

Contributing to Swift

by @neilkimmett



Contributing to Swift

- a little revision
- story of my contribution
- ways you can contribute
- gnarly details



```
struct CGPoint {  
    var x: CGFloat  
    var y: CGFloat  
}
```

```
struct CGSize {  
    var width: CGFloat  
    var height: CGFloat  
}
```

```
struct CGRect {  
    var origin: CGPoint  
    var size: CGSize  
}
```

```
struct CGPoint {  
    static var zero: CGPoint  
}  
  
x: 0  
y: 0  
  
struct CGSize {  
    static var zero: CGSize  
}  
  
width: 0  
height: 0  
  
struct CGRect {  
    static var zero: CGRect  
}  
  
origin: {0, 0}  
size: {0, 0}
```

```
let view = UIView()  
...  
view.frame.origin = .zero
```

```
struct UIEdgeInsets {  
    var top: CGFloat  
    var left: CGFloat  
    var bottom: CGFloat  
    var right: CGFloat  
}
```

```
let view = UIView()  
...  
view.layoutMargins = .safeEdgeInsetsZero
```





Neil Kimmitt 2:02 PM



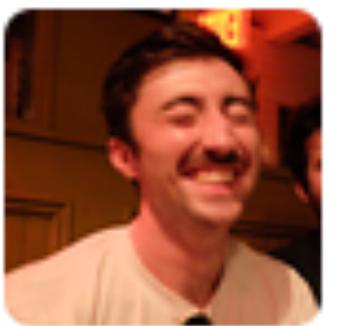
aside: why is there `CGRect.zero` but no `UIEdgeInsets.zero`

2:03 ★ whats that all about



Soroush Khanlou 2:03 PM

that's wack



Neil Kimmitt 2:05 PM

can fix with

```
extension UIEdgeInsets {  
    static var zero: UIEdgeInsets {  
        return UIEdgeInsetsZero  
    }  
}
```

but still

cmon Apple



Soroush Khanlou 2:05 PM

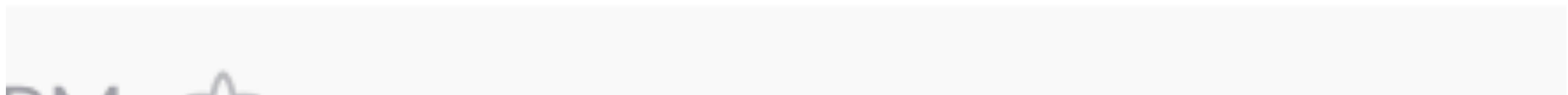
yeah this one's on apple

if you're that kind of person you could make a radar

WPA C07

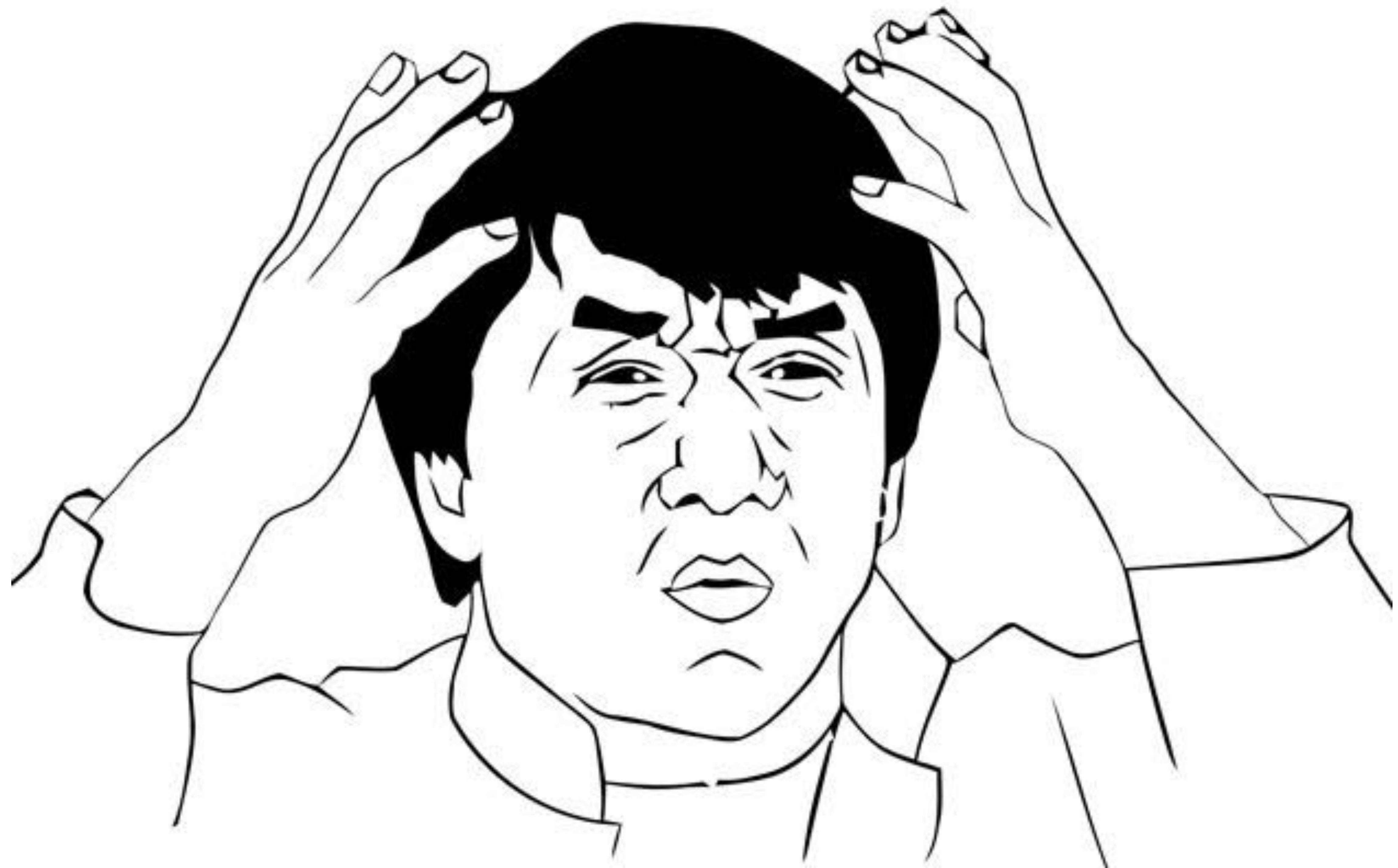
apple

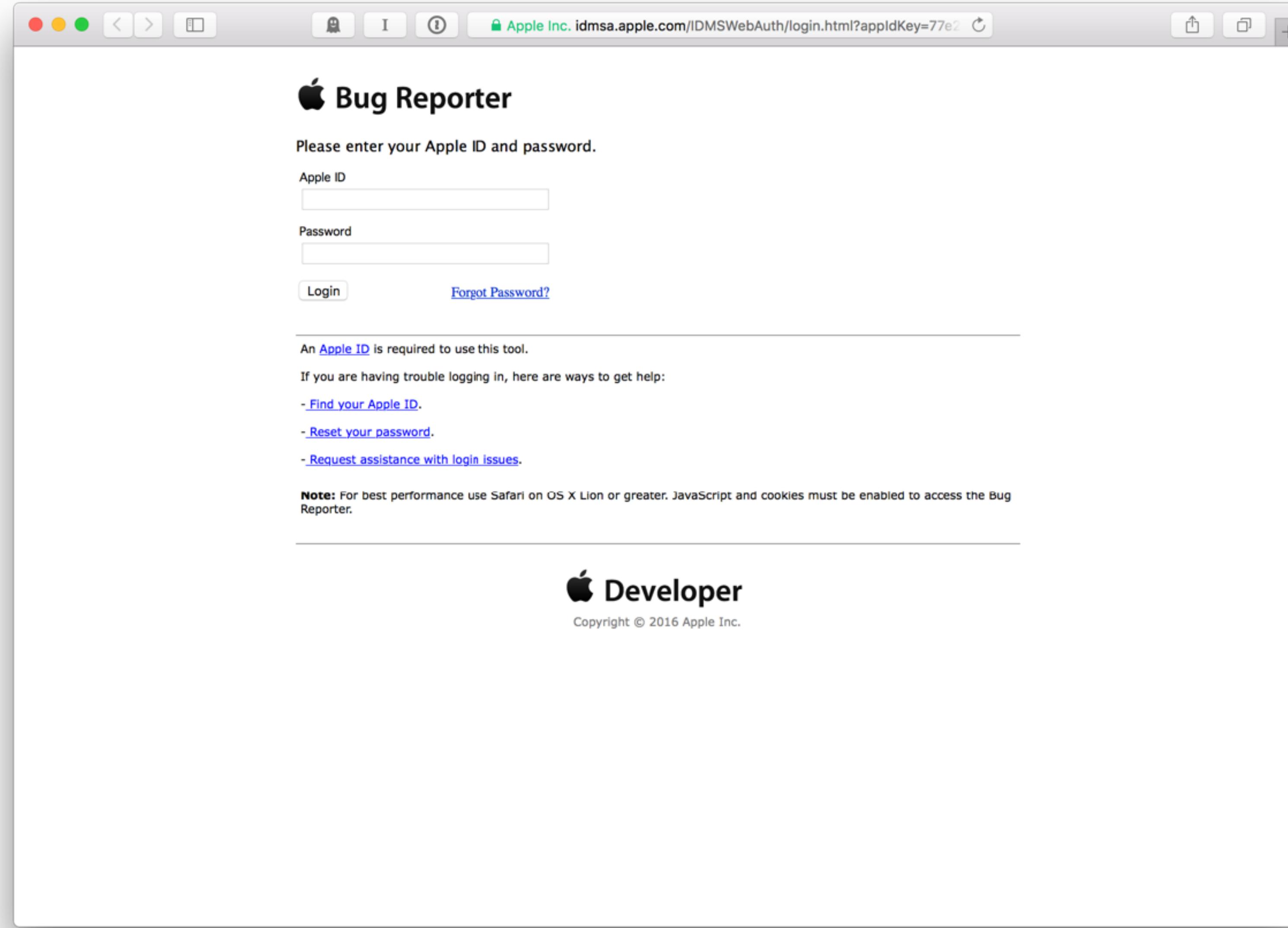
if person you could make a radar



u could make a radar

a radar



A screenshot of a Mac OS X desktop showing a Safari browser window. The window title is "Apple Inc. idmsa.apple.com/IDMSWebAuth/login.html?appIdKey=77e2". The main content is the "Bug Reporter" login page. It features the Apple logo and the title "Bug Reporter". A message says "Please enter your Apple ID and password." Below are two text input fields: "Apple ID" and "Password". Underneath each field is a small placeholder text: "Apple ID" has "appleid@yourdomain.com" and "Password" has "password123". To the right of the fields are two buttons: "Login" and "Forgot Password?". Below the fields is a note: "An [Apple ID](#) is required to use this tool." Another note follows: "If you are having trouble logging in, here are ways to get help:" with three links: "Find your Apple ID.", "Reset your password.", and "Request assistance with login issues.". A note at the bottom states: "Note: For best performance use Safari on OS X Lion or greater. JavaScript and cookies must be enabled to access the Bug Reporter." At the bottom center is the "Developer" logo with "Copyright © 2016 Apple Inc.".

<http://bugreport.apple.com/>

Apple Inc. bugreport.apple.com/web/newproblemform

Bug Reporter

Report a Bug

Title
Enter a descriptive title for your bug report.

Summary
Provide a detailed summary of the issue you are reporting.

How would you categorize this issue?

