问题。

如何避免并发性能问题

这种情况最典型的场景就是数据库读写操作。FMDB是一个开源的第三方数据库框架,通过 FMDatabaseQueue 这个核心类,将与读写数据库相关的磁盘操作都放到一个串行队列里执行,从而避免了线程创建过多导致系统资源紧张的情况。

内存问题

内存开销

创建线程的过程,需要用到物理内存,CPU 也会消耗时间。而且,新建一个线程,系统还需要为这个进程空间分配一定的内存作为线程堆栈。堆栈大小是 4KB 的倍数。在 iOS 开发中,主线程堆栈大小是 1MB,新创建的子线程堆栈大小是 512KB。

寻址开销

除了内存开销外,线程创建得多了,CPU 在切换线程上下文时,还会更新寄存器,更新寄存器的时候需要寻址,而寻址的过程还会有较大的 CPU 消耗。

精品留言

关于NSURLConnection和URLSession那块有一个地方讲的感觉不是很清楚。NSURLConnection也可以设置执行代理的queue,通过setDelegateQueue：方法，本质上是NSURLConnection的机制需要创建此对象的线程的RunLoop去监听网络事件，然后执行注册的回调。所以在网络回来前，需要保持RunLoop的运行状态。NSURLSession应该是内部做了这个事情，所以发请求不需要所在线程“常驻”，而不是因为我们可以设置delegateQueue。从代

01 | 建立你自己的 iOS开发知识体系

04 | 项目大了人员多了，架构怎么设计更合理?

03 | Auto Layout 是怎么进行自动布局的，性能如何?

24 | A-B 测试：验证决策效果的利器

05 | 链接器：符号是怎么绑定到地址上的?

09 | 无侵入的埋点方案如何实现?

10 | 包大小：如何从资源和代码层面实现全方位瘦身?

12 | iOS 崩溃千奇百怪，如何全面监控?

06 | App 如何通过注入动态库的方式实现极速编译调试?

07 | Clang、Infer 和 OCLint ，我们应该使用谁来做静态分析?

08 | 如何利用 Clang 为 App 提质?

13 | 如何利用 RunLoop 原理去监控卡顿?

14 | 临近 OOM，如何获取详细内存分配信息，分析内存问题?