Contribute to Swift.org

A Personal Experience

Swift

- Announced at WWDC2014
    - → Darwin Platforms
- Open sourced at WWDC2015
    - → Ubuntu
- The community is keeping expanding it
    - → BSD
    - → Android
    - → Cygwin

Projects Under Swift.org

- Compiler and Standard Libraies
- Package Manager
- Core Libraries
- REPL, Debugger and Playgrounds

swift-corelibs-foundation

- An implementation of Foundation without Objective-C runtime dependency
- The Foundation.framework for the rest of the platforms
    - Also referenced as → SwiftFoundation
    - The Objective-C Version → DarwinFoundation
- Currently, only interfaces are shared between the two
    - e.g. The Foundation interface update applied in Swift 3
- Projects can use CorelibsFoundation with Swift Package Manager
    - On Darwin Platforms, Swift projects will still use the DarwinFoundation

My Path to Contribute

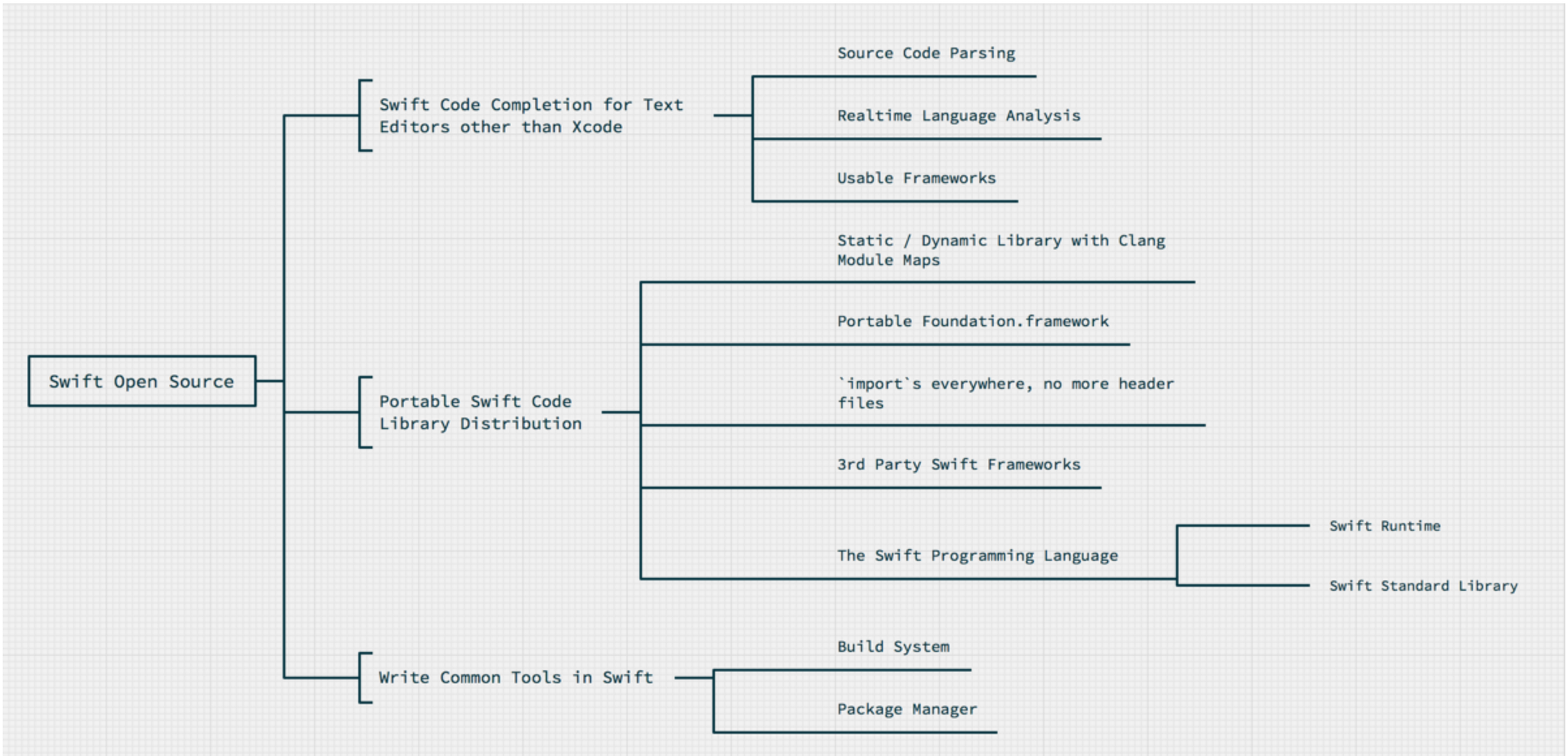
#16 of 194 / 18 commits / 806 ++ / 227 --