What are you seeing?
Choose a product category for your bug report.

What kind of issue is this?
Select the type of issue you are experiencing.

What steps can I take to reproduce this issue?

Steps to Reproduce

Tech Note/Q&A

Cancel **Save** **Submit**

Choose Product
What are you seeing an issue with?

Search Products

- OS and Development
- iOS + SDK**
For issues with iOS utilities and features
- macOS + SDK
For issues with macOS utilities and features (file system, graphics, networking, security, scripting, etc.)
- tvOS + SDK
For issues with tvOS apps and features
- watchOS + SDK
For issues with watchOS apps and features
- Developer Tools
For all Xcode issues (including Xcode tools - Interface Builder, Instruments, compilers, simulators, command line tools, Swift Programming language, etc.)
- Documentation
For documentation issues (misinformation, broken links, typos, etc.)
- iTunes Connect
For all issues regarding iTunes Connect (a tool that allows developers to submit and manage apps for distribution).
- Sample Code
For all Sample Code issues (broken links, missing images, misinformation, typos, etc.)
- Server
For issues with Server administration and features (backups, calendaring, file sharing, user profiles, etc.)
- Tech Note/Q&A

Apple Inc. bugreport.apple.com/web/newproblemform

Bug Reporter

Neil Kimmett | ?

What steps can we take to reproduce this issue?

Steps to Reproduce

Please provide a step-by-step set of instructions to reproduce the problem (if possible)

Expected Results

Describe what you expected to happen after completing the steps above.

Observed Results

Describe what actually happened after completing the steps above.

Configuration optional

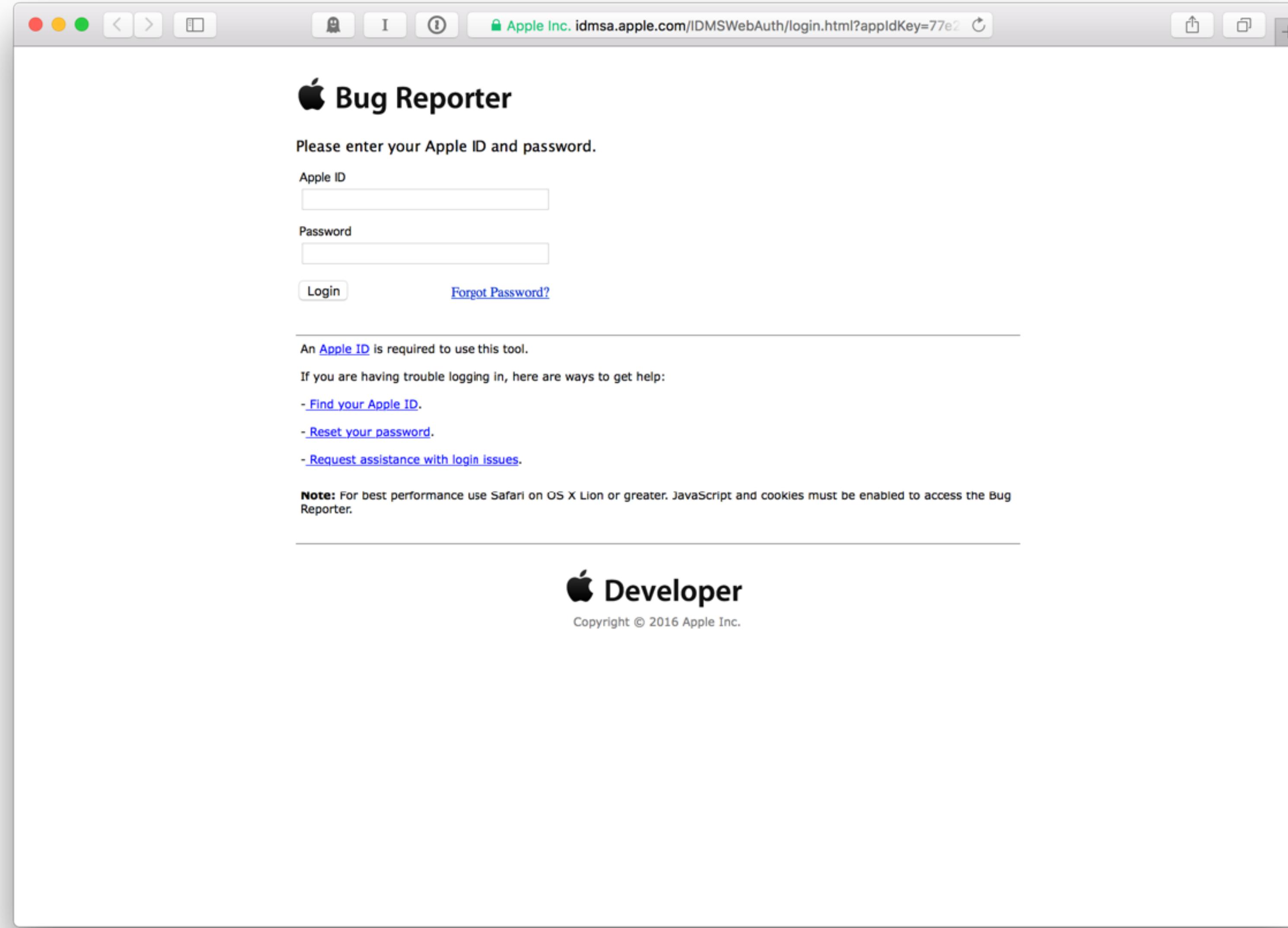
Describe the circumstances where this does or does not occur, including hardware and software configurations.

Cancel **Save** **Submit**

Apple Developer Relations

April 9 2015, 11:15 PM

Engineering has determined that your bug report is a duplicate of another issue and will be closed.

A screenshot of a Mac OS X desktop showing a Safari browser window. The window title is "Apple Inc. idmsa.apple.com/IDMSWebAuth/login.html?appIdKey=77e2". The main content is the "Bug Reporter" login page. It features the Apple logo and the title "Bug Reporter". A message says "Please enter your Apple ID and password." Below are two text input fields: "Apple ID" and "Password". Underneath each field is a small placeholder text: "Apple ID" has "appleid@yourdomain.com" and "Password" has "password123". To the right of the fields are two buttons: "Login" and "Forgot Password?". Below the fields is a note: "An [Apple ID](#) is required to use this tool." Another note follows: "If you are having trouble logging in, here are ways to get help:" with three links: "Find your Apple ID.", "Reset your password.", and "Request assistance with login issues.". A note at the bottom states: "Note: For best performance use Safari on OS X Lion or greater. JavaScript and cookies must be enabled to access the Bug Reporter." At the bottom center is the "Developer" logo with "Copyright © 2016 Apple Inc.".

<http://bugreport.apple.com/>



THAT GIVES ME AN IDEA.

A screenshot of a web browser window showing the homepage of [Swift.org](https://swift.org). The page has a clean, modern design with a white background. On the left side, there is a vertical navigation menu with a red and white logo for Swift. The menu items are: ABOUT SWIFT, BLOG, DOWNLOAD, GETTING STARTED, DOCUMENTATION, MIGRATING TO SWIFT 3, SOURCE CODE, COMMUNITY, CONTRIBUTING, CONTINUOUS INTEGRATION, PROJECTS, COMPILER AND STANDARD LIBRARY, PACKAGE MANAGER, CORE LIBRARIES, and REPL AND DEBUGGER. The main content area on the right features a large heading "Welcome to Swift.org". Below it, a sub-headline says "Swift is now open source!". A paragraph explains the excitement of this new chapter, mentioning its rapid growth and safety by design. Another paragraph discusses the educational value and broad applicability of Swift for students. A final section from the Swift Team expresses their welcome to the community and their goal of building a better programming language for everyone.

Welcome to Swift.org

Swift is now open source!

We are excited by this new chapter in the story of Swift. After Apple unveiled the Swift programming language, it quickly became one of the fastest growing languages in history. Swift makes it easy to write software that is incredibly fast and safe by design. Now that Swift is open source, you can help make the best general purpose programming language available everywhere.

For students, learning Swift has been a great introduction to modern programming concepts and best practices. And because it is now open, their Swift skills will be able to be applied to an even broader range of platforms, from mobile devices to the desktop to the cloud.

Welcome to the Swift community. Together we are working to build a better programming language for everyone.

– The Swift Team

<https://swift.org>

The screenshot shows the GitHub repository page for `apple/swift`. The page includes the repository name, a search bar, navigation links for Pull requests, Issues, and Gist, and a header with Watch (2,223), Star (32,093), Fork (4,418) buttons. Below the header are tabs for Code, Pull requests (99), Pulse, and Graphs. A section titled "The Swift Programming Language" with a link to <https://swift.org/> follows. Key statistics are displayed: 39,327 commits, 21 branches, 36 releases, and 358 contributors. A "Clone or download" button is prominent. The main content area lists recent commits from the `master` branch, starting with a commit from `swift-ci` merged via pull request #3342. Other commits are listed under categories like `.github`, `apinotes`, `benchmark`, etc., with their respective descriptions and timestamps.

Commit Details	Time Ago
swift-ci committed on GitHub Merge pull request #3342 from practicalswift/typo-fixes-20160705	Latest commit be62dd7 a minute ago
.github Add a missing "the" to the pull request template	8 days ago
apinotes [APINotes] NSPinterFunctionsOptions 0-valued members shouldn't be [].	5 days ago
benchmark [gardening] Use '{let,var} c: C' instead of '{let,var} c : C'	4 days ago
bindings/xml Nesting parameter/returns/throws doc comments for closure parameters	3 months ago
cmake [cmake] Refactor check_working_std_regex into its own cmake file Swif...	8 hours ago
docs [docs] LibraryEvolution: add "let vs. var" as a planned proposal.	4 days ago
include Sema: Explicitly set interface type on all AbstractFunctionDecls	4 hours ago
lib Sema: Explicitly set interface type on all AbstractFunctionDecls	4 hours ago
stdlib Merge pull request #3322 from lucianomarisi/chore/sequence-documentati...	2 days ago
test [gardening] Fix recently introduced typo: "acutall" → "actual"	an hour ago
tools Revert "[cmake] Do not set CMAKE_INCLUDE_CURRENT_DIR."	3 days ago
unittests [SourceKit] Remove dependency of SwiftLang on Core	13 days ago
utils [emacs support] add a missing (require 'compile)	3 days ago

<https://github.com/apple/swift>



[swift](#) / [stdlib](#) / [public](#) / [SDK](#) / [CoreGraphics](#) / **CoreGraphics.swift**

[swift](#) / [stdlib](#) / [public](#) / [SDK](#) / [CoreGraphics](#) / **CoreGraphics.swift**

```
12
13 @_exported import CoreGraphics
14 import Darwin
15
16 //=====
17 // CGGeometry
18 //=====
19
20 public extension CGPoint {
21     static var zero: CGPoint {
22         @_transparent // @fragile
23         get { return CGPoint(x: 0, y: 0) }
24     }
25
26     @_transparent // @fragile
27     init(x: Int, y: Int) {
28         self.init(x: CGFloat(x), y: CGFloat(y))
29     }
30
31     @_transparent // @fragile
32     init(x: Double, y: Double) {
33         self.init(x: CGFloat(x), y: CGFloat(y))
34     }
35
36     @available(*, unavailable, renamed="zero")
37     static var zeroPoint: CGPoint {
38         fatalError("can't retrieve unavailable property")
39     }
40 }
41
```

```
public extension CGPoint {  
    static var zero: CGPoint {  
        @_transparent // @fragile  
        get { return CGPoint(x: 0, y: 0) }  
    }  
}
```



Neil Kimmitt 7:07 AM

wait a second

fuck radar

<https://github.com/apple/swift/blob/4d0c060e2189571b0103fa5473ed8d8145d0e231/stdlib/public/S DK/CoreGraphics/CoreGraphics.swift>

GitHub

[apple/swift](#)

swift - The Swift Programming Language

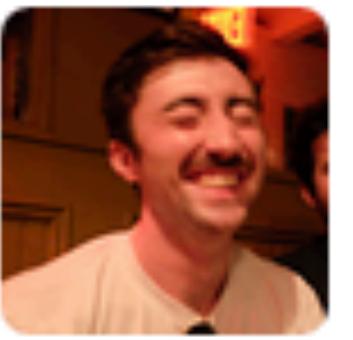




Soroush Khanlou 7:08 AM

hahaha

make a pull request!



