

# Building libraries for iOS

## Going native

Alexander Dodatko  
2014



How you see your app



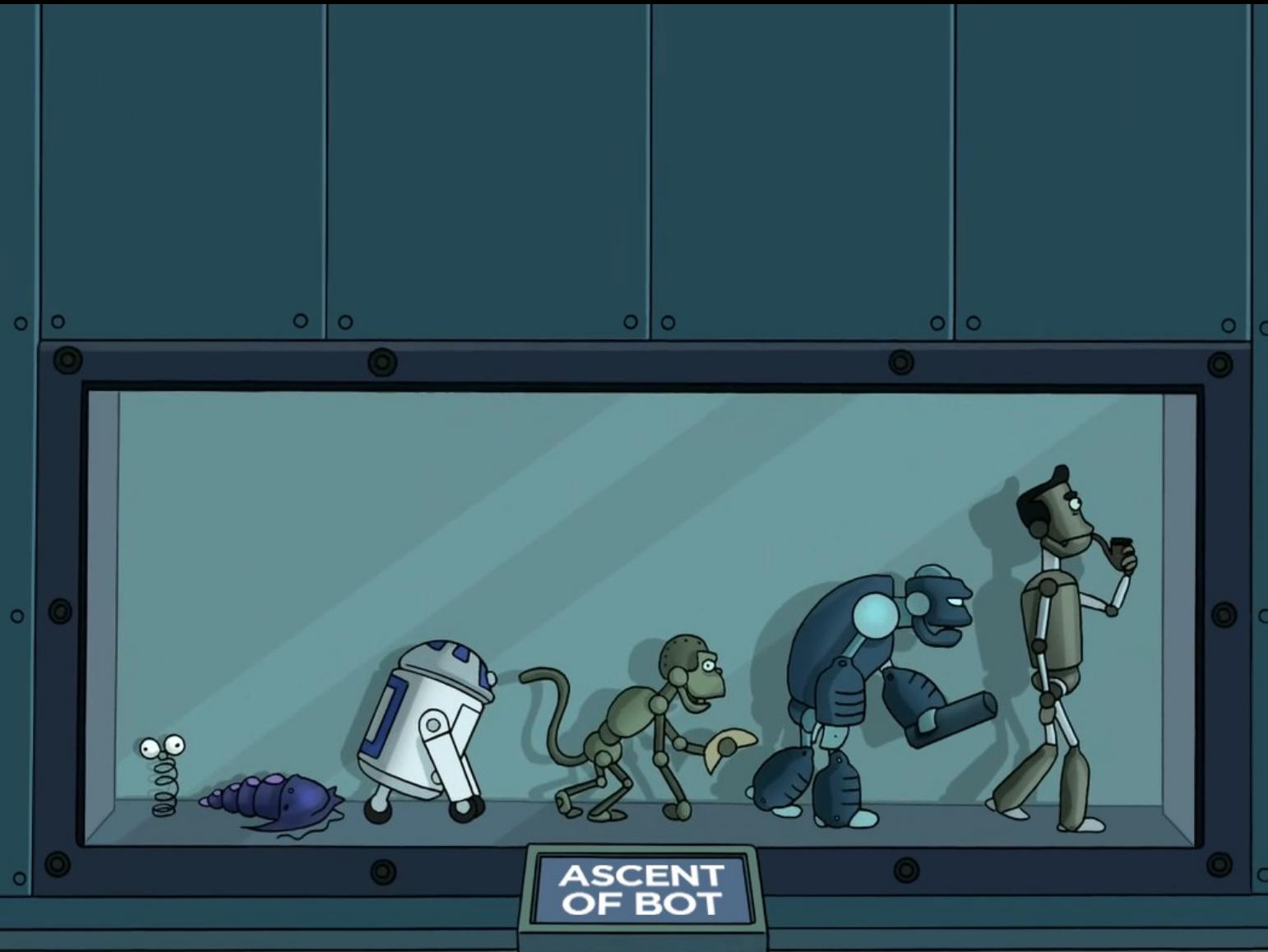
How the user sees your app



# The ONLY VALID MEASUREMENT OF Code QUALITY: WTFs/minute



# Code Reuse is Important

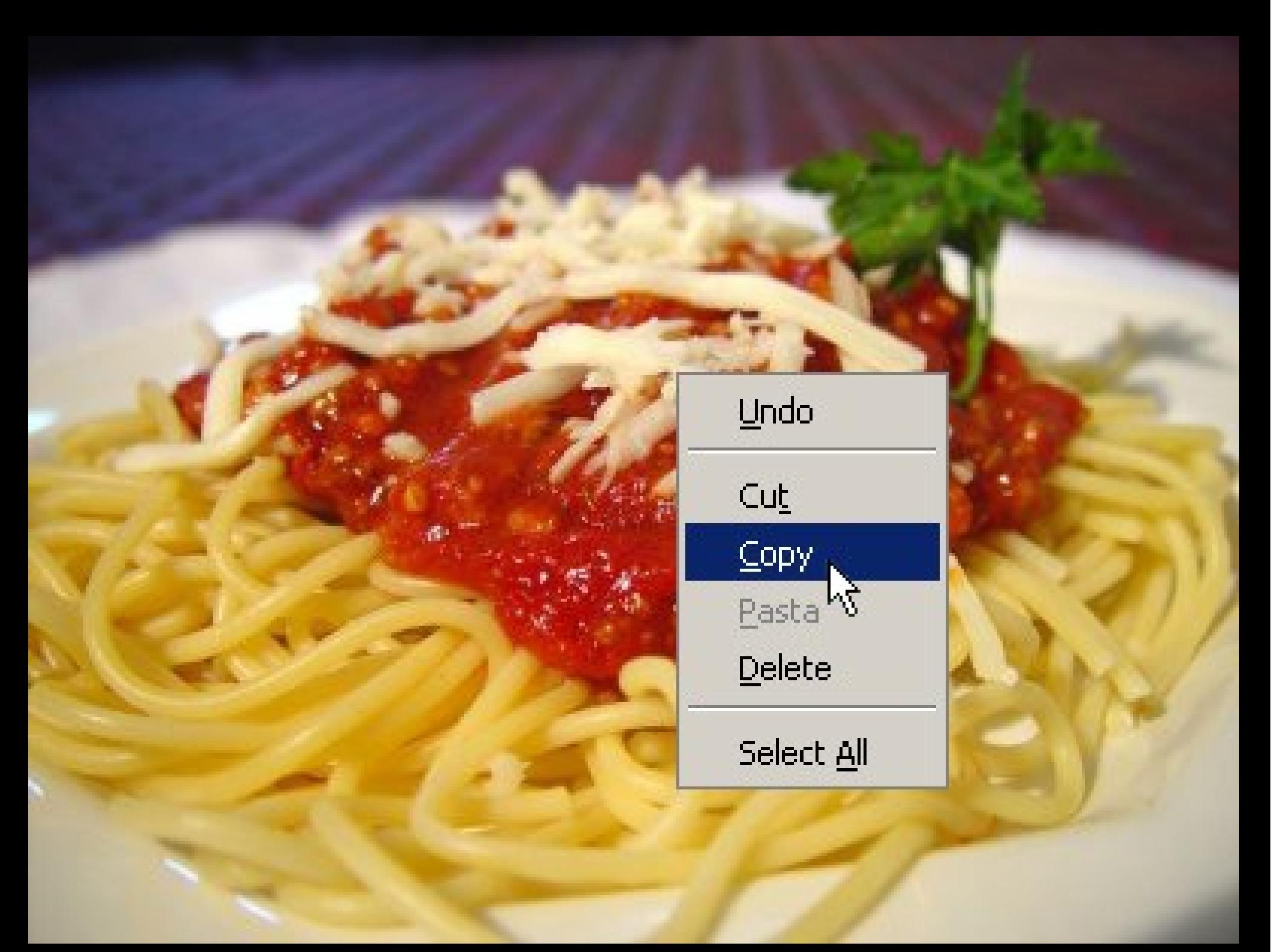


March 2008



<http://engt.co/1pOLKH9>

The fastest code is the code  
that reaches the market first

A close-up photograph of a plate of spaghetti. The pasta is coated in a vibrant red tomato sauce and topped with melted white cheese. A small sprig of green parsley is visible in the background. A context menu is overlaid on the image, centered on the spaghetti. The menu has a light gray background with black text and a dark blue selection bar. The options are: Undo, Cut, Copy, Paste, Delete, and Select All. The "Copy" option is highlighted with a dark blue rectangle and a white cursor arrow is pointing to its right.

Undo

---

Cut

Copy

Paste

Delete

---

Select All

Nobody Cares

August 2011



**CocoaPods**  
@CocoaPods



Follow

@oliverfoggin Thanks! Just over one year old:  
[github.com/CocoaPods/Coco...](https://github.com/CocoaPods/Coco...)

Reply Retweet Favorite More



**CocoaPods**

An Objective-C library dependency manager. Contribute to CocoaPods development by creating an account on GitHub.

[View on web](#)



4:36 AM - 2 Oct 2012

Flag media



**CocoaPods**

pod install SomeAwesomeLibXYZ



Except...