2015.06 - WWDC2015

- Apple announced the open source decision
- I was there at Presidio, 3rd Floor, Moscone West, San Francisco
- GOT REALLY EXCITED!

2015.12 – github.com/apple/swift

- Drew a road map
- Spent a whole day fantasizing
- There was a lot of `NSUnimplemented()`


Swift Open Source
Swift Code Completion for Text Editors other than Xcode
Source Code Parsing
Realtime Language Analysis
Usable Frameworks
Portable Swift Code Library Distribution
Static / Dynamic Library with Clang Module Maps
Portable Foundation.framework
`import`s everywhere, no more header files
3rd Party Swift Frameworks
The Swift Programming Language
Swift Runtime
Swift Standard Library
Write Common Tools in Swift
Build System
Package Manager

But I was really busy.

Back then I was working for a startup as the only iOS and macOS Developer

2016.03 - Changed my job

- Have some free time
- Starter Bugs on Swift Community JIRA
    - Fixed SR-865 (About NSUUID.UUIDString)
    - First Pull Request
- Searched again for `NSUnimplemented()`
    - A lot less than 2015.12
    - Formatters
    - NSAttributedString

Picked one API to work on

- Formatters
  - The underlying implementations depends on libicu
  - Not familiar with libicu
- NSAttributedString
  - CFAttributedString and CFRunArray
  - Easy to get started

Implemented (part of) NSAttributedString

- Basic Initializers
- Is-A Relationship Between NSAttributedString and CFAttributedString
- More Methods...

2016.06 - Met the merging guys

- WWDC2016 - Swift Foundation Labs
- I was there asking questions
    - About dynamic loading NSBundles in Swift

Hey, you are the guy on github!

Actually, how can I implement `enumerateAttributes`?

Let's see the code in DarwinFoundation.

The merging guy said.

2016.09 - Released with Swift 3

- First official release of swift-corelibs-foundation
- Planning more for the Swift 4 release
    - Finishing up NSAttributedString/NSMutableAttributedString

The C.C.I.T.P Loop

Checkout -> Compile -> Implement -> Test -> Pull Request

Checkout

- $ git clone ...apple/swift
- $ ./swift/utils/update-checkout
- path/to/swift.org
  - ├─── clang
  - ├─── lldb
  - ├─── llvm
  - ├─── swift
  - ├─── swift-corelibs-foundation
  - ├─── swift-corelibs-xctest

Compile on macOS

- Swift.org Trunk Development Toolchain
- Xcode
- Open swift-corelibs-foundation/Foundation.xcworkspace
- Build scheme `SwiftFoundation`

Compile on Ubuntu

- $ ./swift/utils/build-script -R --xctest --foundation -j1
  - With `-R -j1` about 8GB of RAM is required at the linking phase
  - With `-r -j1`, even 12GB is not enough
  - llvm 3.4 (default to Ubuntu 14) won't work, use at least llvm 3.6
  - The building artifacts will be several GBs
- $ cd swift-corelibs-foundation/ && ninja build
  - Only builds corelibs-foundation
  - Faster and less resource intensive
  - `-j8`

Implement

- The taxonomy of types
    - Swift only
    - CF only
    - Is-a relationship
    - Has-a relationship
- Do things without Objective-C Runtime support
- Consider cross platform situations

Implement

- Syntax Highlighting
  - Xcode on macOS
  - Text Editor with Swift plugins on any platforms
- Code Completion
  - Xcode on macOS