Neil Kimmett 7:08 AM

oh i'm gonna

now i'm all excited

this has turned out excellently



Soroush Khanlou 7:09 AM

well hm, is UIEdgeInsets part of UIKit?

[swift](#) / [stdlib](#) / [public](#) / [SDK](#) / [UIKit](#) / **UIKit.swift**

[swift](#) / [stdlib](#) / [public](#) / [SDK](#) / [UIKit](#) / **UIKit.swift**

```
11 // -----
12
13 import Foundation
14 @_exported import UIKit
15
16
17 //====-
18 // Equatable types.
19 //====-
20
21 @transparent // @fragile
22 @warn_unused_result
23 public func == (lhs: UIEdgeInsets, rhs: UIEdgeInsets) -> Bool {
24     return lhs.top == rhs.top &&
25         lhs.left == rhs.left &&
26         lhs.bottom == rhs.bottom &&
27         lhs.right == rhs.right
28 }
29
30 extension UIEdgeInsets : Equatable {}
31
32 @transparent // @fragile
33 @warn_unused_result
34 public func == (lhs: UIOffset, rhs: UIOffset) -> Bool {
35     return lhs.horizontal == rhs.horizontal &&
36         lhs.vertical == rhs.vertical
37 }
38
39 extension UIOffset : Equatable {}
```

```
13 import Foundation
14 @_exported import UIKit
15
16 +//=====
17 +// UIGeometry
18 +//=====
19 +
20 +public extension UIEdgeInsets {
21 +    static var zero: UIEdgeInsets {
22 +        @_transparent // @fragile
23 +        get { return UIEdgeInsets(top: 0.0, left: 0.0, bottom: 0.0, right: 0.0) }
24 +    }
25 +}
26 +
27 +public extension UIOffset {
28 +    static var zero: UIOffset {
29 +        @_transparent // @fragile
30 +        get { return UIOffset(horizontal: 0.0, vertical: 0.0) }
31 +    }
32 +}
33
```



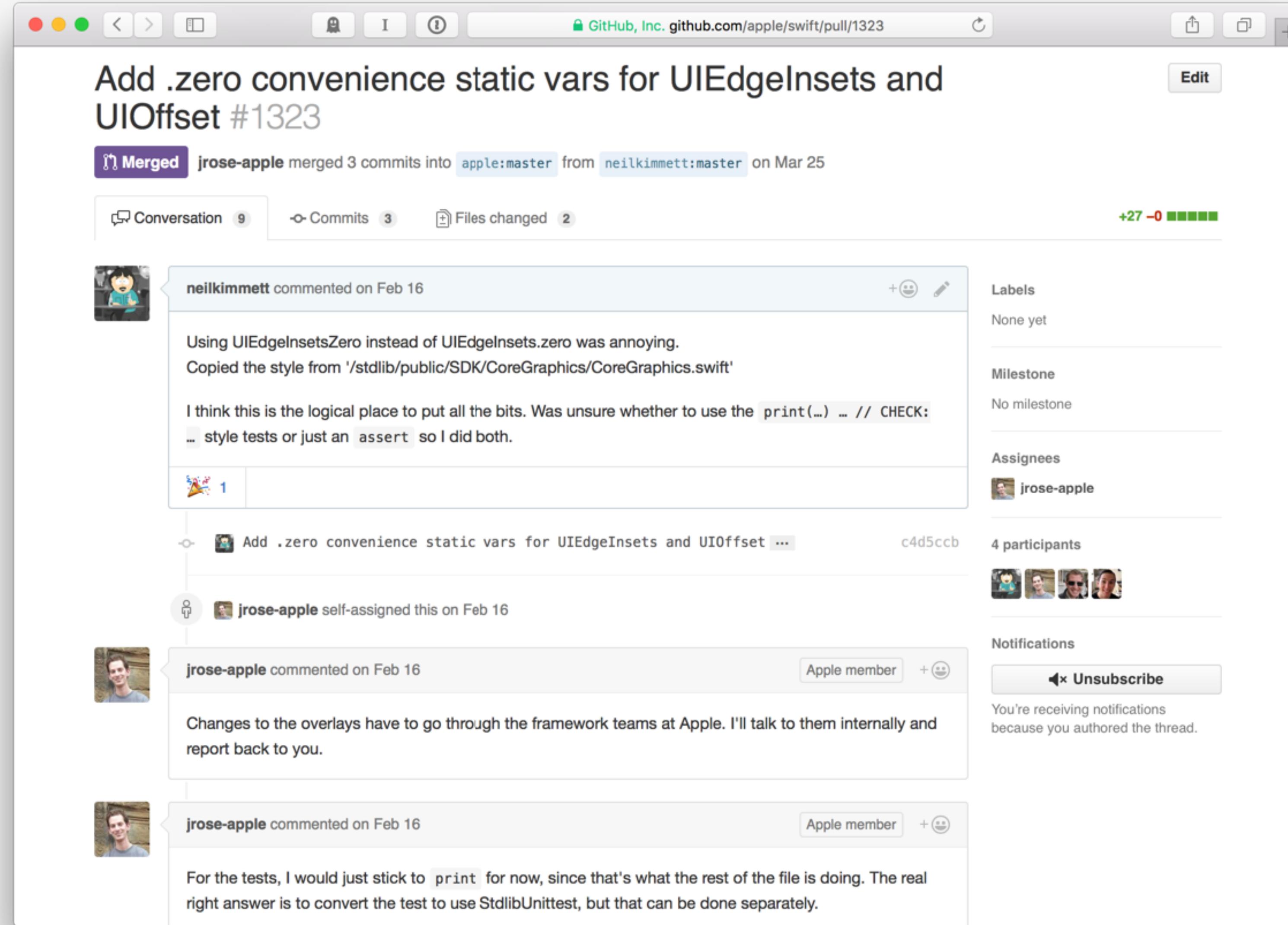


[swift](#) / [test](#) / [1_stdlib](#) / **UIKit.swift**

```
50 var inset1 = UIEdgeInsets(top: 1.0, left: 2.0, bottom: 3.0, right: 4.0)
51 var inset2 = UIEdgeInsets(top: 1.0, left: 2.0, bottom: 3.1, right: 4.0)
52 print("inset1 == inset1: \(inset1 == inset1)")
53 print("inset1 != inset1: \(inset1 != inset1)")
54 print("inset1 == inset2: \(inset1 == inset2)")
55 // CHECK: inset1 == inset1: true
56 // CHECK: inset1 != inset1: false
57 // CHECK: inset1 == inset2: false
58
```

[swift](#) / [test](#) / [1_stdlib](#) / **UIKit.swift**

```
68 +
69 +var inset0 = UIEdgeInsets(top: 0.0, left: 0.0, bottom: 0.0, right: 0.0)
70 +var insetDot0 = UIEdgeInsets.zero
71 +print("inset0 == insetDot0: \(inset0 == insetDot0)")
72 +// CHECK: inset0 == insetDot0: true
73 +
74 +var offset0 = UIOffset(horizontal: 0.0, vertical: 0.0)
75 +var offsetDot0 = UIOffset.zero
76 +print("offset0 == offsetDot0: \(offset0 == offsetDot0)")
77 +// CHECK: offset0 == offsetDot0: true
```

A screenshot of a GitHub pull request interface. The title is "Add .zero convenience static vars for UIEdgeInsets and UIOffset #1323". A purple "Merged" button indicates the pull request has been merged into the `apple:master` branch from the `neilkimmett:master` branch on March 25. The conversation tab shows 9 comments, with 3 commits and 2 files changed. The pull request has a +27 -0 score. The main comment by neilkimmett on Feb 16 discusses using `UIEdgeInsetsZero` instead of `UIEdgeInsets.zero` and copying style from CoreGraphics.swift. jrose-apple self-assigned the issue on Feb 16. jrose-apple commented twice, once stating changes need internal review and another time suggesting sticking to `print` for tests. The right sidebar shows labels (None yet), milestones (No milestone), assignees (jrose-apple), participants (4), and notifications (Unsubscribe).

Add .zero convenience static vars for UIEdgeInsets and UIOffset #1323

Merged jrose-apple merged 3 commits into apple:master from neilkimmett:master on Mar 25

Conversation 9 Commits 3 Files changed 2 +27 -0

Labels None yet

Milestone No milestone

Assignees jrose-apple

Participants 4

Notifications Unsubscribe

<https://github.com/apple/swift/pull/1323>



jrose-apple commented on Feb 16, 2016

Owner + 😊

Changes to the overlays have to go through the framework teams at Apple. I'll talk to them internally and report back to you.



jrose-apple commented on Feb 16, 2016

Owner + 😊

For the tests, I would just stick to `print` for now, since that's what the rest of the file is doing. The real right answer is to convert the test to use `StdlibUnittest`, but that can be done separately.

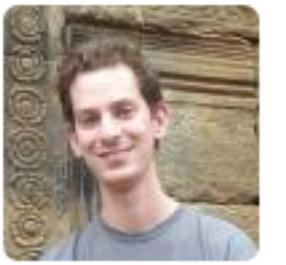


jrose-apple commented on Mar 21

Apple member



Okay, UIKit has approved this change! (Sorry for the turnaround time.) I'll trigger the tests again.



jrose-apple commented on Mar 21

Apple member



@swift-ci Please test

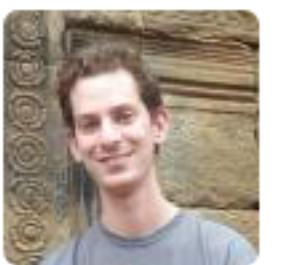


jrose-apple merged commit **d9596ab** into `apple:master` on Mar 25

2 checks passed

[View details](#)

[Revert](#)



jrose-apple commented on Mar 25

Apple member



Thanks, Neil!



```
let scrollView = UIScrollView()  
scrollView.contentInset = UIEdgeInsets.zero
```

UIEdgeInsets zero

What can I do?

“To be a truly great community, Swift.org needs to welcome developers from all walks of life, with different backgrounds, and with a wide range of experience. A diverse and friendly community will have more great ideas, more unique perspectives, and produce more great code. We will work diligently to make the Swift community welcoming to everyone.”

Apple Inc. lists.swift.org

[swift-evolution] Replace throws with Result

Douglas Gregor dgregor@apple.com
Wed May 3 15:03:41 CDT 2017

- Previous message: [swift-evolution] Replace throws with Result
- Next message: [swift-evolution] Replace throws with Result
- Messages sorted by: [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#)

> On May 3, 2017, at 3:31 AM, Jaden Geller via swift-evolution <swift-evolution@swift.org> wrote:
>
> To be frank, there's no way this is happening. The rationale for Swift's error handling is documented <<https://github.com/apple/swift/blob/master/docs/ErrorHandling.rst>>, and this system was designed and implemented *after* optionals already existed in the language. I don't think there's any evidence that this was a horrible decision, especially one that requires a *majorly* source-breaking change to remove.