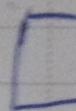




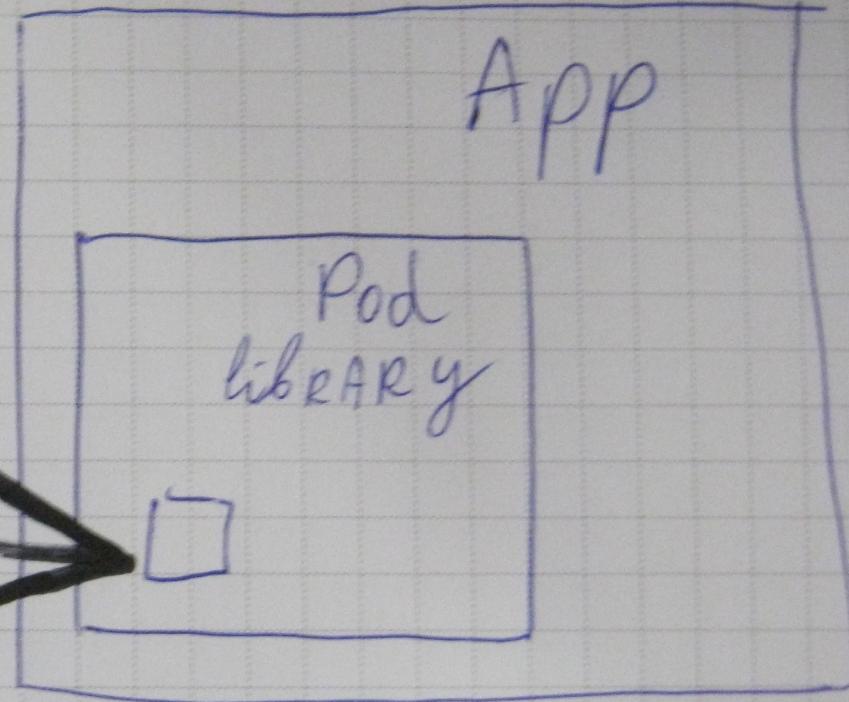
CocoaPods

# Typical Workflow

Github



files  
Pod install



Github

APP

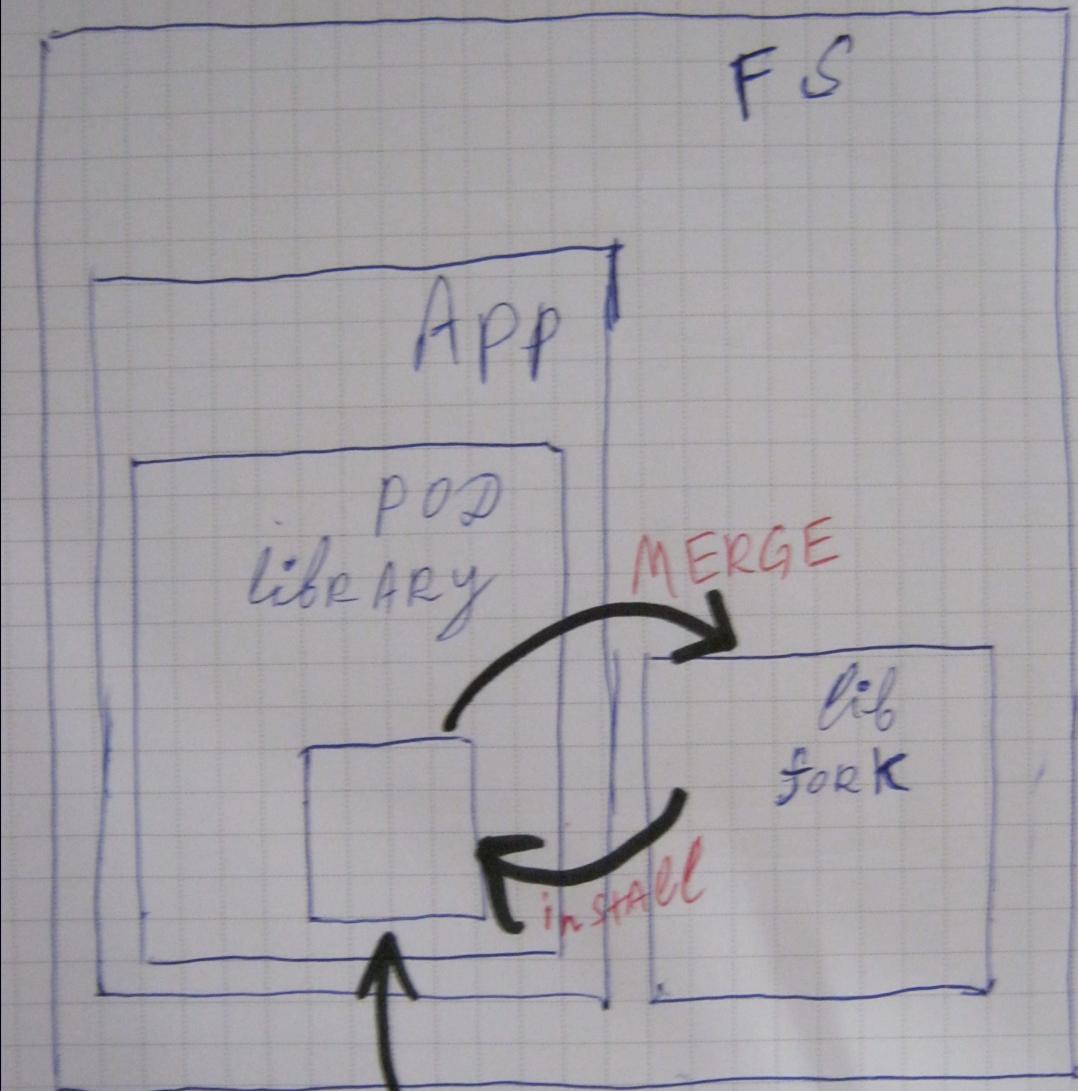
Siles  
Pod Install

Pod  
library



feature

F S



Debug  
and  
Fix

It is a Rare Case

Isn't it ?

