Advanced Xcode
Configuration, Targets, and Schemes

Ashton Williams
Odecee
@AshtonDev

AUC

In Partnership With
RMIT UNIVERSITY

Advanced Xcode

Configurations, Targets, and Schemes

Targets

A target combines a set of input files, instructions in the form of build settings and build phases, to create a single product.

Projects can contain one or more targets.

Targets can depend on other targets. eg: Unit Tests rely on your app or framework.

YourApp
UnitTests
UITests
WatchApp
X Extension

Targets

Build Phases

Compile Sources
Link Libraries
Copy Resources

Targets

Build Phases

Build Settings

Basically the options for build phases. A bunch of variables used when building a target.

You can specify build settings at the project or target level. Each project-level build setting applies to all targets in the project unless explicitly overridden by the build settings for a specific target.

A target inherits the project build settings, but you can override any of the project settings by specifying different settings at the target level.

Configurations

A set of Build Settings

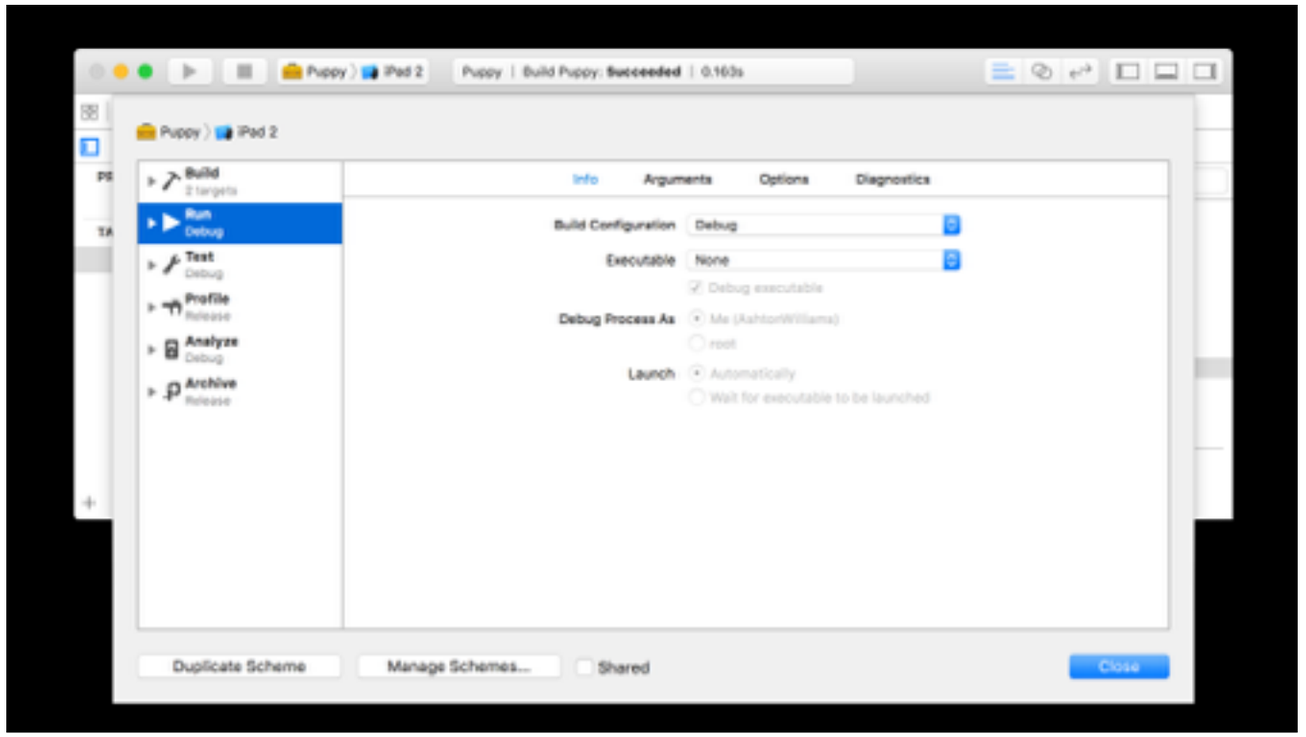
Debug
Release

Schemes

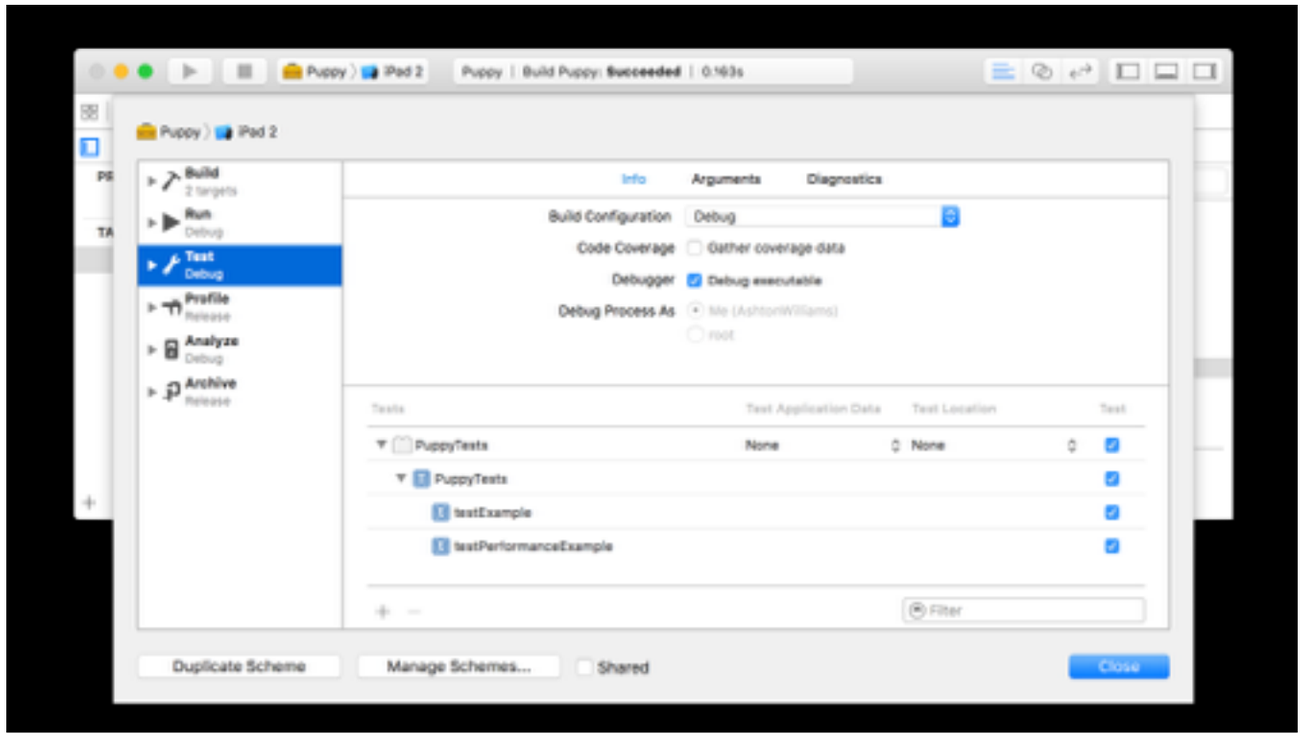
There can be only one active target at a time; the Xcode scheme specifies the active target.

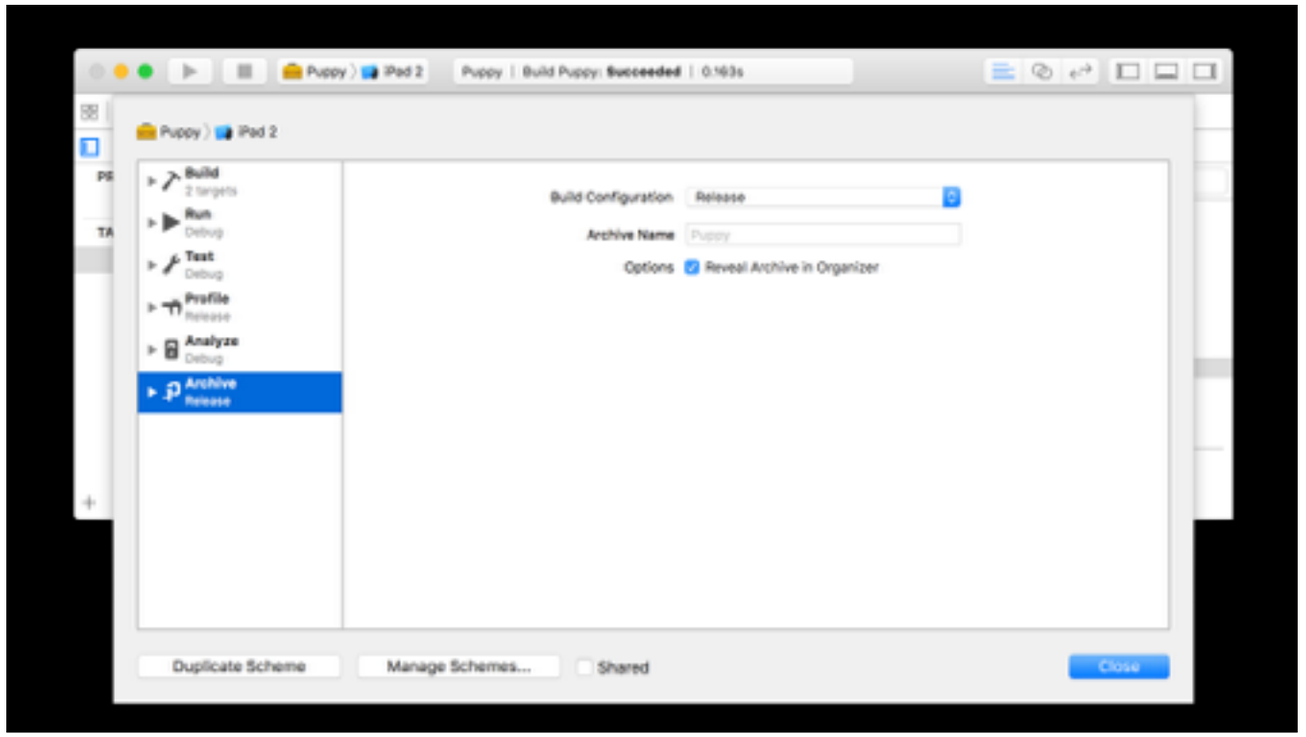
Scheme = Target + Build Configuration + Executable Environment

Run Environment,
Test Environment,
Analyse Environment, ...

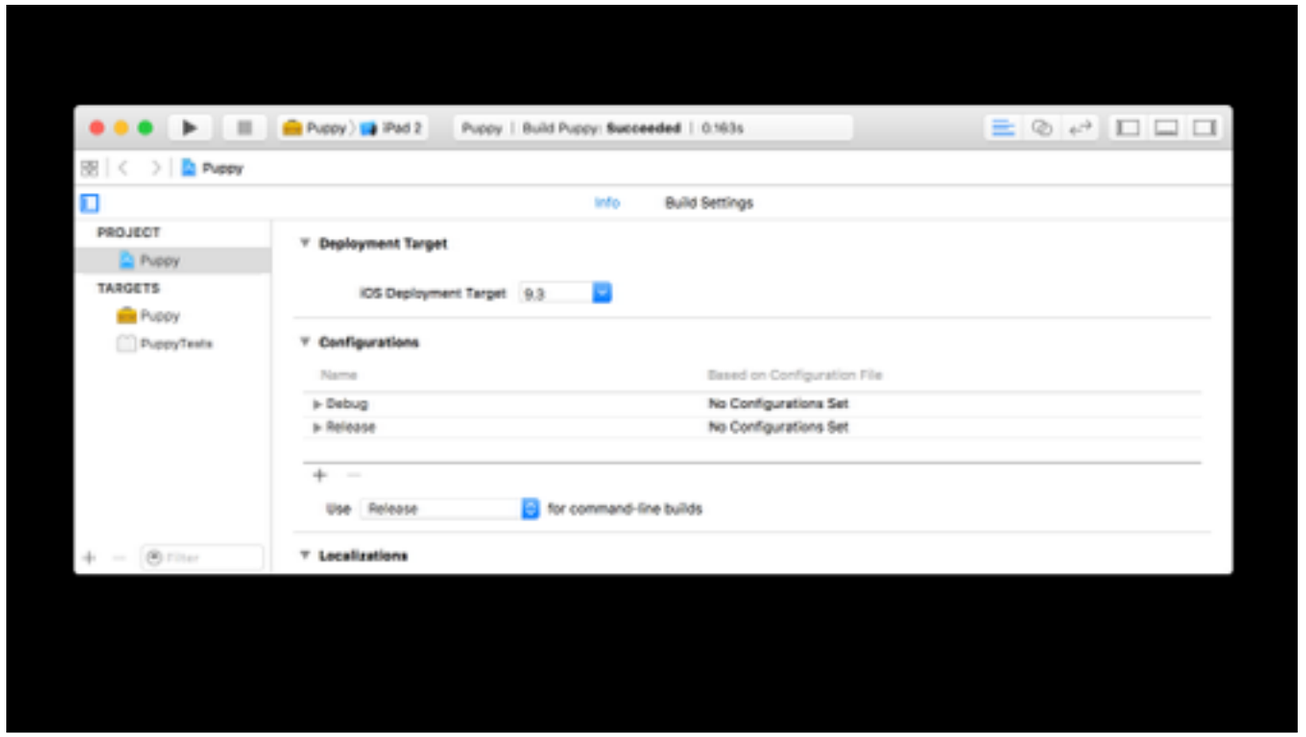


Scheme Actions


Puppy | iPad 2
Puppy | Build Puppy: Succeeded | 0.163s
Puppy 〉 iPad 2
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Info
Arguments
Diagnostics
Build Configuration
Debug
Code Coverage
Gather coverage data
Debugger
Debug executable
Debug Process As
Me (AshtonWilliams)
root
Tests
Test Application Data
Test Location
Test
PuppyTests
None
None
PuppyTests
testExample
testPerformanceExample
Filter
Duplicate Scheme
Manage Schemes...
Shared
Close

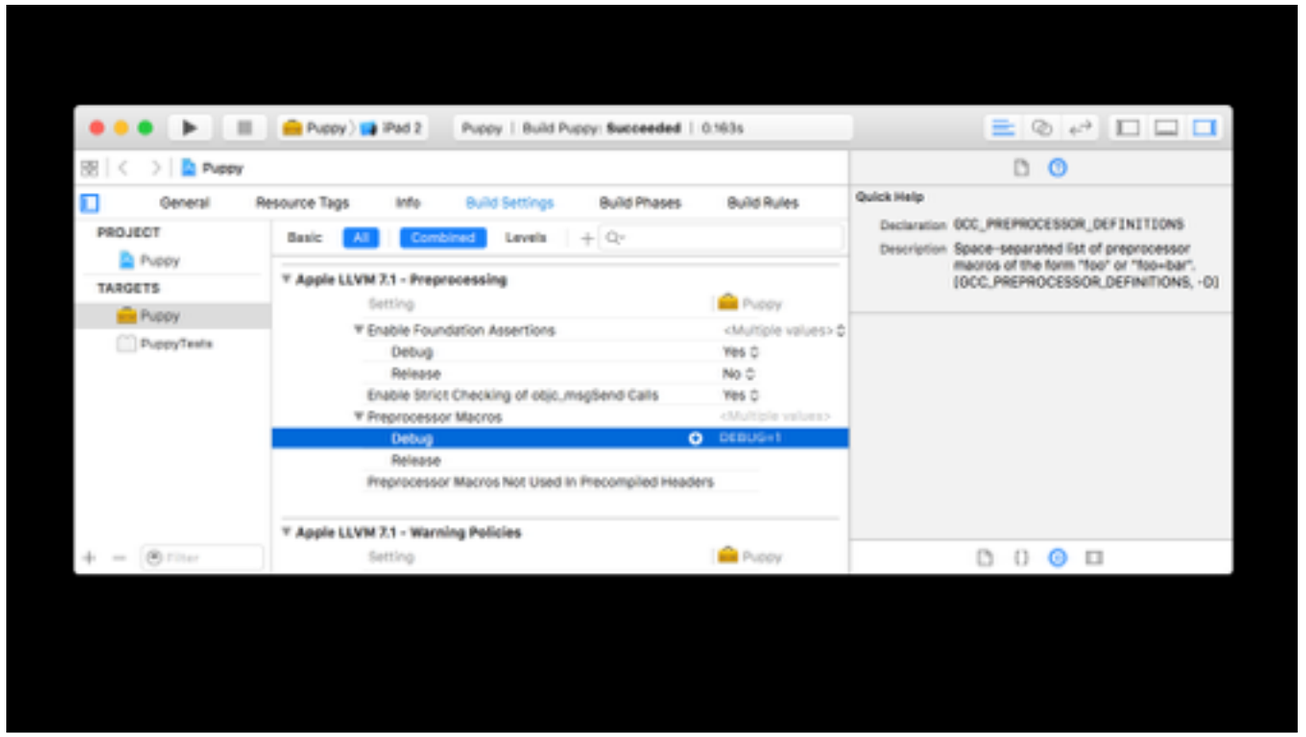

Puppy
iPad 2
Puppy | Build Puppy: Succeeded | 0.163s
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Build Configuration
Release
Archive Name
Puppy
Options
Reveal Archive in Organizer
Duplicate Scheme
Manage Schemes...
Shared
Close

For Visual People



Here we can see a project with:
- two targets
- two configurations
- a scheme

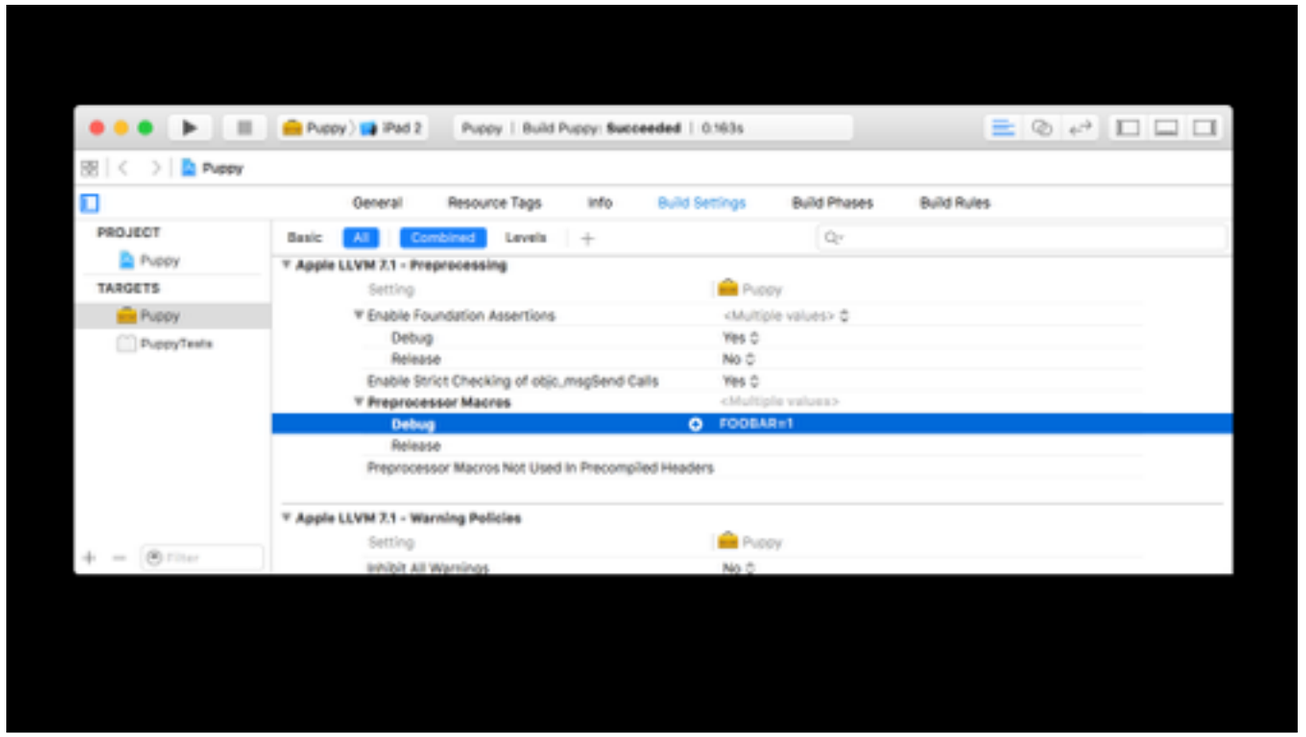
Xcode gives you this in most project templates.



