

Debugging with Xcode

Dave Koziol
Arbormoon Software, Inc.
@DaveKoziol
www.arbormoon.com





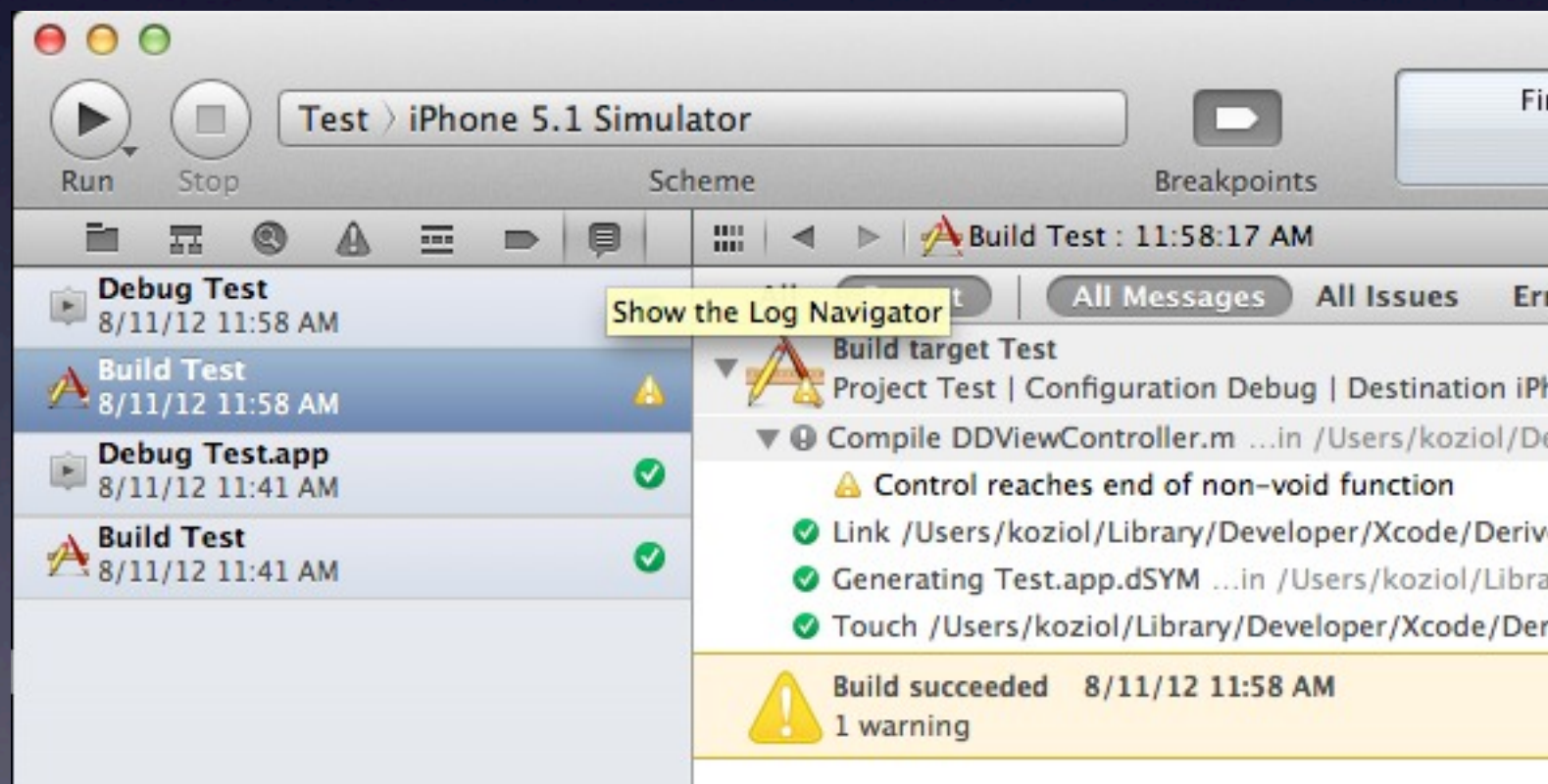
About Me

- Long time Apple Developer (21 WWDCs)
- Organizer Ann Arbor, MI CocoaHeads
- President & iOS Developer at Arbormoon Software Inc.
- Wunder Radio, KNBR, XanEdu, several many other smaller apps.



Getting Started

- Prevent bugs
- Warnings





Static Analyzer

- Prevent Bugs
- Product -> Analyze
- Log Navigator

The screenshot shows a static analyzer window titled "Example.m". The code being analyzed is a C function `foo` that takes two integers `x` and `y`. Inside the function, an Objective-C string object `obj` is allocated on line 13: `id obj = [[NSString alloc] init];`. A `switch` statement follows, with cases for `0` and `1`. In the `case 0` block, `[obj release];` is called, and `break;` is used. In the `case 1` block, there is a comment `// [obj autorelease];` followed by `break;`. The `default` case also contains `break;`. The function ends with a closing brace on line 24. A warning message is displayed at the bottom: "2. Object allocated on line 13 is no longer referenced after this point and has a retain count of +1 (object leaked)". A blue arrow points from the warning message to the allocation line 13. Another blue arrow points from the allocation line 13 to the `break;` statement in the `case 0` block, indicating that the object is released in that path. A third blue arrow points from the allocation line 13 to the closing brace of the `switch` statement, indicating that the object is not released in the `case 1` or `default` paths. A tooltip for the `switch` statement reads: "Method returns an Objective-C object with a +1 retain count (owning reference)".

```
10 }
11
12 void foo(int x, int y) {
13     id obj = [[NSString alloc] init];
14     switch (x) {
15         case 0:
16             [obj release];
17             break;
18         case 1:
19             // [obj autorelease];
20             break;
21         default:
22             break;
23     }
24 }
```