As a core team member, let me amplify this response a bit: the current error-handling system will not be going away, for the reasons Jaden has stated above.

Maybe we will grow typed throws at some point; maybe Result will get added to the standard library with some nice affordances to map between throwing and Result-producing types; but we're well beyond the point where we can rip out major features that have been generally well-received.

- Doug

> - Jaden
>
>> On May 3, 2017, at 3:06 AM, Jose Manuel Sánchez Peñarroja via swift-evolution <swift-evolution@swift.org> wrote:
>>
>> There is currently a discussion open about throws, but it seems that the idea would be to add optionals to throws. In my opinion it would be much better to just get rid of all the throw error system.
>>

<https://lists.swift.org>



A screenshot of a Mac Mail application window. The sidebar on the left shows a list of accounts and mailing lists, with 'swift-users' selected. The main pane displays a list of messages from various users. A specific message from 'brandonb2019@gmail.com' dated May 6, 2017, at 10:21:39 PM is selected, showing a detailed view of the email body and its attachments.

jeschot@xs4all.nl 5/8/17, 7:39 AM
discardable function start?

kelvin13ma@gmail.com 5/7/17, 1:57 PM
● Passing value types or members of value types?

brandonb2019@gmail.com 5/7/17, 8:28 PM
● Swift build-script Smaller Memory Footprint?

swift@hpwooten.com 5/5/17, 9:04 AM
● Compatibility Suite for already failing project

rmann@latencyzero.com 5/4/17, 7:52 PM
● Slicing a [UInt8] into a Data without copying?

dowobeha@gmail.com 5/4/17, 5:12 PM
● Compiling Swift on Ubuntu to run on Centos

geojay@gmail.com 5/3/17, 7:15 PM
● Using .gyb outside of the stdlib

brandonb2019@gmail.com brandonb2019@gmail.com May 6, 2017, 10:21:39 PM

Hello,

I'm installing Swift 3 on FreeBSD 11. After installing the necessary dependencies, I now only need to run the build-script.

The problem is that I cannot proceed beyond:

> Linking CXX executable bin/llvm-lto

because I consistently run out of memory on my 16GB server.

(2 direct replies) ↗

pestov@apple.com pestov@apple.com May 7, 2017, 8:13:10 PM

On Darwin (which uses its own linker and not GNU ld), you can definitely build Swift with less than 16GB of RAM available.

Maybe you could try gold?

Slava

> On May 6, 2017, at 7:21 PM, Brandon B via swift-users<swift-users at swi

<https://stylemac.com/hirundo/>

The screenshot shows a Mac OS X desktop with a browser window open to the Swift Evolution website at <https://apple.github.io/swift-evolution/>. The window has a standard OS X title bar with icons for window control, search, and other system functions. The main content area displays the "Swift Evolution" logo (a red square with a white bird icon) and the title "Swift Evolution". Below the title is a search bar with the placeholder "Search" and a magnifying glass icon. A "Logout" button is visible in the top right corner. The page lists 176 proposals, with the first five shown in detail:

- SE-0176 Enforce Exclusive Access to Memory**
Author: John McCall
Review Manager: Ben Cohen
Status: Active Review
Scheduled: May 2–8
- SE-0175 Package Manager Revised Dependency Resolution**
Author: Rick Ballard
Review Manager: Ankit Aggarwal
Status: Active Review
Scheduled: May 2–9
- SE-0174 Change filter to return an associated type**
Author: Ben Cohen
Review Manager: Doug Gregor
Status: Accepted
- SE-0161 Smart KeyPaths: Better Key-Value Coding for Swift**
Authors: David Smith, Michael LeHew, Joe Groff
Review Manager: Doug Gregor
Status: Accepted
- SE-0173 Add MutableCollection.swapAt(_:_:)**
Author: Ben Cohen
Review Manager: Ted Kremenek
Implemented In: Swift 4
Status: Implemented

<https://apple.github.io/swift-evolution/>

A screenshot of a Mac OS X desktop environment showing a web browser window. The window title is "swiftweekly.github.io/issue-68/". The main content of the page is a red-themed newsletter titled "Swift Weekly Brief". It describes itself as a community-driven weekly newsletter about what's happening in the Swift open source projects at Swift.org. It is curated by Jesse Squires and published for free every Thursday. Below this text are three buttons: "Subscribe" (with an envelope icon), "Sponsor" (with a star icon), and "Podcast" (with a microphone icon). A horizontal line separates this from the main content area. The main content area starts with "Issue #68 04 May 2017" and "Written by: Bas Broek". It then discusses recent work on Swift Evolution proposals and first-time contributors. Below this is a link to learn more about sponsorship. The next section, "Swift Unwrapped", mentions Episode 9 of the String Manifesto, featuring JP Simard and Jesse Squires. The final section, "Commits and pull requests", notes a commit by Slava Pestov implementing SE-0156: Class and Subtype existentials.

Swift Weekly Brief

A [community-driven](#) weekly newsletter about what's happening in the Swift open source projects at [Swift.org](#). Curated by [Jesse Squires](#) and published for free every Thursday.

[Subscribe](#) [Sponsor](#) [Podcast](#)

Issue #68 04 May 2017

Written by: [Bas Broek](#)

Quite a lot of work has been done to implement recently accepted Swift Evolution proposals, as well as improving their diagnostics and error messages. Interestingly, some of this work has been done by first-time contributors, which is amazing to see!

Interested in sponsoring Swift Weekly Brief? [Learn more here.](#)

Swift Unwrapped

Episode 9: [String Manifesto](#)

This week JP Simard and Jesse Squires discuss the String Manifesto!

Commits and pull requests

Slava Pestov [implemented SE-0156: Class and Subtype existentials](#).

<https://swiftweekly.github.io>

A screenshot of a web browser displaying the 'Swift Unwrapped' podcast page on the website spec.fm. The page features a header with the spec logo, a search bar, and links for 'About' and 'Sponsors'. Below the header, the breadcrumb navigation shows 'Home / Podcasts / Swift Unwrapped'. The main content area includes the podcast's logo (an orange square with 'Swift Unwrapped' text), its title 'Swift Unwrapped', a brief description ('A 30-minute spin off of Swift Weekly Brief and other Swifty news.'), and links for 'Subscribe', 'Follow', and 'Sponsor'. The page lists six episodes in a grid format:

- 09: String Manifesto** (May 1, 2017) - We go in way over our heads into Swift's String Manifesto. [Listen to this Episode](#) • [Download](#)
- 08: Archival Serialization & Swift Encoders** (April 24, 2017) - We unpack the recent SE proposals on serialization & encoding. [Listen to this Episode](#) • [Download](#)
- 07: Access Control - The Saga Continues** (April 17, 2017) - There has been a ton of debate on the Swift Evolution mailing lists about access control in Swift. We share our thoughts on the situation. [Listen to this Episode](#) • [Download](#)
- 06: Swift 3.1 Release & SwiftPM Improvements** (April 10, 2017) - Continuing from last week's episode, we discuss the Swift 3.1 release and the improvements in SwiftPM. [Listen to this Episode](#) • [Download](#)
- 05: Objective-C Interoperability** (April 3, 2017) - Continuing from last week's episode, we discuss the Swift 3.1 release and the improvements in SwiftPM. [Listen to this Episode](#) • [Download](#)
- 04: Bridging with Objective-C** (March 27, 2017) - This week we're talking about bridging with Objective-C. [Listen to this Episode](#) • [Download](#)

<https://spec.fm/podcasts/swift-unwrapped>



Joe Groff

@jckarter

Jordan Rose

@UINT_MIN

Doug Gregor

@dgregor79

Ted Kremenek

@tkremenek

Erica Sadun

@ericasadun

Brian Gesiak

@modocache



Brian Gesiak

@modocache

To become one of the top 100 most prolific committers to Swift, you need 15 commits. For LLVM, you'd need 130.

RETWEETS

4

LIKES

24



5:56 PM - 6 May 2017

Apple Inc. bugs.swift.org/browse/SR-4830?filter=10451

Swift Dashboards Projects Issues Search Log In

FILTERS <>

New filter Find filters

My Open Issues Reported by Me Recently Viewed All Issues

FAVORITE FILTERS You must be [logged in](#) to view favorite filters.

All starter tasks Details Export Tools

Swift Type: All Open, Reopened Unassigned Contains text More Advanced Label: StarterBug

Order by Created ↓

- SR-4830 Propagate colors from compiler output
- SR-4824 Add tests to check Collection constraint...
- SR-4793 SwiftPM Gives three different error mes...
- SR-4747 Implement binary search for inserting e...
- SR-4716 Better diagnostics for multifile executab...
- SR-4595 Tests generate lots of warnings
- SR-4589 Add --filter option to swift test
- SR-4411 Make SwiftPM store relative paths in pi...
- SR-4405 Add @escaping to RecoverableError.at...

1 of 82 Export

SR-4830 Propagate colors from compiler output

Details

Type:	Bug
Status:	OPEN
Priority:	Medium
Resolution:	Unresolved
Component/s:	lldbuild, ... (1)
Labels:	StarterBug

Description

Currently, colors are not shown in compiler output. We probably need to detect if stdout is tty and then force colors out of compiler using some driver flag.

Activity

All Comments History Activity

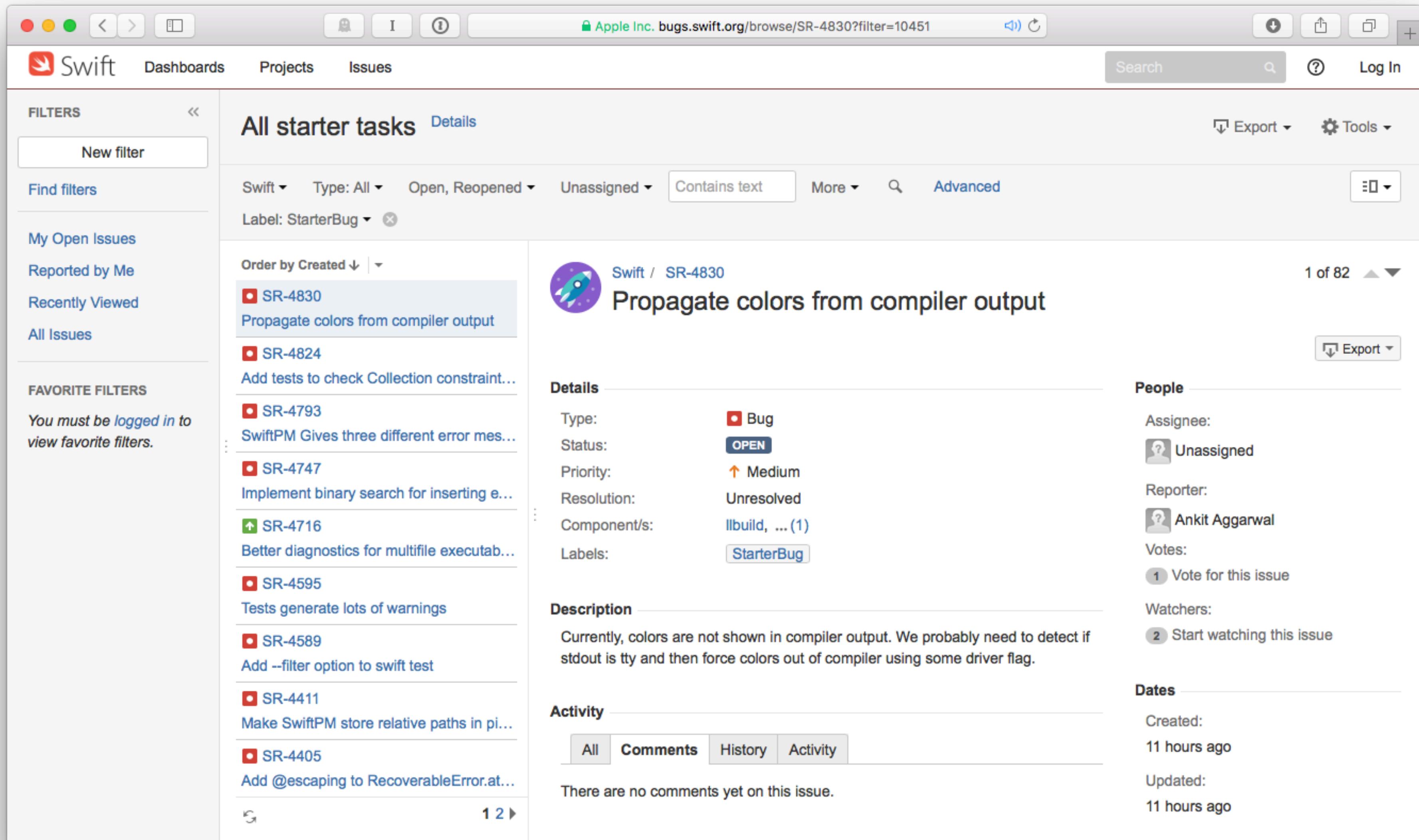
There are no comments yet on this issue.