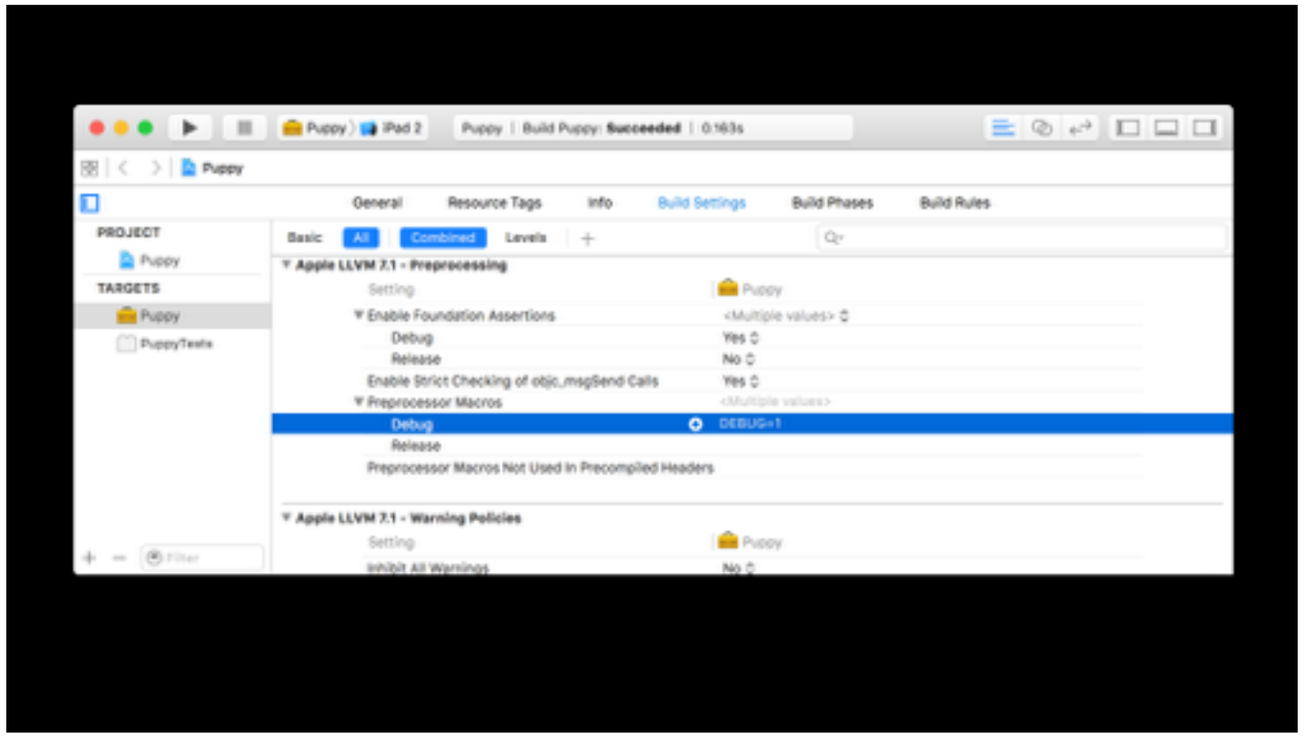
The Quick Help inspector will tell you the identifier and a description of a build setting.

You want to Google the identifier to find more information, the other name is general.

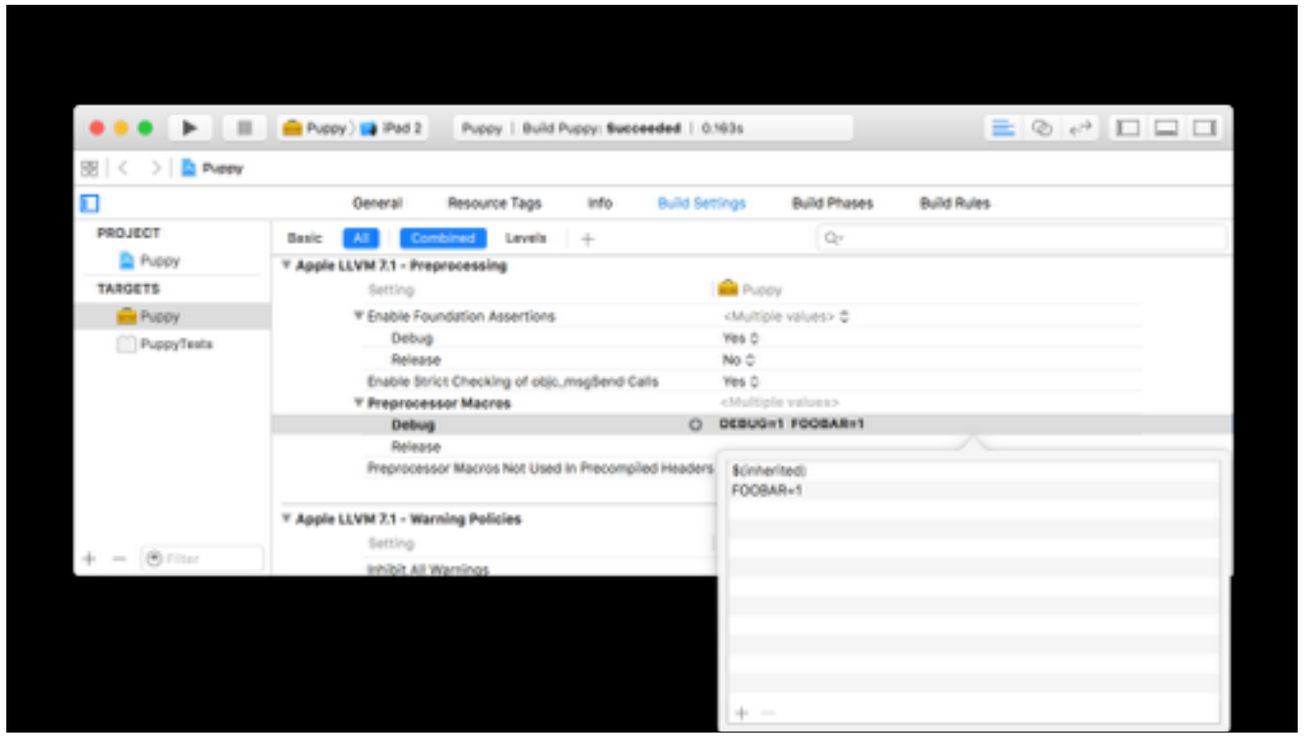
We can also see here that this Target has a Build Setting different for each configuration.



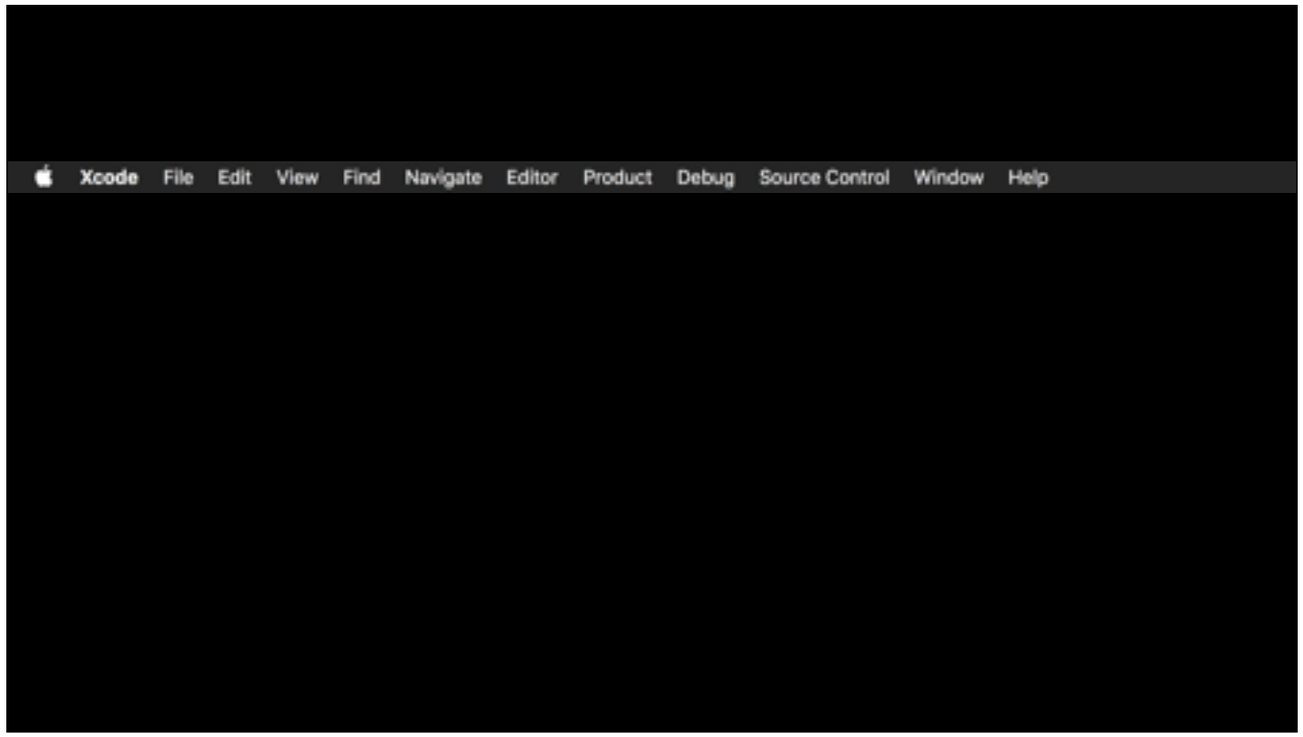
A build setting with BOLD FONT means it has been SET in this target. You can select it and press delete to get the PROJECT level value back.



A build setting with BOLD FONT means it has been SET in this target. You can select it and press delete to get the PROJECT level value back.

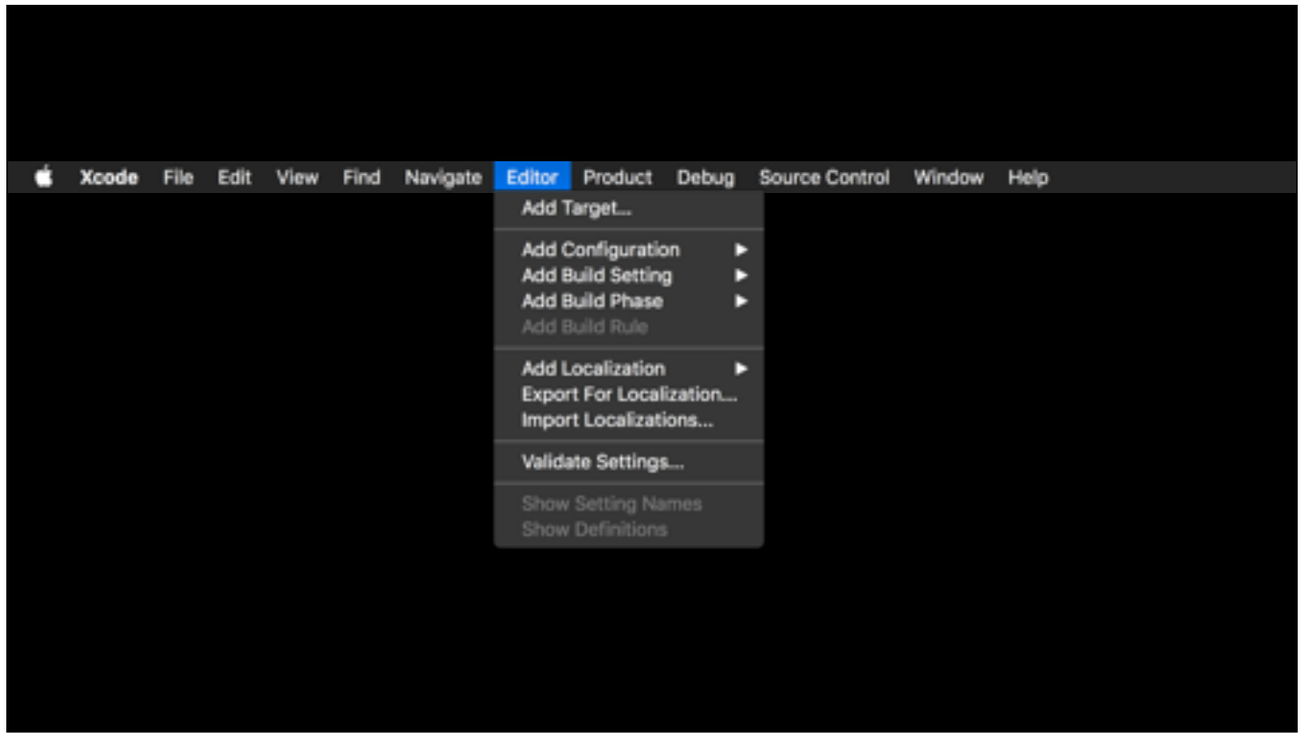


For multivalve build settings, a better option is to use the expression $(inherited)



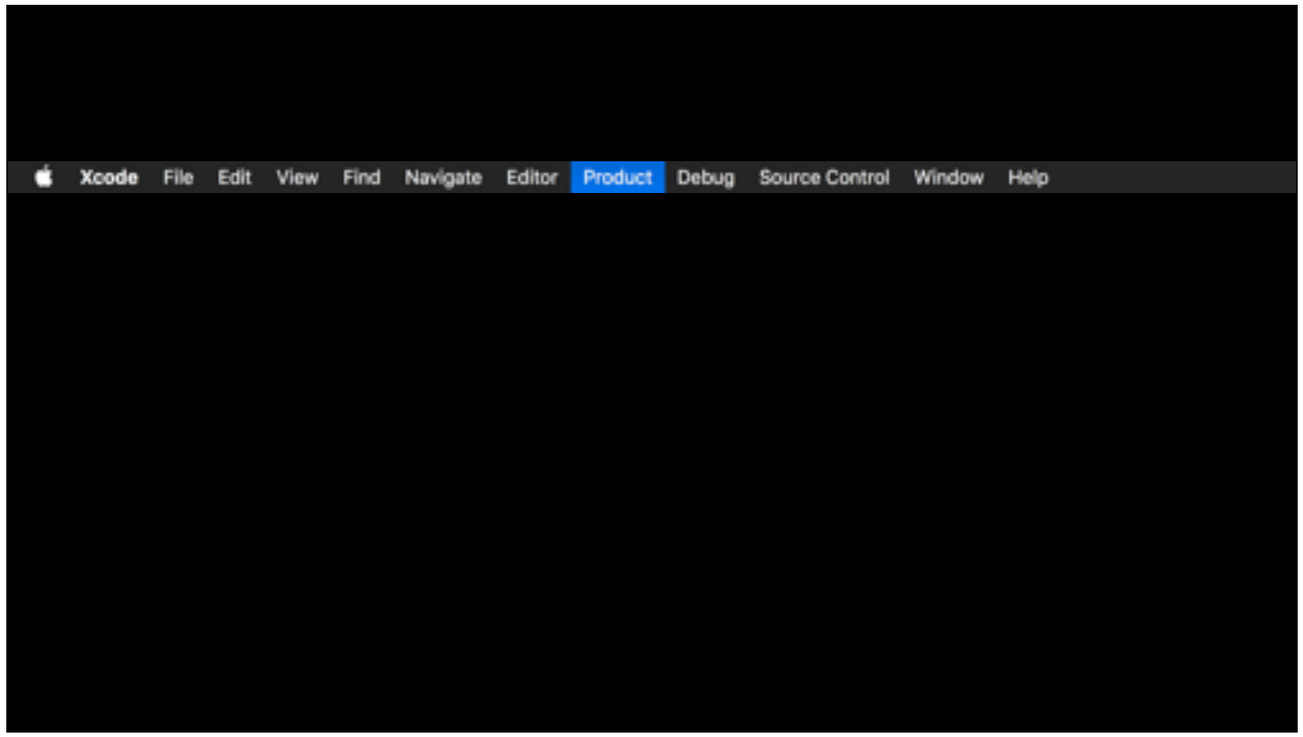
You can create of all of these things from the Xcode menu bar too.

Targets, Configurations, Build Settings, and Build Phases are under the Editor menu.



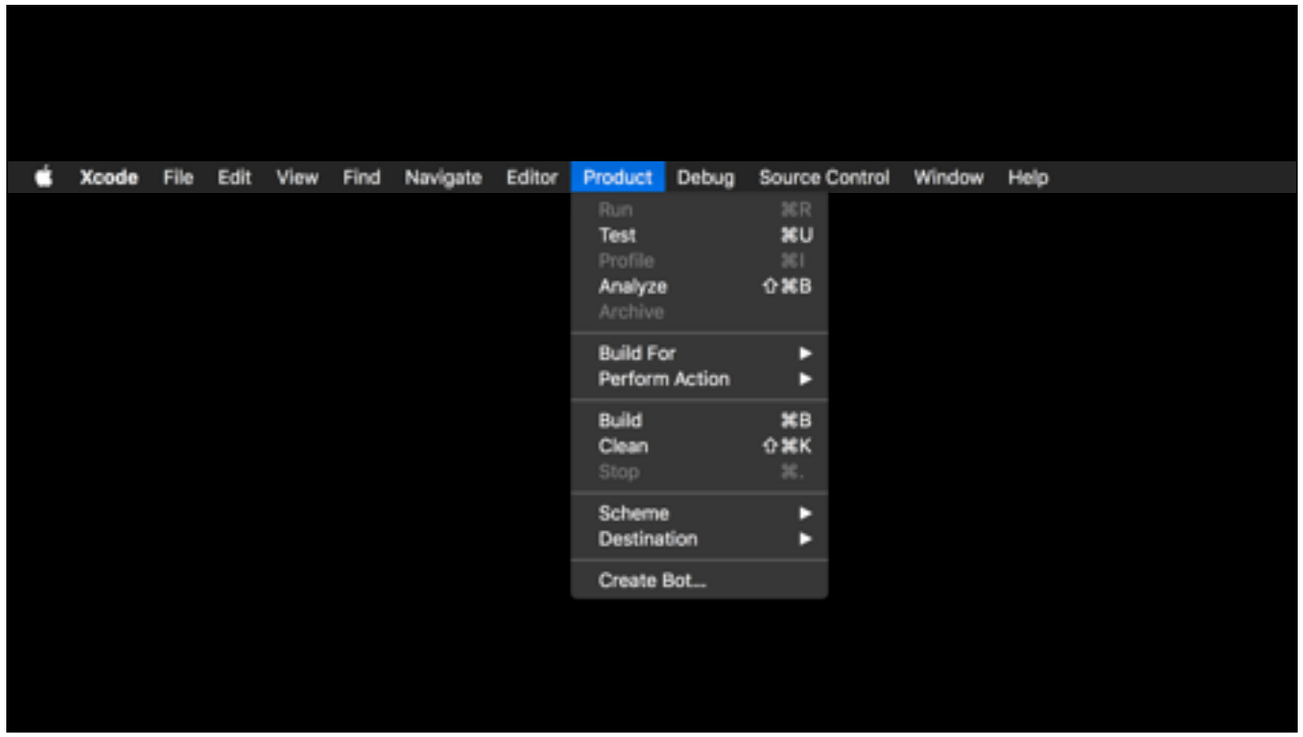
You can create of all of these things from the Xcode menu bar too.

Targets, Configurations, Build Settings, and Build Phases are under the Editor menu.



Schemes are under Product at the bottom here

Destinations are architectures to build for.
Simulators, devices, or generic devices. Some build settings are affected by this.

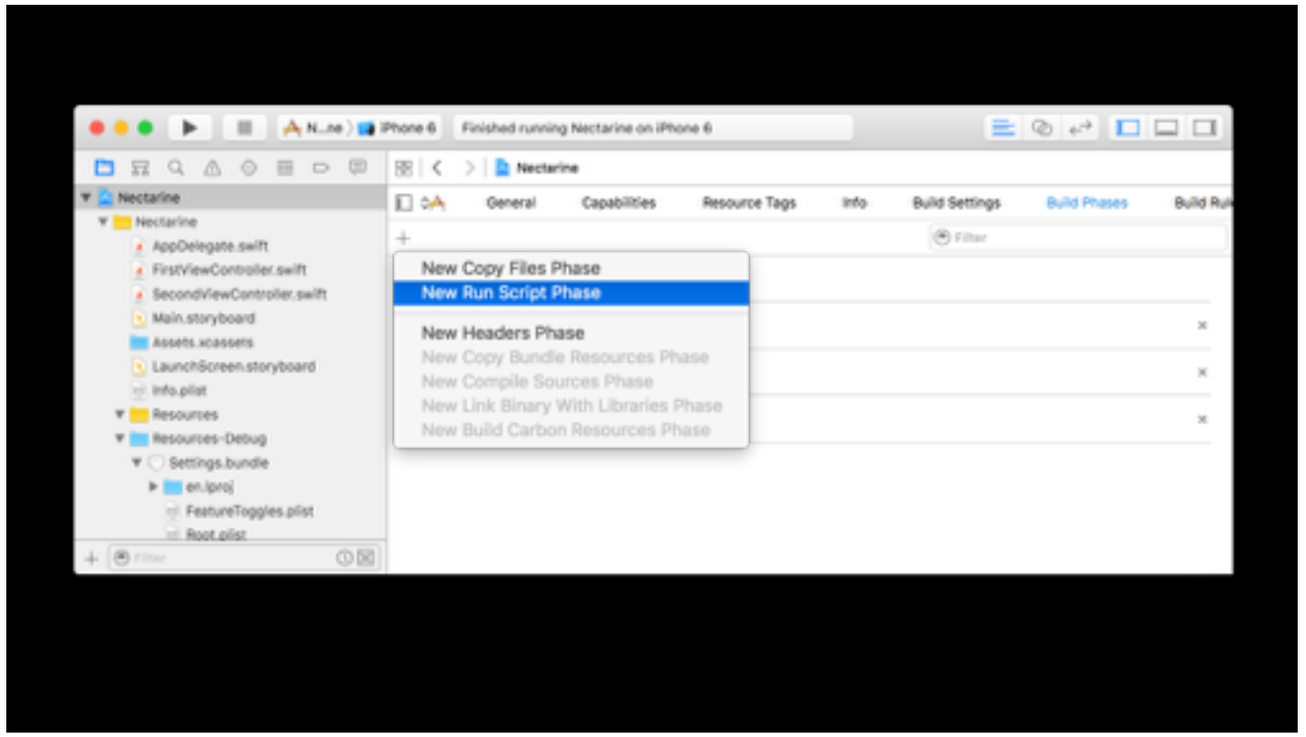


Schemes are under Product at the bottom here

Destinations are architectures to build for.
Simulators, devices, or generic devices. Some build settings are affected by this.

