iContinuousIntegration

Oleksandr Dodatko

Dnepropetrovsk,
Ukraine. 2012

Introduce yourselves please

I came to iOS from C++ and was disappointed

All code in a single project

Hard to reuse common functionality

No unit tests

No build automation

All other developers solved these problems years ago

IOS developers do not use modern engineering practices

Only 10% do unit testing

33% use nothing but Apple supplied components

28% copy-paste third-parties to their projects

Copy-paste style libraries

Regex Kit Lite

Touch XML

Touch JSON

Magical Record

Cocoa Lumberjack

and many more ...

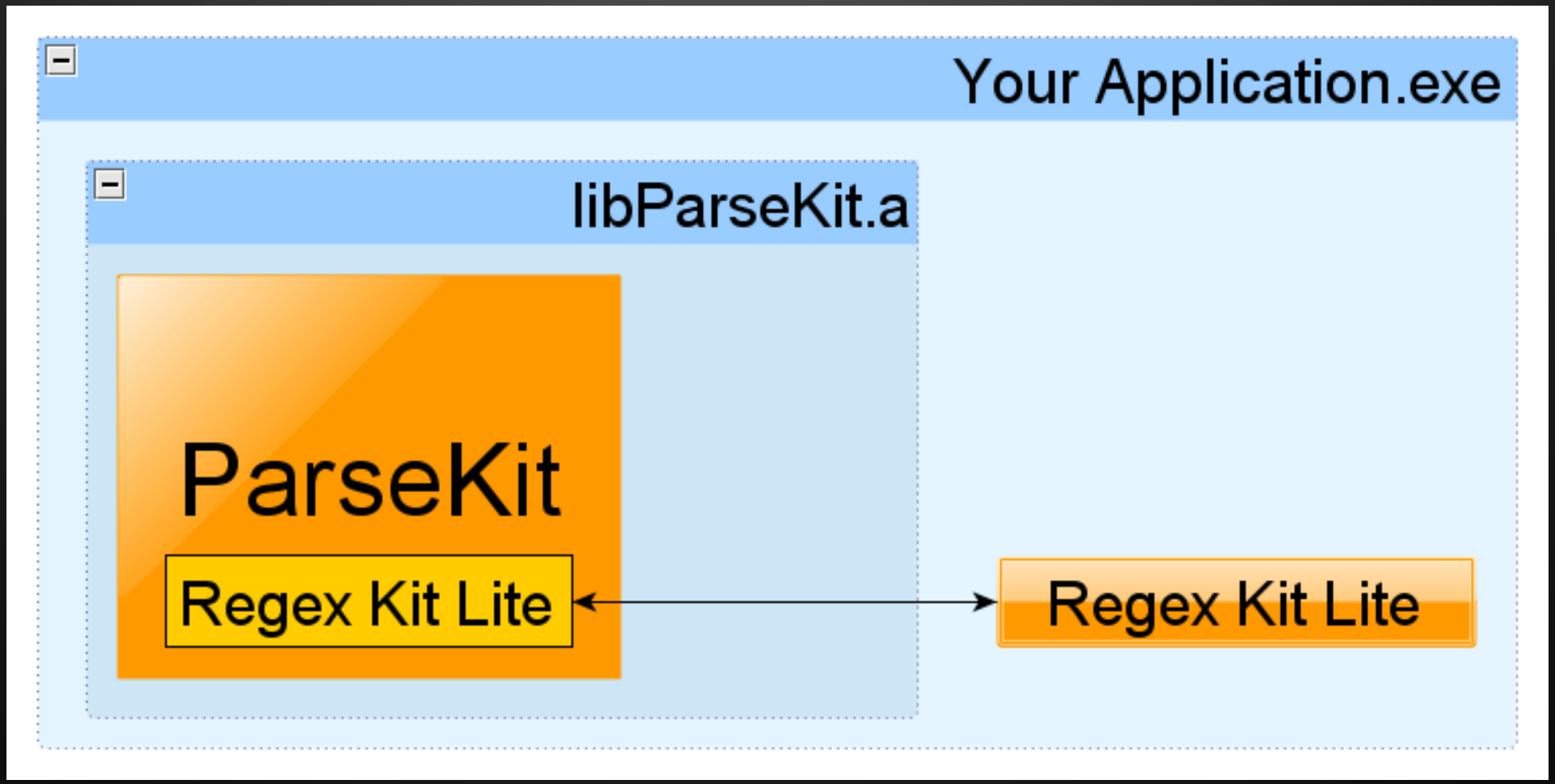
WHY ?

