iOS Libs 与 Frameworks 的理解和使用

axl411

- 高度解耦、模块化
- 每个模块在自己的工程中开发调试
- 用 cocoapods 整合

理解 Libs 与 Frameworks

- 动态库、静态库都是依赖管理的形式
- Mach-O：macOS 与 iOS 使用的二进制文件格式，包含动态库、静态库、可执行文件

静态库

- 源代码 -> 中间对象文件（Intermediate Object File） -> 静态库（对象文件的集合）
- 链接时直接加入到目标中

创建静态库

```
// bar.h
#ifndef __foo__bar__
#define __foo__bar__
#include <stdio.h>

int fizz();

#endif /* defined(__foo__bar__) */
```

```
// bar.c
#include "bar.h"
#include <CoreFoundation/CoreFoundation.h>

int fizz() {
  CFShow(CFSTR("buzz"));
  return 0;
}
```

```
$ clang -c bar.c -o bar.o -arch x86_64
$ file bar.o
bar.o: Mach-O 64-bit object x86_64
```

```
$ libtool -static bar.o -o libfoo_static.a
$ file libfoo_static.a
libfoo_static.a: current ar archive random library
```

```c
// main.c
#include "bar.h"
int main() {
  return fizz();
}
```

```
$ clang -c main.c -o main.o -arch x86_64
$ file main.o
main.o: Mach-O 64-bit object x86_64

$ ld main.o -framework CoreFoundation -lSystem -L. -
lfoo_static -o test_static -arch x86_64
$ file test_static
test_static: Mach-O 64-bit executable x86_64

$ ./test_static
buzz
```

```
$ nm test_static
                 U _CFShow
                 U ___CFConstantStringClassReference
0000000100000000 T __mh_execute_header
0000000100000f70 T _fizz
0000000100000f50 T _main
                 U dyld_stub_binder

$ nm libfoo_static.a

libfoo_static.a(bar.o):
                 U _CFShow
                 U ___CFConstantStringClassReference
0000000000000000 T _fizz

$ nm bar.o
                 U _CFShow
                 U ___CFConstantStringClassReference
0000000000000000 T _fizz
```

典型的静态库 - 微信 SDK

- “SDK文件包括 libWeChatSDK.a，WXApi.h, WXApiObject.h 三个。”
- “开发者需要在工程中链接上:SystemConfiguration.framework, libz.dylib, libsqlite3.0.dylib, libc++.dylib, Security.framework, CoreTelephony.framework, CFNetwork.framework”

动态库

- 源代码 -> 中间对象文件（Intermediate Object File） -> 动态库
- 程序启动时由 dyld 载入到程序的进程中
- 包含依赖信息

创建、使用动态库

```
$ libtool -dynamic bar.o -o libfoo_dynamic.dylib -framework
CoreFoundation -lSystem
$ file libfoo_dynamic.dylib
libfoo_dynamic.dylib: Mach-O 64-bit dynamically linked
shared library x86_64

$ ld main.o -lSystem -L. -lfoo_dynamic -o test_dynamic -arch
x86_64
$ file test_dynamic
test_dynamic: Mach-O 64-bit executable x86_64

$ ./test_dynamic
buzz
```

```
$ nm test_dynamic
0000000100000000 T __mh_execute_header
                 U _fizz
0000000100000f70 T _main
                 U dyld_stub_binder

$ nm libfoo_dynamic.dylib
                 U _CFShow
                 U ___CFConstantStringClassReference
0000000000000f70 T _fizz
                 U dyld_stub_binder

$ nm bar.o
                 U _CFShow
                 U ___CFConstantStringClassReference
0000000000000000 T _fizz
```