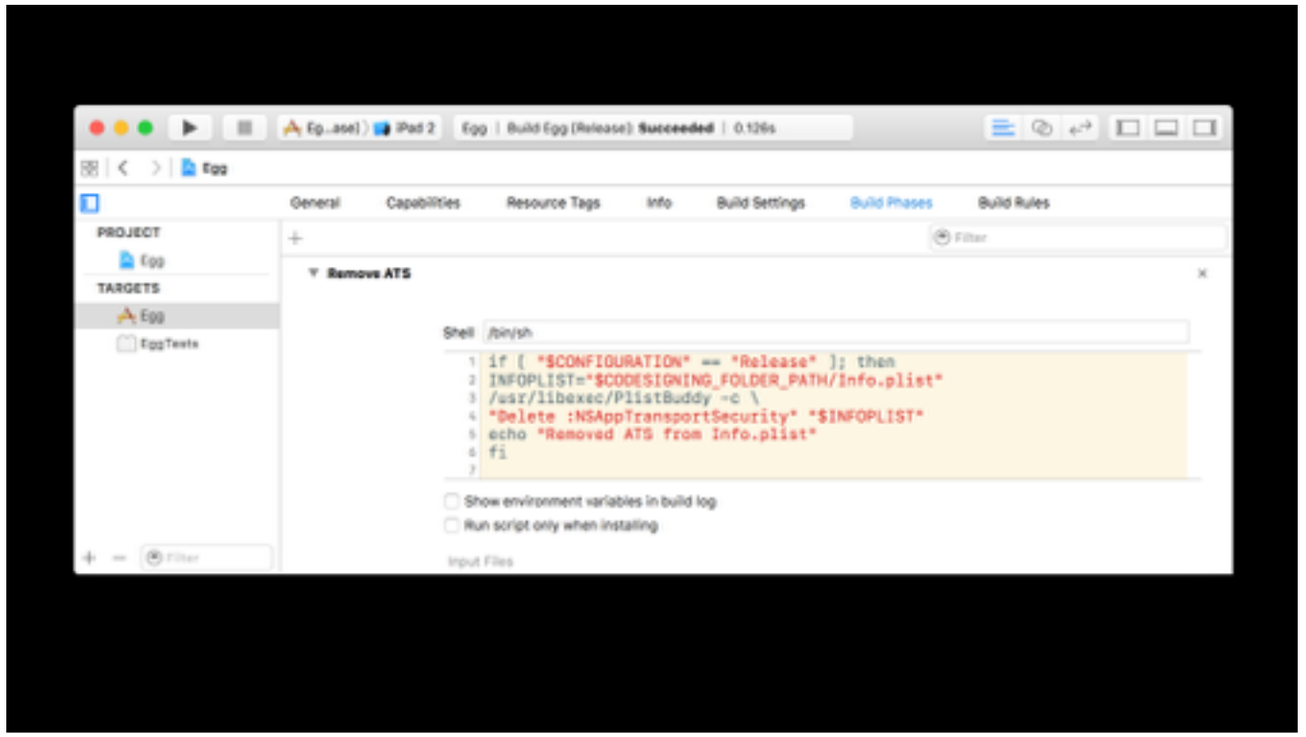
Now that we know the terminology, let's put them to work

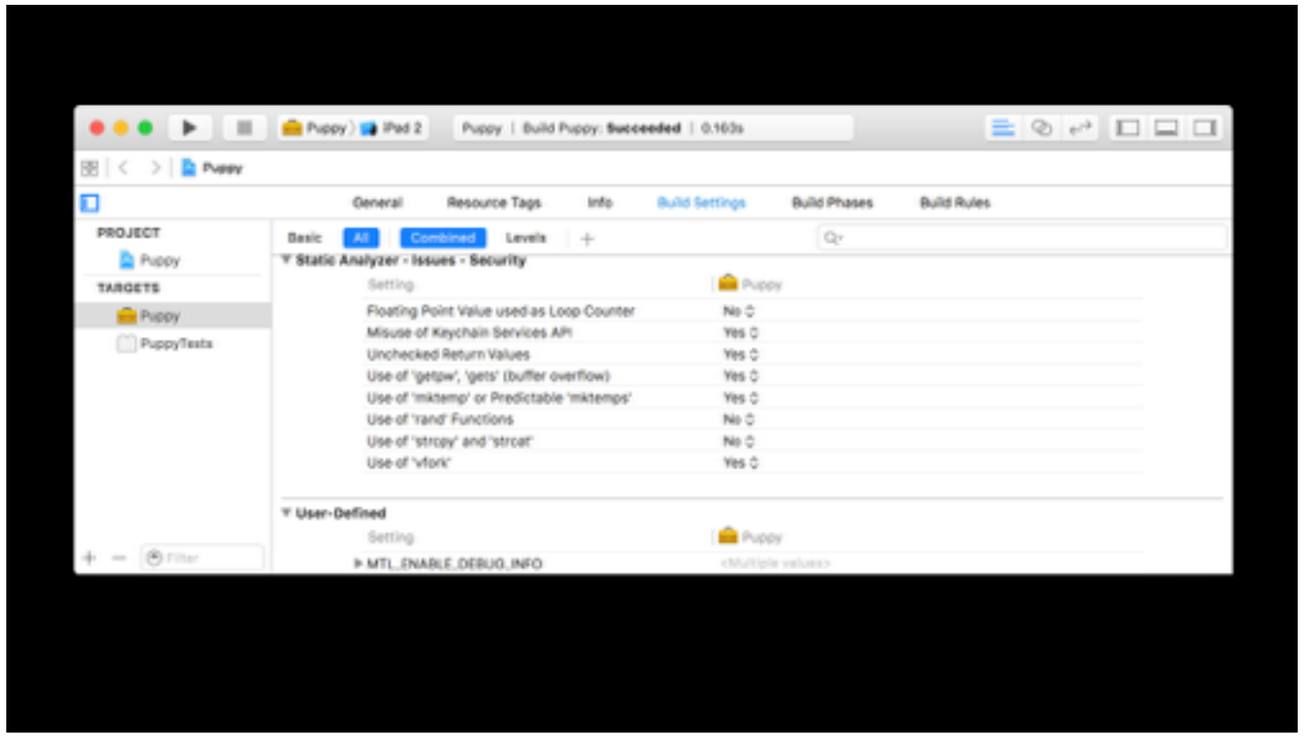
Build Phase Script

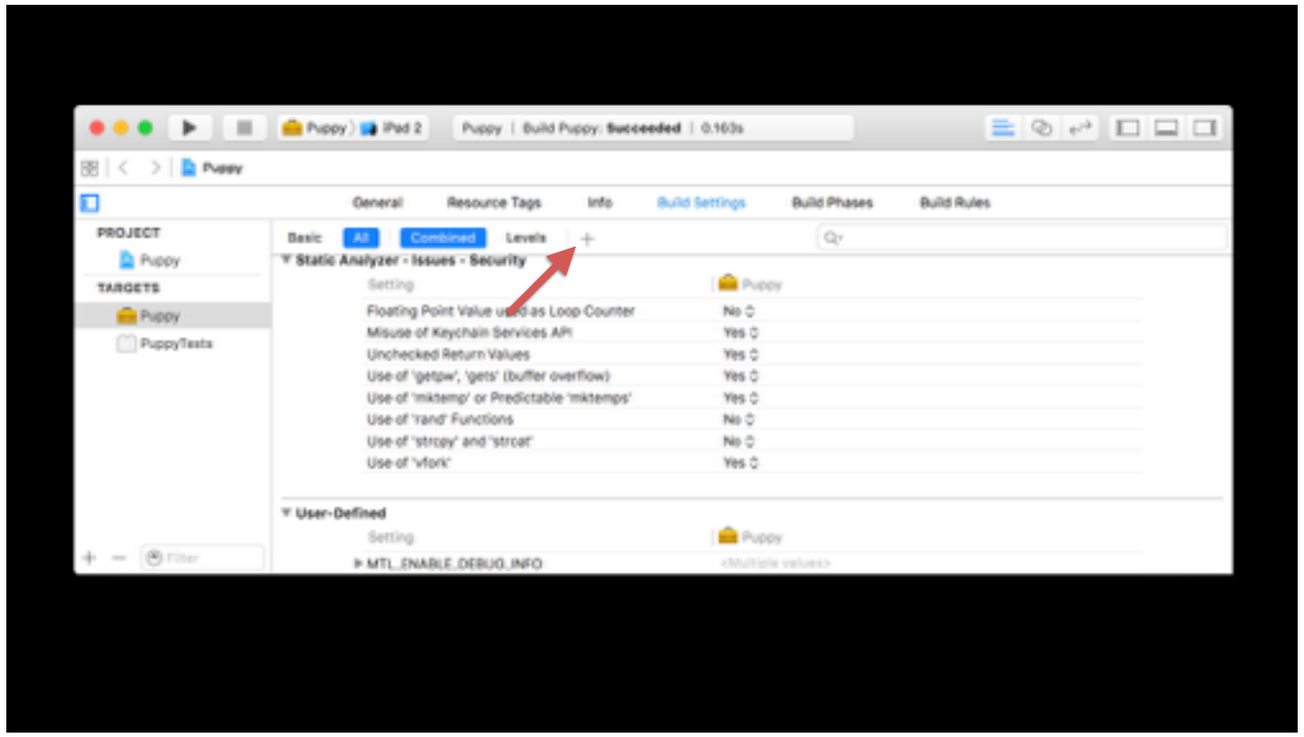

iPhone 6
Finished running Nectarine on iPhone 6
Nectarine
Nectarine
AppDelegate.swift
FirstViewController.swift
SecondViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Resources
Resources-Debug
Settings.bundle
en.lproj
FeatureToggles.plist
Root.plist
Nectarine
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
New Copy Files Phase
New Run Script Phase
New Headers Phase
New Copy Bundle Resources Phase
New Compile Sources Phase
New Link Binary With Libraries Phase
New Build Carbon Resources Phase

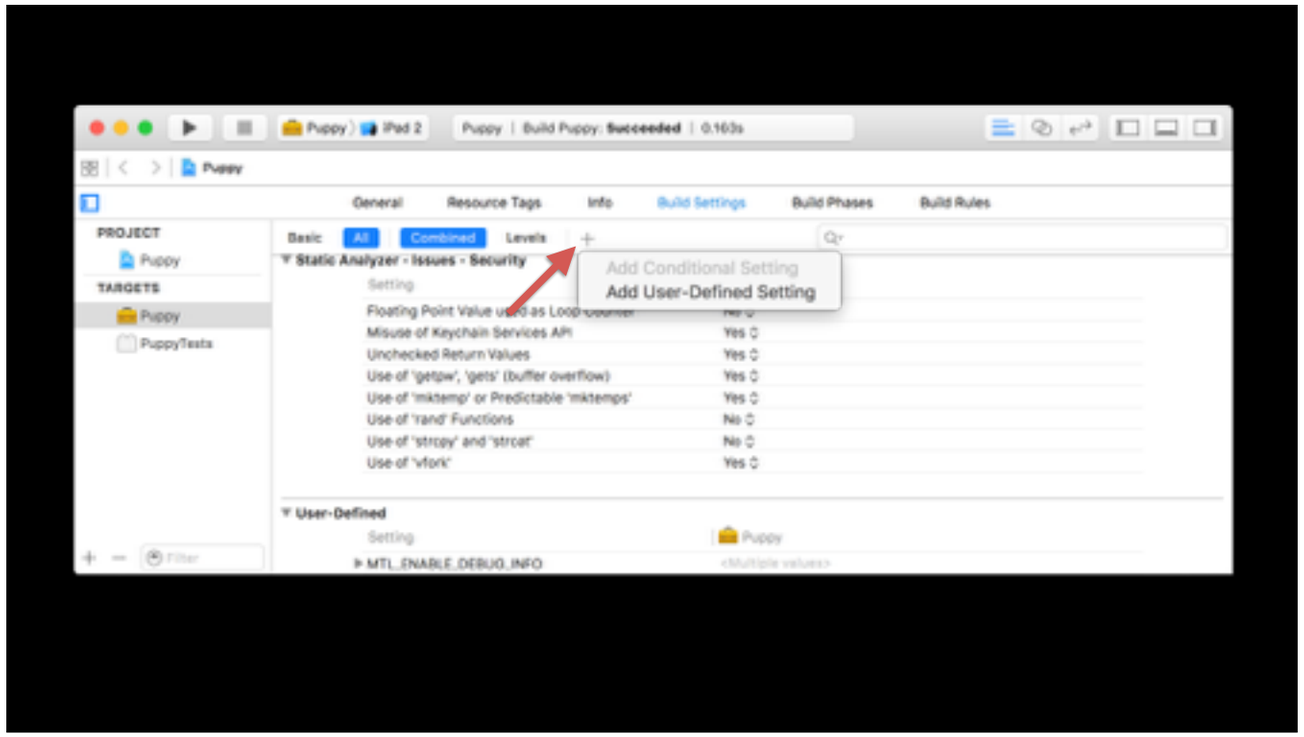
ATS

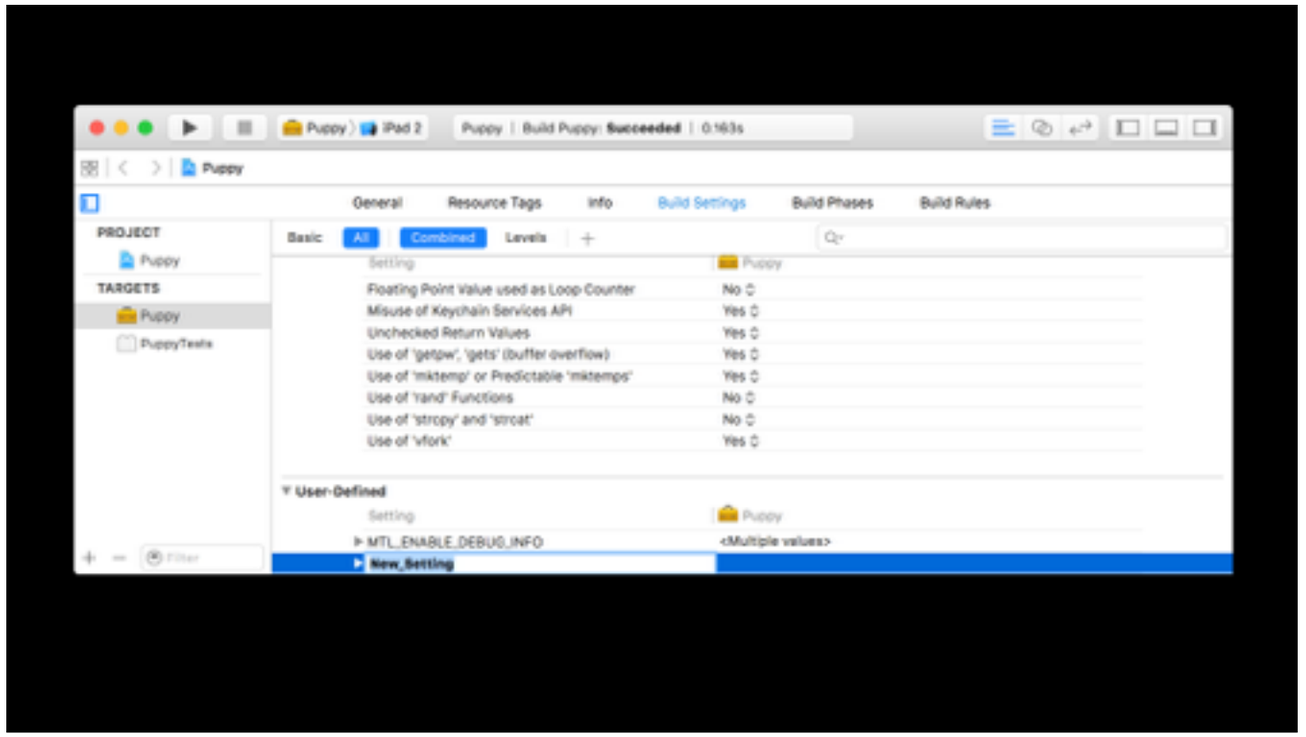
```
if [ "$CONFIGURATION" == "Release" ]; then
   INFOPLIST="$CODESIGNING_FOLDER_PATH/Info.plist"
   /usr/libexec/PlistBuddy -c \
        "Delete :NSAppTransportSecurity" "$INFOPLIST"
    echo "Removed ATS from Info.plist"
fi
```


Eg...ase) iPad 2
Egg | Build Egg (Release): Succeeded | 0.126s
Egg
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
Egg
TARGETS
Egg
EggTests
Filter
Remove ATS
Shell /bin/sh
1 if ["$CONFIGURATION" == "Release"]; then
2 INFOPLIST="$CODESIGNING_FOLDER_PATH/Info.plist"
3 /usr/libexec/PlistBuddy -c \
4 "Delete :NSAppTransportSecurity" "$INFOPLIST"
5 echo "Removed ATS from Info.plist"
6 fi
7
Show environment variables in build log
Run script only when installing
Input Files

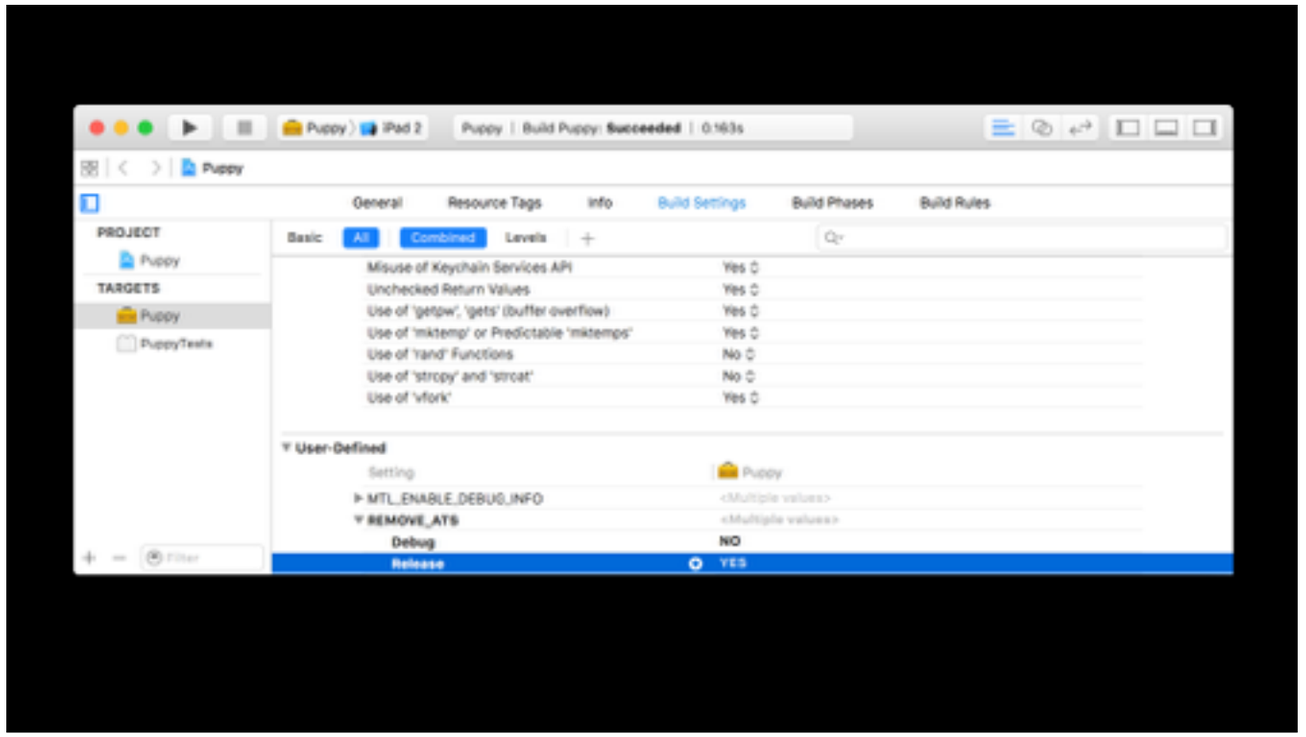

Puppy | iPad 2
Puppy | Build Puppy: Succeeded | 0.163s
Puppy
General
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
Puppy
TARGETS
Puppy
PuppyTests
Basic
All
Combined
Levels
Static Analyzer - Issues - Security
Setting
Puppy
Floating Point Value used as Loop Counter
No
Misuse of Keychain Services API
Yes
Unchecked Return Values
Yes
Use of 'getpw', 'gets' (buffer overflow)
Yes
Use of 'mktemp' or Predictable 'mktemps'
Yes
Use of 'rand' Functions
No
Use of 'strcpy' and 'strcat'
No
Use of 'vfork'
Yes
User-Defined
Setting
Puppy
MTL_ENABLE_DEBUG_INFO