Magicalpanda : “I'm not sure what the benefits are to everyone...”

tonyxiao : “I don't really care to compile from source”

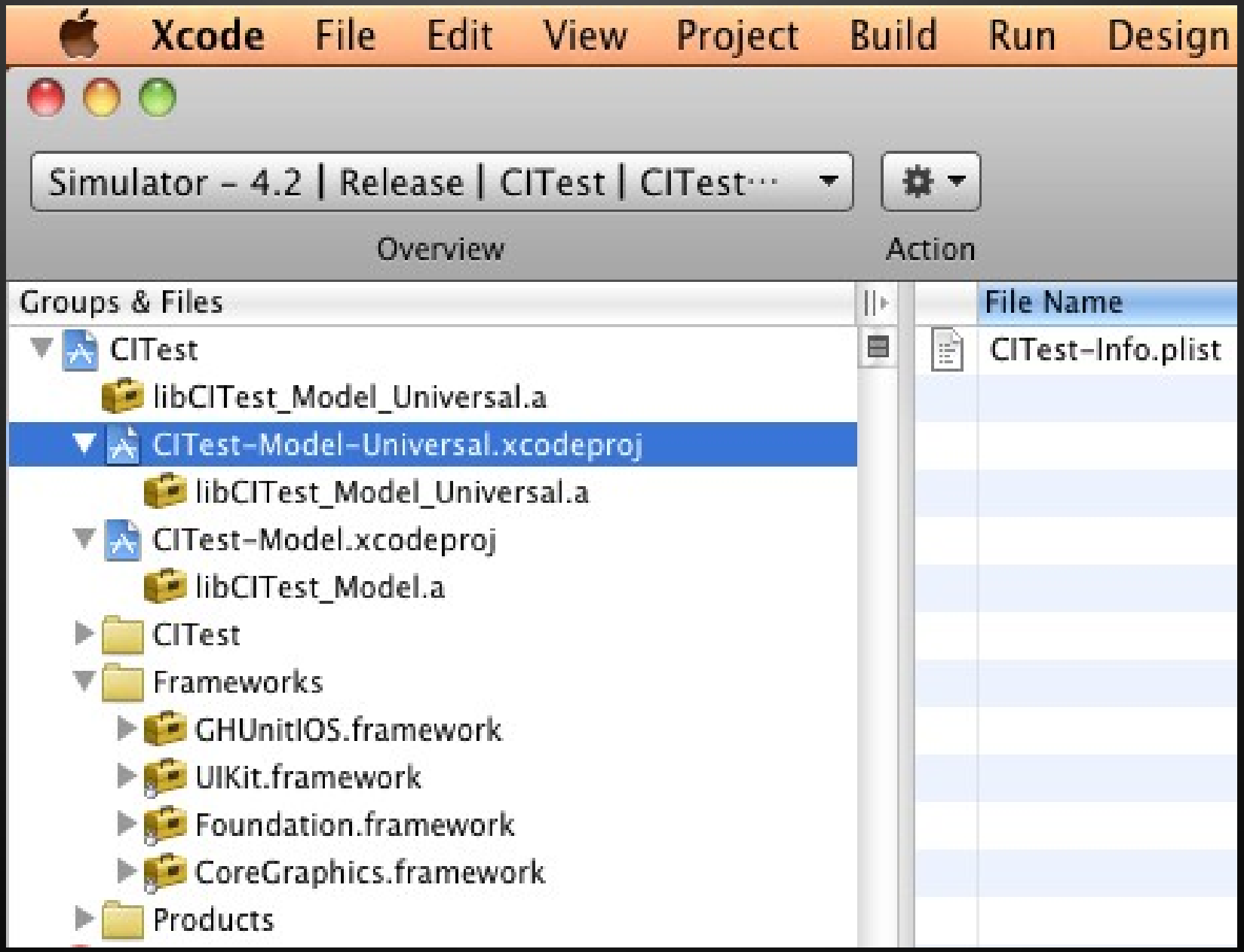
Magicalpanda : “as long as I can still use the Source Code approach, I'm ok with having a separate target in the project that dumps out a static library”

Ok. Why should we care?