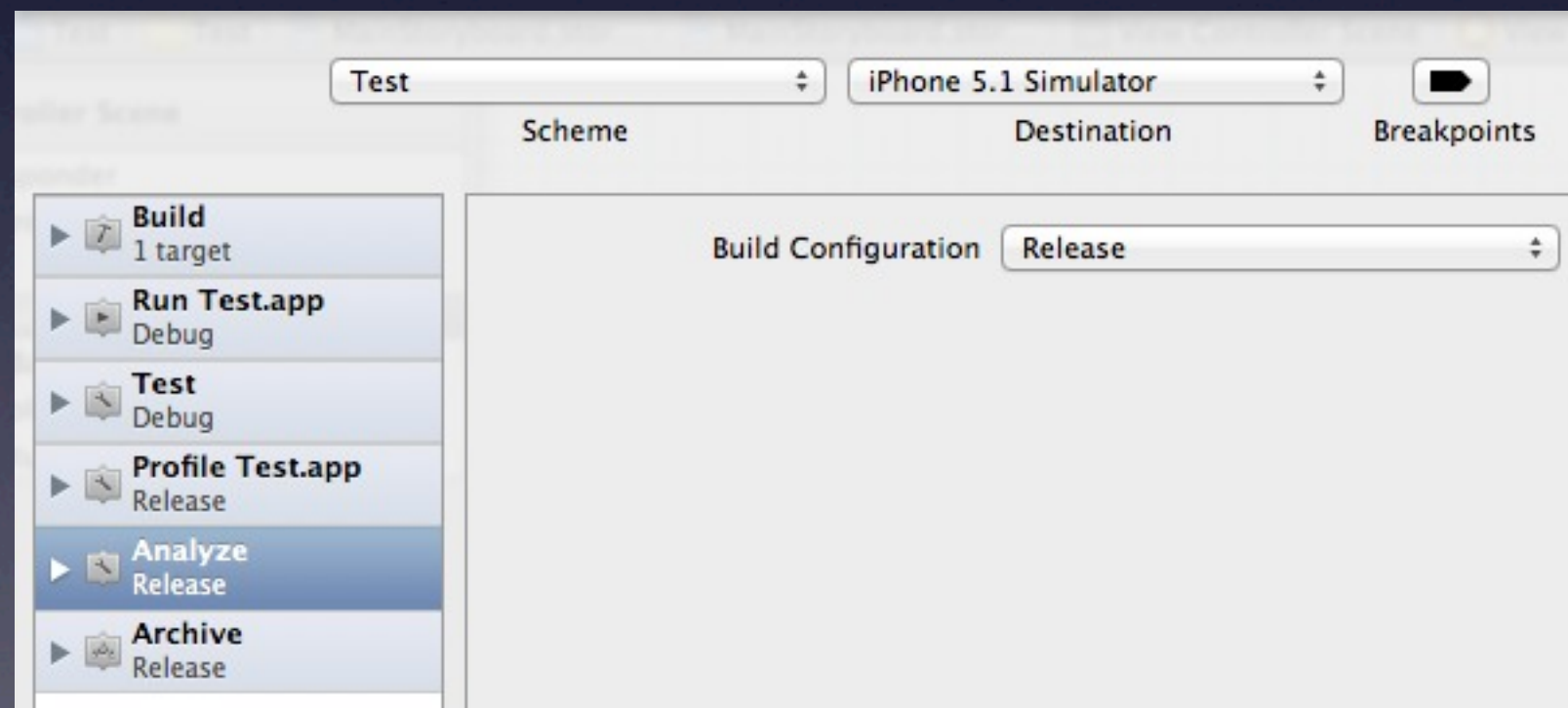
2. Object allocated on line 13 is no longer referenced after this point and has a retain count of +1 (object leaked)



Configurations

- Debug vs Release

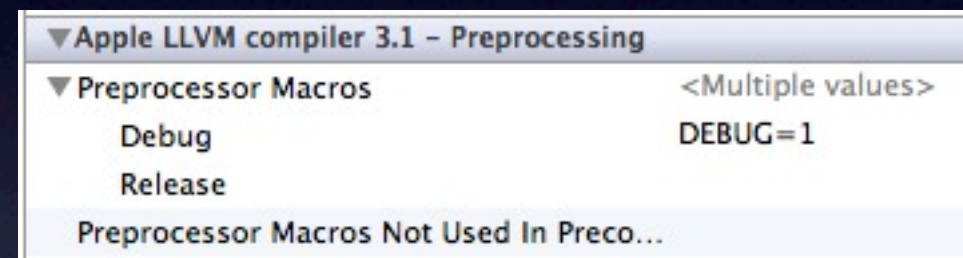
Edit Scheme, Info, Build Configuration



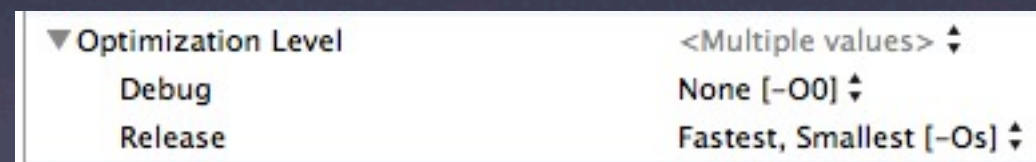


Build Settings

- Preprocessor Macros



- Optimization Level





NSLog

- printf
- Output to Console
- %@ calls Objective-C - (NSString) description method



NSLog 2

- Macros Help

```
NSLog((@"%s [Line %d] " fmt), __PRETTY_FUNCTION__, __LINE__,  
##__VA_ARGS__);
```

```
-[LibraryController awakeFromNib] [Line 364] Hello world
```




NSLog 2

- Macros Help

```
NSLog(@"%s [Line %d] " fmt), __PRETTY_FUNCTION__, __LINE__,  
##__VA_ARGS__);
```

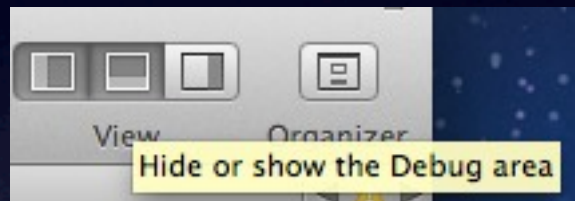
```
-[LibraryController awakeFromNib] [Line 364] Hello world
```

- Beware the watchdog:

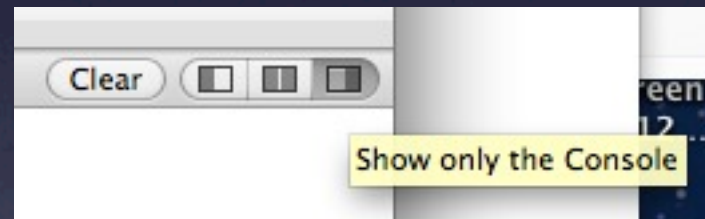
```
#ifdef DEBUG  
    #define DLog(fmt, ...) NSLog(@"%s [Line %d] " fmt),  
    __PRETTY_FUNCTION__, __LINE__, ##__VA_ARGS__);  
#else  
    #define DLog(...)   
#endif
```