App

.UI

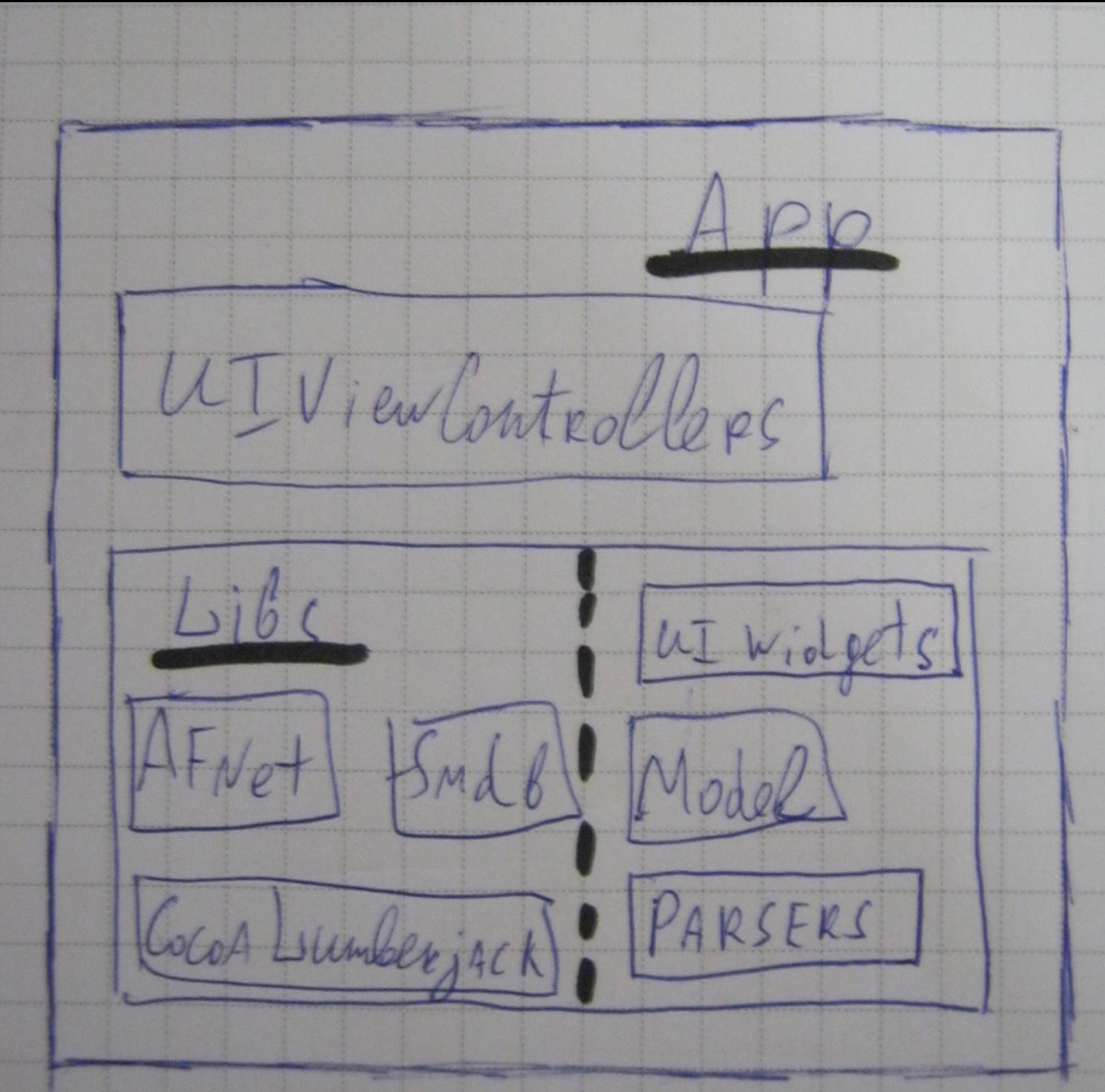
Model

Libs

AFNet

fmdb

CocoALumberjack

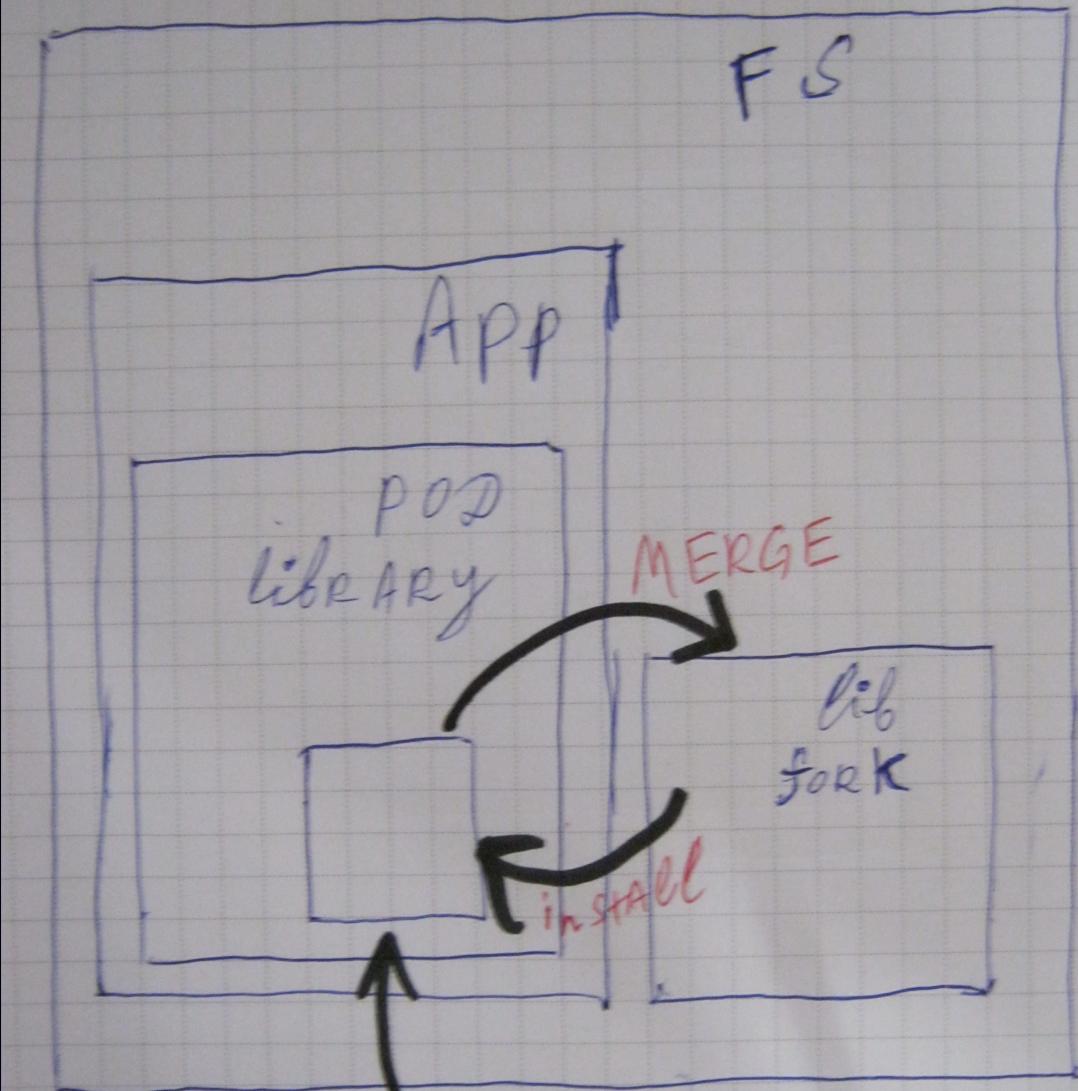




CocoaPods

--no-integrate

F S



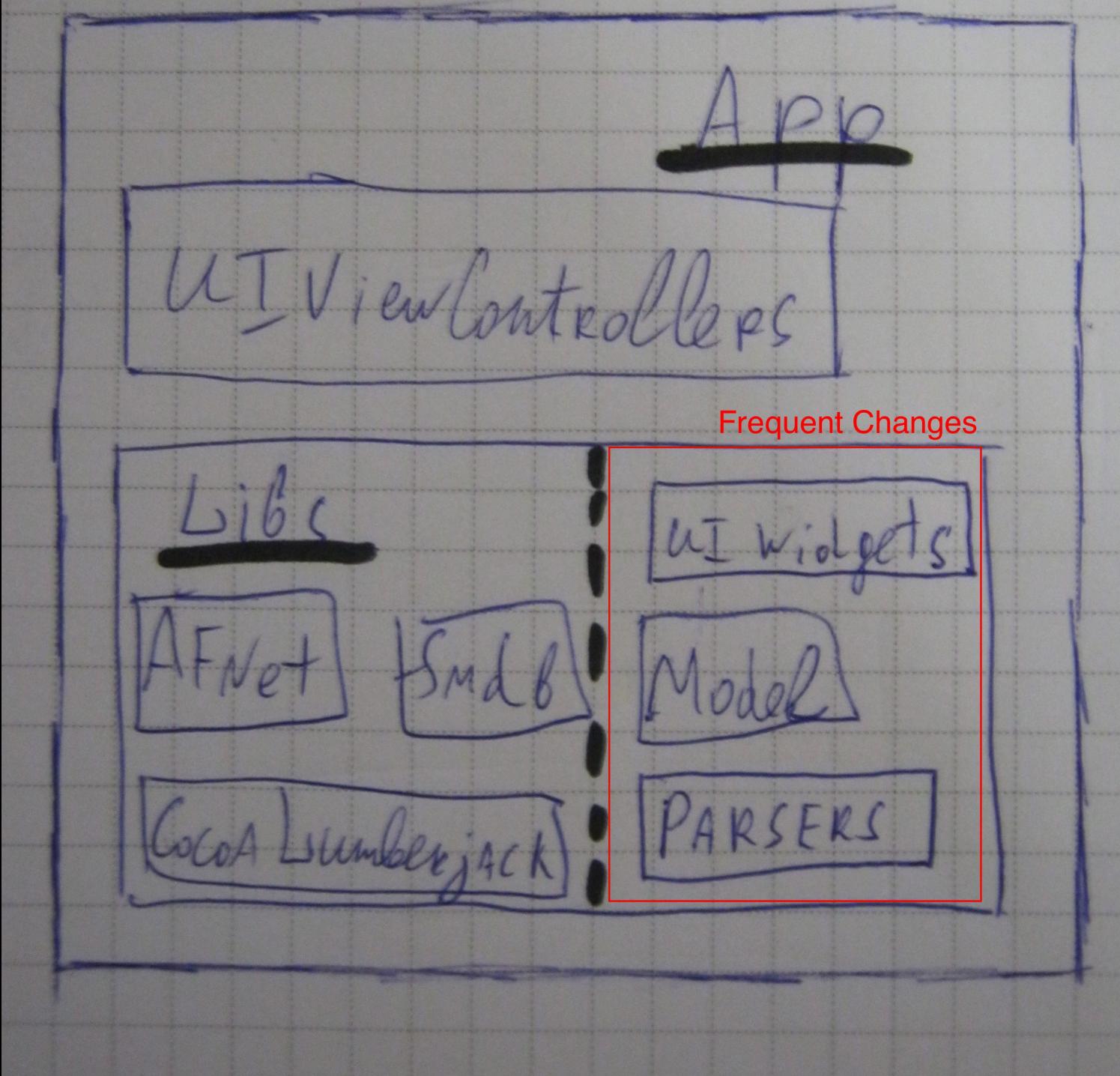
Debug  
and  
Fix

# Modular Architecture Benefits

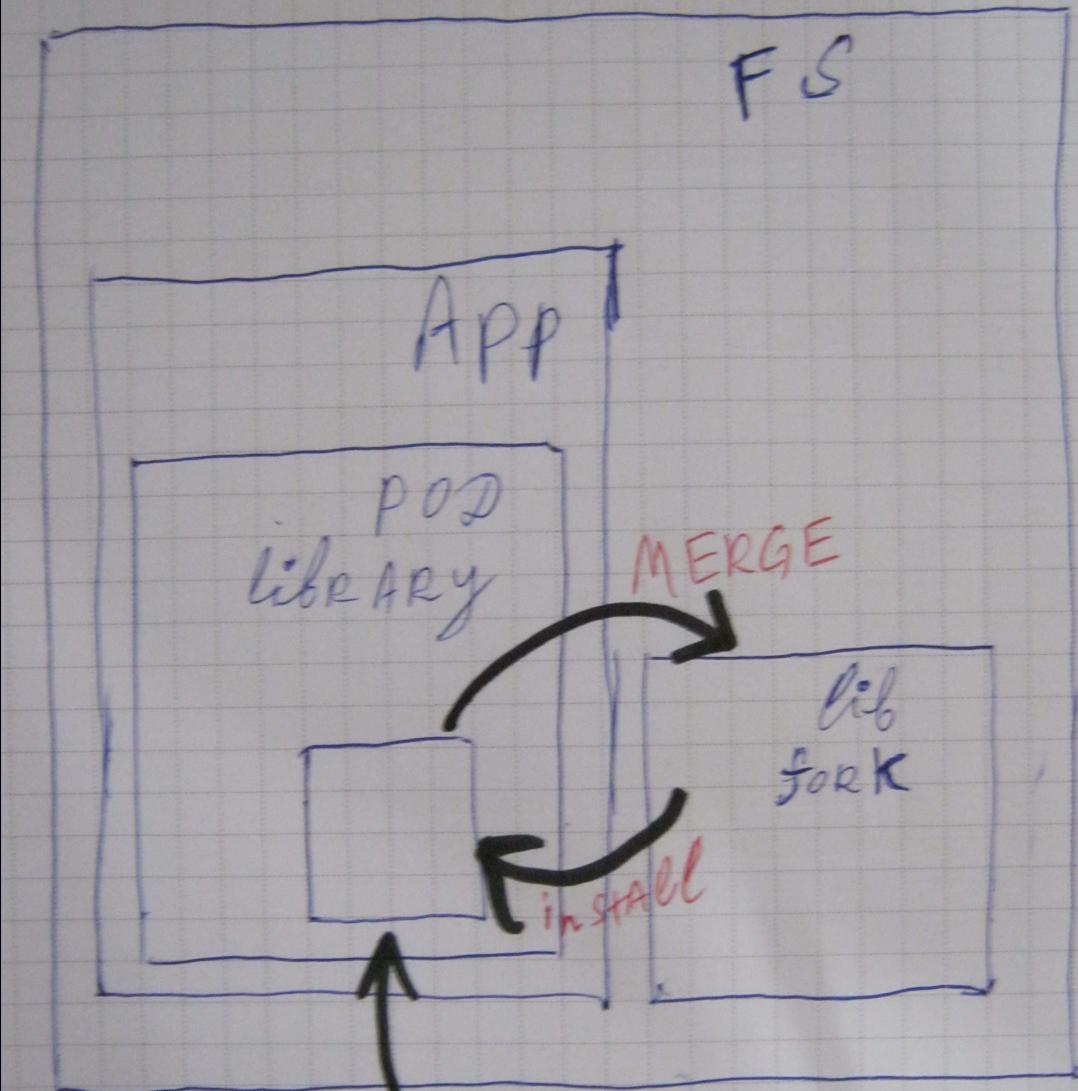
Code Reuse

Test Coverage

Easier to Apply Changes

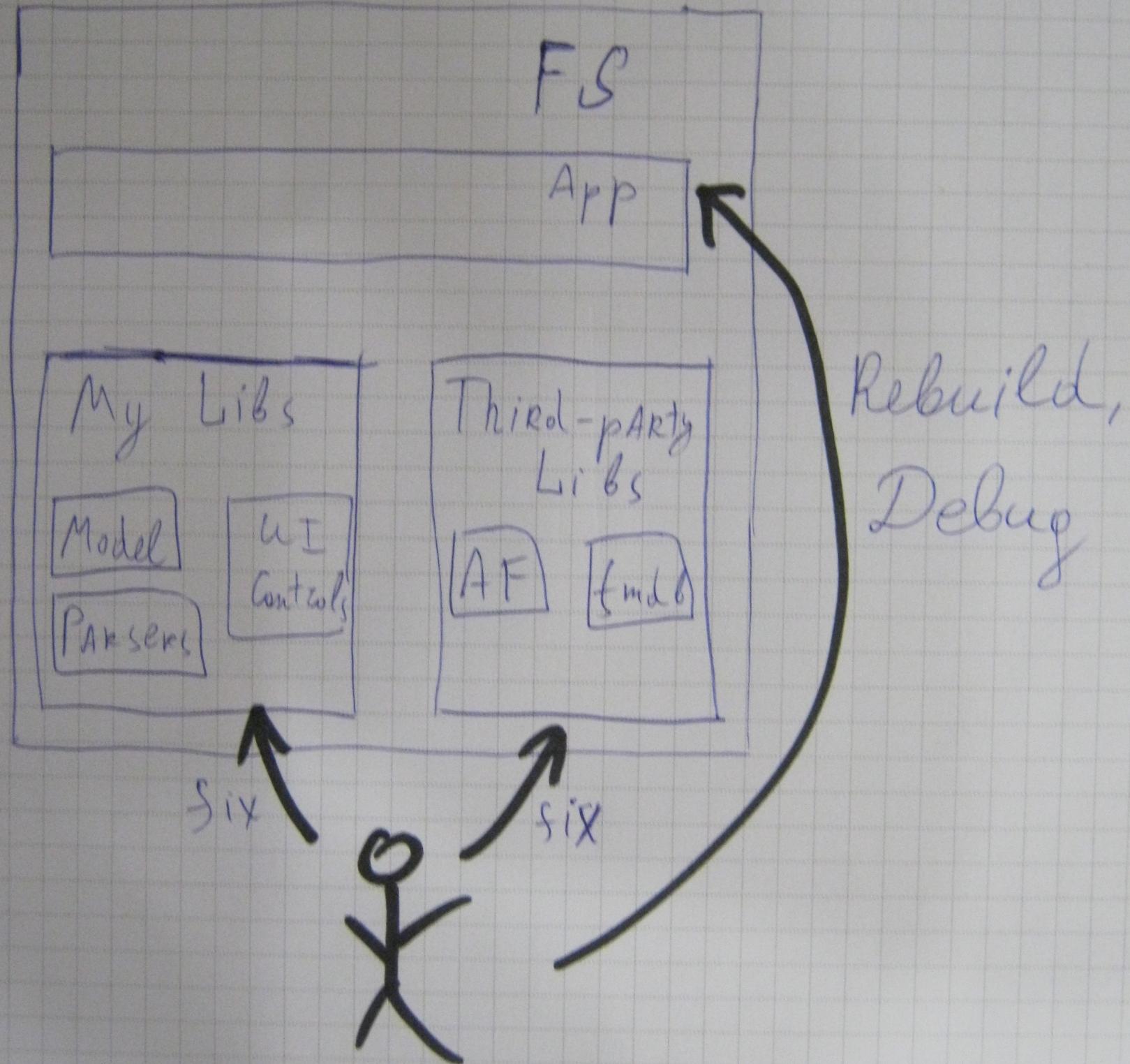


F S



Debug  
and  
Fix