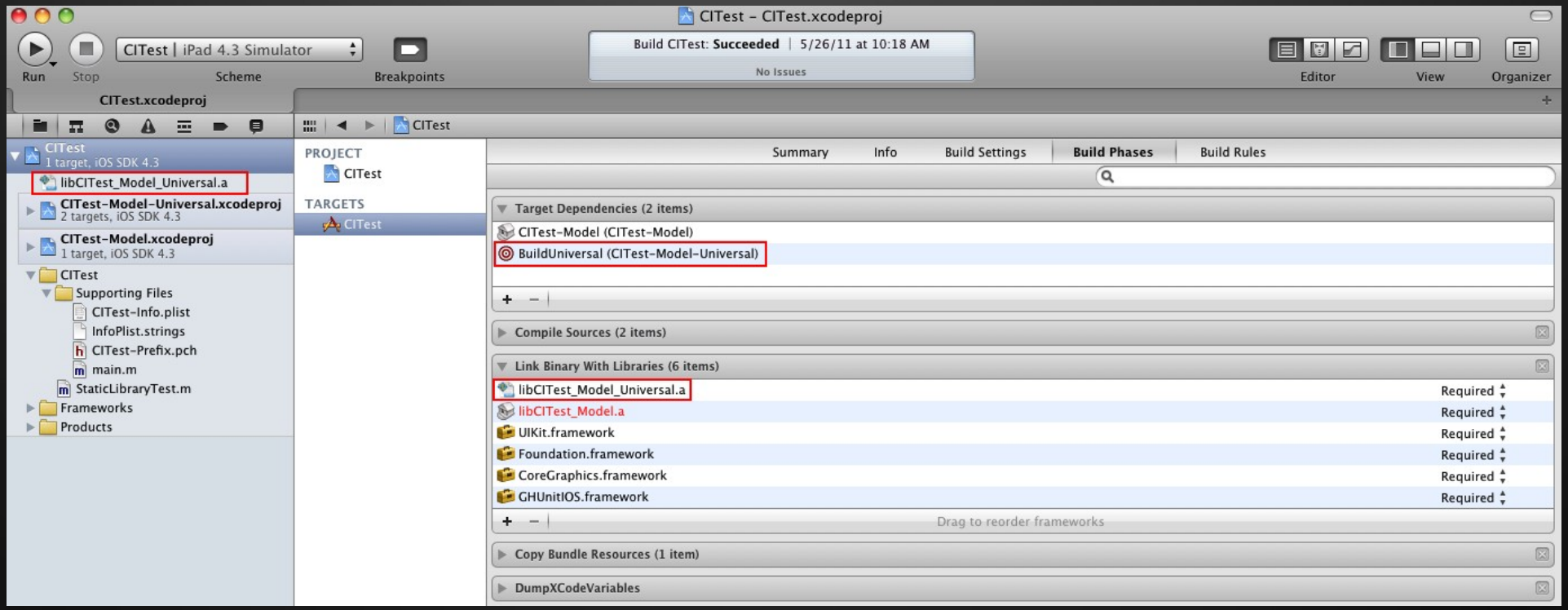


Use Static Libraries

Add a Library Sub-project



Set up Dependencies



Do not forget about the header path

Library Usage Benefits

Clear design

No linker conflicts

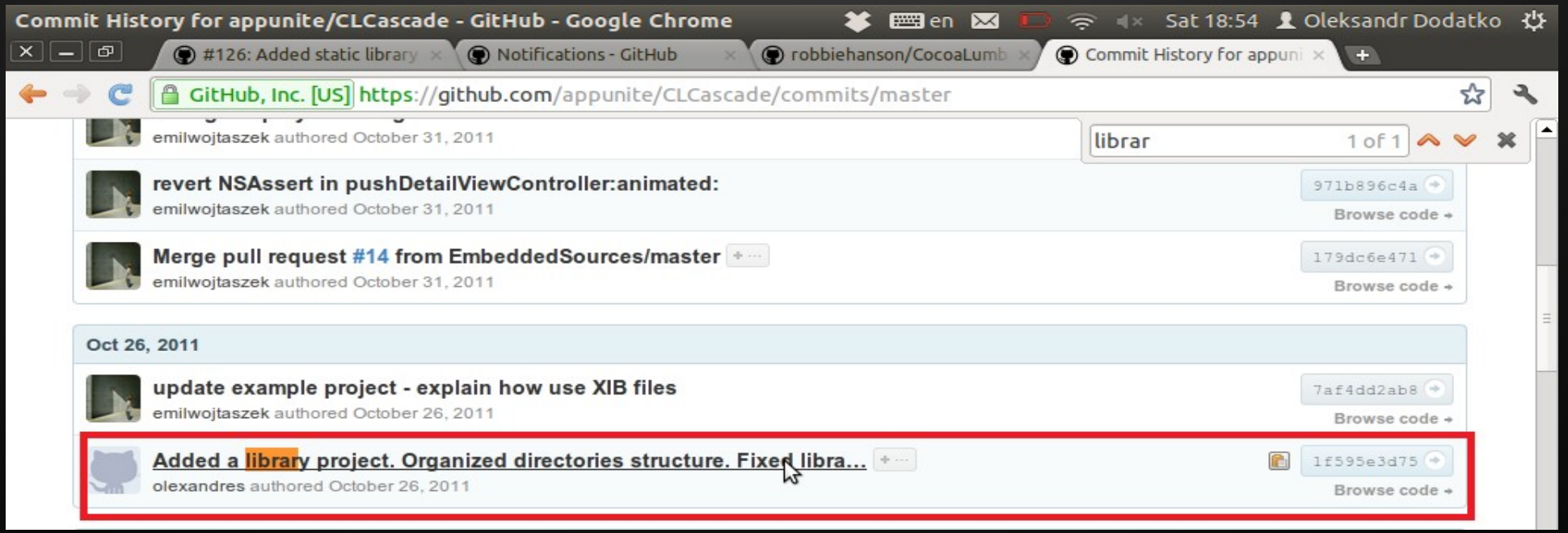
A better way to reuse code

Yes. It works!

“ BTW, thanks for your pull request. I have't had time to review it all yet, but most of it looks good. “

Saul Mora. Founding Panda. saul@magicalpanda.com

appunite / CLCascade has accepted our patch



Find our forks at
github.com/EmbeddedSources

Let's discuss unit tests

I recommend ...

Easy to debug failed tests

Easy to use files with test data bundles

Generates jUnit compatible reports

GHUnit : unit test is an iOS application

Unit test life cycle

Pass test data files to the test program

Launch the test program

Publish test reports

iOS sandbox makes things complicated

Application Launch Demo

Launching app without xCode

```
