# 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 = .zero
```

**Neil Kimmett** 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 Kimmett** 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

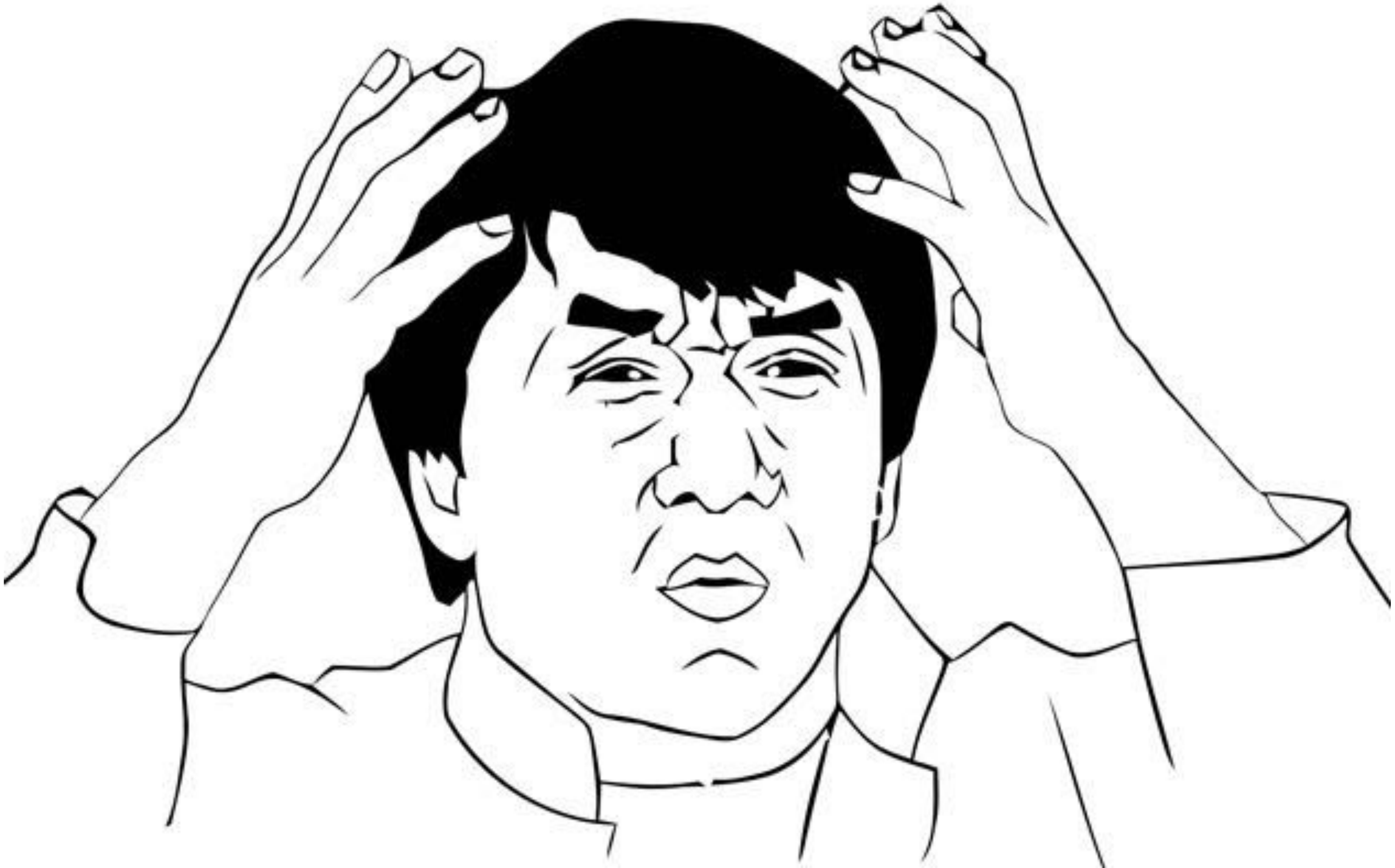
2:05 PM

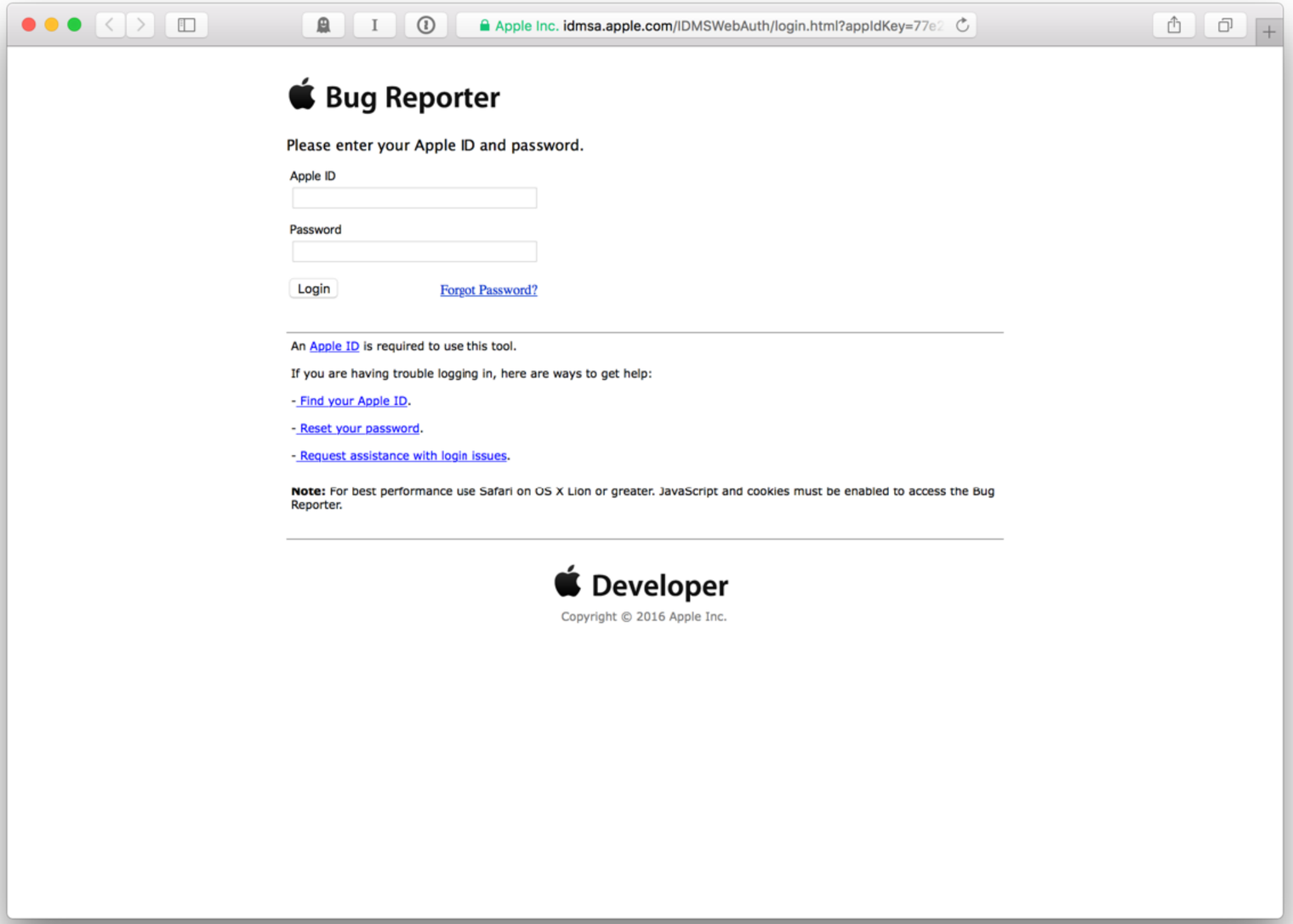