People

Assignee: Unassigned Reporter: Ankit Aggarwal Votes: 1 Vote for this issue Watchers: 2 Start watching this issue

Dates

Created: 11 hours ago Updated: 11 hours ago

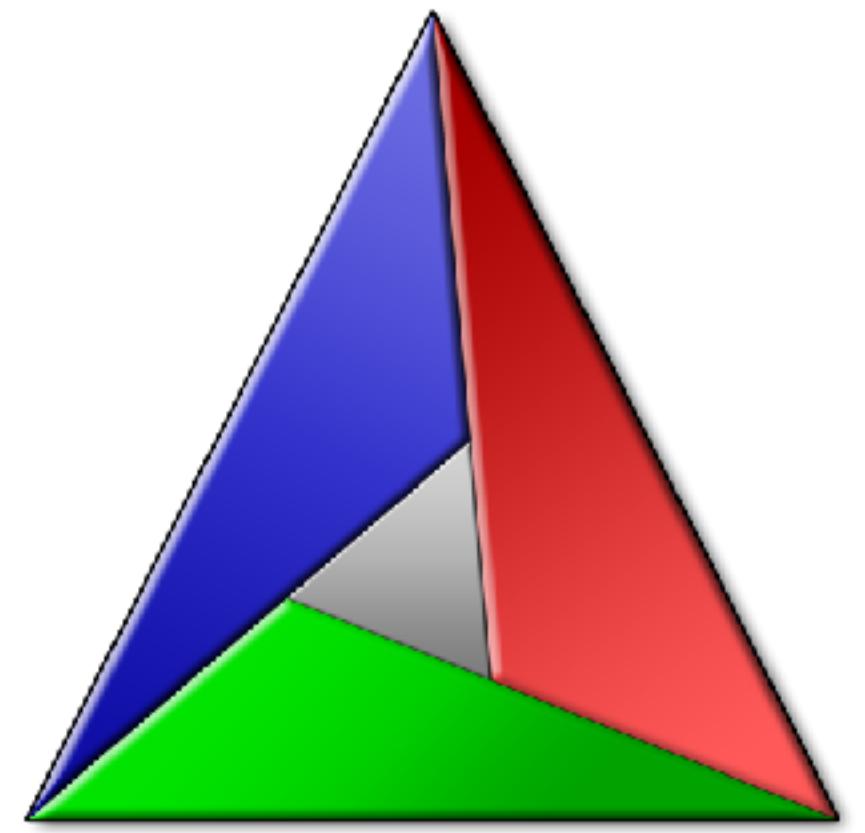
A screenshot of a web browser window displaying the Swift bug tracking system at https://bugs.swift.org. The page shows a list of issues under the 'All starter tasks' filter. The first issue, SR-4830, is selected and shown in detail. The details page includes sections for 'Details' (Type: Bug, Status: OPEN, Priority: Medium, Resolution: Unresolved, Component/s: lldbuild, Labels: StarterBug), 'Description' (a note about colors not appearing in compiler output), 'Activity' (no comments yet), and 'People' (unassigned assignee, reporter Ankit Aggarwal, 1 vote, 2 watchers). The URL in the address bar is https://bugs.swift.org/browse/SR-4830?filter=10451.

<https://bugs.swift.org>

The screenshot shows the GitHub repository page for `apple/swift`. The page includes the repository name, a search bar, navigation links for Pull requests, Issues, and Gist, and a header with Watch (2,223), Star (32,093), Fork (4,418) buttons. Below the header are tabs for Code, Pull requests (99), Pulse, and Graphs. A banner at the top says "The Swift Programming Language <https://swift.org/>". Key statistics are displayed: 39,327 commits, 21 branches, 36 releases, and 358 contributors. A "Clone or download" button is prominent. The main content area lists recent commits from the `master` branch, with the most recent being a merge pull request from `practicalswift/typo-fixes-20160705`.

Commit	Message	Time Ago
<code>.github</code>	Add a missing "the" to the pull request template	8 days ago
<code>apinotes</code>	[APINotes] NSPointerFunctionsOptions 0-valued members shouldn't be [].	5 days ago
<code>benchmark</code>	[gardening] Use '{let,var} c: C' instead of '{let,var} c : C'	4 days ago
<code>bindings/xml</code>	Nesting parameter/returns/throws doc comments for closure parameters	3 months ago
<code>cmake</code>	[cmake] Refactor check_working_std_regex into its own cmake file Swif...	8 hours ago
<code>docs</code>	[docs] LibraryEvolution: add "let vs. var" as a planned proposal.	4 days ago
<code>include</code>	Sema: Explicitly set interface type on all AbstractFunctionDecls	4 hours ago
<code>lib</code>	Sema: Explicitly set interface type on all AbstractFunctionDecls	4 hours ago
<code>stdlib</code>	Merge pull request #3322 from lucianomarisi/chore/sequence-documentati...	2 days ago
<code>test</code>	[gardening] Fix recently introduced typo: "acutall" → "actual"	an hour ago
<code>tools</code>	Revert "[cmake] Do not set CMAKE_INCLUDE_CURRENT_DIR."	3 days ago
<code>unittests</code>	[SourceKit] Remove dependency of SwiftLang on Core	13 days ago
<code>utils</code>	[emacs support] add a missing (require 'compile)	3 days ago

<https://github.com/apple/swift>



CMake
Cross-platform Make

Ninja



```
brew install cmake ninja
```

```
mkdir swift-source
```

```
cd swift-source
```

```
git clone https://github.com/apple/swift.git
```

```
./swift/utils/update-checkout --clone
```

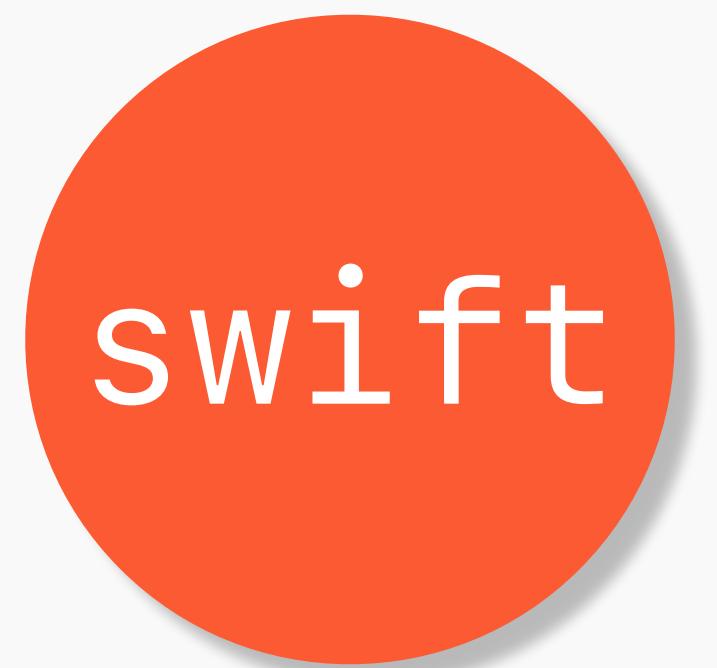
```
→ swift-source ls
build
clang
cmark
compiler-rt
llbuild
lldb
llvm
ninja
swift
swift-corelibs-foundation
swift-corelibs-libdispatch
swift-corelibs-xctest
swift-integration-tests
swift-xcode-playground-support
swiftpm
```

```
./swift/utils/build-script -h
```

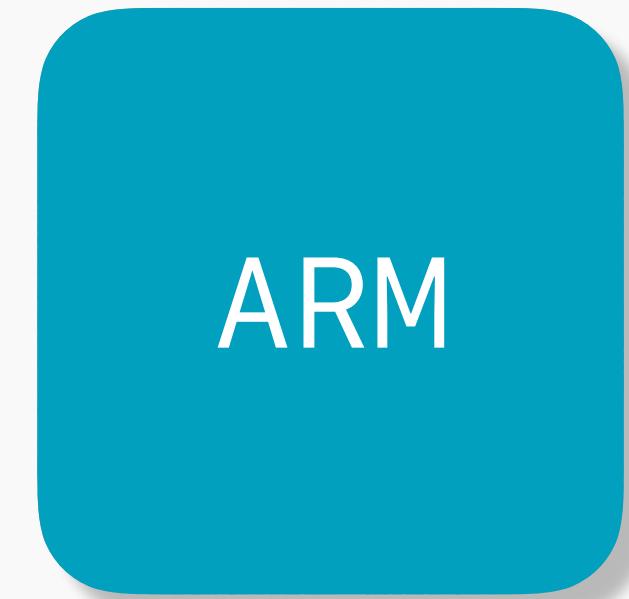
```
./swift/utils/build-script -R -t
```