Framework

- 包含额外资源的文件夹 - 头文件、nib、图片、文档、多语言资源等

静态 vs 动态：依赖关系

```
$ otool -L test_static
test_static:
   /System/Library/Frameworks/CoreFoundation.framework/
Versions/A/CoreFoundation (compatibility version 150.0.0,
current version 1348.28.0)
   /usr/lib/libSystem.B.dylib (compatibility version 1.0.0,
current version 1238.0.0)

$ otool -L libfoo_static.a
Archive : libfoo_static.a
libfoo_static.a(bar.o):
```

```
$ otool -L test_dynamic
test_dynamic:
   /usr/lib/libSystem.B.dylib (compatibility version 1.0.0,
current version 1238.0.0)
   libfoo_dynamic.dylib (compatibility version 0.0.0,
current version 0.0.0)

$ otool -L libfoo_dynamic.dylib
libfoo_dynamic.dylib:
   libfoo_dynamic.dylib (compatibility version 0.0.0,
current version 0.0.0)
   /System/Library/Frameworks/CoreFoundation.framework/
Versions/A/CoreFoundation (compatibility version 150.0.0,
current version 1348.28.0)
   /usr/lib/libSystem.B.dylib (compatibility version 1.0.0,
current version 1238.0.0)
```

静态 vs 动态： 架构

- [模拟器] i386 x86_64
- [真机] armv7 arm64

- 静态链接：从静态库中直接取出目标需要的架构
- 动态链接：嵌入 - embedded library）

```
eleme.app
├── Frameworks
|   └── NVMWeChatSDK.framework
```

- 动态链接：需要包含所有的架构（模拟器、真机）

静态 vs 动态：对启动时间的影响

- 静态：载入 app 的二进制
- 动态：dyld 还需要载入动态库的依赖
- WWDC 2016 Session 406：一个 app 启动会加载数百个系统的动态库，这是优化过的；而自己的动态库是越少越好（26个 => 200+ms VS 2个 => 20+ms）

使用 Libs 与 Frameworks

- “将 libWeChatSDK.a，WXApi.h，WXApiObject.h 这三个拖到主工程...”
- “开发者需要在工程中链接上:a.framework, b.dylib, c.dylib, d.dylib, e.framework, f.framework, g.framework...”
- “将 a.png，b.png，c.js 拖到主工程...”

cocoapods

```
Pod::Spec.new do |spec|
  spec.name         = 'Reachability'
  spec.version      = '3.1.0'
  spec.license      = { :type => 'BSD' }
  spec.homepage     = 'https://github.com/tonymillion/
Reachability'
  spec.authors      = { 'Tony Million' =>
'tonymillion@gmail.com' }
  spec.summary      = 'ARC and GCD Compatible Reachability
Class for iOS and OS X.'
  spec.source       = { :git => 'https://github.com/
tonymillion/Reachability.git', :tag => 'v3.1.0' }
  spec.source_files = 'Reachability.{h,m}'
  spec.framework    = 'SystemConfiguration'
end
```

用 cocoapods 封装库

```
spec.vendored_frameworks = 'a.framework', 'b.framework'
spec.vendored_libraries = 'liba.a', 'libb.a'
spec.source_files = ‘Headers/Public/*.h'
spec.libraries = 'xml2', 'z'
spec.frameworks = 'QuartzCore', ‘CoreData'
spec.weak_framework = ‘UserNotifications’ # weak linking
```