apple

of person you could make a radar

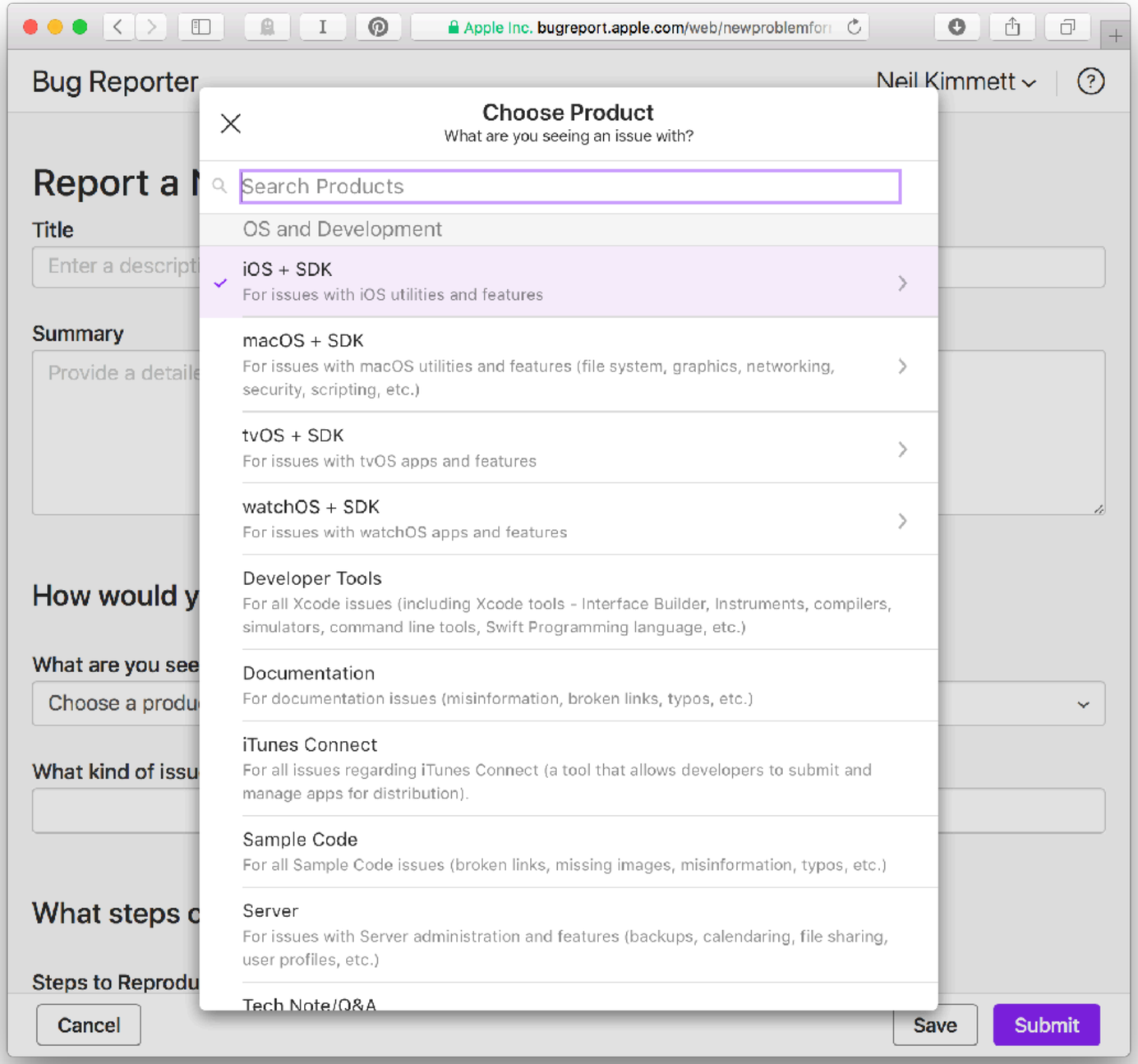
u could make a radar

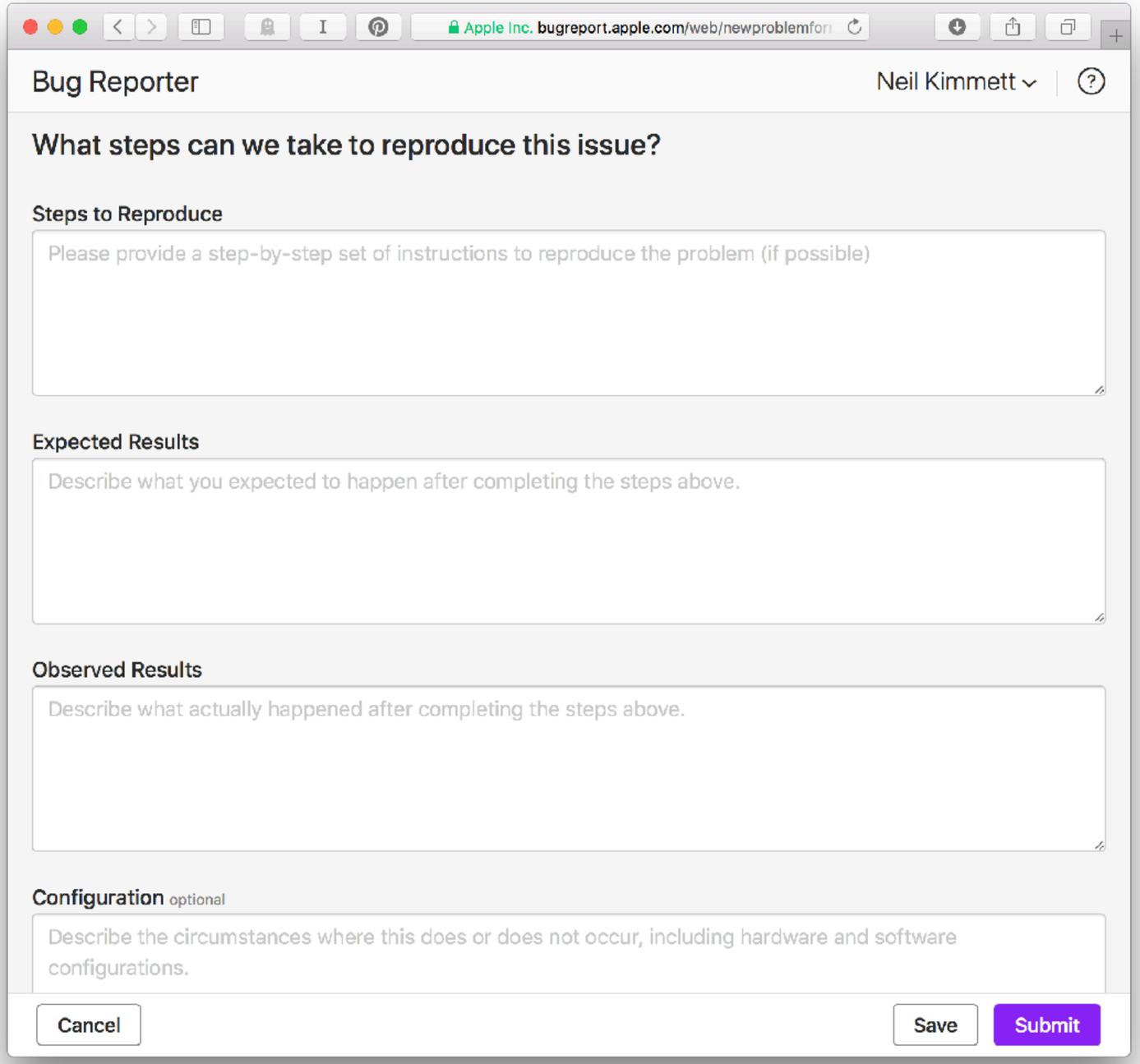
a radar





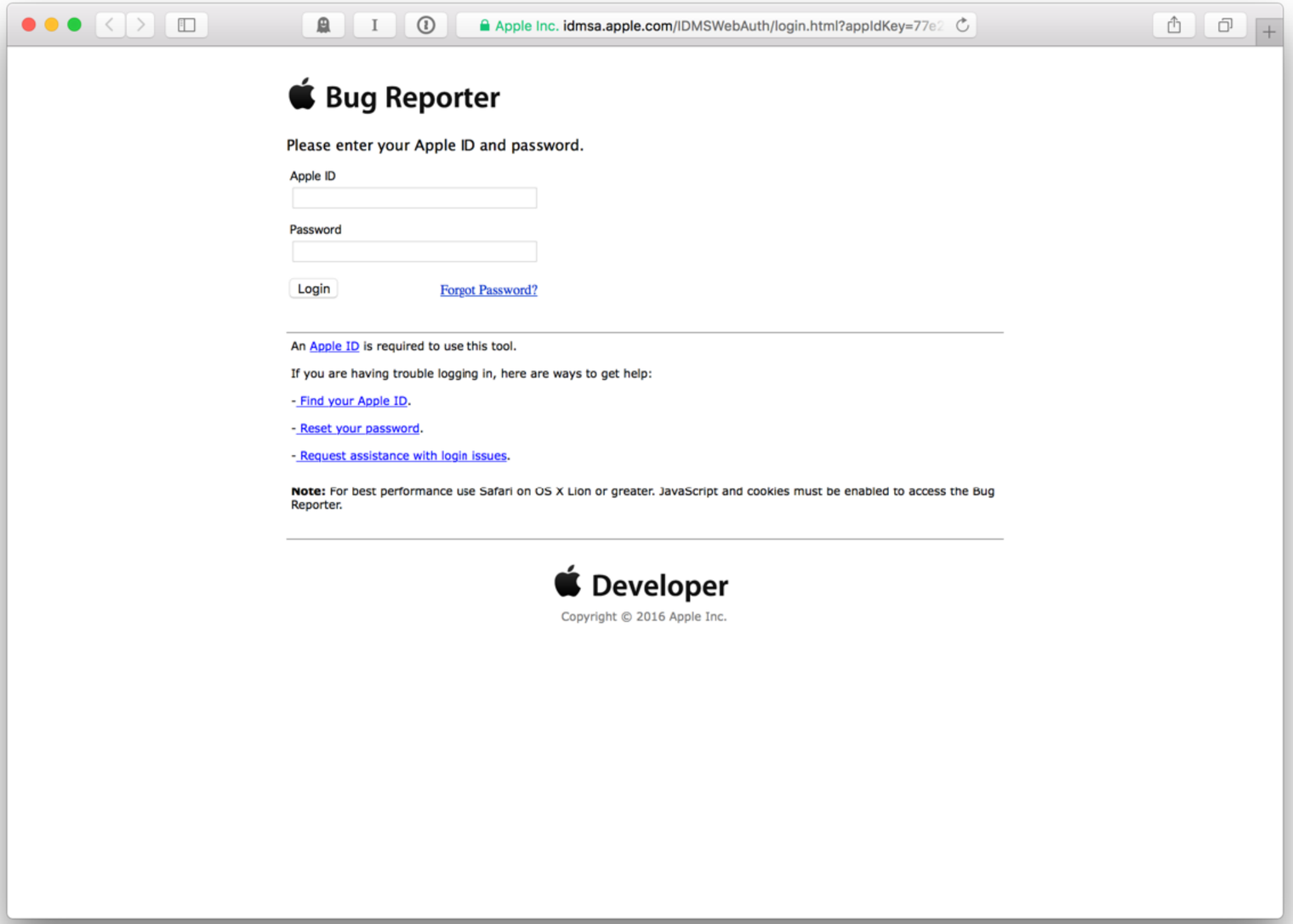

http://bugreport.apple.com/


Apple Inc. bugreport.apple.com/web/newproblemfor