L L V M

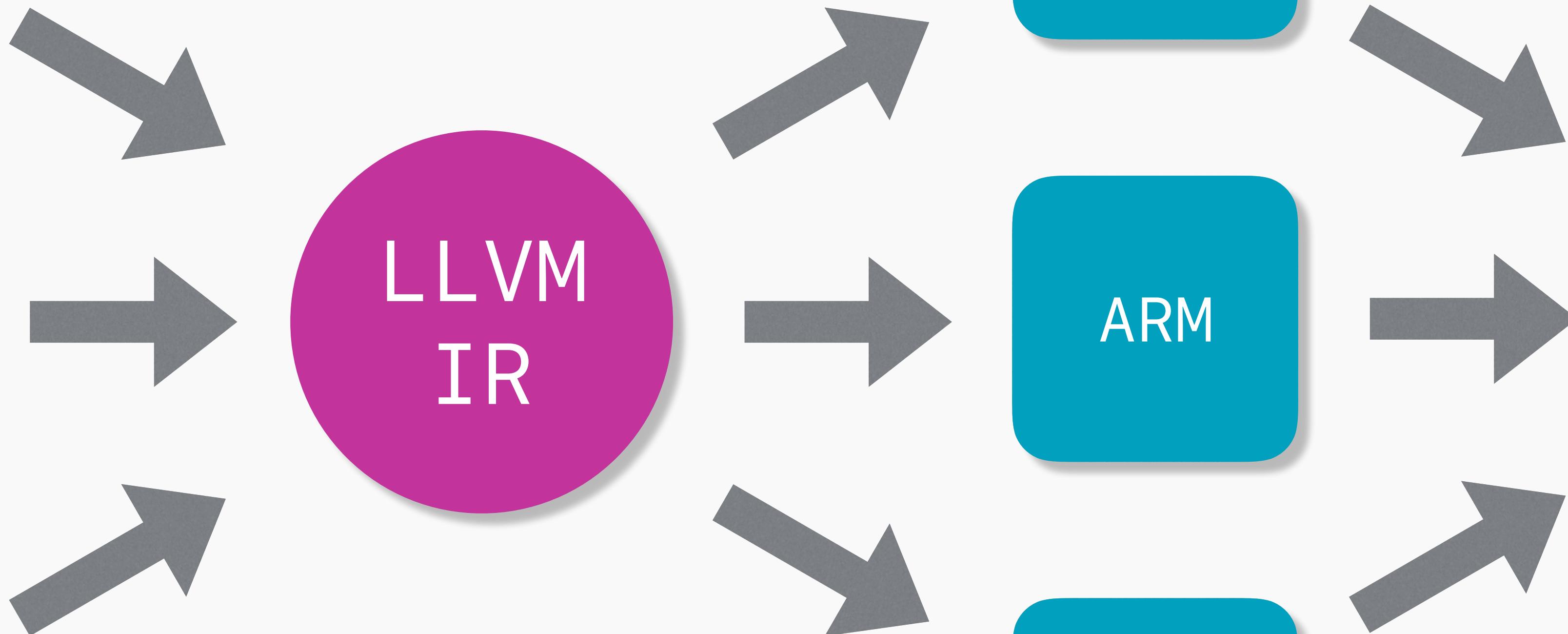
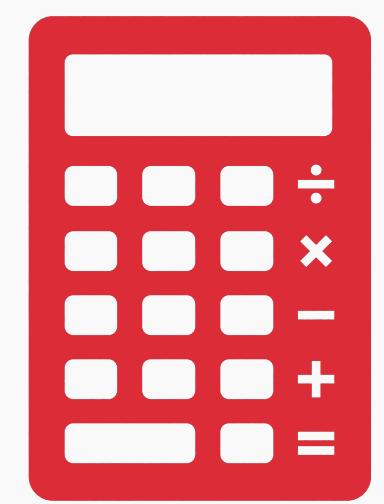
frontend

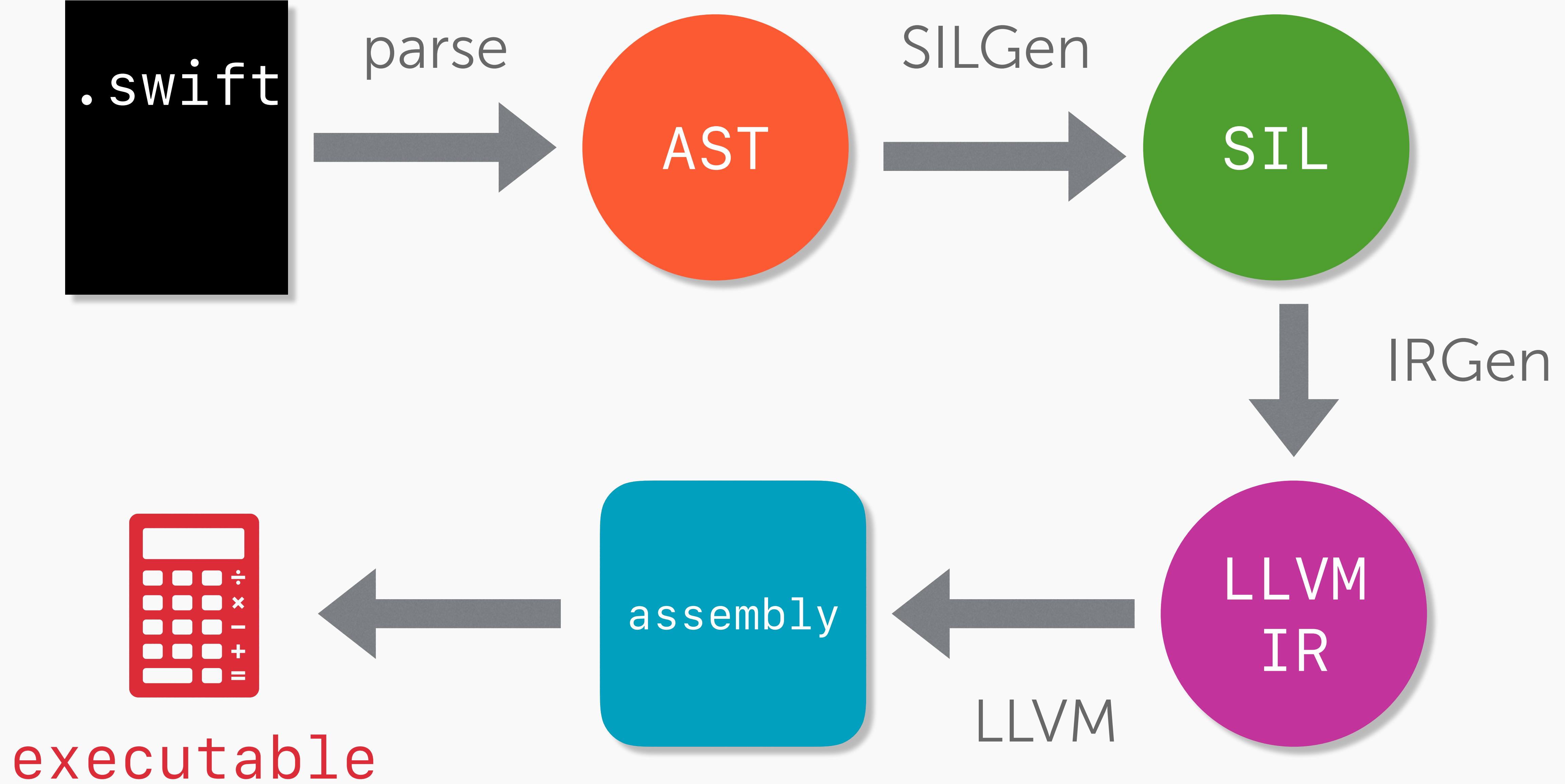


backend



executable





```
swiftc -dump-ast  
        -emit-silgen  
        -emit-sil      add.swift  
        -emit-ir  
        -emit-assembly
```

add.swift

```
let x = 1 + 1  
print(x)
```

swiftc -dump-ast add.swift

```
→ swift-test swiftc -dump-ast add.swift
(source_file
  (top_level_code_decl
    (brace_stmt
      (pattern_binding_decl
        (pattern_named type='Int' 'x')
        (binary_expr type='Int' location=add.swift:1:11 range=[add.swift:1:9 - line:1:13] nothrow
          (declref_expr type='(Int, Int) -> Int' location=add.swift:1:11 range=[add.swift:1:11 - line:1:11] decl=Swift.(file).+ function_ref=unapplied specialized=no)
          (tuple_expr implicit type='(Int, Int)' location=add.swift:1:9 range=[add.swift:1:9 - line:1:13]
            (call_expr implicit type='Int' location=add.swift:1:9 range=[add.swift:1:9 - line:1:9] nothrow arg_labels=_builtinIntegerLiteral:
              (constructor_ref_call_expr implicit type='(Int2048) -> Int' location=add.swift:1:9 range=[add.swift:1:9 - line:1:9] nothrow
                (declref_expr implicit type='(Int.Type) -> (Int2048) -> Int' location=add.swift:1:9 range=[add.swift:1:9 - line:1:9] decl=Swift.(file).Int.init(_builtinIntegerLiteral:) function_ref=single specialized=no)
                  (type_expr implicit type='Int.Type' location=add.swift:1:9 range=[add.swift:1:9 - line:1:9] typerepr='Int'))
              (tuple_expr implicit type='(_builtinIntegerLiteral: Int2048)' location=add.swift:1:9 range=[add.swift:1:9 - line:1:9] names=_builtinIntegerLiteral
                (integer_literal_expr type='Int2048' location=add.swift:1:9 range=[add.swift:1:9 - line:1:9] value=1)))
            (call_expr implicit type='Int' location=add.swift:1:13 range=[add.swift:1:13 - line:1:13] nothrow arg_labels=_builtinIntegerLiteral:
              (constructor_ref_call_expr implicit type='(Int2048) -> Int' location=add.swift:1:13 range=[add.swift:1:13 - line:1:13] nothrow
                (declref_expr implicit type='(Int.Type) -> (Int2048) -> Int' location=add.swift:1:13 range=[add.swift:1:13 - line:1:13] decl=Swift.(file).Int.init(_builtinIntegerLiteral:) function_ref=single specialized=no)
                  (type_expr implicit type='Int.Type' location=add.swift:1:13 range=[add.swift:1:13 - line:1:13] typerepr='Int'))
              (tuple_expr implicit type='(_builtinIntegerLiteral: Int2048)' location=add.swift:1:13 range=[add.swift:1:13 - line:1:13] names=_builtinIntegerLiteral
                (integer_literal_expr type='Int2048' location=add.swift:1:13 range=[add.swift:1:13 - line:1:13] value=1))))))
        (var_decl "x" type='Int' interface type='Int' access=internal let storage_kind=stored)
      (top_level_code_decl
        (brace_stmt
          (call_expr type='()' location=add.swift:2:1 range=[add.swift:2:1 - line:2:8] nothrow arg_labels=_:
            (declref_expr type='(Any..., String, String) -> ()' location=add.swift:2:1 range=[add.swift:2:1 - line:2:1] decl=Swift.(file).print(_:_separator:_terminator:) function_ref=single specialized=no)
            (tuple_shuffle_expr implicit type='(Any..., separator: String, terminator: String)' location=add.swift:2:7 range=[add.swift:2:6 - line:2:8] source_is_scalar elements=[-2, -1, -1] variadic_sources=[0] default_args_owner=Swift.(file).print(_:_separator:_terminator:)
              (paren_expr type='Any' location=add.swift:2:7 range=[add.swift:2:6 - line:2:8]
                (erasure_expr implicit type='Any' location=add.swift:2:7 range=[add.swift:2:7 - line:2:7]
                  (declref_expr type='Int' location=add.swift:2:7 range=[add.swift:2:7 - line:2:7] decl=add.(file).x@add.swift:1:5 direct_to_storage function_ref=unapplied specialized=no)))))))
```

```
swiftc -dump-ast add.swift
```

```
→ swift-test swiftc -dump-ast add.swift
(source_file
  (top_level_code_decl
    (brace_stmt
      (pattern_binding_decl
        (pattern_named type='Int' 'x')
        (binary_expr type='Int' location=add.swift:1:11 n
          (declref_expr type='(Int, Int) -> Int' location=
```