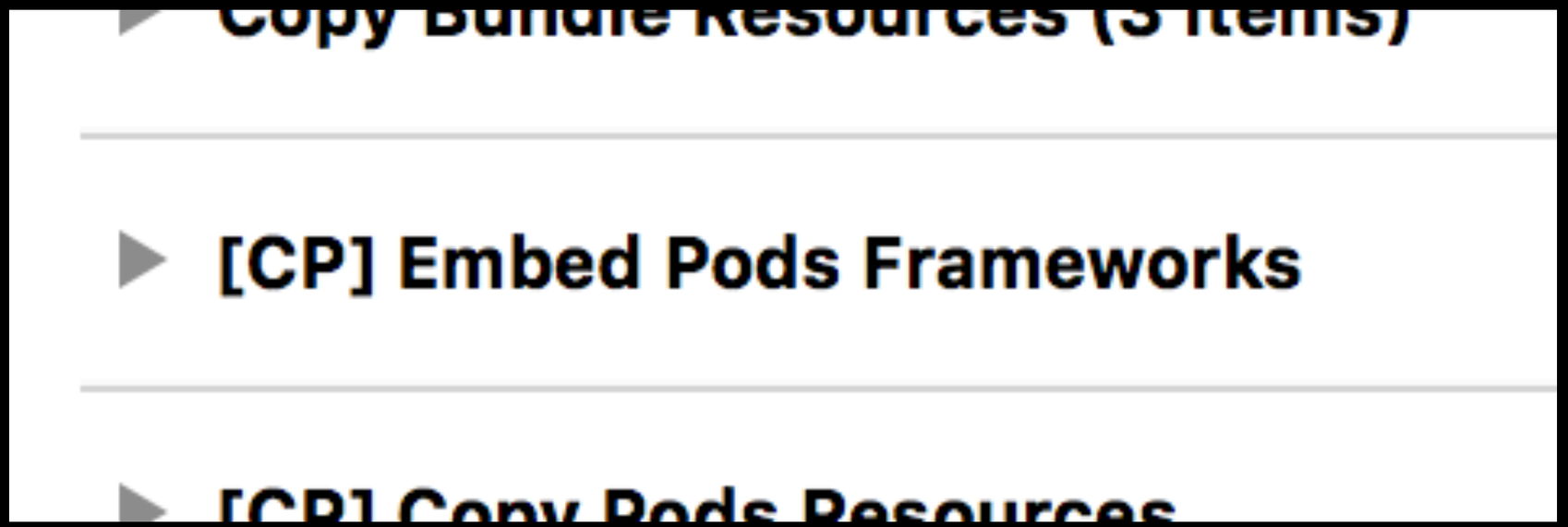
cocoapods 切换动、静态链接

```
# Podfile

# use_frameworks!

use_frameworks!
```

- import 头文件的写法：<AFNetworking.h> -> <AFNetworking/AFNetworking.h>
- Transitive dependencies include static binaries：直接把 static binary 放在直接的依赖中；静态库 -> 动态库
- 资源的读取
- weak dependency

静态库 -> 动态库

- 创建“Cocoa Touch Framework”项目，正常引入静态库
- 为模拟器编译（i386 x86_64）
- 为真机编译（armv7 arm64）
- 用 lipo 合并


Copy Bundle Resources (3 items)
[CP] Embed Pods Frameworks
[CP] Copy Pods Resources

```
# Strip invalid architectures
strip_invalid_archs() {
  binary="$1"
  # Get architectures for current file
  archs="$(lipo -info "$binary" | rev | cut -d ':' -f1 |
rev)"
  stripped=""
  for arch in $archs; do
    if ! [[ "${VALID_ARCHS}" == *"$arch"* ]]; then
      # Strip non-valid architectures in-place
      lipo -remove "$arch" -output "$binary" "$binary" ||
exit 1
      stripped="$stripped $arch"
    fi
  done
  if [[ "$stripped" ]]; then
    echo "Stripped $binary of architectures:$stripped"
  fi
}
```

资源的读取

```
s.name = ‘MyPod’

# strongly recommend against…
s.resources = “#{PATH_TO_RESOURCE}”

# strongly recommend…
s.resource_bundles = { 'MyPod' => "#{PATH_TO_RESOURCE}" }
```

```
s.resources

// use_frameworks!
MyApp.app/Frameworks/MyPod.framework/img.png
// # use_frameworks!
MyApp.app/img.png
// # use_frameworks! 安装到 Cocoa Touch Framework target 下，然后把这个 framework 给主工程使用
MyApp.app/Frameworks/MyHandMadeFramework.framework/img.png
```

```
[[NSBundle mainBundle] pathForResource:@"img"
                                ofType:@“png”]; ❌
MyApp.app/img.png


[[NSBundle bundleForClass:[self class]]
    pathForResource:@"img" ofType:@“png”]; ✅
MyApp.app/Frameworks/MyPod.framework/img.png
MyApp.app/img.png
MyApp.app/Frameworks/MyHandMadeFramework.framework/img.png
```