Puppy | iPad 2
Puppy | Build Puppy: Succeeded | 0.163s
Puppy
General
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
Puppy
TARGETS
Puppy
PuppyTests
Basic
All
Combined
Levels
Static Analyzer - Issues - Security
Setting
Puppy
Floating Point Value used as Loop Counter
No
Misuse of Keychain Services API
Yes
Unchecked Return Values
Yes
Use of 'getpw', 'gets' (buffer overflow)
Yes
Use of 'mktemp' or Predictable 'mktemps'
Yes
Use of 'rand' Functions
No
Use of 'strcpy' and 'strcat'
No
Use of 'vfork'
Yes
User-Defined
Setting
Puppy
MTL_ENABLE_DEBUG_INFO

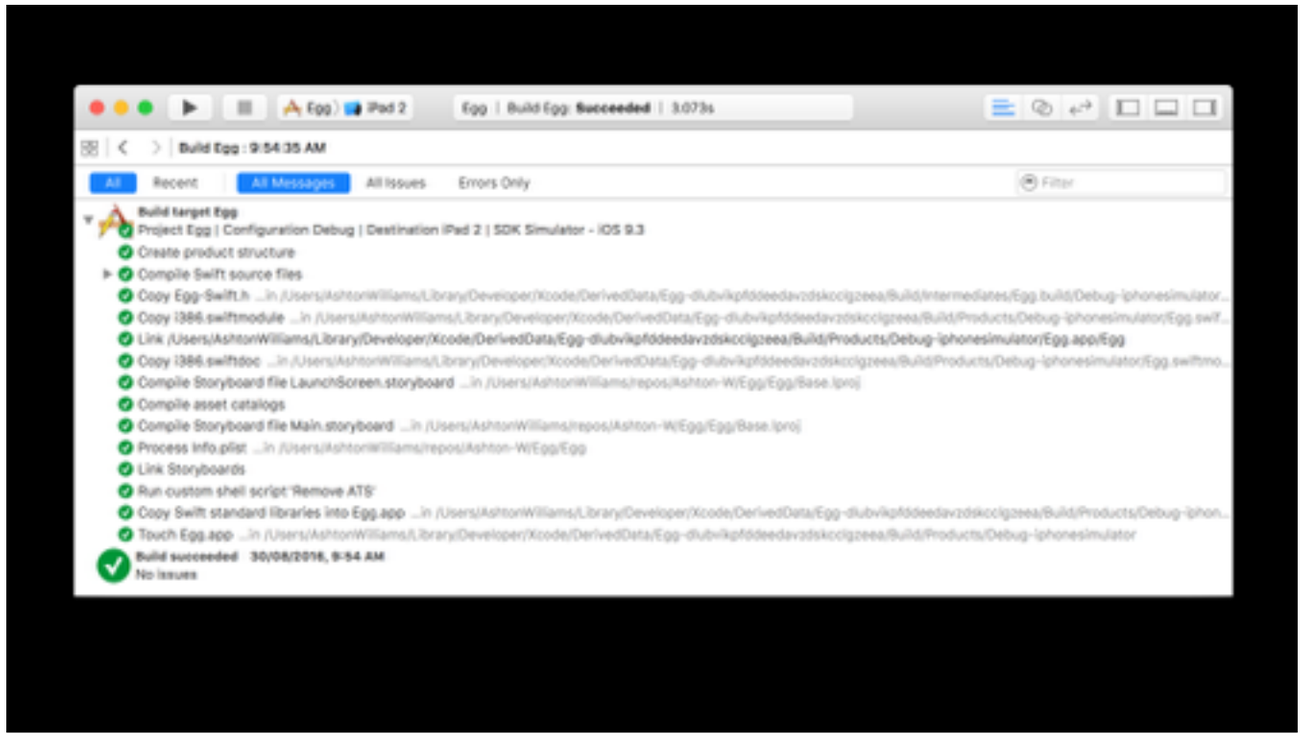

Puppy | iPad 2
Puppy | Build Puppy: Succeeded | 0.163s
Puppy
General
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
Puppy
TARGETS
Puppy
PuppyTests
Basic
All
Combined
Levels
Static Analyzer - Issues - Security
Setting
Add Conditional Setting
Add User-Defined Setting
Misuse of Keychain Services API
Yes
Unchecked Return Values
Yes
Use of 'getpw', 'gets' (buffer overflow)
Yes
Use of 'mktemp' or Predictable 'mktemps'
Yes
Use of 'rand' Functions
No
Use of 'strcpy' and 'strcat'
No
Use of 'vfork'
Yes
User-Defined
Setting
Puppy
MTL_ENABLE_DEBUG_INFO

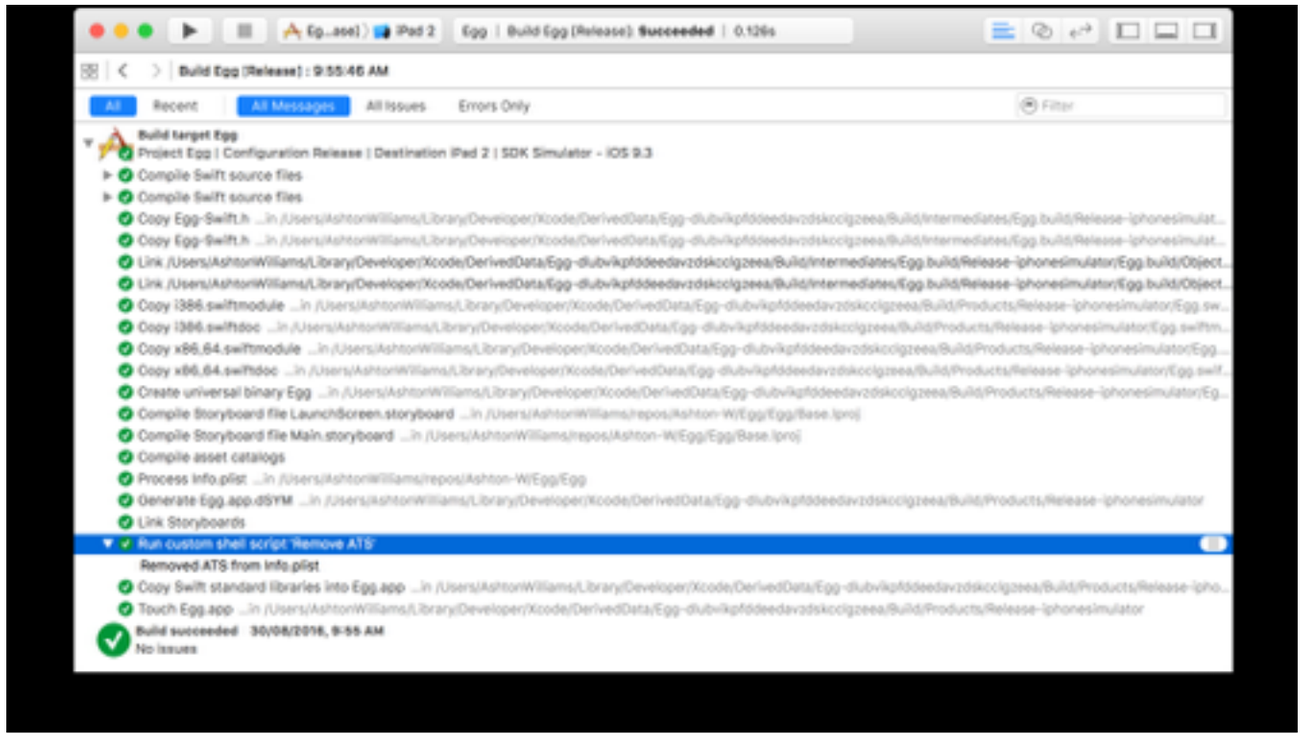


Our script can now check this Build Setting, which is available in the script as environment variables



Our script can now check this Build Setting, which is available in the script as environment variables


Egg | iPad 2
Egg | Build Egg: Succeeded | 3.073s
Build Egg : 9:54:35 AM
All
Recent
All Messages
All Issues
Errors Only
Filter
Build target Egg
Project Egg | Configuration Debug | Destination iPad 2 | SDK Simulator - iOS 9.3
Create product structure
Compile Swift source files
Copy Egg-Swift.h ...in /Users/AshtonWilliams/Library/Developer/Xcode/DerivedData/Egg-dlubvlkpfddeedavzdskcclgzeea/Build/Intermediates/Egg.build/Debug-iphonesimulator...
Copy i386.swiftmodule ...in /Users/AshtonWilliams/Library/Developer/Xcode/DerivedData/Egg-dlubvlkpfddeedavzdskcclgzeea/Build/Products/Debug-iphonesimulator/Egg.swif...
Link /Users/AshtonWilliams/Library/Developer/Xcode/DerivedData/Egg-dlubvlkpfddeedavzdskcclgzeea/Build/Products/Debug-iphonesimulator/Egg.app/Egg
Copy i386.swiftdoc ...in /Users/AshtonWilliams/Library/Developer/Xcode/DerivedData/Egg-dlubvlkpfddeedavzdskcclgzeea/Build/Products/Debug-iphonesimulator/Egg.swiftmo...
Compile Storyboard file LaunchScreen.storyboard ...in /Users/AshtonWilliams/repos/Ashton-W/Egg/Egg/Base.lproj
Compile asset catalogs
Compile Storyboard file Main.storyboard ...in /Users/AshtonWilliams/repos/Ashton-W/Egg/Egg/Base.lproj
Process Info.plist ...in /Users/AshtonWilliams/repos/Ashton-W/Egg/Egg
Link Storyboards
Run custom shell script 'Remove ATS'
Copy Swift standard libraries into Egg.app ...in /Users/AshtonWilliams/Library/Developer/Xcode/DerivedData/Egg-dlubvlkpfddeedavzdskcclgzeea/Build/Products/Debug-iphon...
Touch Egg.app ...in /Users/AshtonWilliams/Library/Developer/Xcode/DerivedData/Egg-dlubvlkpfddeedavzdskcclgzeea/Build/Products/Debug-iphonesimulator
Build succeeded 30/08/2016, 9:54 AM
No issues


Egg | Build Egg (Release): Succeeded | 0.126s
Build Egg (Release) : 9:55:46 AM
All
Recent
All Messages
All Issues
Errors Only
Filter
Build target Egg
Project Egg | Configuration Release | Destination iPad 2 | SDK Simulator - iOS 9.3
Compile Swift source files
Compile Swift source files
Copy Egg-Swift.h
Copy Egg-Swift.h
Link /Users/AshtonWilliams/Library/Developer/Xcode/DerivedData/Egg-dubvlkpfddeedavzdskcclgzeea/Build/Intermediates/Egg.build/Release-iphonesimulator/Egg.build/Object...
Link /Users/AshtonWilliams/Library/Developer/Xcode/DerivedData/Egg-dubvlkpfddeedavzdskcclgzeea/Build/Intermediates/Egg.build/Release-iphonesimulator/Egg.build/Object...
Copy i386.swiftmodule
Copy i386.swiftdoc
Copy x86_64.swiftmodule
Copy x86_64.swiftdoc
Create universal binary Egg
Compile Storyboard file LaunchScreen.storyboard
Compile Storyboard file Main.storyboard
Compile asset catalogs
Process Info.plist
Generate Egg.app.dSYM
Link Storyboards
Run custom shell script 'Remove ATS'
Removed ATS from Info.plist
Copy Swift standard libraries into Egg.app
Touch Egg.app
Build succeeded 30/08/2016, 9:55 AM
No issues

Configurations & Resources

Test and Beta Builds

Use different resources for pre-release builds

- App Icons
- Settings Bundle
- Test Data

You can create different builds of your app for different purposes, for example: Test and Release. Release is your regular production version of the app but Test might include debug features.

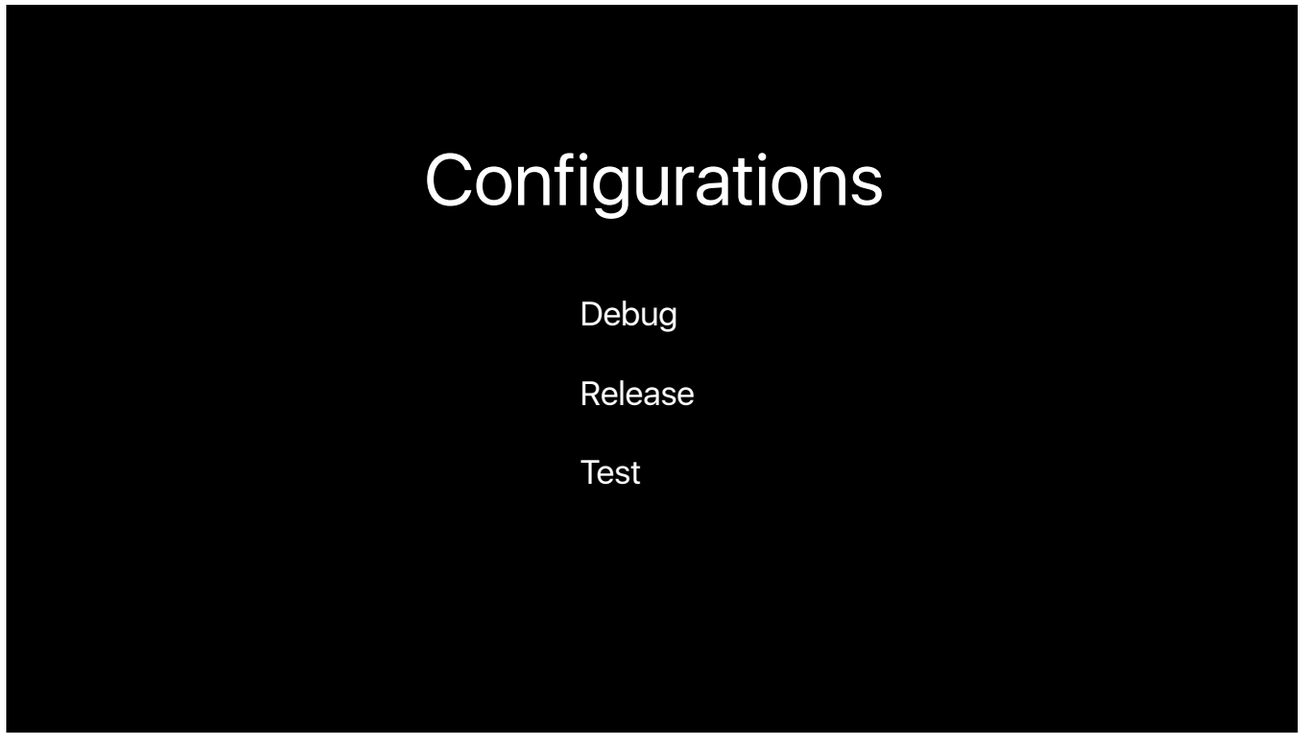
Some examples of what you can do with this is to include a `Settings.bundle` with dev switches or debug flags only in the Test version.
If you are creating seperate Beta and Release builds you can include correctly named App icon assets (Check your build product `.app`) and have different icon variations for the different builds too.

Configurations

Debug

Release

To achieve this you will want to leverage configurations in Xcode to create variants of your app in addition to the default Debug and Release builds.



To achieve this you will want to leverage configurations in Xcode to create variants of your app in addition to the default Debug and Release builds.

Configurations

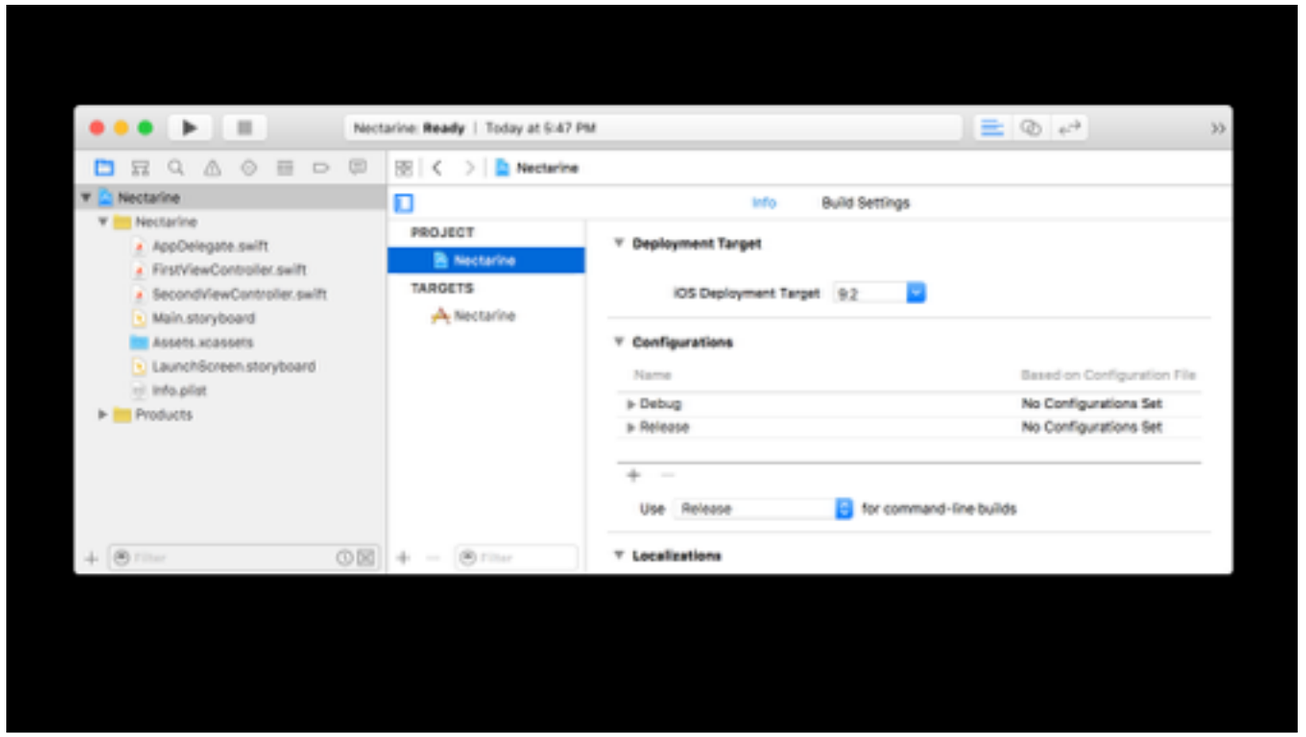
Debug

Release

Test

Beta

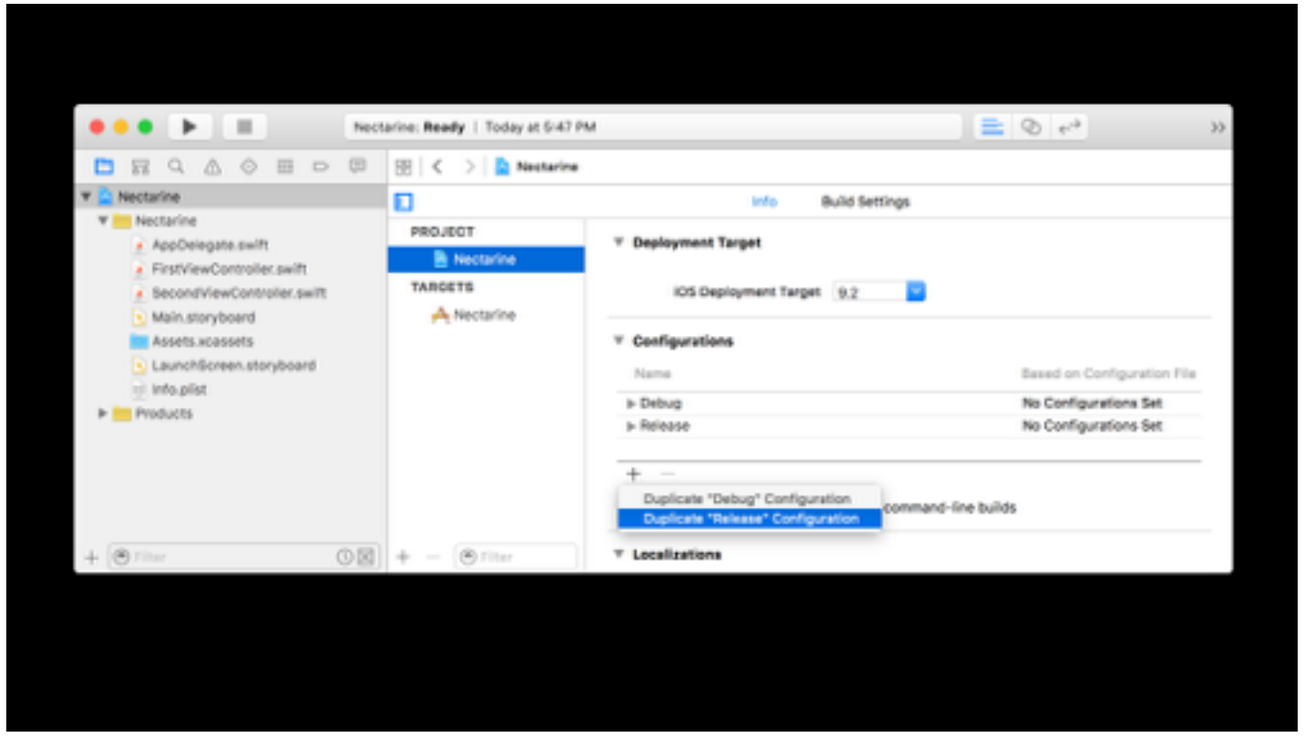
To achieve this you will want to leverage configurations in Xcode to create variants of your app in addition to the default Debug and Release builds.