Bug Reporter
Neil Kimmett
Report a
Title
Enter a descript
Summary
Provide a detail
How would y
What are you see
Choose a produ
What kind of issu
What steps c
Steps to Reprodu
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/newproblemfor
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

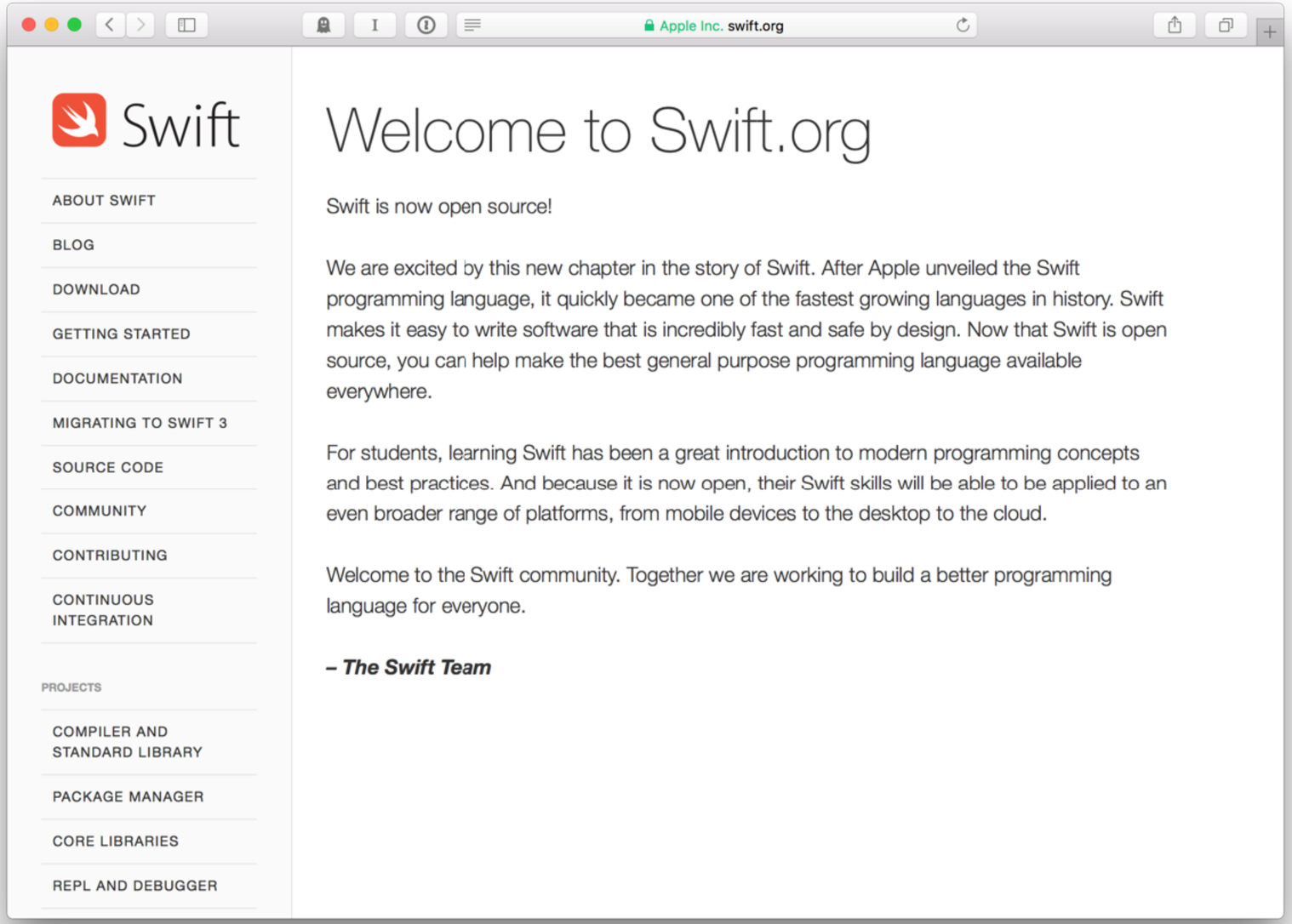
**Apple Developer Relations** April 9 2015, 11:15 PM

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

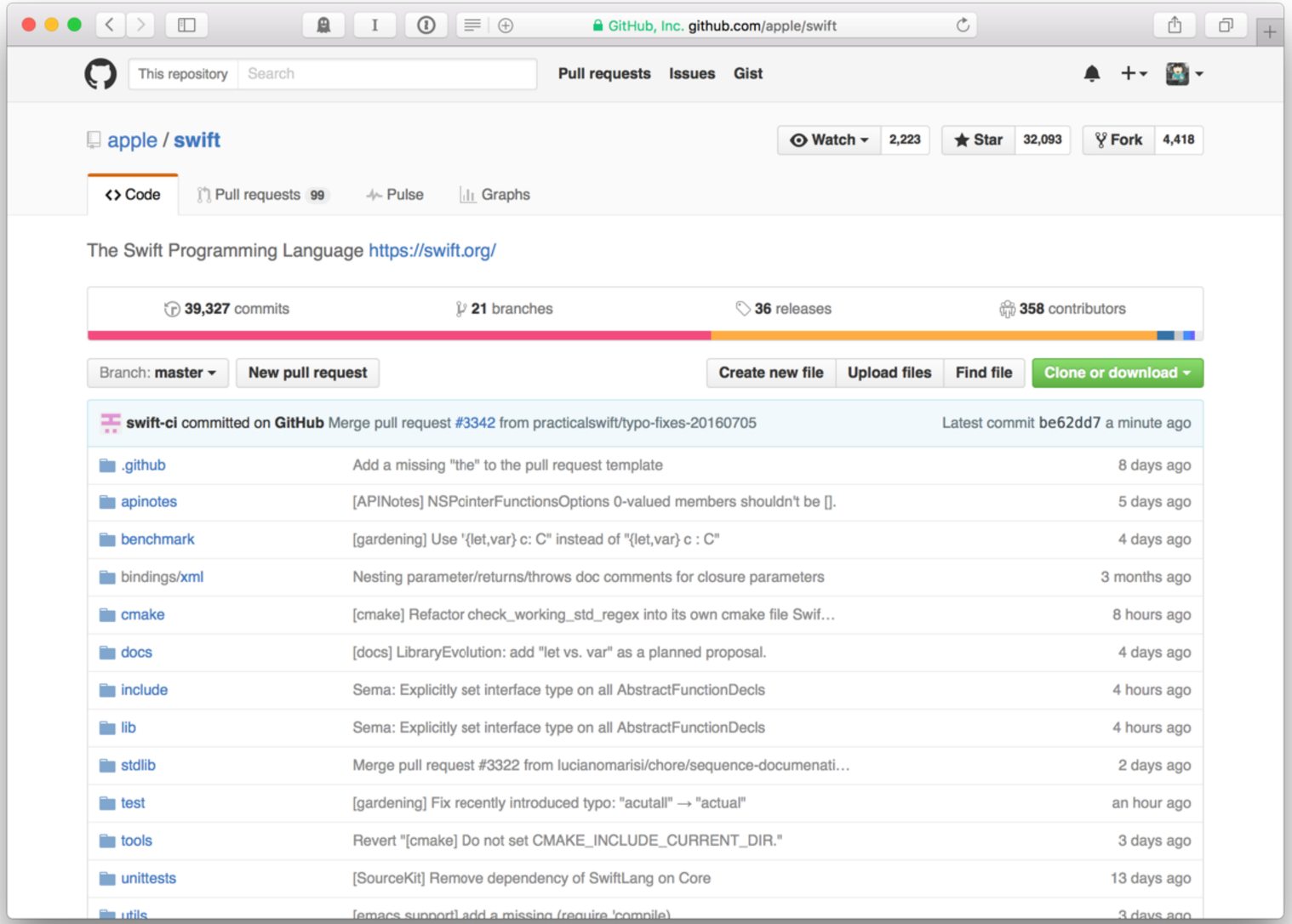



http://bugreport.apple.com/