Console

- Show/Hide Debug Area



- Only Console or Variable View and Console



- Organizer, Devices, Console



- /Applications/Utilities/Console.app

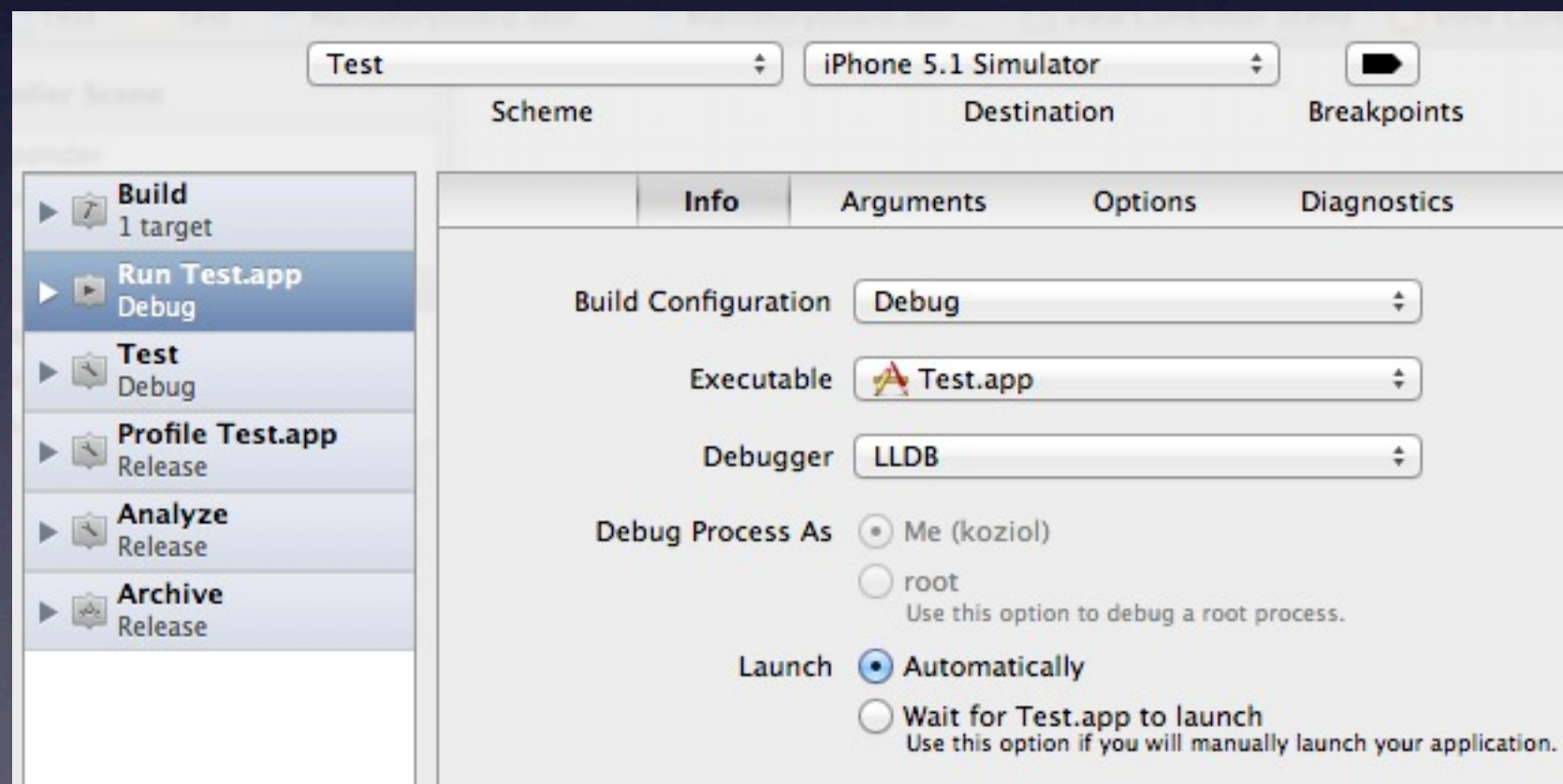




Which Debugger?

- LLDB vs GDB

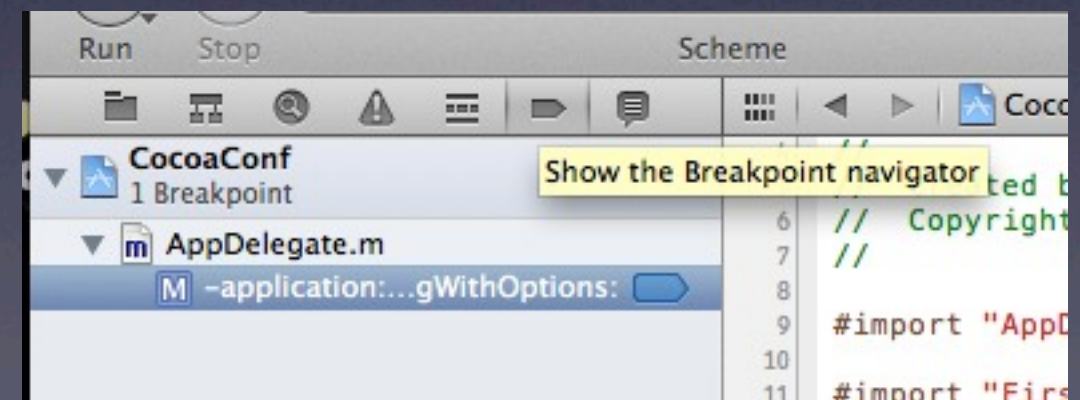
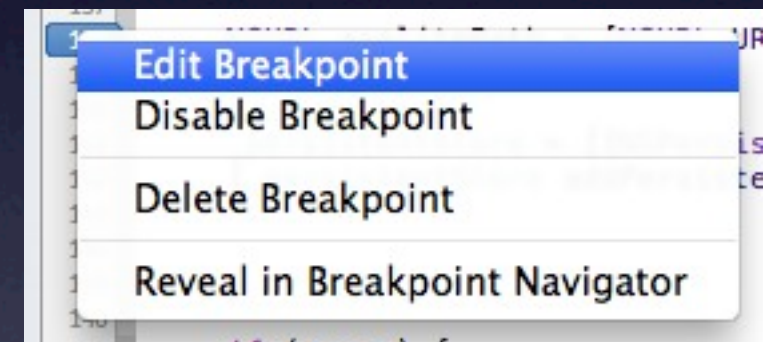
Edit Scheme, Info, Debugger