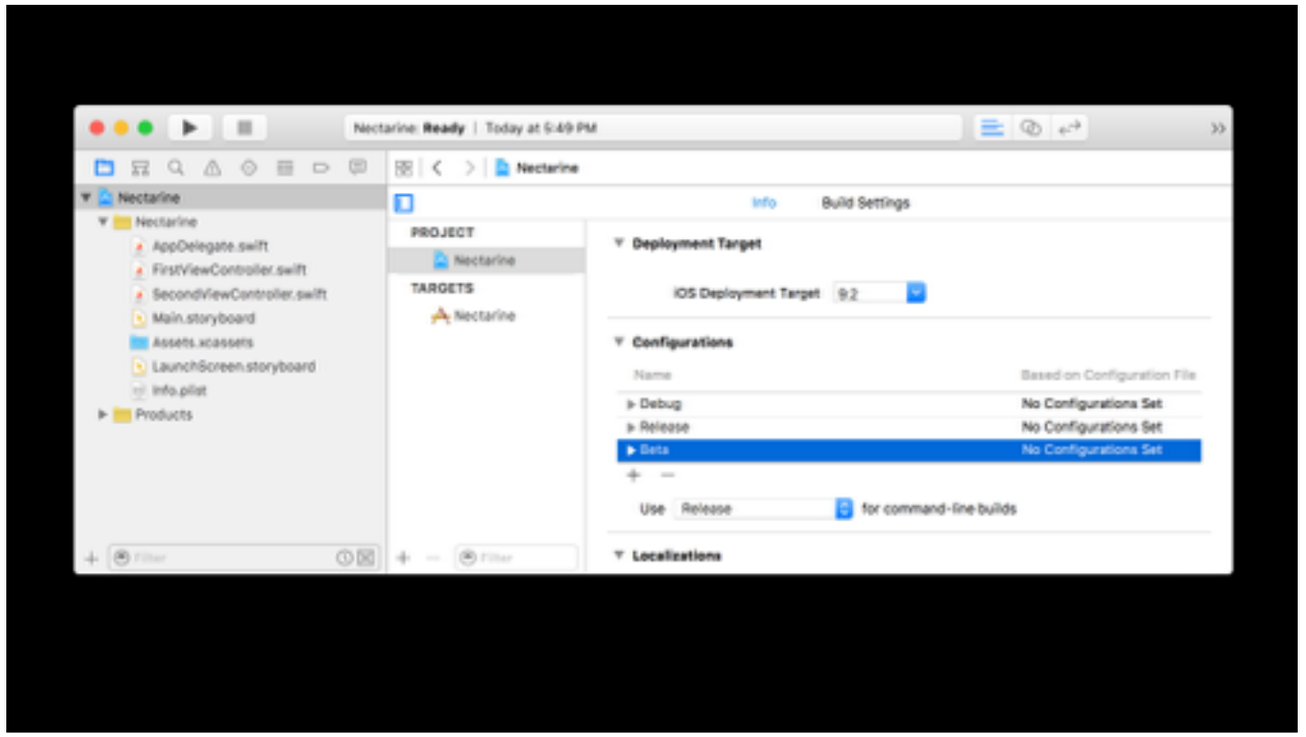


Project's Info pane

Configurations List

Click +


Nectarine: Ready | Today at 6:47 PM
Nectarine
Nectarine
AppDelegate.swift
FirstViewController.swift
SecondViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Nectarine
Info
Build Settings
PROJECT
Nectarine
TARGETS
Nectarine
Deployment Target
iOS Deployment Target 9.2
Configurations
Name
Based on Configuration File
Debug
No Configurations Set
Release
No Configurations Set
Duplicate "Debug" Configuration
Duplicate "Release" Configuration
command-line builds
Localizations


Nectarine: Ready | Today at 5:49 PM
Nectarine
Nectarine
Nectarine
AppDelegate.swift
FirstViewController.swift
SecondViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products
Info
Build Settings
PROJECT
Nectarine
TARGETS
Nectarine
Deployment Target
iOS Deployment Target
9.2
Configurations
Name
Based on Configuration File
Debug
No Configurations Set
Release
No Configurations Set
Beta
No Configurations Set
Use
Release
for command-line builds
Localizations

Resources

- `Resources/`

Build settings can be changed per configuration but resources can't - they are assigned to targets in the Xcode project.

Resources

- `Resources/`
- `Resources-Release/`

Build settings can be changed per configuration but resources can't - they are assigned to targets in the Xcode project.

Resources

- `Resources/`
- `Resources-Release/`
- `Resources-Test/`

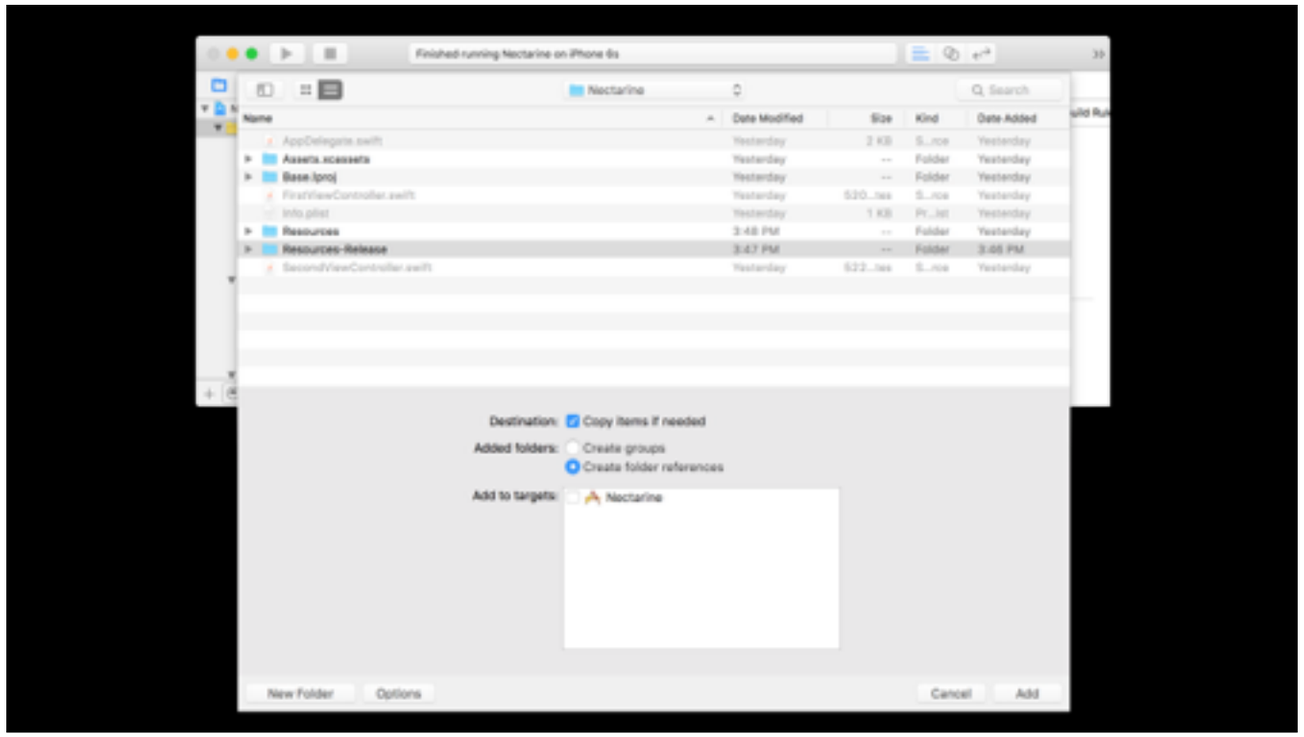
Build settings can be changed per configuration but resources can't - they are assigned to targets in the Xcode project.

```
RESOURCES_PATH="${RESOURCES_PATH:-$SOURCE_ROOT/$TARGET_NAME/Resources}"

#
copy_common_resources() {
  cp -vRf "$RESOURCES_PATH/" "$CODESIGNING_FOLDER_PATH/"
}

#
copy_config_resources() {
  local config_path="$RESOURCES_PATH-$CONFIGURATION/"
  if [ -d "$config_path" ]; then
    cp -vRf "$config_path" "$CODESIGNING_FOLDER_PATH/"
  fi
}

copy_common_resources && copy_config_resources
```

tinyurl.com/ConfigurationResources

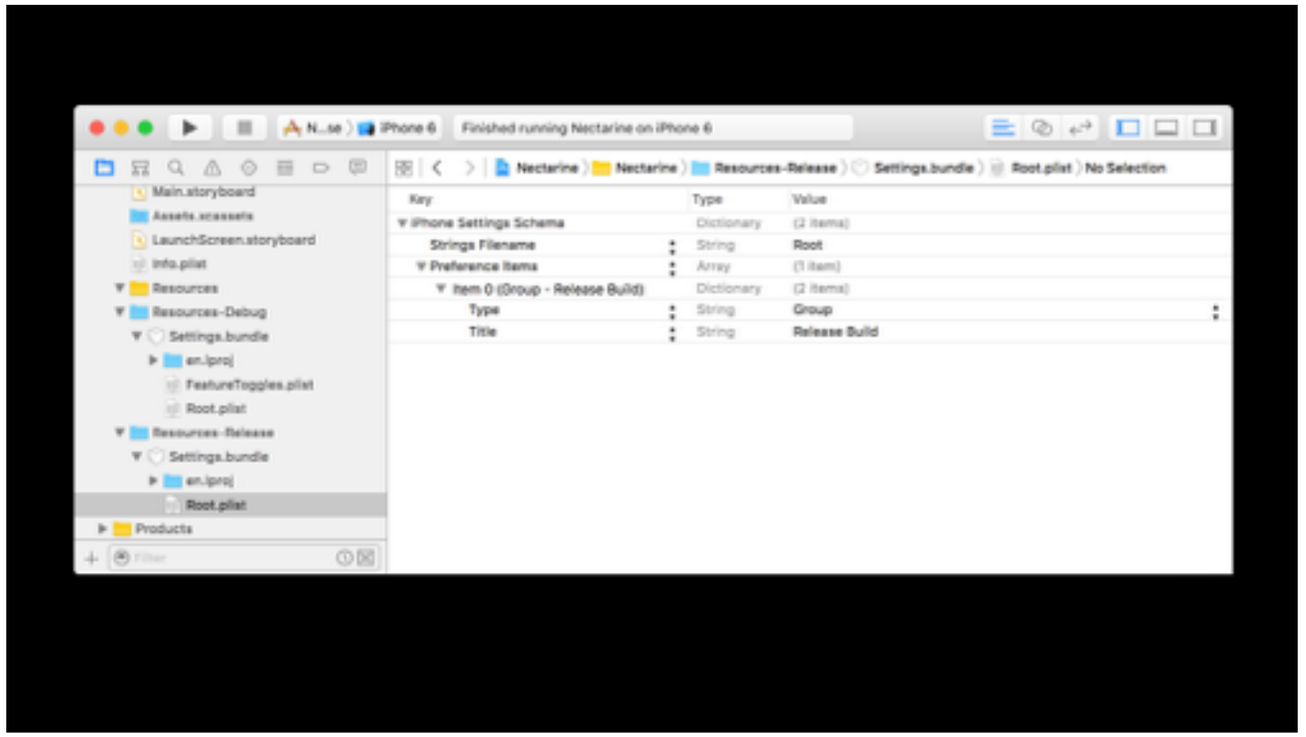
So I wrote a *Build Phase Script* to add resources at build time. It chooses them by which configuration we build.
With this script configurations can produce builds with different bundled resources.

Check out the full gist and you'll see I even wrote tests for it!

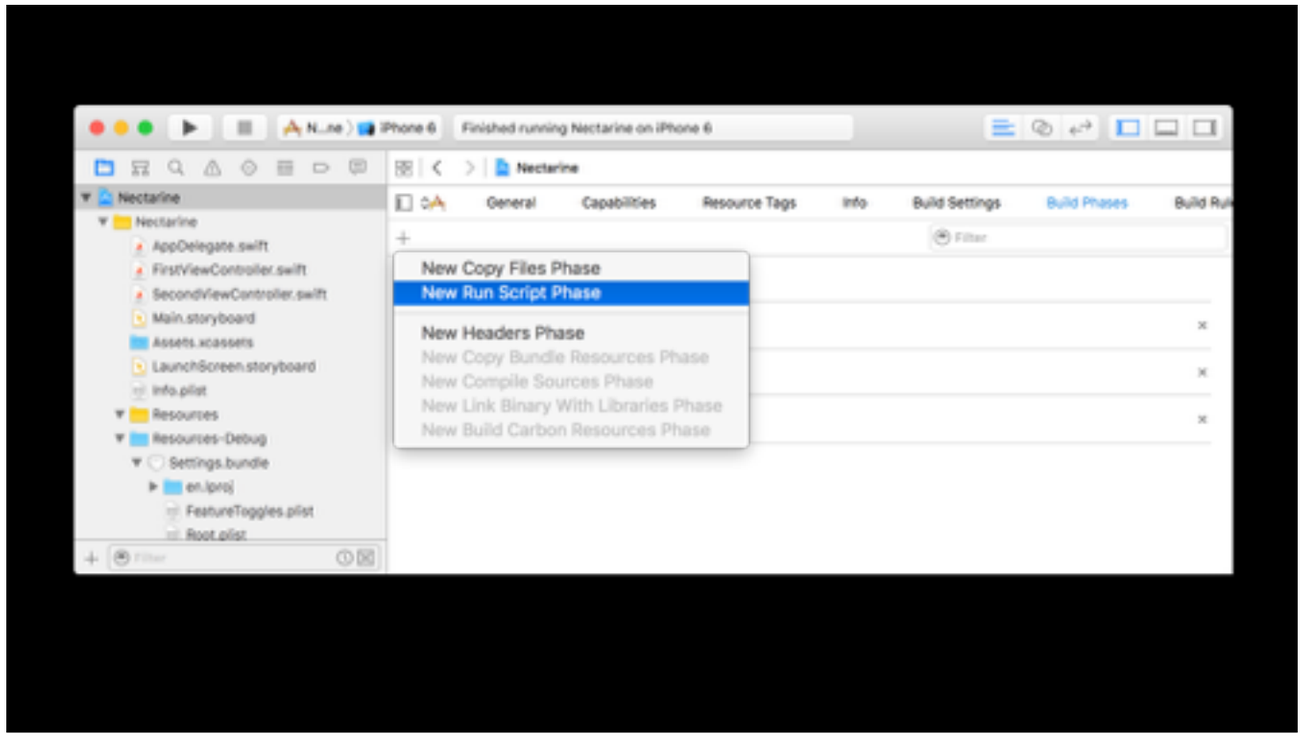


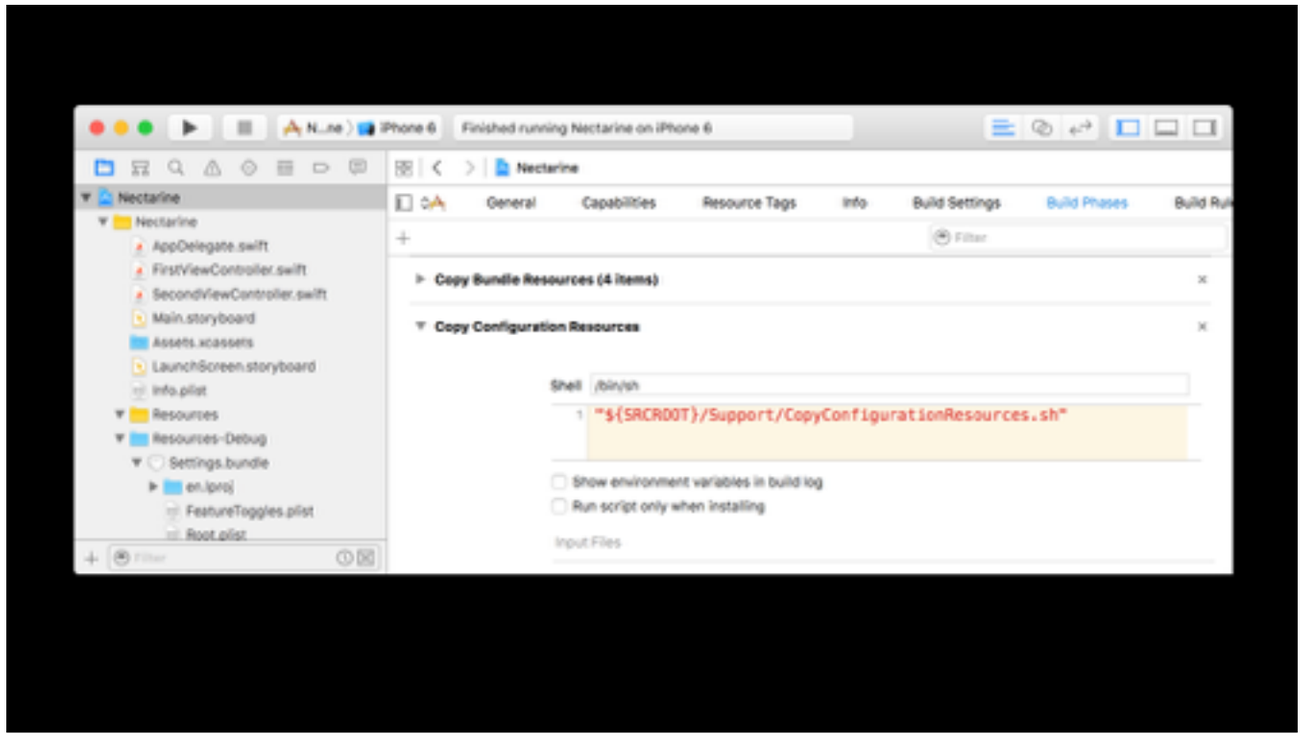
Don't add extra Resources to a TARGET
This would copy your resources by default

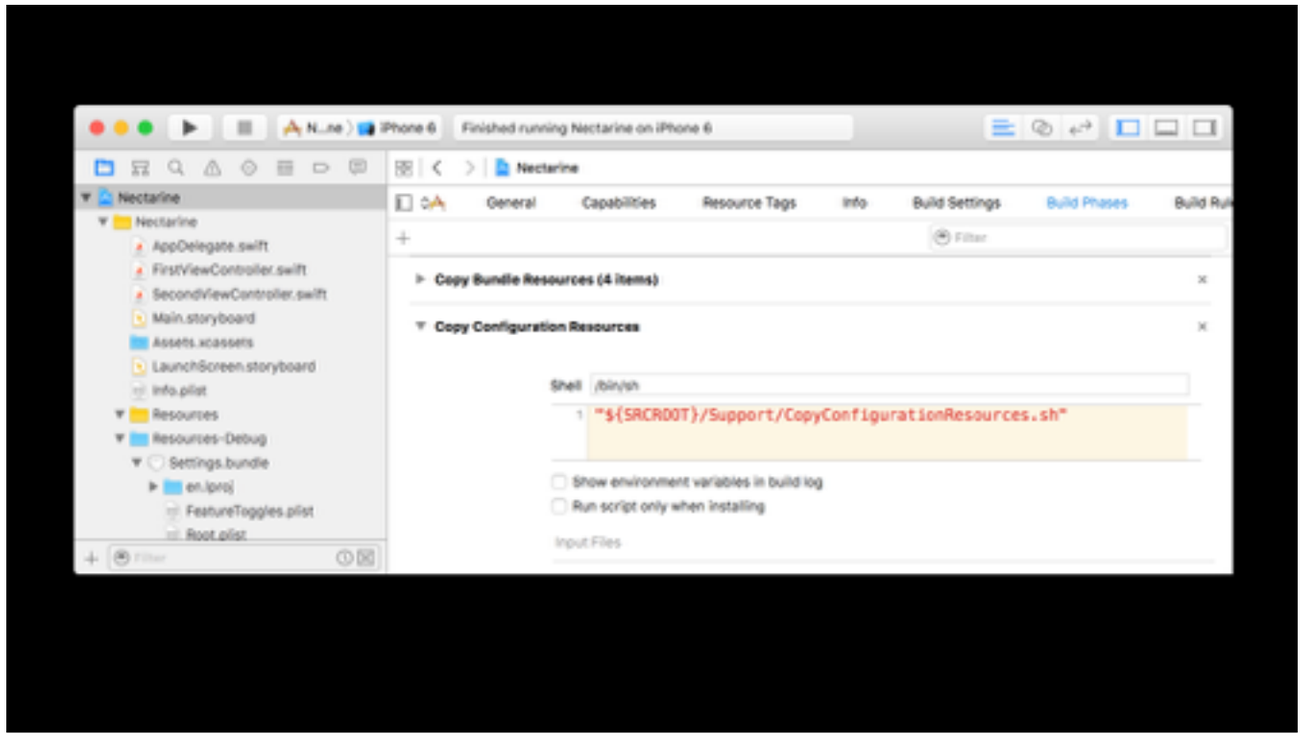
Blue Folder Reference means we can see and modify the directory