Test on macOS

- Same as the previous steps
- Xcode is your friend
- Build the SwiftFoundation scheme
- Run Test with the TestFoundation scheme

Test on Ubuntu

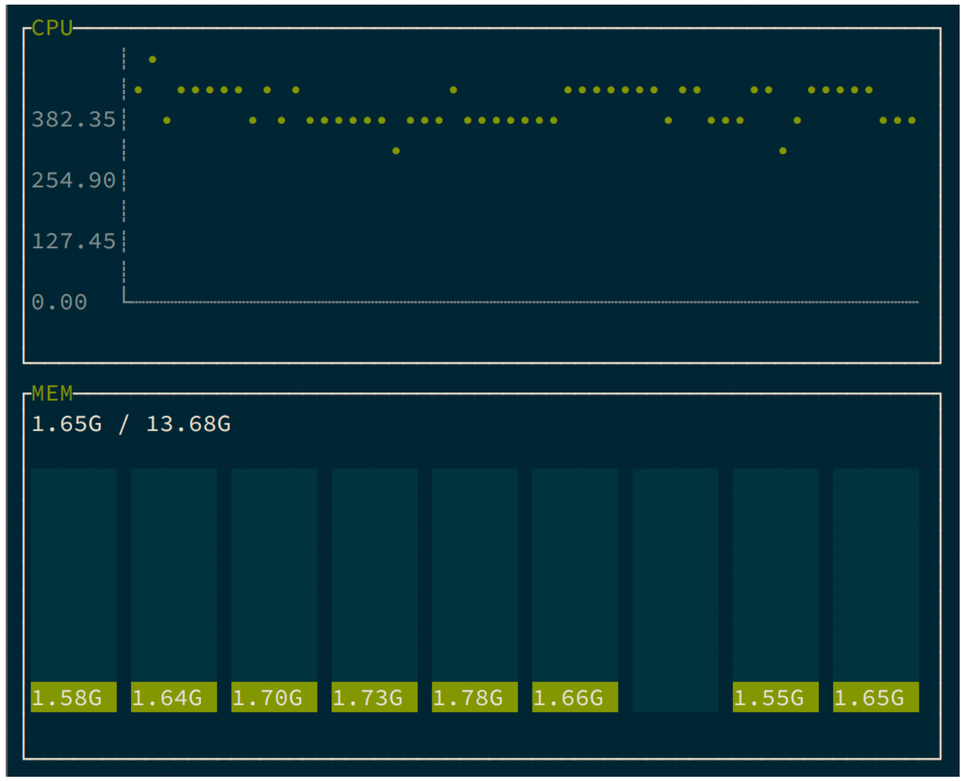
- The compiled swift binaries must be available
- $ cd swift-corelibs-foundation/ && ninja test
- $ LD_LIBRARY_PATH=./build/Ninja-ReleaseAssert/foundation-linux-x86_64/Foundation/:./build/Ninja-ReleaseAssert/xctest-linux-x86_64/ ./build/Ninja-ReleaseAssert/foundation-linux-x86_64/TestFoundation/TestFoundation

Compile and Test for Ubuntu on macOS?

Yes, with Docker.

Swift.org on Docker

- Edit in OS X with Xcode
- Build and test in containers
- $ docker pull eyeplum/swift-foundation-dev
    - This Docker image has the latest development env built-in
    - llvm-3.9, cmake, ninja, etc.
- $ docker run -it -v /path/to/swift.org/:/swift.org eyeplum/swift-foundation-dev /bin/bash



VirtualBox -> Hypervisor.framework

Pull Request

- All implementations should be paired with tests
- Follow the Swift.org API guideline
- Wait for code review
- @swift-ci please test and merge
    - If the test fails, don't panic
    - Check the jenkins build log to find out the issue
    - Sometimes it's unrelated to the pull request
- Get merged! 🎉

Some Personal Thoughts

- The open side of Apple
    - It was really hard to peek inside anything inside the 'Apple stuff'
    - Now it is possible
- Working on the wheel makers' problems
    - Helps you understand the 'wheels' better
    - Other people may benifit from your work
- Have Fun!

Thank you!

@eyeplum