s.resource_bundles

```
// use_frameworks!
MyApp.app/Frameworks/MyPod.framework/MyPod.bundle/img.png
// # use_frameworks!
MyApp.app/MyPod.bundle/img.png
// # use_frameworks! 安装到 Cocoa Touch Framework target 下，然后把这个 framework 给主工程使用
MyApp.app/Frameworks/MyHandMadeFramework.framework/
MyPod.bundle/img.png
```

```
NSBundle *libBundle = [NSBundle bundleForClass:[self
class]];
NSString *resourceBundlePath = [[libBundle bundlePath]
stringByAppendingPathComponent:@"MyPod.bundle"];
resourceBundle = [NSBundle
bundleWithPath:resourceBundlePath];✅
MyApp.app/Frameworks/MyPod.framework/MyPod.bundle/img.png
MyApp.app/MyPod.bundle/img.png
MyApp.app/Frameworks/MyHandMadeFramework.framework/
MyPod.bundle/img.png
```

+[UIImage imageNamed:
inBundle:
compatibleWithTraitCollection:]

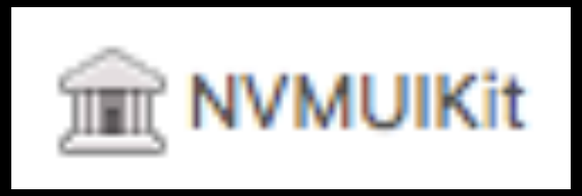
nibs also works
images specified in nibs(IB) also works when they are in the same directory

Weak Dependency

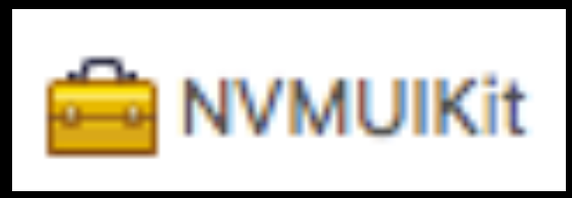
- MyLib 要用 Crashlytics 的代码，又不想添加对 Crashlytics 的依赖

```
#if __has_include(<Crashlytics/Crashlytics.h>)
#endif
```