Breakpoints

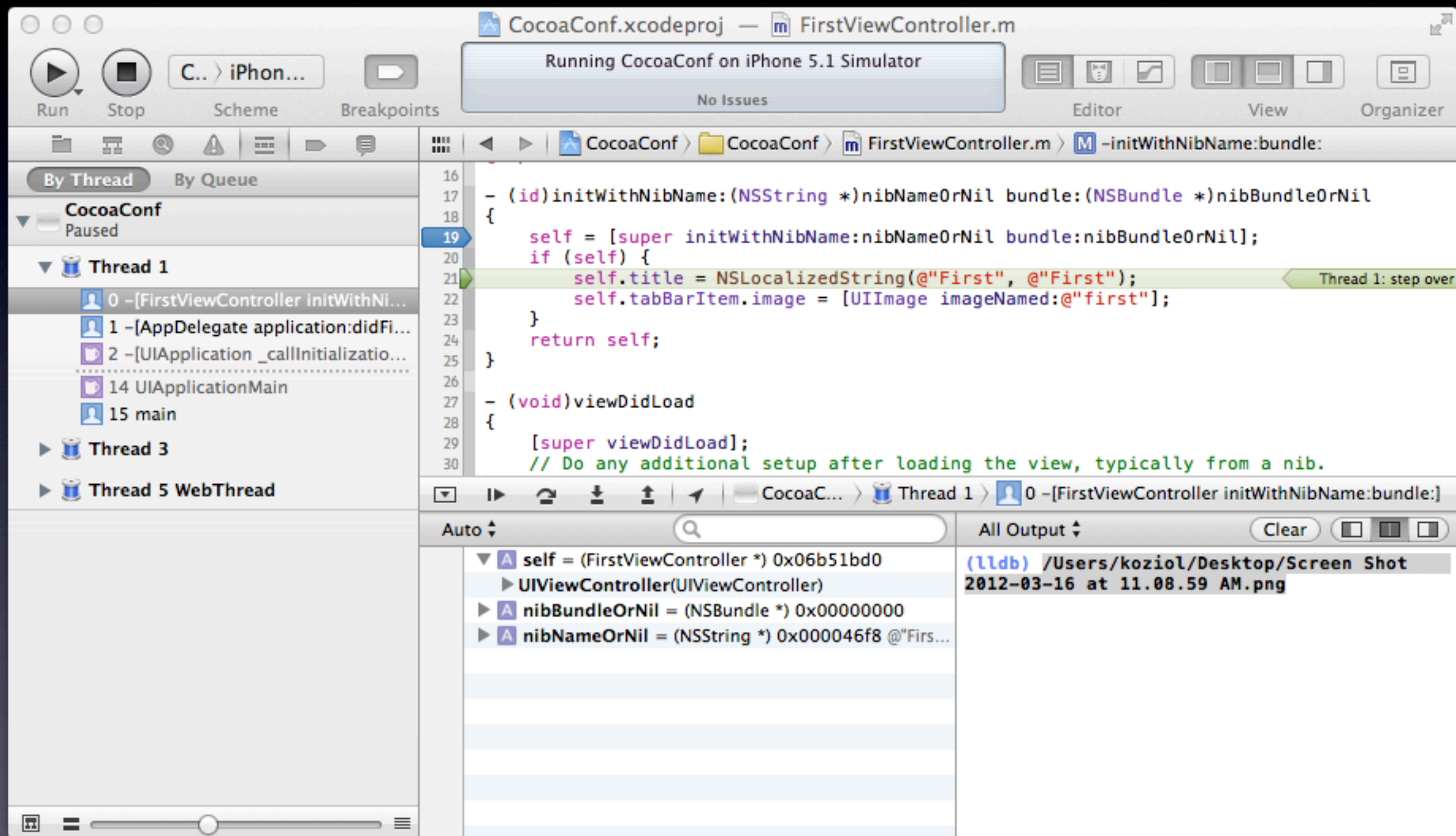
- At line numbers
- Edit - Right Click or Drag
- Breakpoints Navigator

```
134 - (NSPersistentStoreCoordinator*)pe
135     if (_persistentStore)
136         return _persistentStore;
137
138     NSURL *sqlitePath = [NSURL URLW
139     NSError *error = nil;
140
141     _persistentStore = [[NSPersiste
142     [_persistentStore addPersisten
```