swiftc -emit-silgen add.swift | xcrun swift-demangle

```
+ swift-test swiftc -emit-silgen add.swift | xcrun swift-demangle
sil_stage raw

import Builtin
import Swift
import SwiftShims

// x
sil_global hidden @_let @_add : Swift.Int : $Int

sil_scope 1 { parent @main : $@convention(c) (Int32, UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>>) -> Int32 }

// main
sil @main : $@convention(c) (Int32, UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>>) -> Int32 {
    bb0(%0 : $Int32, %1 : $UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>>):
        alloc_global @_add : Swift.Int, loc "@dd.swift":1:5, scope 1 // id: %2
        %3 = global_addr @_add, loc "@dd.swift":1:5, scope 1 // users: %23, %14
        // Function_ref + infix(Int, Int) -> Int
        %4 = function_ref @_Swift.+ infix(Swift.Int, Swift.Int) -> Swift.Int : $@convention(thin) (Int, Int) -> Int, loc "add.swift":1:11, scope 1 // user: %13
        // function_ref Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Int
        %5 = function_ref @_Swift.Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Swift.Int : $@convention(method) (Builtin.Int2048, @thin Int.Type) -> Int, loc "add.swift":1:9, scope 1 // user: %8
        %6 = metatype @_thin Int.Type, loc "add.swift":1:9, scope 1 // user: %8
        %7 = integer_literal $Builtin.Int2048, 1, loc "add.swift":1:9, scope 1 // user: %8
        %8 = apply %5(%7, %6) : $@convention(method) (Builtin.Int2048, @_thin Int.Type) -> Int, loc "add.swift":1:9, scope 1 // user: %13
        // Function_ref Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Int
        %9 = function_ref @_Swift.Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Swift.Int : $@convention(method) (Builtin.Int2048, @_thin Int.Type) -> Int, loc "add.swift":1:13, scope 1 // user: %12
        %10 = metatype @_thin Int.Type, loc "add.swift":1:13, scope 1 // user: %12
        %11 = integer_literal $Builtin.Int2048, 1, loc "add.swift":1:13, scope 1 // user: %12
        %12 = apply %9(%11, %10) : $@convention(method) (Builtin.Int2048, @_thin Int.Type) -> Int, loc "add.swift":1:13, scope 1 // user: %13
        %13 = apply %6(%8, %12) : $@convention(thin) (Int, Int) -> Int, loc "add.swift":1:11, scope 1 // user: %14
        store %13 to [trivial] %3 : $Int, loc "add.swift":1:11, scope 1 // id: %14
        // Function_ref print([Any], separator : String, terminator : String) -> ()
        %15 = function_ref @_Swift.print ([Any], separator : Swift.String, terminator : Swift.String) -> (), loc "add.swift":2:1, scope 1 // user: %29
        %16 = integer_literal $Builtin.Word, 1, loc "add.swift":2:1, scope 1 // user: %18
        // Function_ref _allocateUninitializedArray<A> (Builtin.Word) -> ([A], Builtin.RawPointer)
        %17 = function_ref @_Swift._allocateUninitializedArray<A> (Builtin.Word) -> ([A], Builtin.RawPointer) : $@convention(thin) <r_0_0> (Builtin.Word) -> (@owned Array<r_0_0>, Builtin.RawPointer), loc "add.swift":2:7, scope 1 // user: %18
        %18 = apply %17<Any>(%16) : $@convention(thin) <r_0_0> (Builtin.Word) -> (@owned Array<r_0_0>, Builtin.RawPointer), loc "add.swift":2:7, scope 1 // users: %20, %19
        %19 = tuple_extract %18 : ($Array<Any>, Builtin.RawPointer), 0, loc "add.swift":2:7, scope 1 // user: %20
        %20 = tuple_extract %18 : ($Array<Any>, Builtin.RawPointer), 1, loc "add.swift":2:7, scope 1 // user: %21
        %21 = pointer_to_address %20 : Builtin.RawPointer to [strict] @_Any, loc "add.swift":2:7, scope 1 // user: %22
        %22 = init_existential_addr %21 : @_Any, SInt, loc "add.swift":2:7, scope 1 // user: %22
        %23 = load [trivial] %3 : $Int, loc "add.swift":2:7, scope 1 // user: %24
        store %23 to [trivial] %22 : @_Any, loc "add.swift":2:7, scope 1 // id: %24
        // Function_ref (print([Any], separator : String, terminator : String) -> ()).(default argument 1)
        %25 = function_ref @_Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 1) : $@convention(thin) () -> @owned String, loc "add.swift":2:7, scope 1 // user: %26
        %26 = apply %25() : $@convention(thin) () -> @owned String, loc "add.swift":2:7, scope 1 // user: %29
        // Function_ref (print([Any], separator : String, terminator : String) -> ()).(default argument 2)
        %27 = function_ref @_Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 2) : $@convention(thin) () -> @owned String, loc "add.swift":2:7, scope 1 // user: %28
        %28 = apply %27() : $@convention(thin) () -> @owned String, loc "add.swift":2:7, scope 1 // user: %28
        %29 = apply %15(%19, %26, %28) : $@convention(thin) (@owned Array<Any>, @owned String, @owned String) -> (), loc "add.swift":2:8, scope 1
        %30 = integer_literal $Builtin.Int32, 0, scope 1 // user: %30
        %31 = struct @_Int32 (%30 : $Builtin.Int32), scope 1 // user: %32
        return %31 : $Int32, scope 1 // id: %32
    } // end sil function 'main'

    // + infix(Int, Int) -> Int
    sil [transparent] @_Swift.+ infix (Swift.Int, Swift.Int) -> Swift.Int : $@convention(thin) (Int, Int) -> Int

    // Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Int
    sil [transparent] @_Swift.Int.init (_builtinIntegerLiteral : Builtin.Int2048) -> Swift.Int : $@convention(method) (Builtin.Int2048, @_thin Int.Type) -> Int

    // print([Any], separator : String, terminator : String) -> ()
    sil [nonline] [_semantics "stdlib_binary_only"] @_Swift.print ([Any], separator : Swift.String, terminator : Swift.String) -> () : $@convention(thin) (@owned Array<Any>, @owned String, @owned String) -> ()

    // _allocateUninitializedArray<A> (Builtin.Word) -> ([A], Builtin.RawPointer)
    sil [fragile] [always_inline] @_Swift._allocateUninitializedArray<A> (Builtin.Word) -> ([A], Builtin.RawPointer) : $@convention(thin) <r_0_0> (Builtin.Word) -> (@owned Array<r_0_0>, Builtin.RawPointer)

    // (print([Any], separator : String, terminator : String) -> ()).(default argument 1)
    sil [nonline] [_semantics "stdlib_binary_only"] @_Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 1) : $@convention(thin) () -> @owned String

    // (print([Any], separator : String, terminator : String) -> ()).(default argument 2)
    sil [nonline] [_semantics "stdlib_binary_only"] @_Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 2) : $@convention(thin) () -> @owned String
```

```
swiftc -emit-silgen add.swift | xcrun swift-demangle
```

```
// main
sil @main : $@convention(c) (Int32, UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>,
bb0(%0 : $Int32, %1 : $UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>>):
    alloc_global @add.x : Swift.Int, loc "add.swift":1:5, scope 1 // id: %2
    %3 = global_addr @add.x : Swift.Int : $*Int, loc "add.swift":1:5, scope 1 // users: %23, %
    // function_ref + infix(Int, Int) -> Int
    %4 = function_ref @Swift.+ infix (Swift.Int, Swift.Int) -> Swift.Int : $@convention(thin)
    // function_ref Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Int
    %5 = function_ref @Swift.Int.init (_builtinIntegerLiteral : Builtin.Int2048) -> Swift.Int
```

```
swiftc -emit-silgen add.swift | xcrun swift-demangle
```

```
// main
sil @main : $@convention(c) (Int32, UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>,
bb0(%0 : $Int32, %1 : $UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>>):
    alloc_global @add.x : Swift.Int, loc "add.swift":1:5, scope 1 // id: %2
    %3 = global_addr @add.x : Swift.Int : $*Int, loc "add.swift":1:5, scope 1 // users: %23, %
    // function_ref + infix(Int, Int) -> Int
    %4 = function_ref @Swift.+ infix (Swift.Int, Swift.Int) -> Swift.Int : $@convention(thin)
    // function_ref Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Int
    %5 = function_ref @Swift.Int.init (_builtinIntegerLiteral : Builtin.Int2048) -> Swift.Int
```

```
swiftc -emit-silgen add.swift | xcrun swift-demangle
```

```
// main
sil @main : $@convention(c) (Int32, UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>,
bb0(%0 : $Int32, %1 : $UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>>):
    alloc_global @add.x : Swift.Int, loc "add.swift":1:5, scope 1 // id: %2
    %3 = global_addr @add.x : Swift.Int : $*Int, loc "add.swift":1:5, scope 1 // users: %23, %
    // function_ref + infix(Int, Int) -> Int
    %4 = function_ref @Swift.+ infix (Swift.Int, Swift.Int) -> Swift.Int : $@convention(thin)
    // function_ref Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Int
    %5 = function_ref @Swift.Int.init (_builtinIntegerLiteral : Builtin.Int2048) -> Swift.Int
```

```
swiftc -emit-silgen add.swift | xcrun swift-demangle
```

```
// main
sil @main : $@convention(c) (Int32, UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>,
bb0(%0 : $Int32, %1 : $UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>>):
    alloc_global @add.x : Swift.Int, loc "add.swift":1:5, scope 1 // id: %2
    %3 = global_addr @add.x : Swift.Int : $*Int, loc "add.swift":1:5, scope 1 // users: %23, %
    // function_ref + infix(Int, Int) -> Int
    %4 = function_ref @Swift.+ infix (Swift.Int, Swift.Int) -> Swift.Int : $@convention(thin)
    // function_ref Int.init(_builtinIntegerLiteral : Builtin.Int2048) -> Int
    %5 = function_ref @Swift.Int.init (_builtinIntegerLiteral : Builtin.Int2048) -> Swift.Int
```

```
swiftc -emit-sil add.swift | xcrun swift-demangle
```

```
swiftc -emit-ir add.swift | xcrun swift-demangle
```

```
➔ swiftc -emit-ir add.swift | xcrun swift-demangle
; ModuleID = '-'
source_filename = "-"
target datalayout = "e-m:o-i64:64-f80:128-n8:16:32:64-S128"
target triple = "x86_64-apple-macosx10.9"

%Si = type <( i64 )>
%struct._SwiftEmptyArrayStorage = type { %struct.HeapObject, %struct._SwiftArrayBodyStorage }
%struct.HeapObject = type { %struct.HeapMetadata*, %struct.StrongRefCount, %struct.WeakRefCount }
%struct.HeapMetadata = type opaque
%struct.StrongRefCount = type { i32 }
%struct.WeakRefCount = type { i32 }
%struct._SwiftArrayBodyStorage = type { i64, i64 }
%swift.type = type { i64 }
%swift.protocol = type { i8*, i8*, i8*, i8*, i8*, i8*, i8*, i8*, i32, i32, i16, i16, i32 }
%swift.bridge = type opaque
%Any = type { [24 x i8], %swift.type* }

@add.x : Swift.Int = hidden global %Si zeroinitializer, align 8
 @_swiftEmptyArrayStorage = external global %struct._SwiftEmptyArrayStorage, align 8
@lazy cache variable for type metadata for Any = linkonce_ode hidden global %swift.type* null, align 8
 @_swift_getExistentialTypeMetadata = external global %swift.type* (i64, %swift.protocol**)
@type metadata for Swift.Int = external global %swift.type, align 8
@__swift_reflection_version = linkonce_ode hidden constant i16 1
@llvm.used = appending global [1 x i8*] [i8* bitcast (i16* @_swift_reflection_version to i8*)], section "llvm.metadata", align 8

define i32 @main(i32, i8**) #0 {
entry:
    %2 = bitcast i8** %1 to i8*
    store i64 2, i64* getelementptr inbounds (%Si, %Si* @add.x : Swift.Int, i32 0, i32 0), align 8
    %3 = call %swift.type* @type metadata accessor for Any() #5
    %4 = call { %swift.bridge*, i8* } @Swift._allocateUninitializedArray <@> (Builtin.Word) -> ([A], Builtin.RawPointer)(i64 1, %swift.type* %3)
    %5 = extractvalue { %swift.bridge*, i8* } %4, 0
    %6 = extractvalue { %swift.bridge*, i8* } %4, 1
    %7 = bitcast i8* %6 to %Any*
    %8 = getelementptr inbounds %Any, %Any* %7, i32 0, i32 1
    store %swift.type* @type metadata for Swift.Int, %swift.type** %8, align 8
    %9 = getelementptr inbounds %Any, %Any* %7, i32 0, i32 0
    %object = bitcast [24 x i8]* %9 to %Si*
    %10 = load i64, i64* getelementptr inbounds (%Si, %Si* @add.x : Swift.Int, i32 0, i32 0), align 8
    %object._value = getelementptr inbounds %Si, %Si* %object, i32 0, i32 0
    store i64 %10, i64* %object._value, align 8
    %11 = call { i64, i64, i64 } @Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 1)()
    %12 = extractvalue { i64, i64, i64 } %11, 0
    %13 = extractvalue { i64, i64, i64 } %11, 1
    %14 = extractvalue { i64, i64, i64 } %11, 2
    %15 = call { i64, i64, i64 } @Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 2)()
    %16 = extractvalue { i64, i64, i64 } %15, 0
    %17 = extractvalue { i64, i64, i64 } %15, 1
    %18 = extractvalue { i64, i64, i64 } %15, 2
    call void @Swift.print ([Any], separator : Swift.String, terminator : Swift.String) -> ()(%swift.bridge* %5, i64 %12, i64 %13, i64 %14, i64 %16, i64 %17, i64 %18)
    ret i32 0
}

; Function Attrs: noinline
declare void @Swift.print ([Any], separator : Swift.String, terminator : Swift.String) -> ()(%swift.bridge*, i64, i64, i64, i64, i64) #1

declare { %swift.bridge*, i8* } @Swift._allocateUninitializedArray <@> (Builtin.Word) -> ([A], Builtin.RawPointer)(i64, %swift.type*) #0

; Function Attrs: nounwind readnone
define linkonce_ode hidden %swift.type* @type metadata accessor for Any() #2 {
entry:
    %protocols = alloca [0 x %swift.protocol*], align 8
```