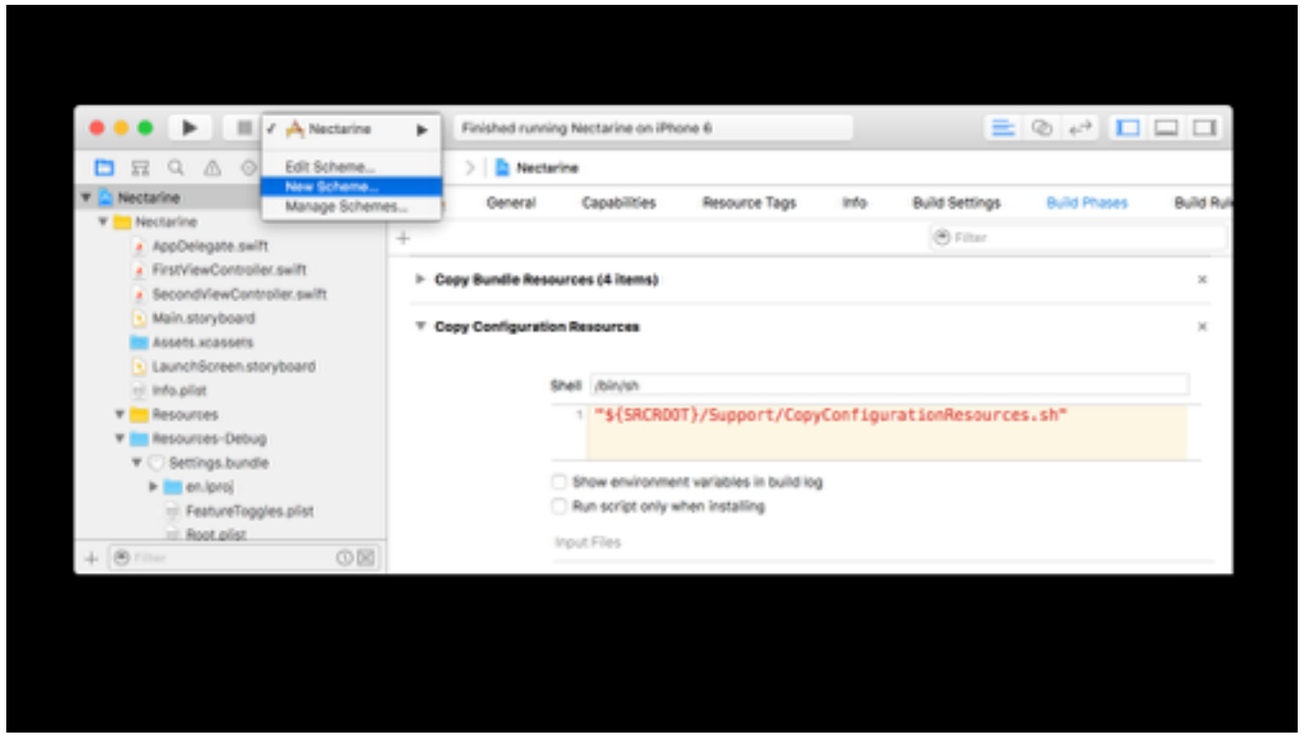


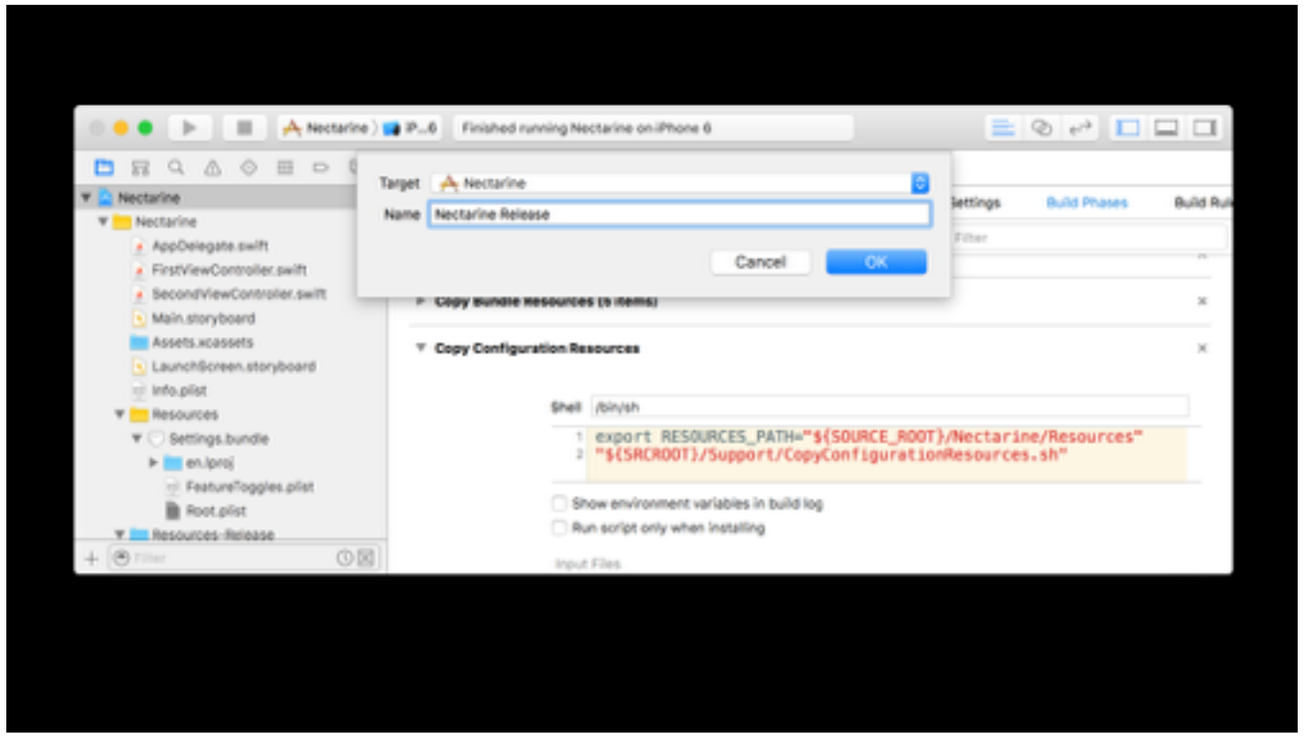
Regular resources are included in the target, these are default, overwritten by Configuration Resources.
Be aware bundles are directories, so they are merged.

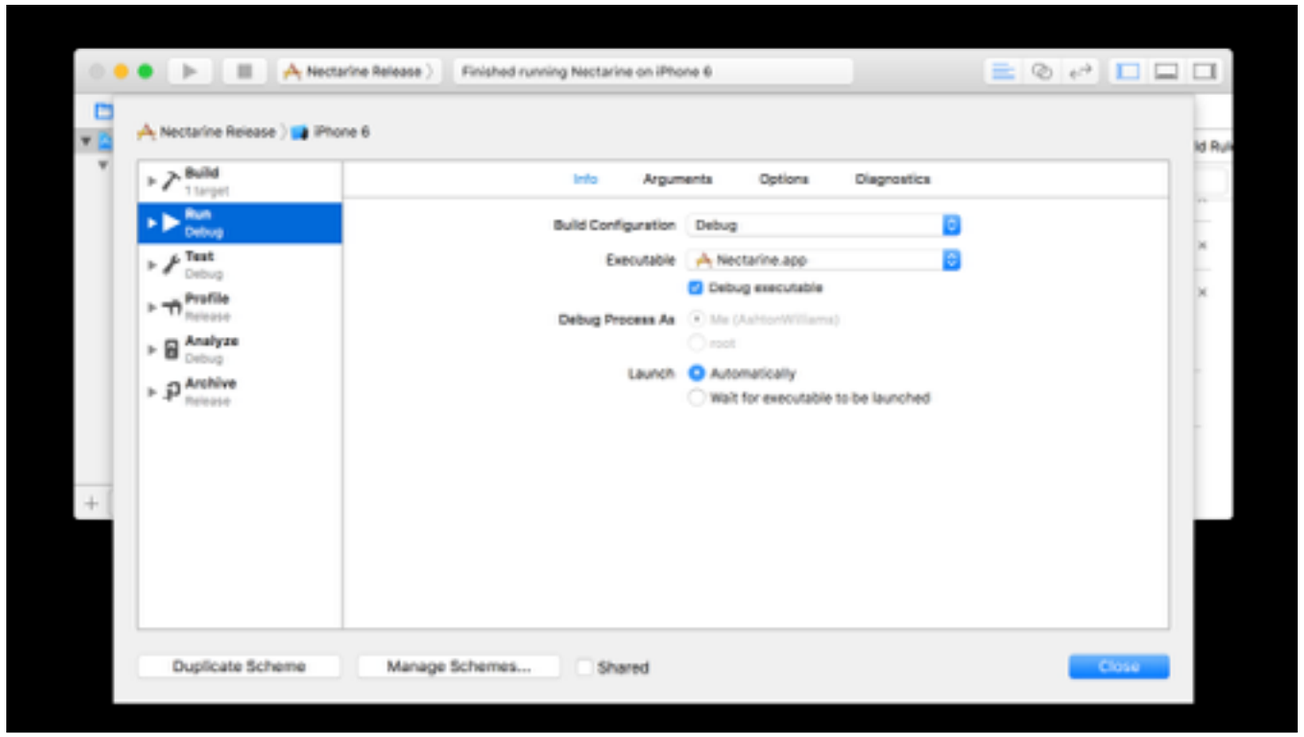

iPhone 6
Finished running Nectarine on iPhone 6
Nectarine
Nectarine
AppDelegate.swift
FirstViewController.swift
SecondViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Resources
Resources-Debug
Settings.bundle
en.lproj
FeatureToggles.plist
Root.plist
Nectarine
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
New Copy Files Phase
New Run Script Phase
New Headers Phase
New Copy Bundle Resources Phase
New Compile Sources Phase
New Link Binary With Libraries Phase
New Build Carbon Resources Phase

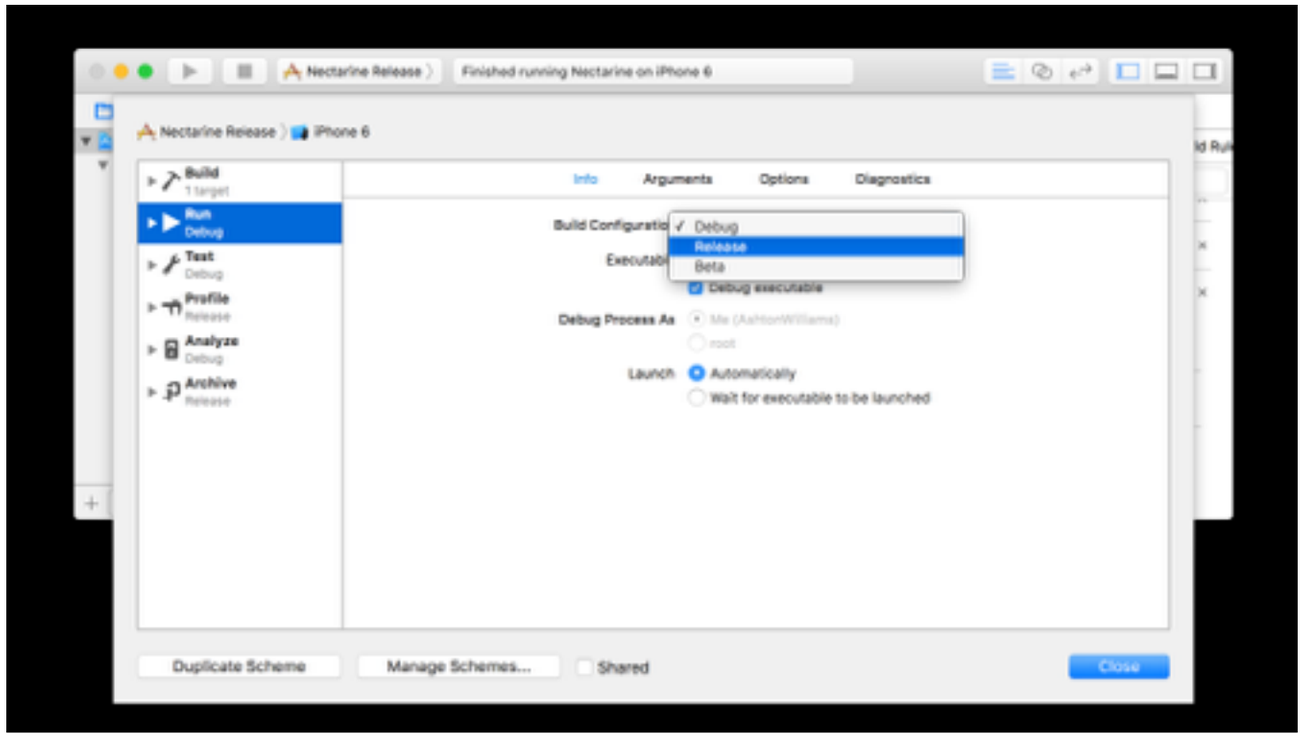

Finished running Nectarine on iPhone 6
Nectarine
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Copy Bundle Resources (4 items)
Copy Configuration Resources
Shell /bin/sh
"${SRCROOT}/Support/CopyConfigurationResources.sh"
Show environment variables in build log
Run script only when installing
Input Files
AppDelegate.swift
FirstViewController.swift
SecondViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Resources
Resources-Debug
Settings.bundle
FeatureToggles.plist
Root.plist

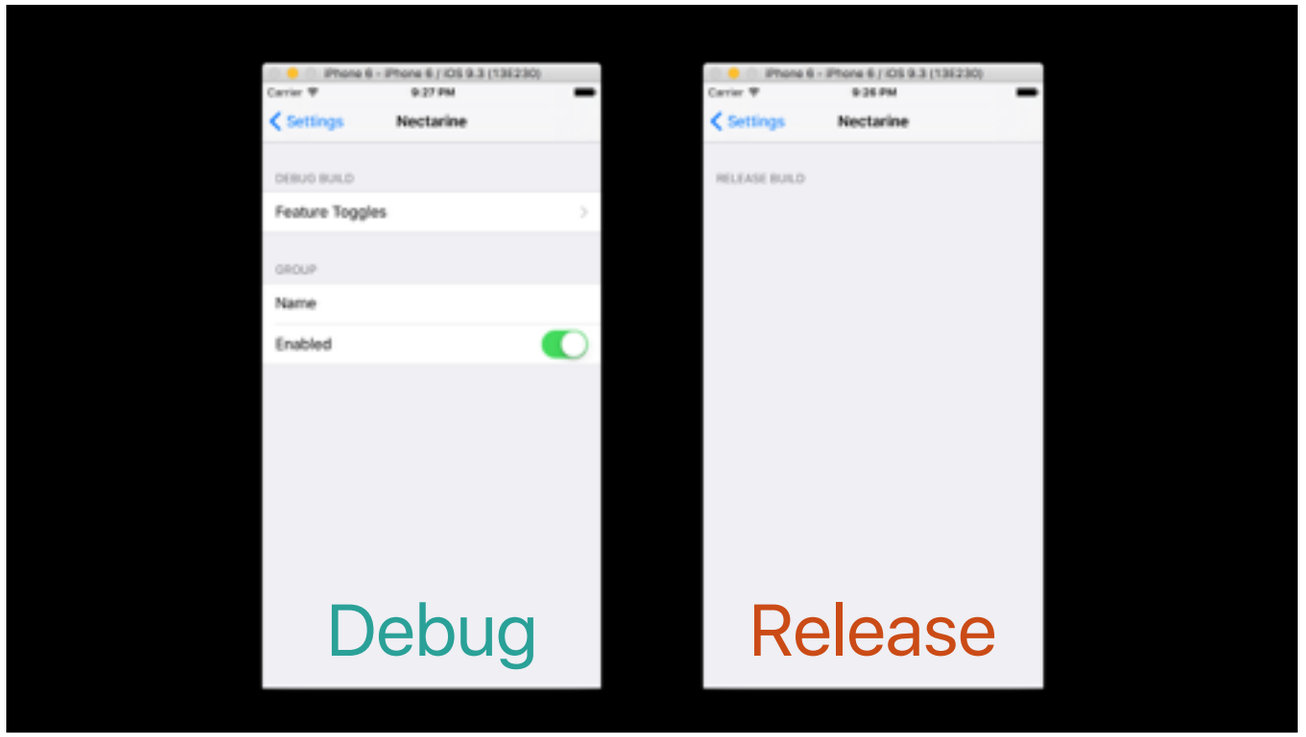

Finished running Nectarine on iPhone 6
Nectarine
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Copy Bundle Resources (4 items)
Copy Configuration Resources
Shell /bin/sh
"${SRCROOT}/Support/CopyConfigurationResources.sh"
Show environment variables in build log
Run script only when installing
Input Files
AppDelegate.swift
FirstViewController.swift
SecondViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Resources
Resources-Debug
Settings.bundle
FeatureToggles.plist
Root.plist


Nectarine
Finished running Nectarine on iPhone 6
Edit Scheme...
New Scheme...
Manage Schemes...
Nectarine
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Nectarine
AppDelegate.swift
FirstViewController.swift
SecondViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Resources
Resources-Debug
Settings.bundle
en.lproj
FeatureToggles.plist
Root.plist
Copy Bundle Resources (4 items)
Copy Configuration Resources
Shell
/bin/sh
"${SRCROOT}/Support/CopyConfigurationResources.sh"
Show environment variables in build log
Run script only when installing
Input Files


Nectarine
Finished running Nectarine on iPhone 6
Target
Nectarine
Name
Nectarine Release
Cancel
OK
Nectarine
AppDelegate.swift
FirstViewController.swift
SecondViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Resources
Settings.bundle
en.lproj
FeatureToggles.plist
Root.plist
Resources-Release
Build Phases
Copy Configuration Resources
Shell
/bin/sh
export RESOURCES_PATH="${SOURCE_ROOT}/Nectarine/Resources"
"${SRCROOT}/Support/CopyConfigurationResources.sh"
Show environment variables in build log
Run script only when installing
Input Files


Nectarine Release
Finished running Nectarine on iPhone 6
Nectarine Release
iPhone 6
Build
1 target
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Info
Arguments
Options
Diagnostics
Build Configuration
Debug
Executable
Nectarine.app
Debug executable
Debug Process As
Launch
Automatically
Wait for executable to be launched
Duplicate Scheme
Manage Schemes...
Shared
Close


Nectarine Release
Finished running Nectarine on iPhone 6
Nectarine Release
iPhone 6
Build
1 target
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Info
Arguments
Options
Diagnostics
Build Configuratio
Debug
Release
Beta
Executabl
Debug executable
Debug Process As
Launch
Automatically
Wait for executable to be launched
Duplicate Scheme
Manage Schemes...
Shared
Close


Settings
Nectarine
DEBUG BUILD
Feature Toggles
GROUP
Name
Enabled
Debug
Settings
Nectarine
RELEASE BUILD
Release

Schemes & User Defaults

User Defaults

NSUserDefaults

A.K.A. "the defaults system"
A.K.A. "Preferences and Settings"

User Defaults

- NSUserDefaults
- Runtime Arguments
- Settings app

Programmatically.

Runtime Arguments.

Settings bundle - Settings app.

User Defaults Domains

```
FOUNDATION_EXPORT NSString * const NSGlobalDomain;

FOUNDATION_EXPORT NSString * const NSArgumentDomain;

FOUNDATION_EXPORT NSString * const NSRegistrationDomain;
```

User Defaults Domains

`NSGlobalDomain`
Defaults meant to be seen by all applications. (OS X)

`NSArgumentDomain`
Defaults parsed from the application’s arguments.

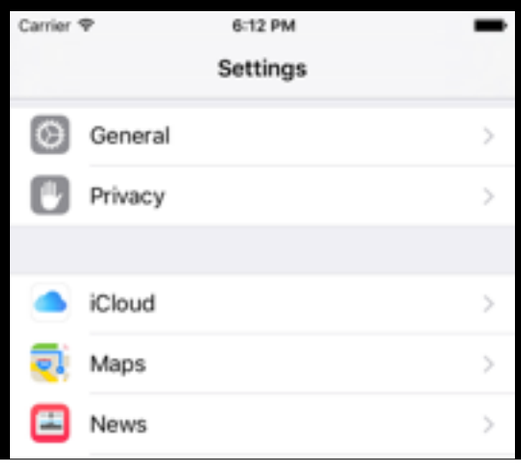
`NSRegistrationDomain`
Temporary defaults.

NSGlobalDomain
The domain consisting of defaults meant to be seen by all applications.

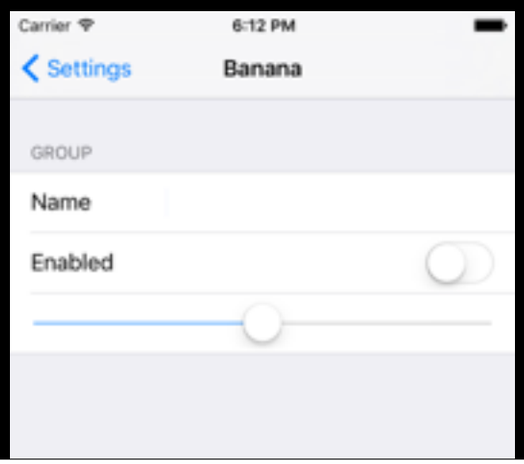
NSArgumentDomain
The domain consisting of defaults parsed from the application’s arguments. These are one or more pairs of the form -default value included in the command-line invocation of the application.