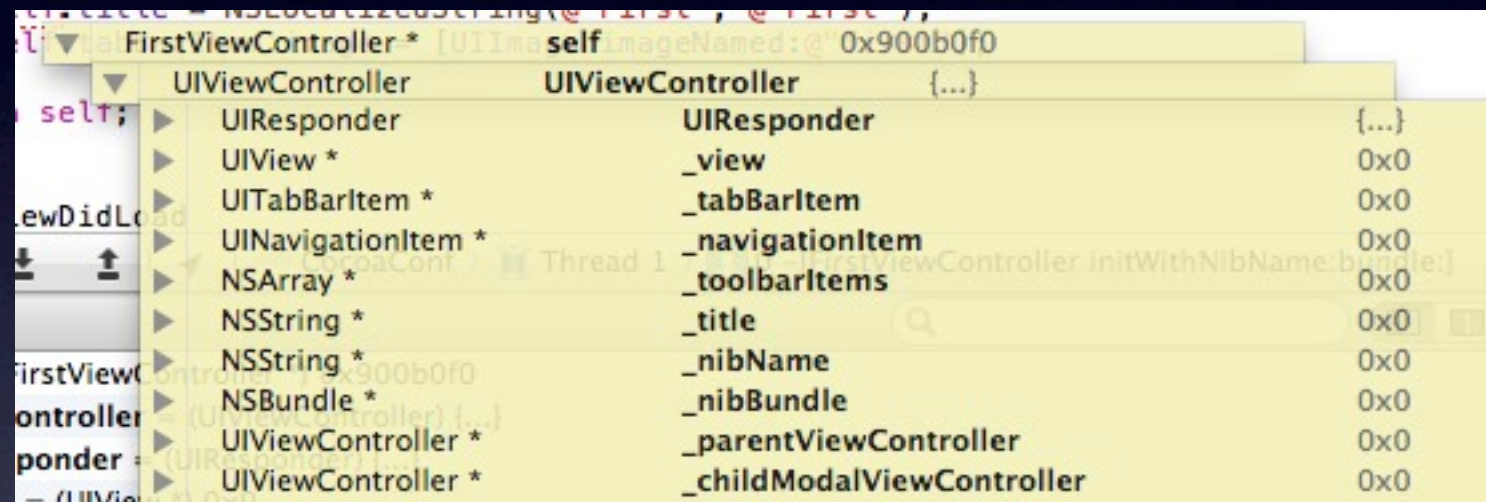


Debug Navigator

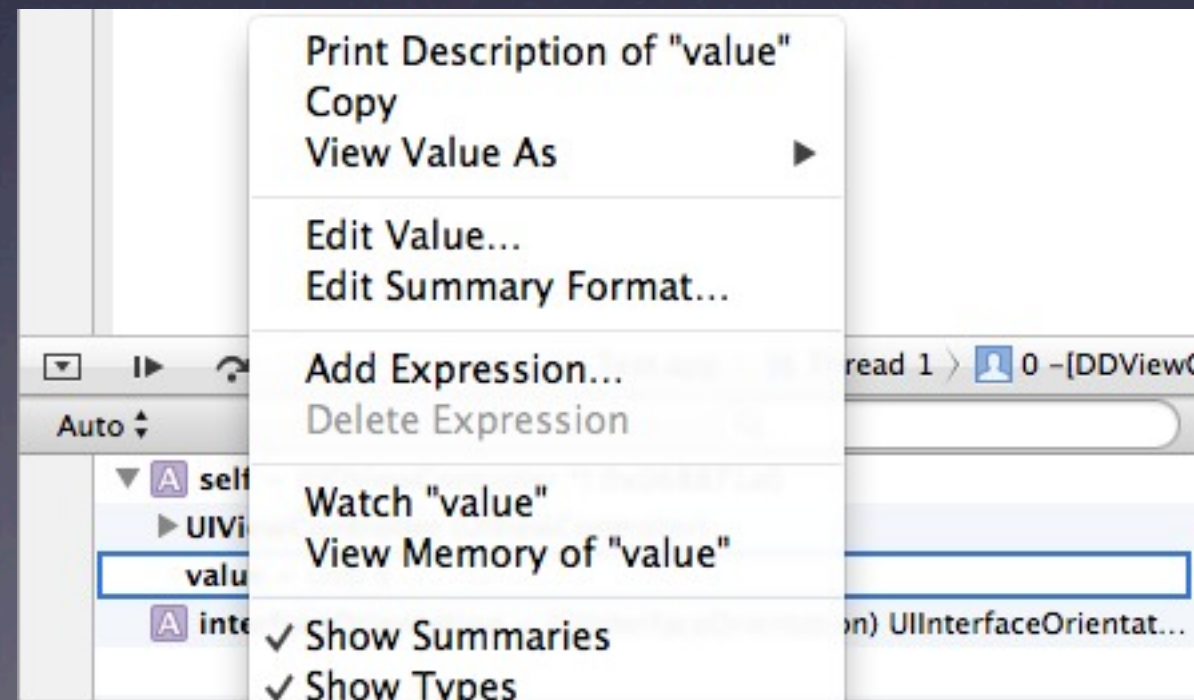


Variables

- Hover over in editor



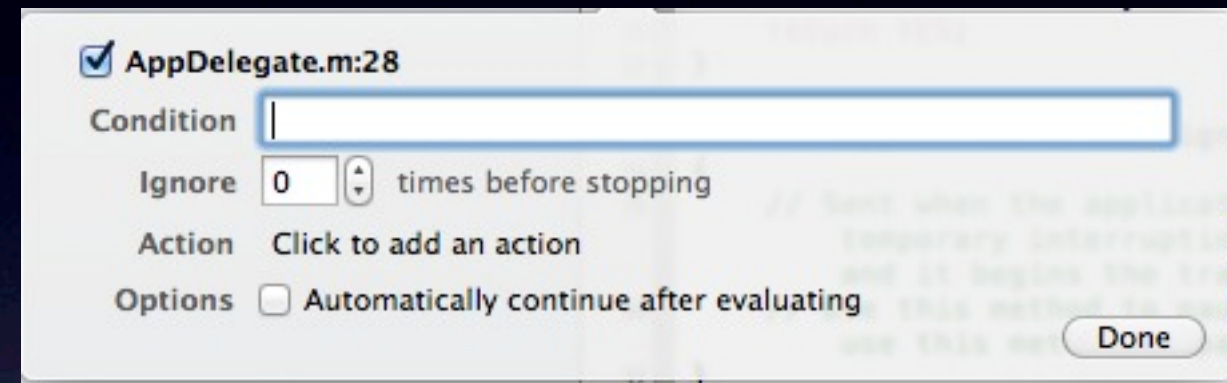
- View Value As
- View Memory
- Watch





Conditional Breakpoints

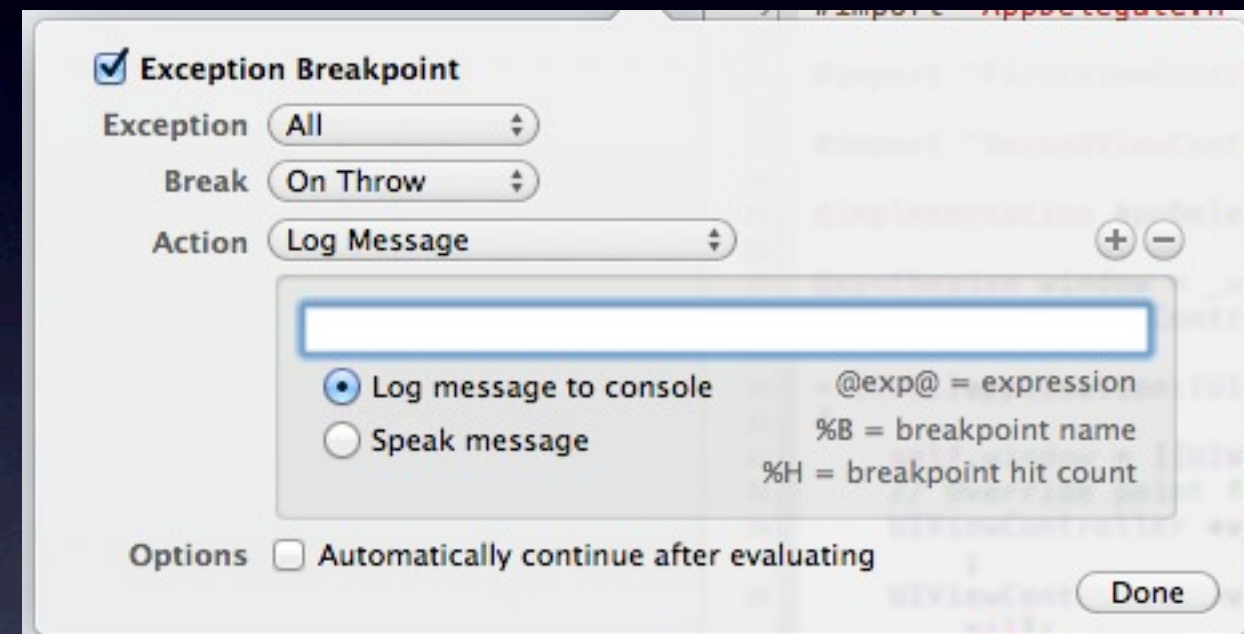
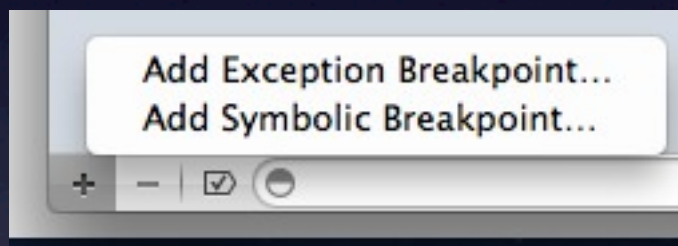
- Ignore multiple times
- Log without stopping
- Break on expression