THAT GIVES ME AN IDEA.




https://swift.org




https://github.com/apple/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 Kimmett** 7:07 AM

wait a second

fuck radar

https://github.com/apple/swift/blob/4d0c060e2189571b0103fa5473ed8d8145d0e231/stdlib/public/SDK/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**

```
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 {}
```

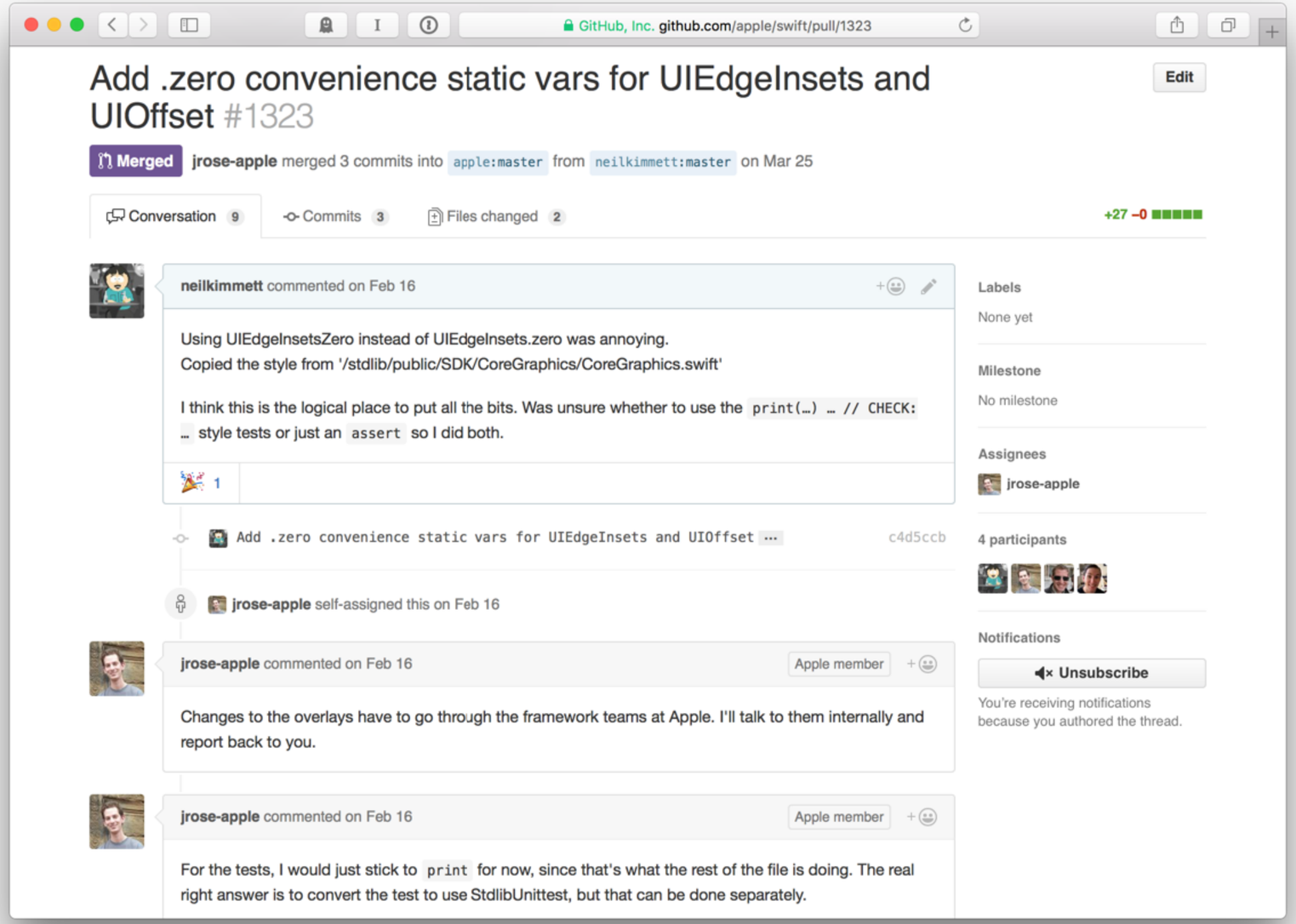
```
13   import Foundation
14   @_exported import UIKit
15
16  +//===----------------------------------------------------------------------===//