iphonesim launch
 "$DEPLOYMENT_DIR/CITest.app"
 4.2
 ipad
```

NOTE : Use only FULL PATH to the app
as shown above

Collecting Test Results

TEMP_DIR=$(/usr/bin/getconf DARWIN_USER_TEMP_DIR)

All Test results are here :

$TEMP_DIR/test-results

Before you run a test...

killall -KILL -c "iphonesim"

killall -KILL -c "iPhone Simulator"

GHUNIT_AUTORUN

WRITE_JUNIT_XML

GHUNIT_AUTOEXIT

Unit tests reduce risks due to early error discovery

Jenkins job should build and deploy in one click

There should be no interaction with the user

Building without xCode

xcodebuild -project CITest.xcodeproj
-sdk iphonesimulator4.3
-configuration Release
-target CITest
-parallelizeTargets
clean build

Deliver your beta builds as *.ipa files using TestFlight

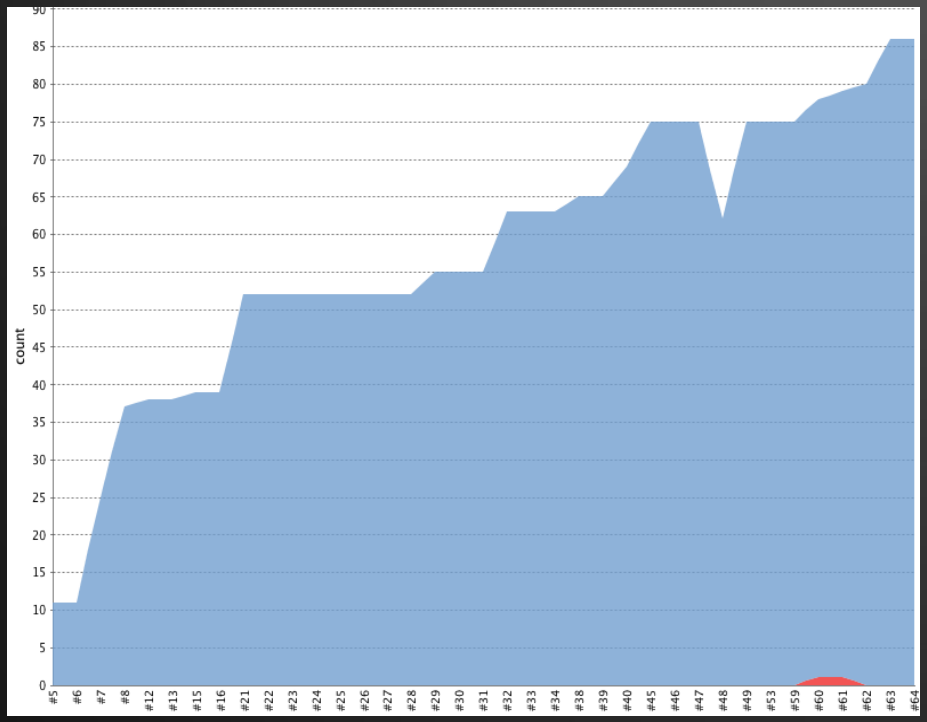
Creating Installable *.ipa File

```
/usr/bin/xcrun -sdk iphoneos PackageApplication
-v "${BUILD_DIR}/Release-iphoneos/CITest.app"
-o "${DEPLOYMENT_DIR}/CITest.ipa"
```