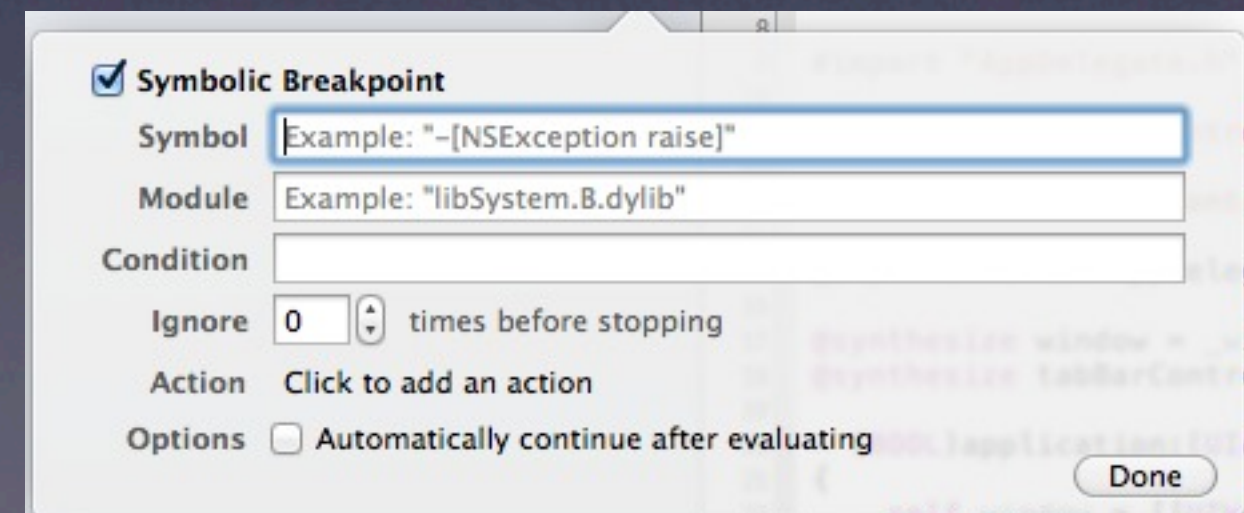
More Breakpoints

- Exception Breakpoint



- Symbolic Breakpoint

`-[NSObject doesNotRecognizeSelector:]`



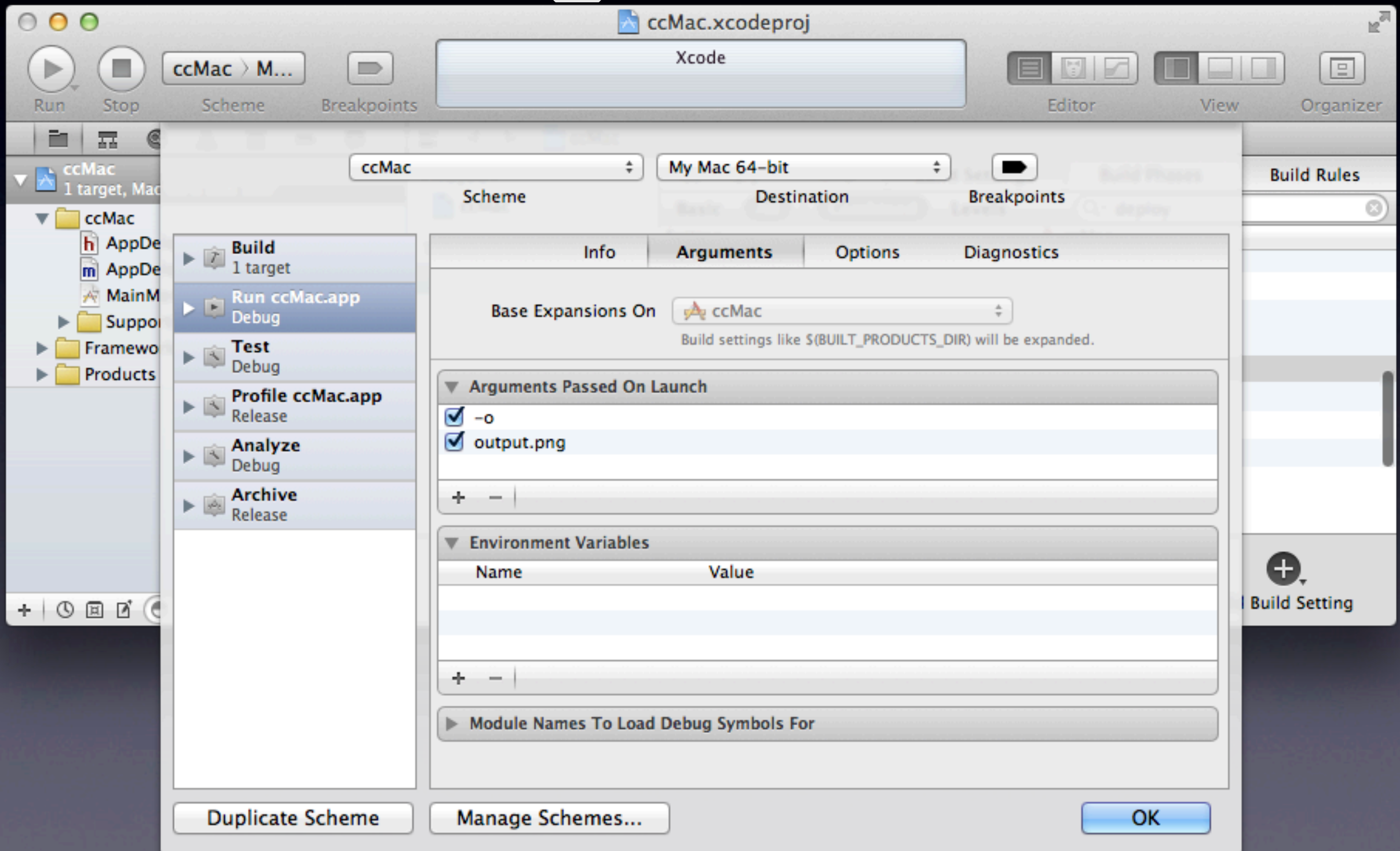


Debugger Console

- `po {object}` - calls description
- method calls
- `p (CGRect)[[self view] frame]`