swiftc -emit-assembly add.swift | xcrun swift-demangle

```
↳ swift-test swiftc -emit-assembly add.swift | xcrun swift-demangle
    .section      __TEXT,__text,regular,pure_instructions
    .macosx_version_min 10, 9
    .globl  _main
    .p2align     4, 0x90
_main:
    .cfi_startproc
    pushq  %rbp
.Ltmp0:
    .cfi_offset %rbp, 16
.Ltmp1:
    .cfi_offset %rbp, -16
    movq  %rsp, %rbp
.Ltmp2:
    .cfi_def_cfa_register %rbp
    subq  $80, %rsp
    movq  $2, _add.x : Swift.Int(%rip)
    movl  %edi, -4(%rbp)
    movq  %rsi, -16(%rbp)
    callq _type metadata accessor for Any
    movl  $1, %edi
    movq  %rax, %rsi
    callq _Swift._allocateUninitializedArray <A> (Builtin.Word) -> ([A], Builtin.RawPointer)
    movq  _type metadata for Swift.Int@GOTPCREL(%rip), %rsi
    movq  %rsi, 24(%rdx)
    movq  _add.x : Swift.Int(%rip), %rsi
    movq  %rsi, (%rdx)
    movq  %rax, -24(%rbp)
    callq _Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 1)
    movq  %rax, -32(%rbp)
    movq  %rdx, -40(%rbp)
    movq  %rcx, -48(%rbp)
    callq _Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 2)
    movq  -24(%rbp), %rdi
    movq  -32(%rbp), %rsi
    movq  -40(%rbp), %r8
    movq  %rdx, -56(%rbp)
    movq  %r8, %rdx
    movq  -48(%rbp), %r9
    movq  %rcx, -64(%rbp)
    movq  %r9, %rcx
    movq  %rax, %r8
    movq  -56(%rbp), %r9
    movq  -64(%rbp), %rax
    movq  %rax, (%rsp)
    callq _Swift.print ([Any], separator : Swift.String, terminator : Swift.String) -> ()
    xorl  %eax, %eax
    addq  $80, %rsp
    popq  %rbp
    retq
    .cfi_endproc

.private_extern _type metadata accessor for Any
.globl  _type metadata accessor for Any
.weak_definition _type metadata accessor for Any
.p2align     4, 0x90
_type metadata accessor for Any:
    pushq  %rbp
    movq  %rsp, %rbp
    subq  $16, %rsp
    movq  _lazy cache variable for type metadata for Any(%rip), %rax
    cmpq  $0, %rax
    movq  %rax, -16(%rbp)
    jne   LBB1_2
    xorl  %eax, %eax
    movl  %eax, %edi
    leaq  -8(%rbp), %rsi
    callq _swift_rt_swift_getExistentialTypeMetadata
    movq  %rax, %rsi
    movq  %rax, _lazy cache variable for type metadata for Any(%rip)
    movq  %rsi, -16(%rbp)
LBB1_2:
    movq  -16(%rbp), %rax
    addq  $16, %rsp
    popq  %rbp
    retq

.private_extern _swift_rt_swift_getExistentialTypeMetadata
.globl  _swift_rt_swift_getExistentialTypeMetadata
.weak_definition _swift_rt_swift_getExistentialTypeMetadata
.p2align     4, 0x90
.swift_rt_swift_getExistentialTypeMetadata:
    movq  __swift_getExistentialTypeMetadata@GOTPCREL(%rip), %rax
    movq  (%rax), %rax
    jmpq  *%rax

.private_extern _add.x : Swift.Int
.globl  _add.x : Swift.Int
.zerofill __DATA,__common,_add.x : Swift.Int,8,3
.private_extern _lazy cache variable for type metadata for Any
.section      __DATA,__data
.globl  _lazy cache variable for type metadata for Any
.weak_definition _lazy cache variable for type metadata for Any
.p2align     3
.lazy cache variable for type metadata for Any:
    .quad  0

.private_extern __swift_reflection_version
.section      __TEXT,__const
.globl  __swift_reflection_version
.weak_definition __swift_reflection_version
.p2align     1
__swift_reflection_version:
    .short  1

.no_dead_strip __swift_reflection_version
.linker_option "-lswiftCore"
.linker_option "-lobjc"
.section      __DATA,__objc_imageinfo,regular,no_dead_strip
L_OBJC_IMAGE_INFO:
    .long  0
    .long  1088

.subsections_via_symbols
```

nicoleorchard.com/blog

An Intro to Compilers

How to Speak to Computers, Pre-Siri

August 13, 2017

tl;dr: Learning new meanings for front-end and back-end.

A compiler is just a program that translates other programs. Traditional compilers translate source code into executable machine code that your computer understands. (Some compilers translate source code into another programming language. These compilers are called source-to-source translators or transpilers.)

LLVM is a widely used compiler project, consisting of many modular compiler tools.

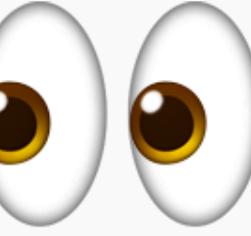
Traditional compiler design comprises three parts:

```
graph LR; SC[SOURCE CODE] --> FE[FRONTEND]; FE --> IR1[IR]; IR1 --> OPT[OPTIMIZER]; OPT --> IR2[IR]; IR2 --> BE[BACKEND]; BE --> MC[MACHINE CODE]
```

The diagram illustrates the traditional three-stage architecture of a compiler. It is labeled "COMPILER" at the top. Stage 1, the "FRONTEND", is represented by an orange box. Stage 2, the "OPTIMIZER", is represented by a grey box. Stage 3, the "BACKEND", is represented by a green box. Arrows indicate the flow of data from "SOURCE CODE" through the "FRONTEND" to produce "IR" (Intermediate Representation), then through the "OPTIMIZER" to produce another "IR", and finally through the "BACKEND" to produce "MACHINE CODE".

<https://nicoleorchard.com/blog/compilers>

Swift 5



GitHub, Inc. github.com/apple/swift-evolution/blob/9c

Development major version: Swift 5.0

Expected release date: Late 2018

Primary Focus: ABI Stability

The Swift 5 release will provide [ABI stability](#) for the Swift Standard Library. ABI stability enables OS vendors to embed a Swift Standard Library and runtime in the OS that is compatible with applications built with Swift 5 or later. Progress towards achieving ABI stability will be tracked at a high level on the [ABI Dashboard](#).

ABI stability is only one of two pieces needed to support binary frameworks. The second half is *module stability* (see "[The Big Picture](#)" of the [ABI Stability Manifesto](#) for more information). While we'd like to support this for Swift 5, it will be a stretch goal, and may not make it in time.

The need to achieve ABI stability in Swift 5 will guide most of the priorities for the release. In addition, there are important goals to complete that carry over from Swift 4 that are prerequisites to locking down the ABI of the standard library:

- **Generics features needed for standard library.** We will finish implementing [conditional conformances](#) and [recursive protocol requirements](#), which are needed for the standard library to achieve ABI stability. Both of these have gone through the evolution proposal process and there are no known other generics enhancements needed for ABI stability.
- **API resilience.** We will implement the essential pieces need to support API resilience, in order to allow public APIs for a library to evolve over time while maintaining a stable ABI.
- **Memory ownership model.** An (opt-in) Cyclone/Rust-inspired memory [ownership model](#) is strongly desirable for systems programming and for other high-performance applications that require predictable and deterministic performance. Part of this model was introduced in Swift 4 when we began to enforce [exclusive access to memory](#). In Swift 5 our goal is to tackle the [pieces of the ownership model that are key to ABI stability](#).

Other Improvements

Beyond ABI stability (which focuses mostly on getting a bunch of low-level implementation details of the language finalized), in Swift 5 the evolution process welcomes additions that improve the overall usability of the language and standard library, including but not restricted to:

<https://github.com/apple/swift-evolution>

Swift 5

- ABI stability *required*
- Generics and String features for stdlib
- Memory ownership model
- Source breaking changes have even higher bar than Swift 4
- Every evolution proposal requires *working implementation with test cases*

No-one has any idea
what they're doing



▶ ▶| 🔍 16:30 / 38:26

CC HD 🎞️ 📺 🎵

Do iOS 2016 - Ayaka Nonaka, Contributing to Swift



Xebia Group International

Subscribe 326

659 views

+ Add to

Share

More

17 0

<https://www.youtube.com/watch?v=GnT2ZeHVJe4>

The image is a composite of two parts. On the left, a young man with short brown hair and a black t-shirt is speaking into a microphone at a conference. He has a tattoo on his right arm. The background features a green and white checkered banner with logos for try! Swift, Realm, and CyberAgent. On the right, a screenshot of a presentation slide from Speaker Deck. The slide has a dark header with 'Speaker Deck' and 'Talk by Jesse Squires'. The main title 'CONTRIBUTING TO OPEN SOURCE SWIFT' is in large, bold, green capital letters. Below the title, 'JESSE SQUIRES' is written in smaller white capital letters. At the bottom, there's a decorative bar made of a grid of green squares.

Jesse Squires - Contributing to Open Source Swift

<https://news.realm.io/news/tryswift-jesse-squires-contributing-open-source-swift/>



Speaker Deck

Talk by Realm

CONTRIBUTING TO SWIFT

Russ Bishop
My views are my own and are not endorsed by my employer in any way.

share

Russ Bishop - Contributing to Swift: From Proposal to Shipped

<https://news.realm.io/news/slug-russ-bishop-contributing-open-source-swift-proposal/>







Thanks!
from @neilkimmett