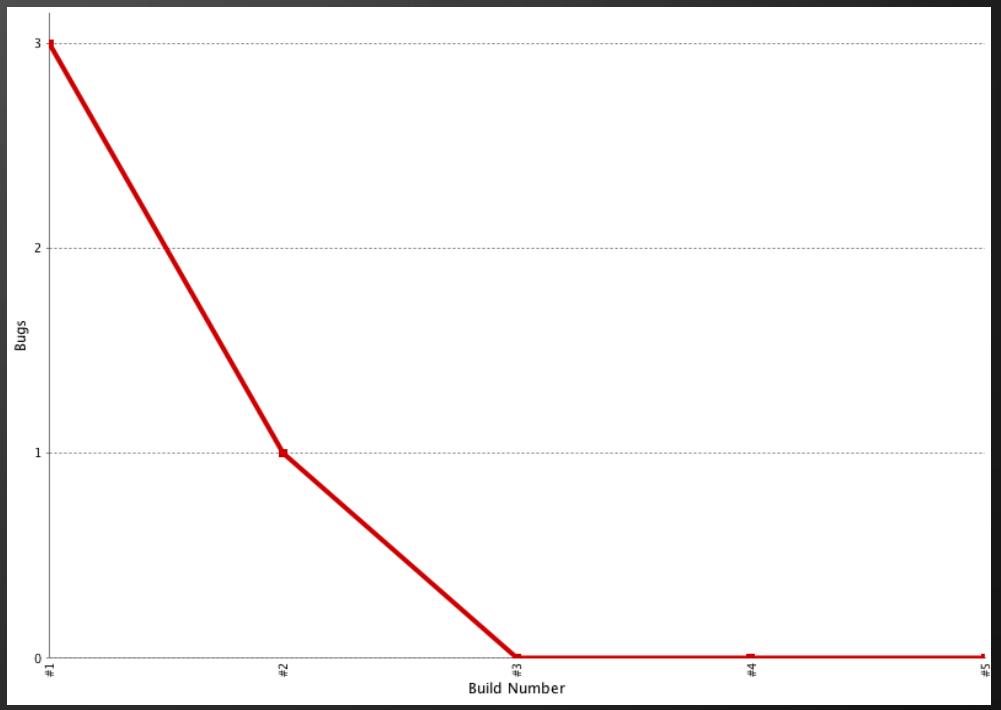
Create one more target for static analyzer in each of your projects

You'll get

Unit test trend



Clang trend



Oh, One more thing...

Custom iOS frameworks – a more native way to reuse the code

A framework is

A special kind of NSBundle

A directory with a special structure

A fat universal binary for BOTH the device AND the simulator

Framework directory structure

MyFramework.framework
|-------> MyFramework (universal static library)
|------->Headers (symlink)
|------->Resources (symlink, optional)
|------->Versions Actual files should be here
|-----------|
|-----------|--->A (all symlinks lead here)

Creating a universal binary

lipo -create

-output <OUTPUT_PATH>

Thank you for your time

You apply	You receive
Unit tests	Early errors discovery
Automated builds	Reduced project risks
TestFlight deployment	High application quality

Contacts

Oleksandr Dodatko

mail/jabber : dodikk88.reg@gmail.com

Skype : alexander.dodatko.work@skype.com

Github page :

https://github.com/dodikk

https://github.com/EmbeddedSources