NSRegistrationDomain
The domain consisting of a set of temporary defaults whose values can be set by the application to ensure that searches will always be successful.

Settings



Settings

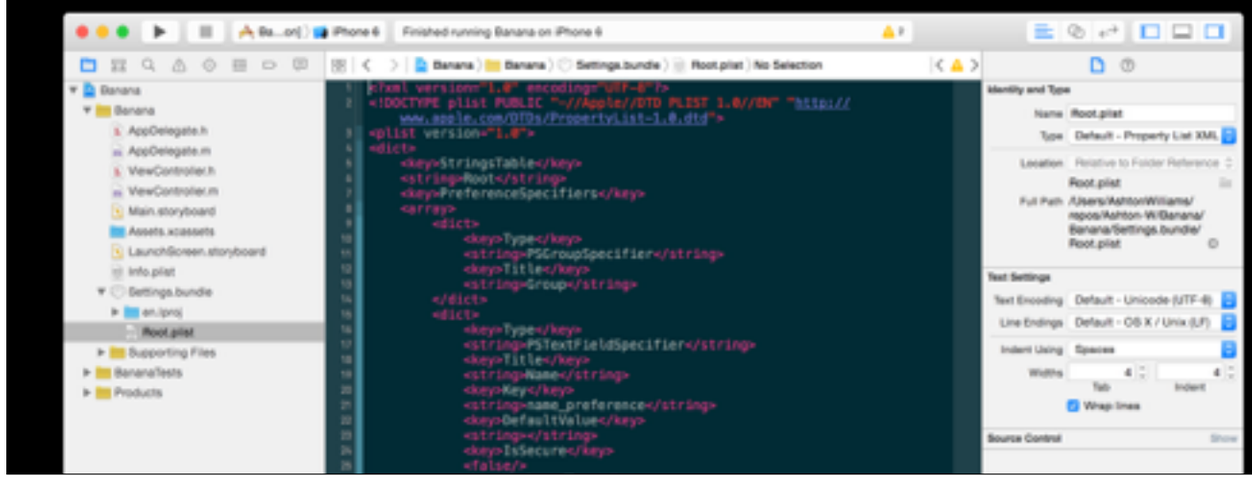


Settings Bundle

How to make Settings page?

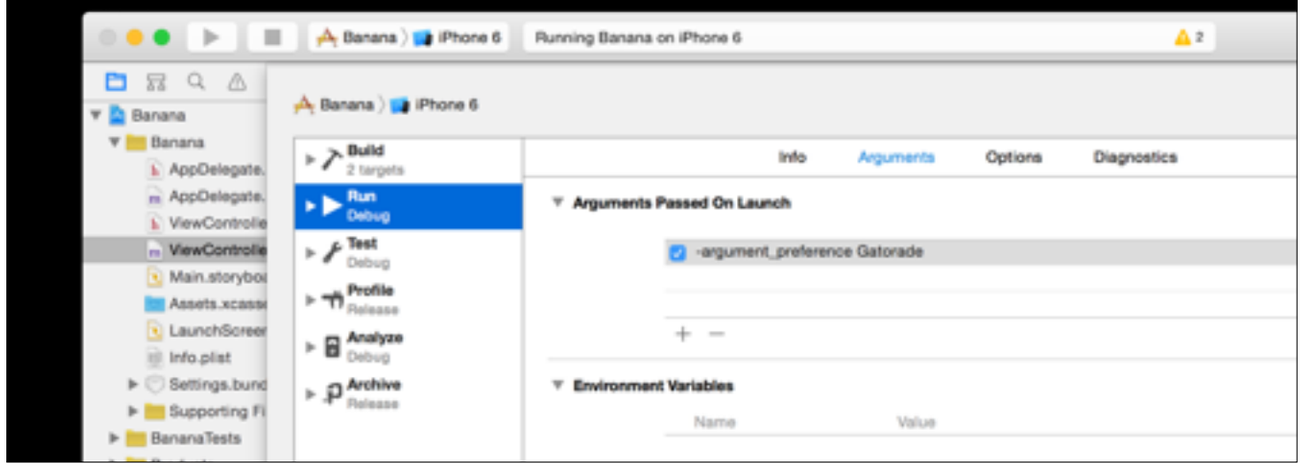
Bundle
Property List

Settings Bundle



Edit as XML.

Runtime Arguments



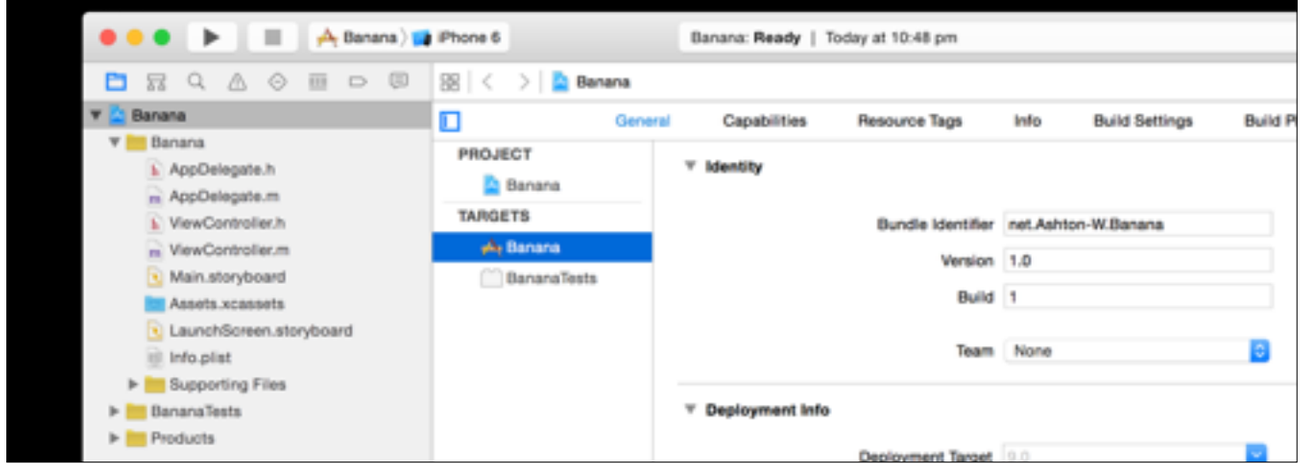
Registered Defaults

```
NSString *path = [[NSBundle mainBundle] pathForResource:@"Defaults" ofType:@"plist"];
NSDictionary *bundledDefaults = [NSDictionary dictionaryWithContentsOfFile:path];
[[NSUserDefaults standardUserDefaults] registerDefaults:bundledDefaults];
```

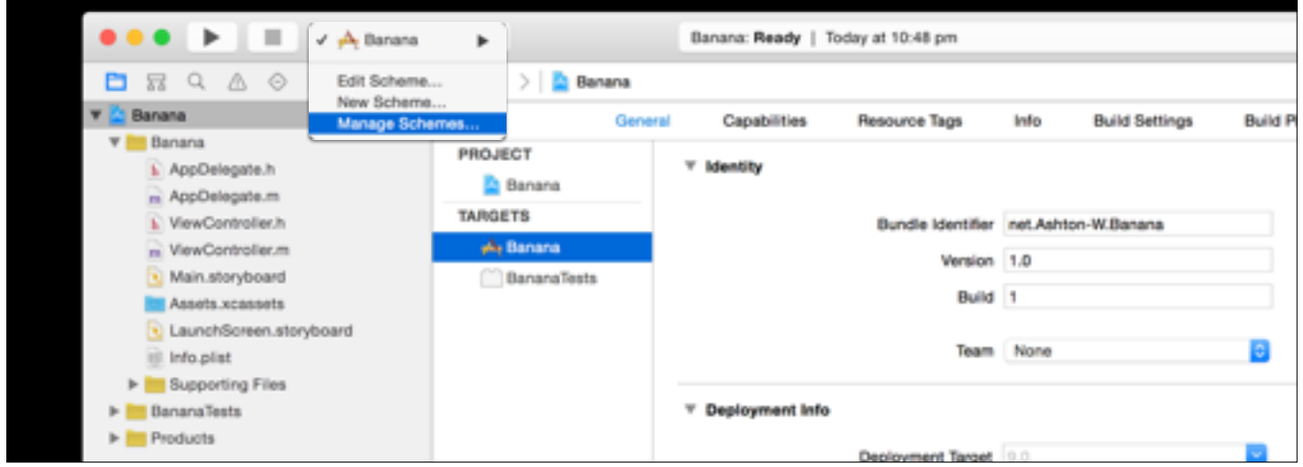
Schemes

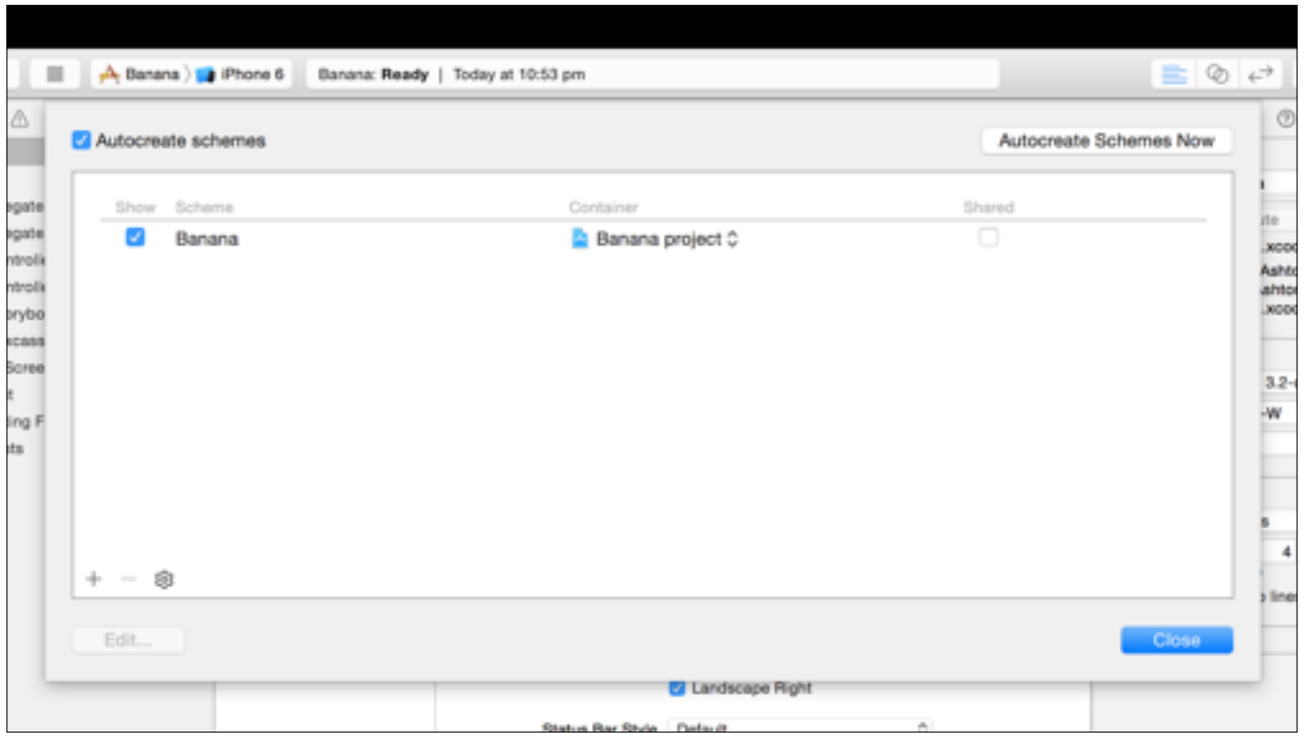
Schemes

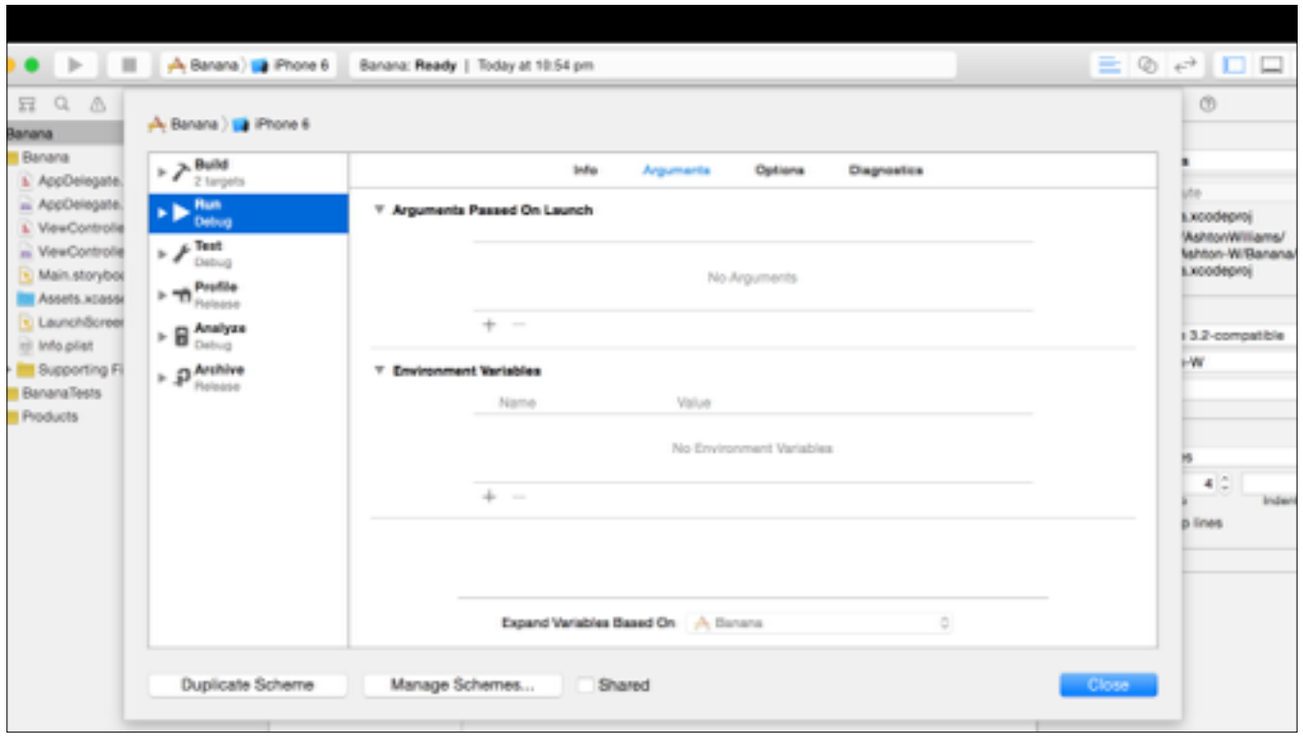
Schemes

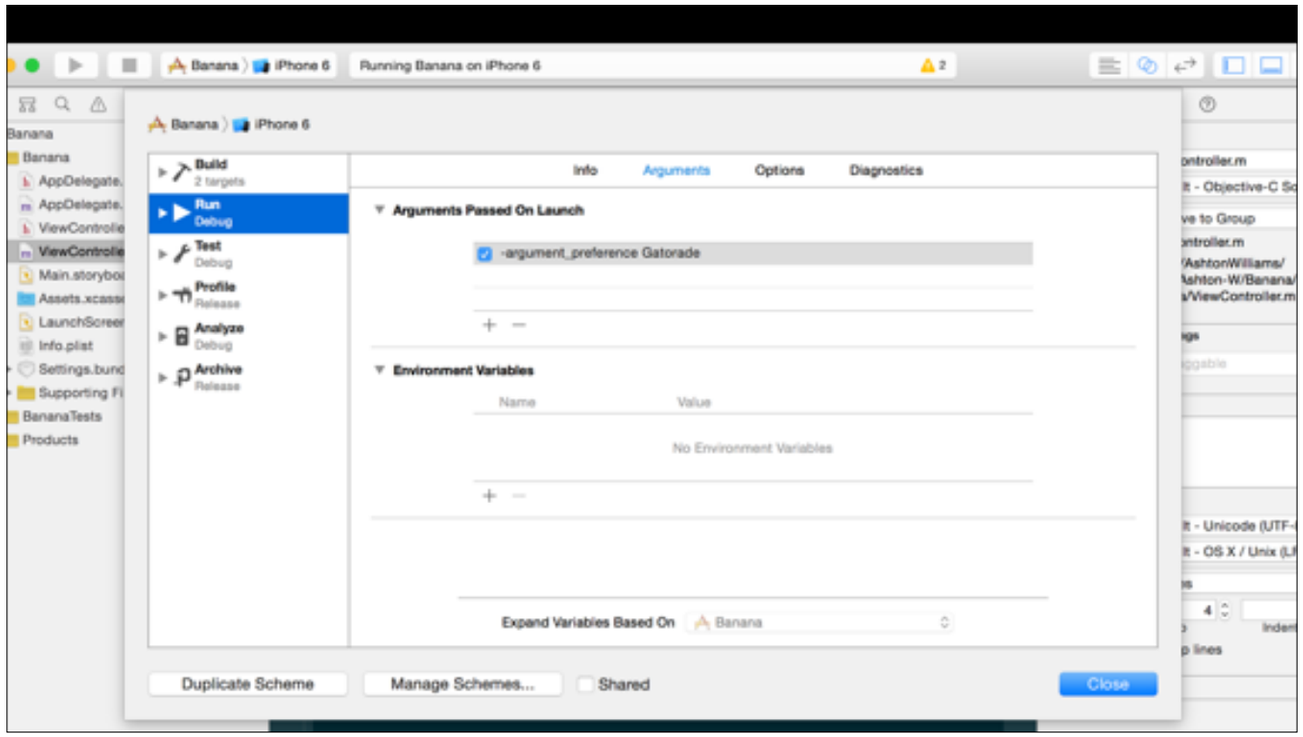

Banana
iPhone 6
Banana: Ready | Today at 10:48 pm
Banana
Banana
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Supporting Files
BananaTests
Products
General
Capabilities
Resource Tags
Info
Build Settings
PROJECT
Banana
TARGETS
Banana
BananaTests
Identity
Bundle Identifier
net.Ashton-W.Banana
Version
1.0
Build
1
Team
None
Deployment Info

Schemes


Banana
Edit Scheme...
New Scheme...
Manage Schemes...
Banana: Ready | Today at 10:48 pm
Banana
Banana
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Supporting Files
BananaTests
Products
General
Capabilities
Resource Tags
Info
Build Settings
PROJECT
Banana
TARGETS
Banana
BananaTests
Identity
Bundle Identifier
net.Ashton-W.Banana
Version
1.0
Build
1
Team
None
Deployment Info

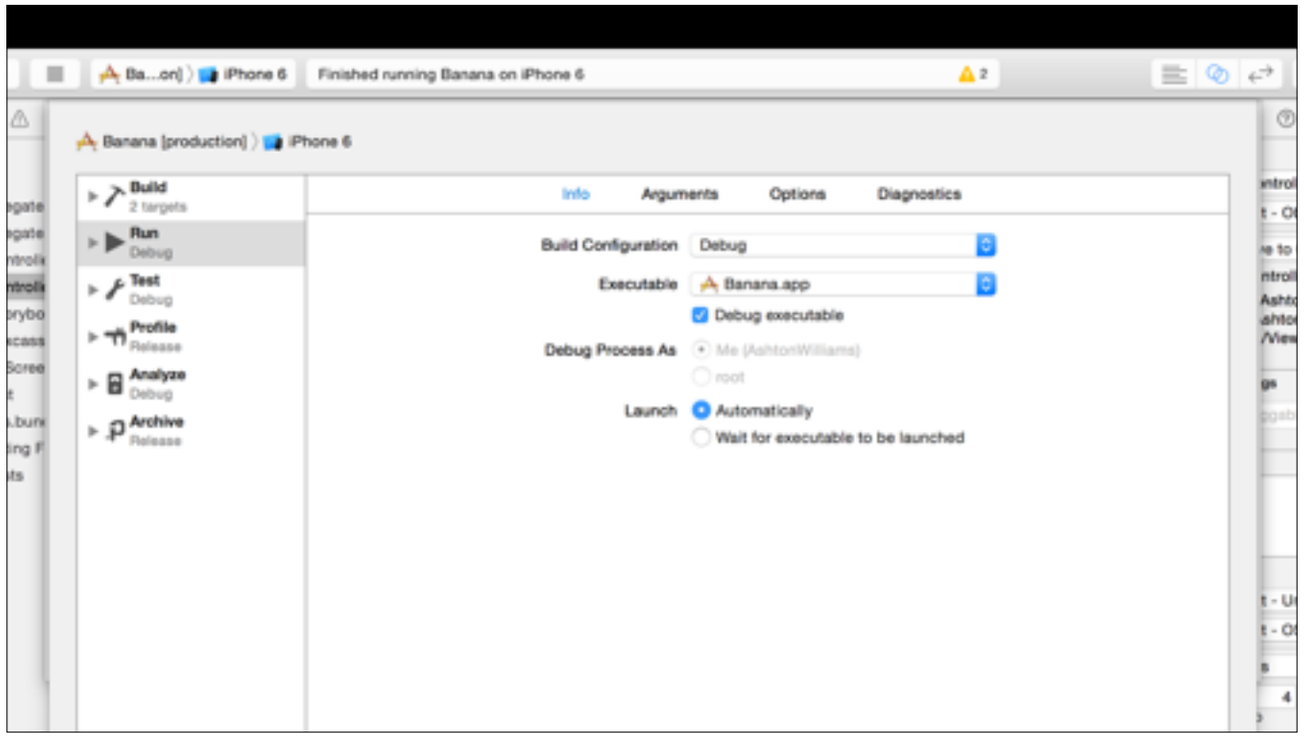

Banana
iPhone 6
Banana: Ready | Today at 10:53 pm
Autocreate schemes
Autocreate Schemes Now
Show
Scheme
Container
Shared
Banana
Banana project
Edit...
Close
Landscape Right