# Going Native



The screenshot shows the Xcode interface with the project 'ListItemsBrowserDemo' selected. The top bar displays the project name and target 'iPhone Retina (4-inch)'. The toolbar below has various icons for file operations. The left sidebar lists project files:

- ListItemsBrowserDemo**: 3 targets, iOS SDK 7.0
- lib**:
  - SitecoreItemsBrowser.framework
  - SitecoreMobileSDK.framework
  - SCItemsBrowser.xcodeproj**: 3 targets, iOS SDK 7.0
- ListItemsBrowserDemo**
- ListItemsBrowserDemoTests**
- Frameworks**
- Products**
- ListItemsBrowserDemo copy-Info.plist**

The 'SCItemsBrowser.xcodeproj' item is highlighted with a red border.

The right side of the screen shows the code editor with partial code snippets:

```
#imp  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
#cla  
@cla  
@cla  
/**  
This  
The  
The
```

# Not Just Library Target

```
#import <AFNetworking/AFNetworking.h>
```

VS

```
#import "AFHTTPRequestOperation.h"
```

```
#import "AFHTTPRequestOperation.h"
```

▼ Search Paths

Setting	
Always Search User Paths	No ▾
Framework Search Paths	..../frameworks
Header Search Paths	/Applications/Xcode.app/Cont
Library Search Paths	..../frameworks
Rez Search Paths	
Sub-Directories to Exclude in Recursive Searches	*.nib *.lproj *.framework *.gch
Sub-Directories to Include in Recursive Searches	
User Header Search Paths	

```
#import <AFNetworking/AFNetworking.h>
```