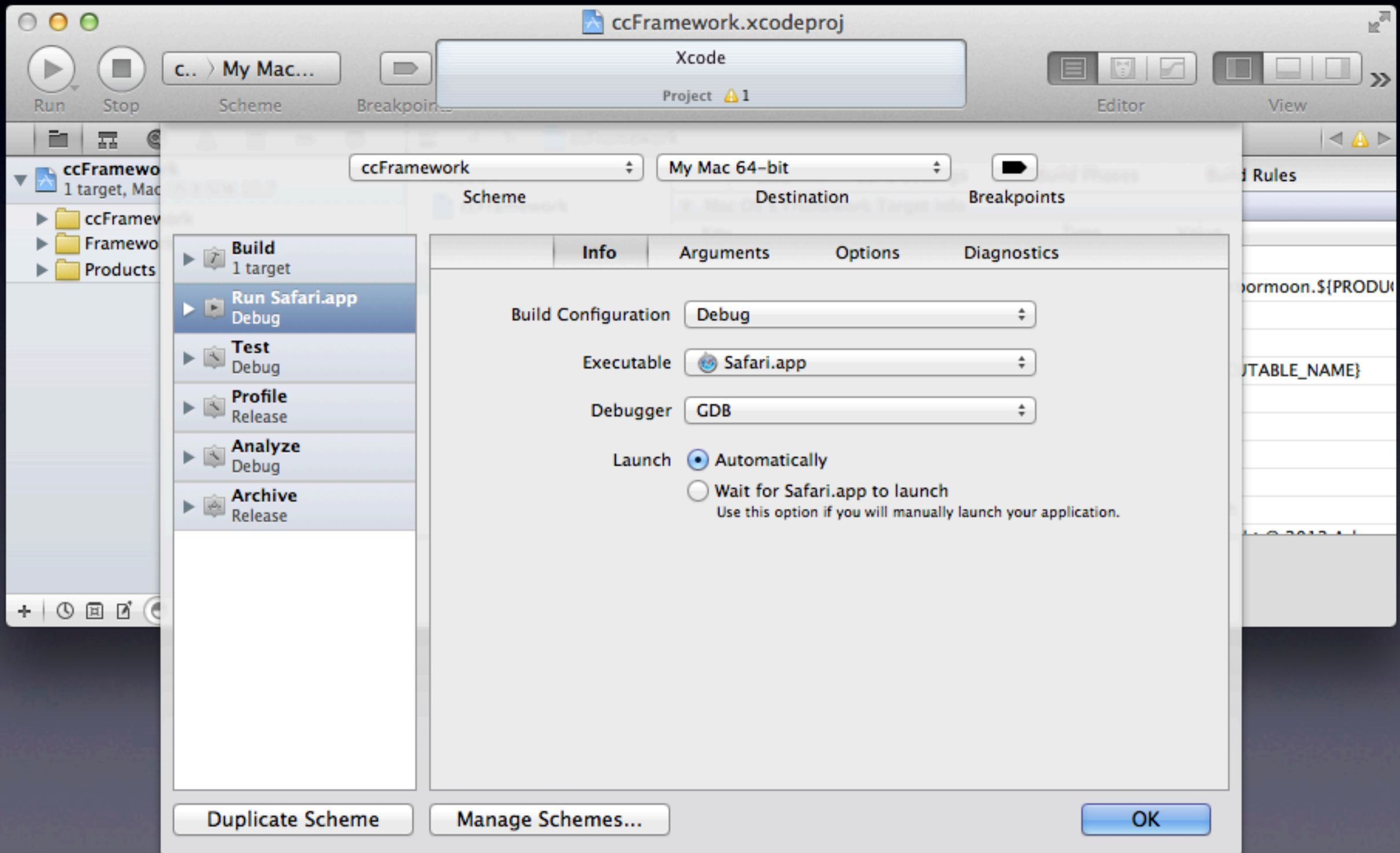


Command Line Arguments





Framework/Plugin Debugging





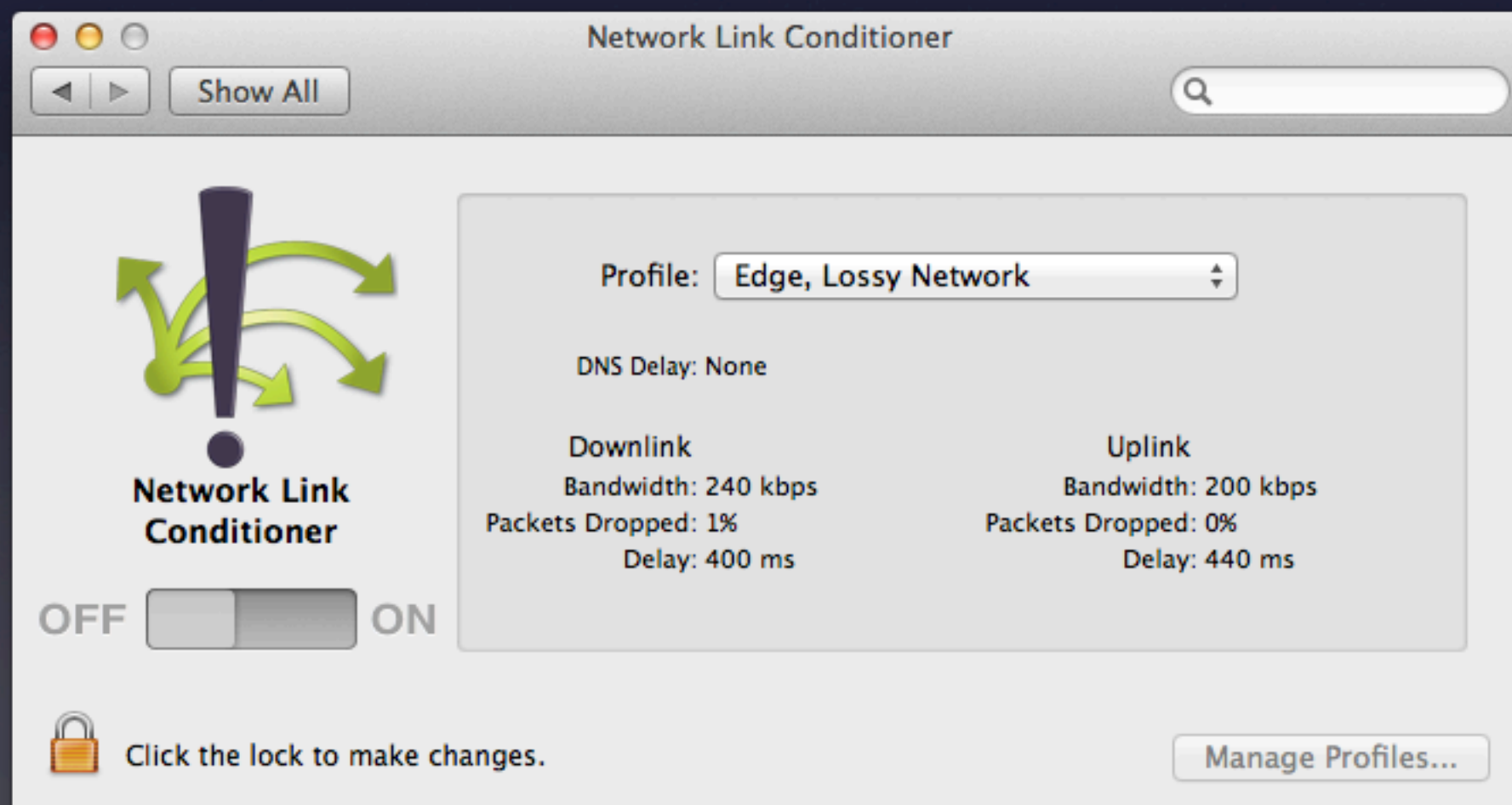
Location Debugging

- Product, Debug, Simulate Location
- GPX Files
- Edit Scheme, Options, Default Location



Network Link Conditioner

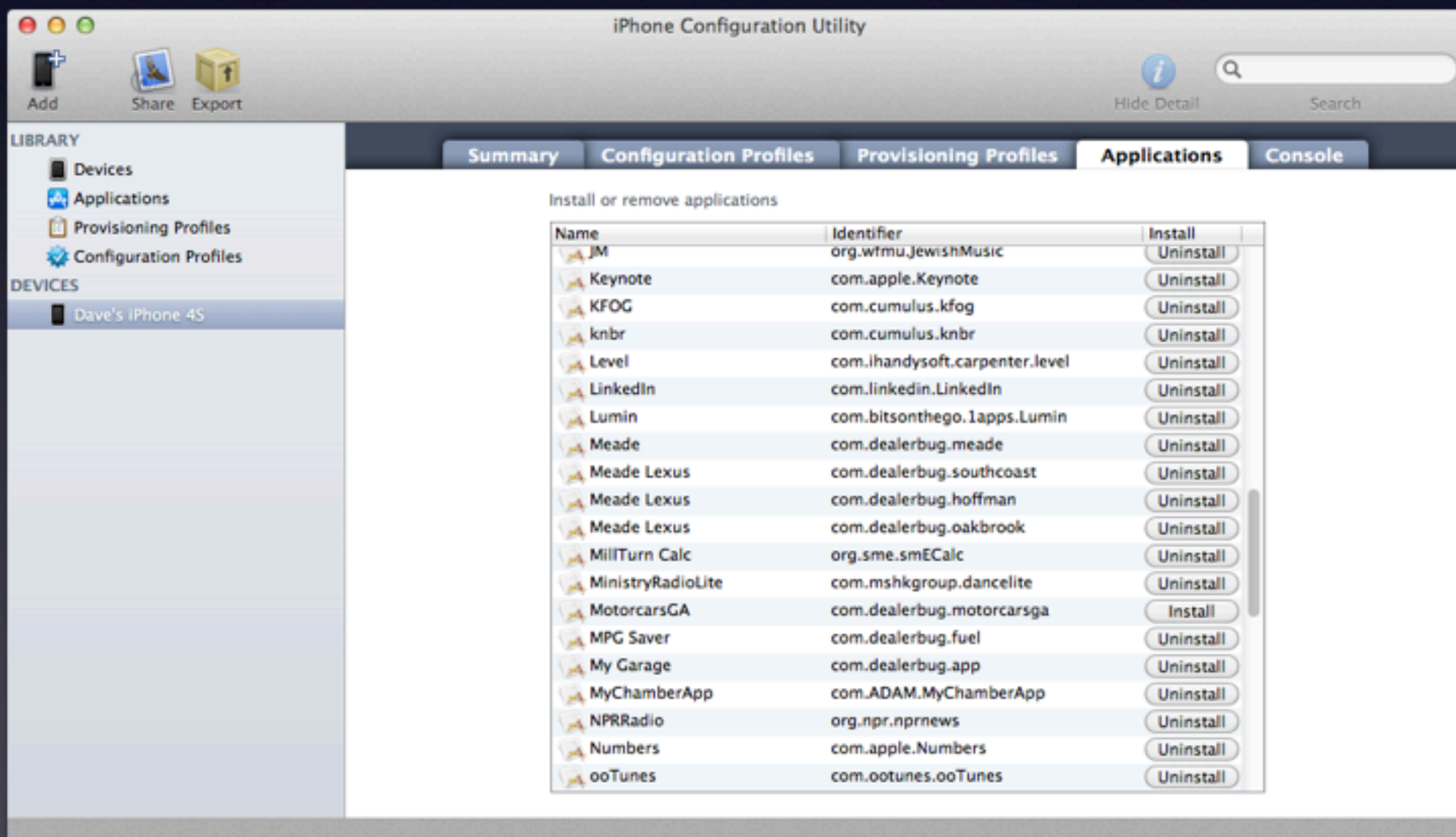
- Xcode, Open Developer Tool, More Developer Tools...
- Download Hardware IO Tools for Xcode





iPhone Configuration Utility

- http://support.apple.com/kb/DLI465?viewlocale=en_US&locale=en_US





Finding Crash Reports iOS

- Organizer, Devices, Device Logs
- `~/Library/Logs/CrashReporter/
MobileDevice/{DeviceName}/`
- iTunes Connect, Manage Your Applications,
{Application}, View Details, Crash Reports,
Refresh Now



Finding Crash Reports

Mac

- Console.app, User Diagnostic Reports and System Diagnostic Reports
- ~/Library/Logs/DiagnosticReports/ or /Library/Logs/DiagnosticReports
- iTunes Connect, Manage Your Applications, {Application}, View Details, Crash Reports, Refresh Now ??



Anatomy of a Crash Report

- TN2123
- Symbolication

Archive is your friend

Drag the crash report to the organizer!



Profile (aka Instruments)

- Simulator:

Leaks, Allocations, Zombies, Time Profiler, Automation, File Activity, Activity Monitor, System Trace, Core Data

- Device:

Energy Diagnostics, Allocations, Leaks, Activity Monitor, Time Profiler, System Trace, Automation



Q&A