Banana
iPhone 6
Banana: Ready | Today at 10:54 pm
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Info
Arguments
Options
Diagnostics
Arguments Passed On Launch
No Arguments
Environment Variables
Name
Value
No Environment Variables
Expand Variables Based On
Duplicate Scheme
Manage Schemes...
Shared
Close


Banana
iPhone 6
Running Banana on iPhone 6
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Info
Arguments
Options
Diagnostics
Arguments Passed On Launch
-argument_preference Gatorade
Environment Variables
Name
Value
No Environment Variables
Expand Variables Based On
Banana
Duplicate Scheme
Manage Schemes...
Shared
Close

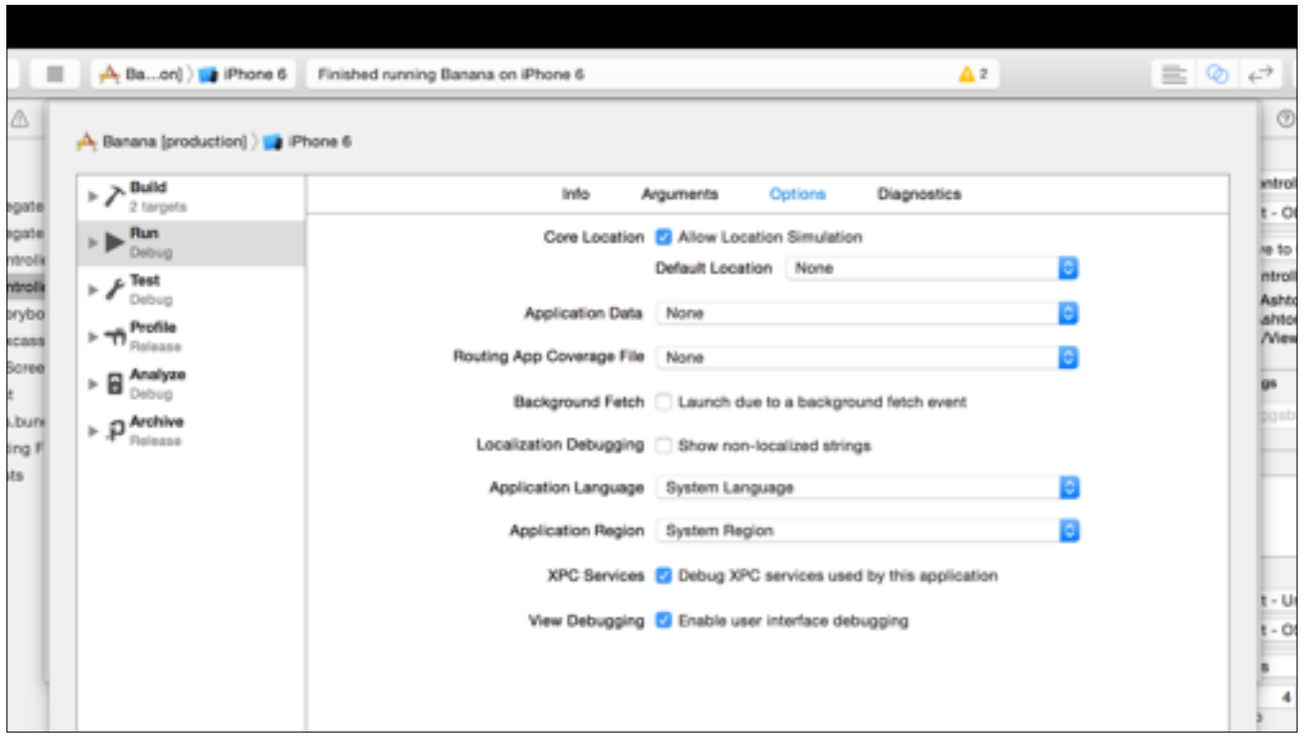
Arguments Passed On Launch

```
-key value
```

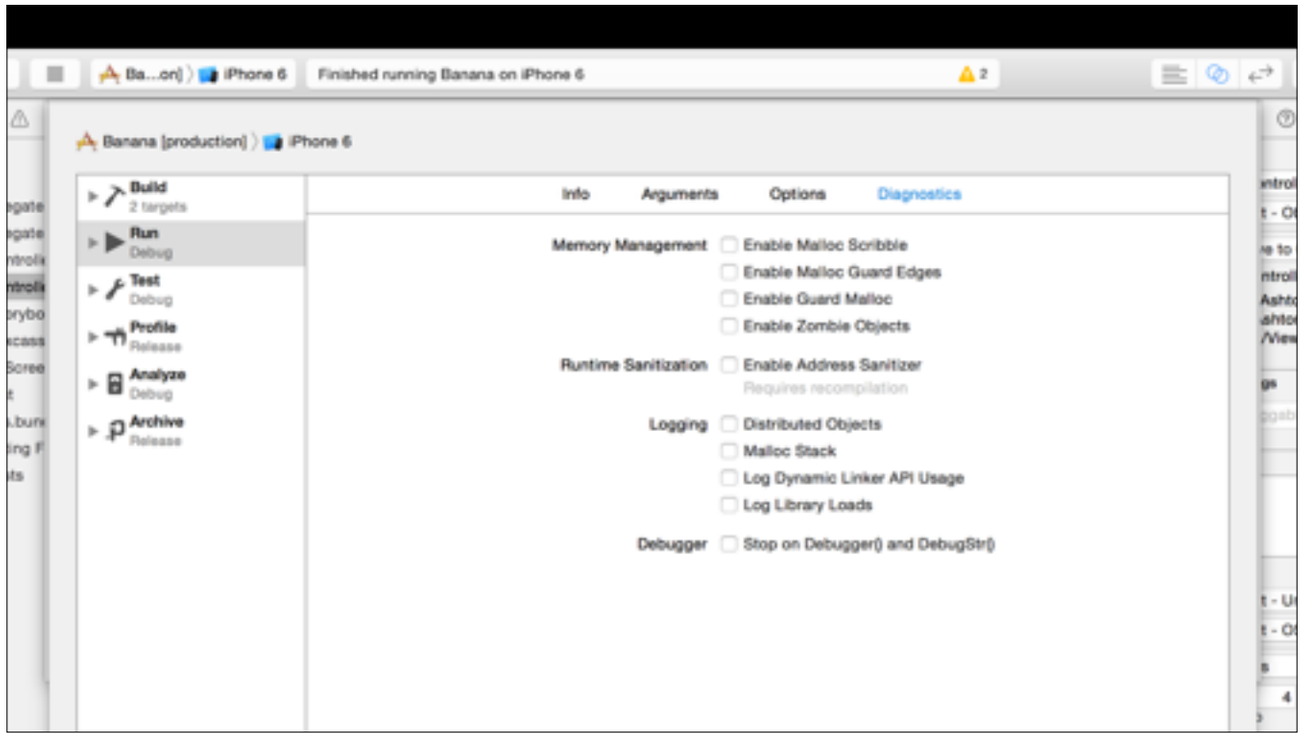


Other tabs.

INFO


Banana [production] iPhone 6
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Info
Arguments
Options
Diagnostics
Core Location
Allow Location Simulation
Default Location
None
Application Data
None
Routing App Coverage File
None
Background Fetch
Launch due to a background fetch event
Localization Debugging
Show non-localized strings
Application Language
System Language
Application Region
System Region
XPC Services
Debug XPC services used by this application
View Debugging
Enable user interface debugging

OPTIONS



Diagnostics

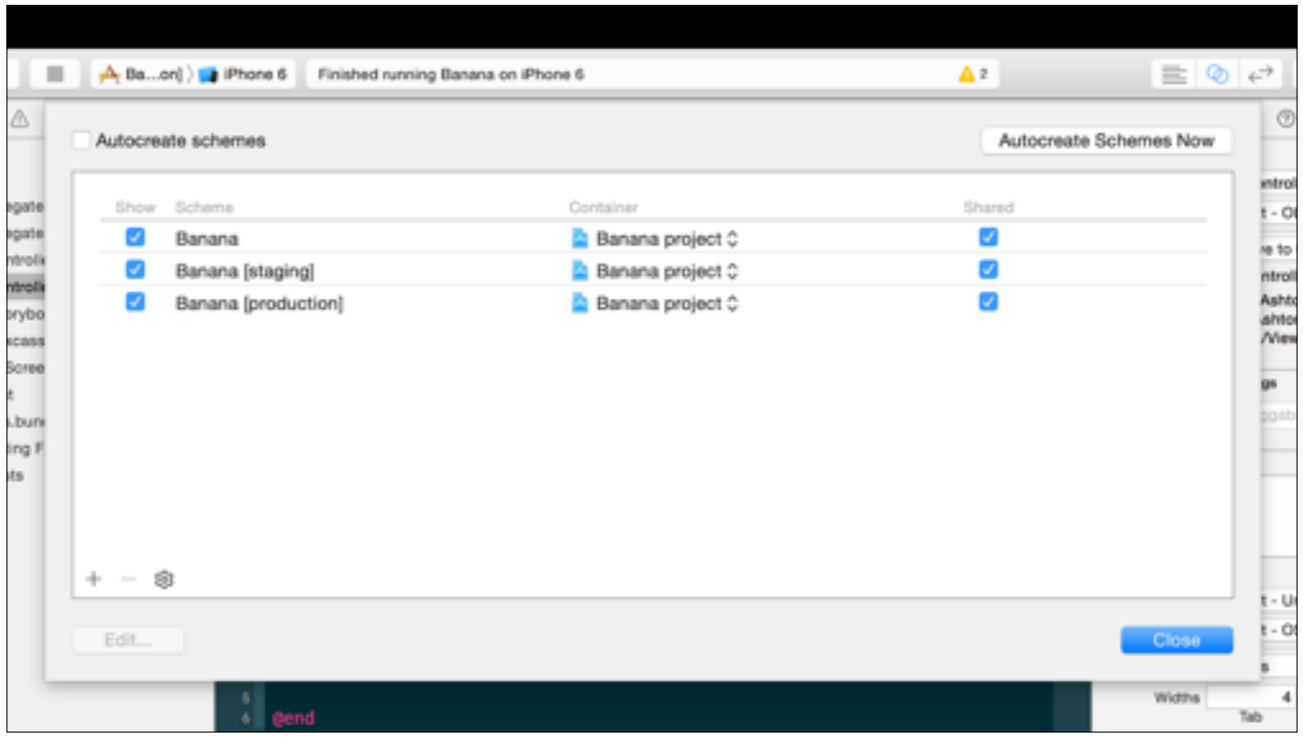
make a debugging scheme

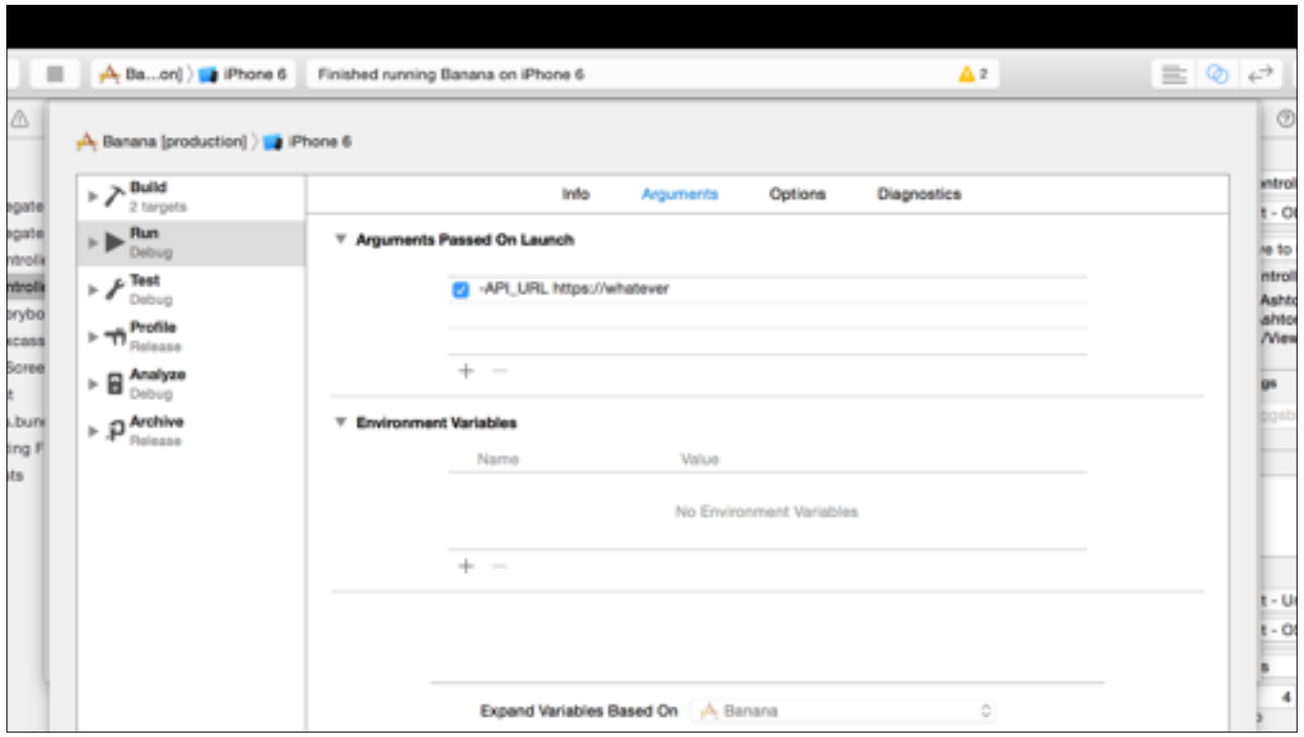
Schemes & User Defaults

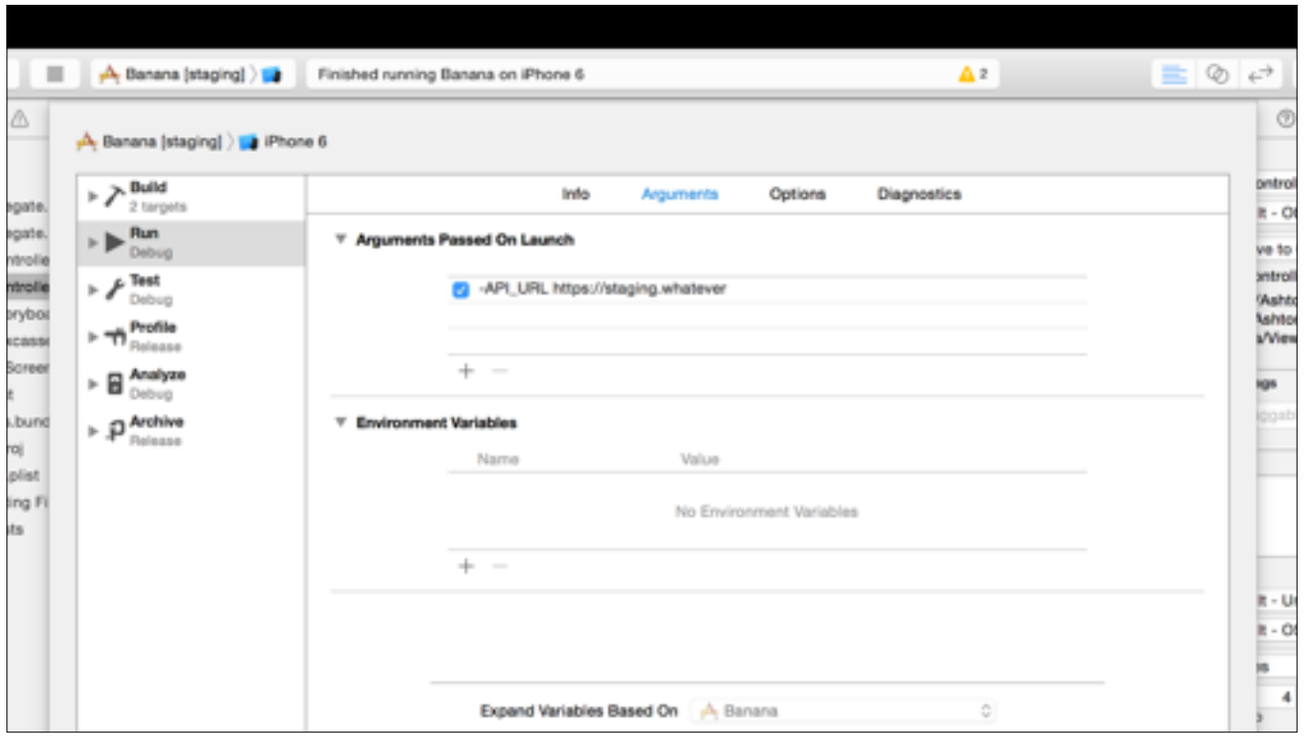
App Configuration

API_URL

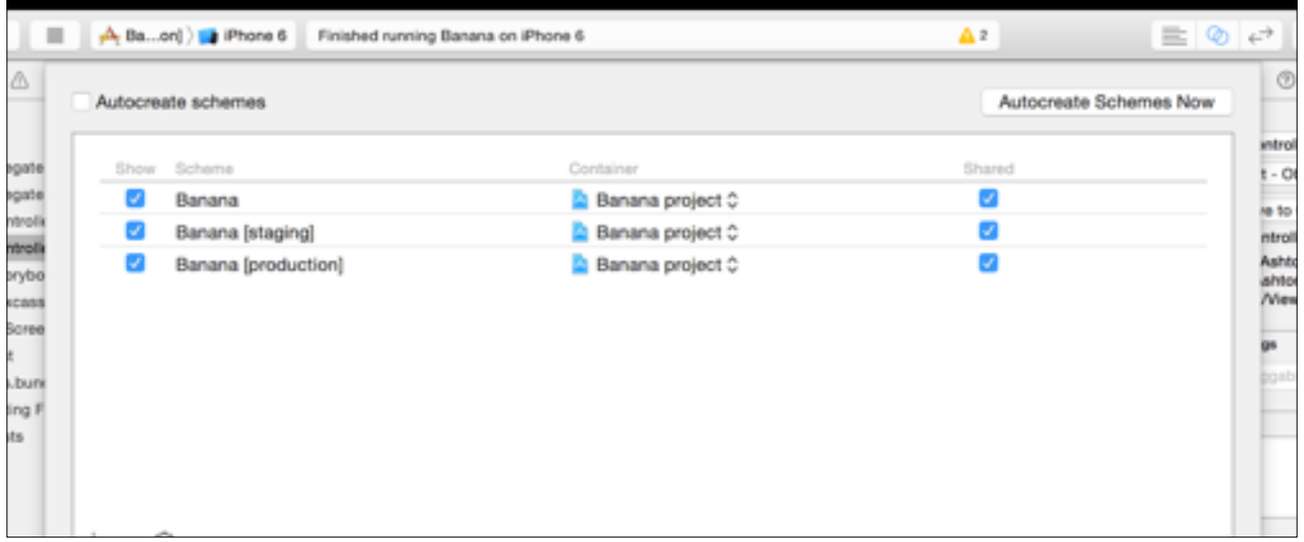
- Provide default in code
- Can set in Settings.app
- Override in Scheme


Ba...on) iPhone 6
Finished running Banana on iPhone 6
2
Autocreate schemes
Autocreate Schemes Now
Show
Scheme
Container
Shared
Banana
Banana project
Banana [staging]
Banana project
Banana [production]
Banana project
Edit...
Close
@end
Widths
4
Tab


Ba...on) iPhone 6
Finished running Banana on iPhone 6
2
Banana [production] iPhone 6
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Info
Arguments
Options
Diagnostics
Arguments Passed On Launch
-API_URL https://whatever
Environment Variables
Name
Value
No Environment Variables
Expand Variables Based On
Banana


Banana [staging]
Finished running Banana on iPhone 6
2
Banana [staging] iPhone 6
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Info
Arguments
Options
Diagnostics
Arguments Passed On Launch
-API_URL https://staging.whatever
Environment Variables
Name
Value
No Environment Variables
Expand Variables Based On
Banana

Sharing



Shared schemes

shared - include in git

unshared - ignore in git, go crazy here

Configuration & Framework

- Xcode UI Tests run in a seperate process
- Need network stubs for UI Tests
- Stubs only affect the process you stub in

- Put stubbing code in a Framework
- Inject it!
- Works outside of tests too

DEMO IF TIME

- Script based on
  http://tinyurl.com/RevealBuildPhase

Advanced Xcode

Configurations, Targets, and Schemes

Breakpoints are also awesome