## ▼ Search Paths

Setting



Always Search User Paths

No ↕

Framework Search Paths

..../frameworks

Header Search Paths

/Applications/Xcode.app/Cont

Library Search Paths

..../frameworks

Rez Search Paths

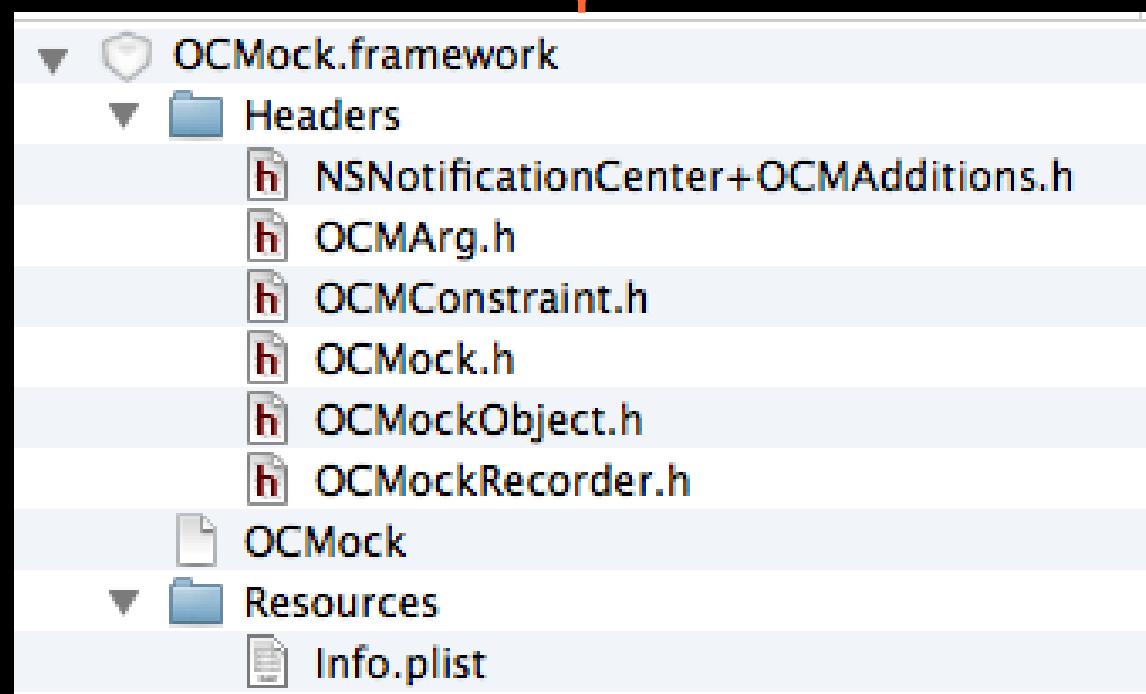
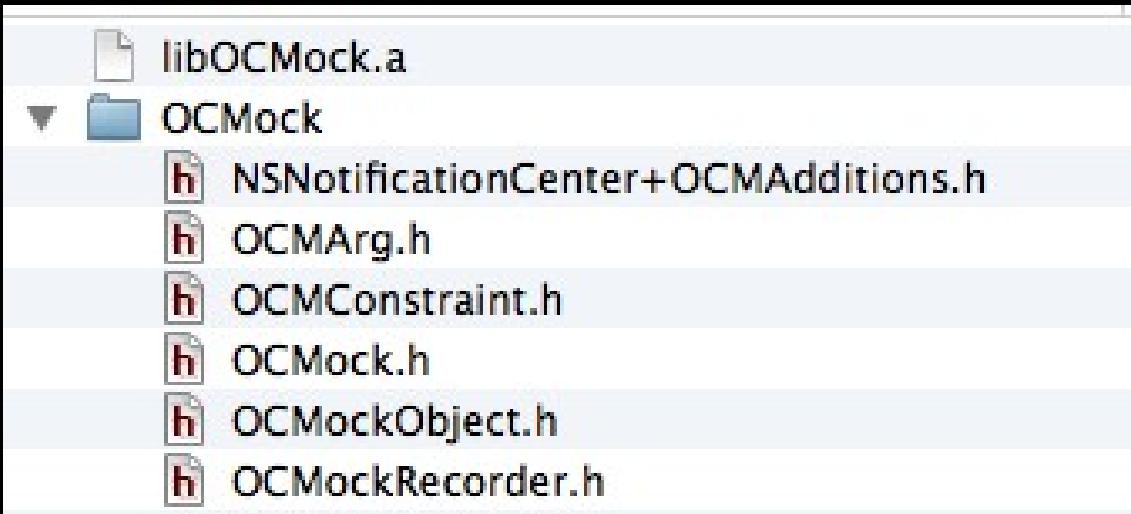
Sub-Directories to Exclude in Recursive Searches

\*.nib \*.lproj \*.framework \*.gch

Sub-Directories to Include in Recursive Searches

User Header Search Paths

# Static Framework for iOS



# Drag & Drop

Framework Search Path  
is updated by Xcode

```
<plist version="1.0">
<dict>
    <key>CFBundleDevelopmentRegion</key>
    <string>English</string>
    <key>CFBundleIdentifier</key>
    <string>org.ocmock</string>
    <key>CFBundleInfoDictionaryVersion</key>
    <string>6.0</string>
    <key>CFBundlePackageType</key>
    <string>FMWK</string>
    <key>CFBundleSignature</key>
    <string>????</string>
    <key>CFBundleVersion</key>
    <string>2.2.3</string>
</dict>
</plist>
```

One More Thing

```
// Pure C headers go here
#include <math.h>

#ifndef __cplusplus
    // Pure C++ headers go here
    #include <vector>
#endif

#ifndef __OBJC__
    // Objective-C headers go here
    #import <Foundation/Foundation.h>

#endif

#ifndef __cplusplus
    #import "MyObjectiveCppClass.h"
#endif

#endif
```

But I do not Need C++

# Print Message for Low Level Error

```
NSDictionary* errorMessages;  
NSLog( @"%@", errorMessages[ @( errorCode ) ] );
```

C++ ==> No Boxing

```
std::map< NSInteger, NSString* > errorMessages;  
NSLog( @"%@", errorMessages[ errorCode ] );
```

# Scoped Guard

```
void bad(const char* p)
{
    FILE* fh = fopen(p,"r"); // acquire

    // use f
    if ( someCondition )
    {
        // Oops! File handle leaks
        return;
    }

    fclose( fh ); // release
}
```

```
void good(const char* p)
{
    FILE* fh = fopen(p,"r"); // acquire

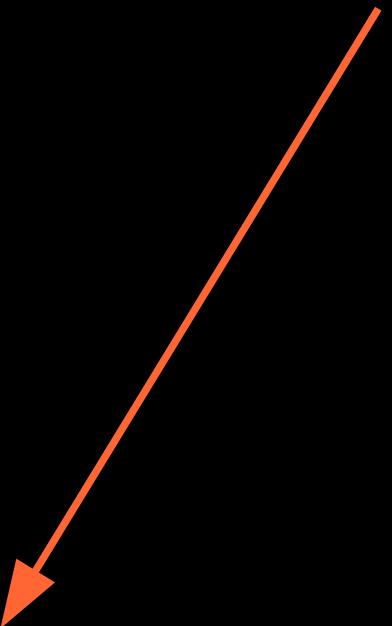
    // the block to perform cleanup actions
    GuardCallbackBlock releaseBlock_ = ^void( void )
    {
        fclose( fh );
    };

    // creating a guard
    ObjcScopedGuard guard( releaseBlock_ );

    if ( someCondition )
    {
        // Now the scoped guard will release the resource
        return;
    }
}
```

# Exception-Safe Resource Deallocation

malloc()



std::vector<unsigned char>

# Native Rulezzz

Alexander Dodatko

@dodikk88