第8节课内容总结

程序启动

在启动App时，真正的加载过程是从exec()函数开始，系统会调用exec()函数创建进程，并且分配内存。然后会执行以下的操作

1. 把App对应的可执行文件加载到内存。
2. 把dyld加载到内存。dyld也是一个可执行的程序
3. dyld进行动态链接。

dyld的具体工作内容

1. dyld会找到可执行文件的依赖动态库。接着dyld会将所依赖的动态库加载到内存中。这是一个递归的过程，依赖的动态库可能还会依赖别的动态库，所以dyld会递归每个动态库，直至所有的依赖库都被加载完毕。
2. Rebase和Bind。rebase修复的是指向当前镜像内部的资源指针；而bind指向的是镜像外部的资源指针
3. 调起main函数，也就是我们程序的入口，然后我们的程序就开始执行了。

dyld进行初始化的流程

dyld是用来加载可执行文件所依赖的动态库的。然后会对可执行文件和可执行文件所依赖的动态库进行初始化的操作。

在进行初始化的操作的时候首先会初始化libsystem，否则就会报错。因为在进行libsystem初始化的时候，会初始化libdispatch，在进行libdispatch初始化的时候，会初始化libobjc，其他的库，可能需要依赖runtime基础或者线程相关的基础。所以libsystem的初始化必须放在第一位。

在libobjc进行初始化的时候，会调用一个_dyld_objc_notify_register函数，这个函数会给dyld传递三个回调函数。

1. map_images: dyld将image镜像文件加载进内存时，会触发该函数
2. load_images: dyld初始化image会触发该函数
3. unmap_image: dyld将image移除时会触发该函数

然后，dyld会调用`map_images`和`load_images`来对image进行初始化的操作。