- 只在静态链接下 work

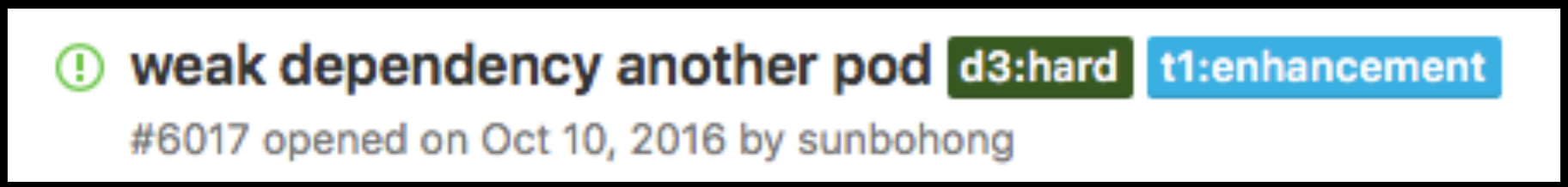


HEADER_SEARCH_PATHS = "${PODS_ROOT}/Headers/Private" "${PODS_ROOT}/Headers/Private/NVMUIKit" "${PODS_ROOT}/Headers/Public" "${PODS_ROOT}/Headers/Public/1PasswordExtension" "${PODS_ROOT}/Headers/Public/AFNetworking" "${PODS_ROOT}/Headers/Public/APFConfigManager" "${PODS_ROOT}/Headers/Public/APFFoundation" "${PODS_ROOT}/Headers/Public/APFRegistry" "${PODS_ROOT}/Headers/Public/AndurilPatch" "${PODS_ROOT}/Headers/Public/Appirater" "${PODS_ROOT}/Headers/Public/BlocksKit" "${PODS_ROOT}/Headers/Public/CocoaLumberjack" "${PODS_ROOT}/Headers/Public/Crashlytics" "${PODS_ROOT}/Headers/Public/DeadPool" "${PODS_ROOT}/Headers/Public/ELMEnvironment" "${PODS_ROOT}/Headers/Public/ELMFoundation" "${PODS_ROOT}/Headers/Public/ELMUpgradeManager" "${PODS_ROOT}/Headers/Public/ELMWebViewJavascripBridge-nevermore" "${PODS_ROOT}/Headers/Public/EPSPay" "${PODS_ROOT}/Headers/Public/FBSnapshotTestCase" "${PODS_ROOT}/Headers/Public/FLEX" "${PODS_ROOT}/Headers/Public/Fabric" "${PODS_ROOT}/Headers/Public/GZIP" "${PODS_ROOT}/Headers/Public/Geohash" "${PODS_ROOT}/Headers/Public/IGListKit" "${PODS_ROOT}/Headers/Public/JSPatch" "${PODS_ROOT}/Headers/Public/Kiwi" "${PODS_ROOT}/Headers/Public/LDNetDiagnoService" "${PODS_ROOT}/Headers/Public/MBProgressHUD" "${PODS_ROOT}/Headers/Public/MD5Digest" "${PODS_ROOT}/Headers/Public/Mantle" "${PODS_ROOT}/Headers/Public/Masonry" "${PODS_ROOT}/Headers/Public/NVMAccountModule" "${PODS_ROOT}/Headers/Public/NVMAlichat" "${PODS_ROOT}/Headers/Public/NVMApplicationModulePublic" "${PODS_ROOT}/Headers/Public/NVMAttributedStringMaker" "${PODS_ROOT}/Headers/Public/NVMBookingModule" "${PODS_ROOT}/Headers/Public/NVMBookingModulePublic" "${PODS_ROOT}/Headers/Public/NVMBreakfastModule" "${PODS_ROOT}/Headers/Public/NVMCartServiceModule" "${PODS_ROOT}/Headers/Public/NVMCartServiceModulePublic" "${PODS_ROOT}/Headers/Public/NVMCommonService" "${PODS_ROOT}/Headers/Public/NVMCoreLocation" "${PODS_ROOT}/Headers/Public/NVMDebugModule" "${PODS_ROOT}/Headers/Public/NVMDebugModulePublic" "${PODS_ROOT}/Headers/Public/NVMEnvironment" "${PODS_ROOT}/Headers/Public/NVMExtensionUtils" "${PODS_ROOT}/Headers/Public/NVMFoundation" "${PODS_ROOT}/Headers/Public/NVMImageMaker" "${PODS_ROOT}/Headers/Public/NVMJSONModel" "${PODS_ROOT}/Headers/Public/NVMLocationService" "${PODS_ROOT}/Headers/Public/NVMModuleManager" "${PODS_ROOT}/Headers/Public/NVMNetwork" "${PODS_ROOT}/Headers/Public/NVMOTPAuth" "${PODS_ROOT}/Headers/Public/NVMOrderModule" "${PODS_ROOT}/Headers/Public/NVMOrderModulePublic" "${PODS_ROOT}/Headers/Public/NVMPOIAddressService" "${PODS_ROOT}/Headers/Public/NVMPayModule" "${PODS_ROOT}/Headers/Public/NVMPayModulePublic" "${PODS_ROOT}/Headers/Public/NVMPerf" "${PODS_ROOT}/Headers/Public/NVMPush" "${PODS_ROOT}/Headers/Public/NVMRouter" "${PODS_ROOT}/Headers/Public/NVMShoppingModule" "${PODS_ROOT}/Headers/Public/NVMShoppingModulePublic" "${PODS_ROOT}/Headers/Public/NVMTableViewKit" "${PODS_ROOT}/Headers/Public/NVMThirdPartySDKWrapper" "${PODS_ROOT}/Headers/Public/NVMTracker" "${PODS_ROOT}/Headers/Public/NVMUIKit" "${PODS_ROOT}/Headers/Public/NVMUIKitCore" "${PODS_ROOT}/Headers/Public/NVMUserModulePublic" "${PODS_ROOT}/Headers/Public/NVMWebImage" "${PODS_ROOT}/Headers/Public/NVMWechatSDK" "${PODS_ROOT}/Headers/Public/NVMWeex" "${PODS_ROOT}/Headers/Public/OAStackView" "${PODS_ROOT}/Headers/Public/RegExCategories" "${PODS_ROOT}/Headers/Public/Reveal-iOS-SDK" "${PODS_ROOT}/Headers/Public/SAMKeychain" "${PODS_ROOT}/Headers/Public/SDWebImage" "${PODS_ROOT}/Headers/Public/SZTextView" "${PODS_ROOT}/Headers/Public/SocketRocket" "${PODS_ROOT}/Headers/Public/StingSSLPin" "${PODS_ROOT}/Headers/Public/Toast" "${PODS_ROOT}/Headers/Public/UICollectionViewLeftAlignedLayout" "${PODS_ROOT}/Headers/Public/UITableView+FDTemplateLayoutCell" "${PODS_ROOT}/Headers/Public/UIView+Positioning" "${PODS_ROOT}/Headers/Public/WBWebViewConsole" "${PODS_ROOT}/Headers/Public/WeexSDK" "${PODS_ROOT}/Headers/Public/YYModel" "${PODS_ROOT}/Headers/Public/libextobjc" "${PODS_ROOT}/Headers/Public/libwebp"




```
FRAMEWORK_SEARCH_PATHS = $(inherited) "$PODS_CONFIGURATION_BUILD_DIR/AFNetworking"
"$PODS_CONFIGURATION_BUILD_DIR/APFConfigManager" "$PODS_CONFIGURATION_BUILD_DIR/APFFoundation"
"$PODS_CONFIGURATION_BUILD_DIR/BlocksKit" "$PODS_CONFIGURATION_BUILD_DIR/CocoaLumberjack"
"$PODS_CONFIGURATION_BUILD_DIR/ELMFoundation" "$PODS_CONFIGURATION_BUILD_DIR/ELMWebViewJavascripBridge-
nevermore" "$PODS_CONFIGURATION_BUILD_DIR/GZIP" "$PODS_CONFIGURATION_BUILD_DIR/Geohash"
"$PODS_CONFIGURATION_BUILD_DIR/MBProgressHUD" "$PODS_CONFIGURATION_BUILD_DIR/Mantle"
"$PODS_CONFIGURATION_BUILD_DIR/Masonry" "$PODS_CONFIGURATION_BUILD_DIR/NVMAttributedStringMaker"
"$PODS_CONFIGURATION_BUILD_DIR/NVMEnvironment" "$PODS_CONFIGURATION_BUILD_DIR/NVMFoundation"
"$PODS_CONFIGURATION_BUILD_DIR/NVMImageMaker" "$PODS_CONFIGURATION_BUILD_DIR/NVMJSONModel"
"$PODS_CONFIGURATION_BUILD_DIR/NVMNetwork" "$PODS_CONFIGURATION_BUILD_DIR/NVMPerf"
"$PODS_CONFIGURATION_BUILD_DIR/NVMRouter" "$PODS_CONFIGURATION_BUILD_DIR/NVMTableViewKit"
"$PODS_CONFIGURATION_BUILD_DIR/NVMTracker" "$PODS_CONFIGURATION_BUILD_DIR/NVMUIKitCore"
"$PODS_CONFIGURATION_BUILD_DIR/NVMWebImage" "$PODS_CONFIGURATION_BUILD_DIR/RegExCategories"
"$PODS_CONFIGURATION_BUILD_DIR/SAMKeychain" "$PODS_CONFIGURATION_BUILD_DIR/SDWebImage"
"$PODS_CONFIGURATION_BUILD_DIR/SZTextView" "$PODS_CONFIGURATION_BUILD_DIR/UITableView+FDTemplateLayoutCell"
"$PODS_CONFIGURATION_BUILD_DIR/libextobjc" "$PODS_CONFIGURATION_BUILD_DIR/libwebp"
```

```
s.dependency ‘Crashlytics'
s.weak_dependency 'Crashlytics'
```


weak dependency another pod
d3:hard
t1:enhancement
#6017 opened on Oct 10, 2016 by sunbohong