17  +// UIGeometry
18  +//===----------------------------------------------------------------------===//
19  +
20  +public extension UIEdgeInsets {
21  +  static 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
```

**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
```




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!

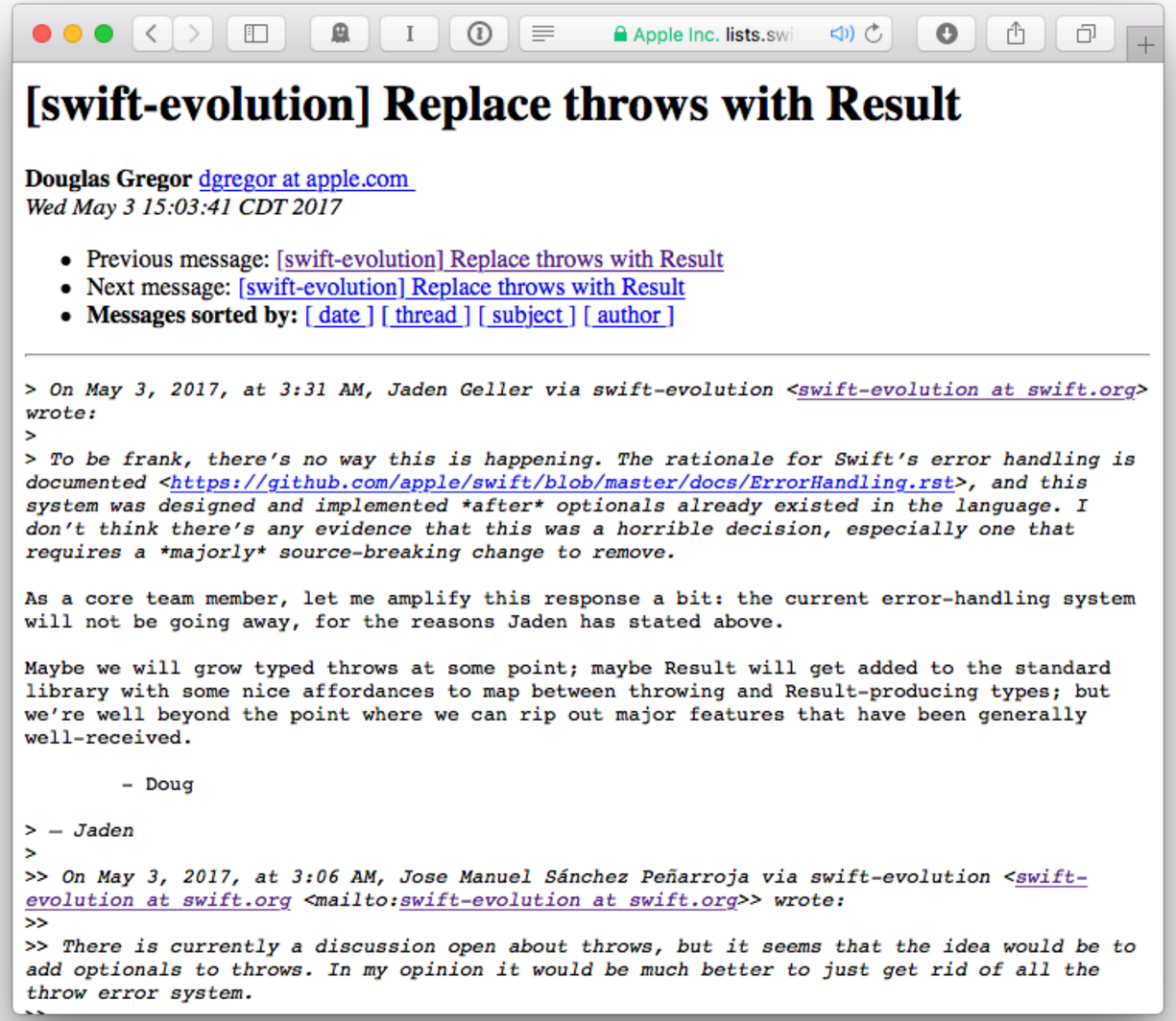

BETA
XCODE

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

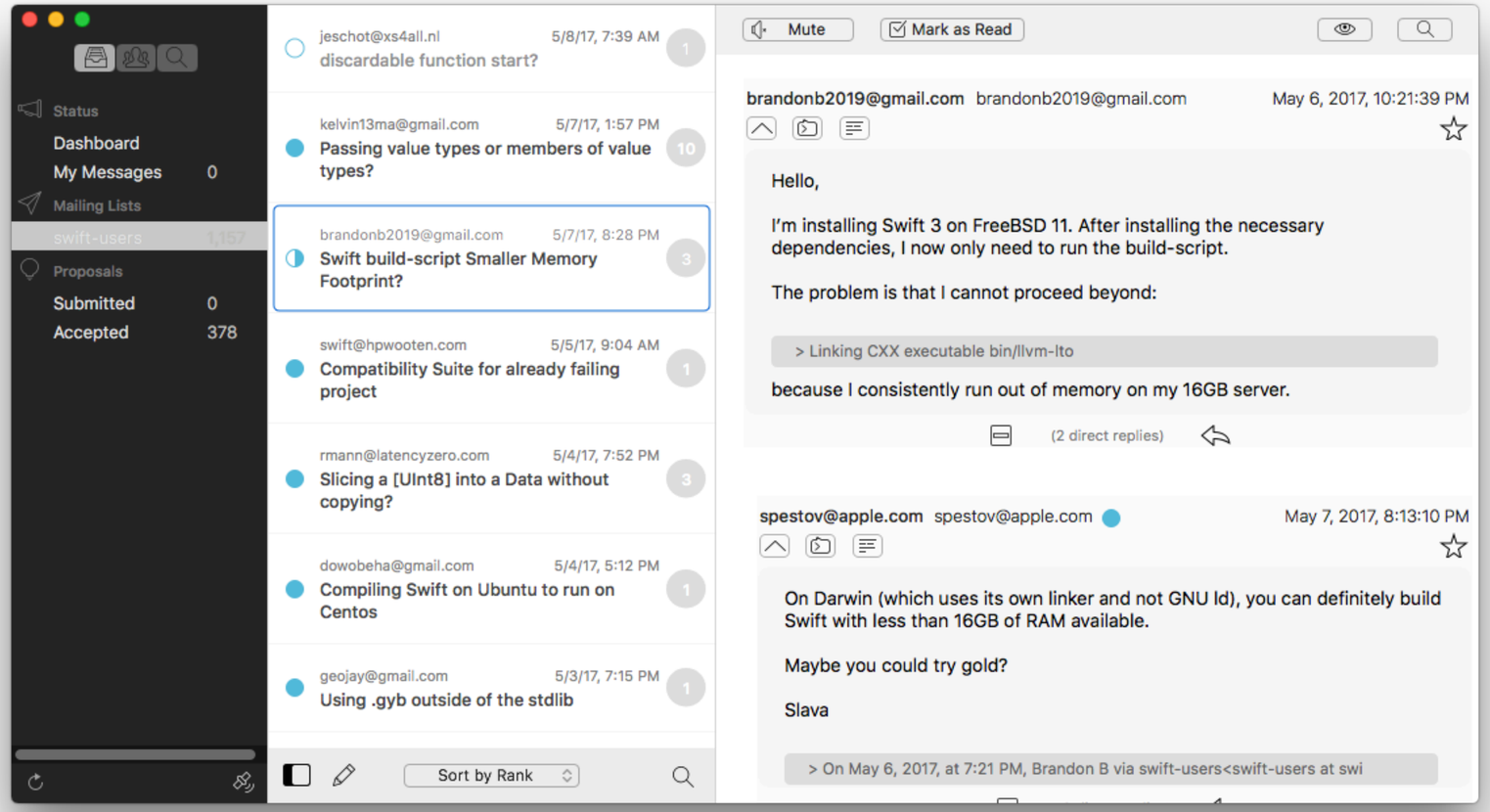
UIEdgeInsets **zero**

# What can I do?

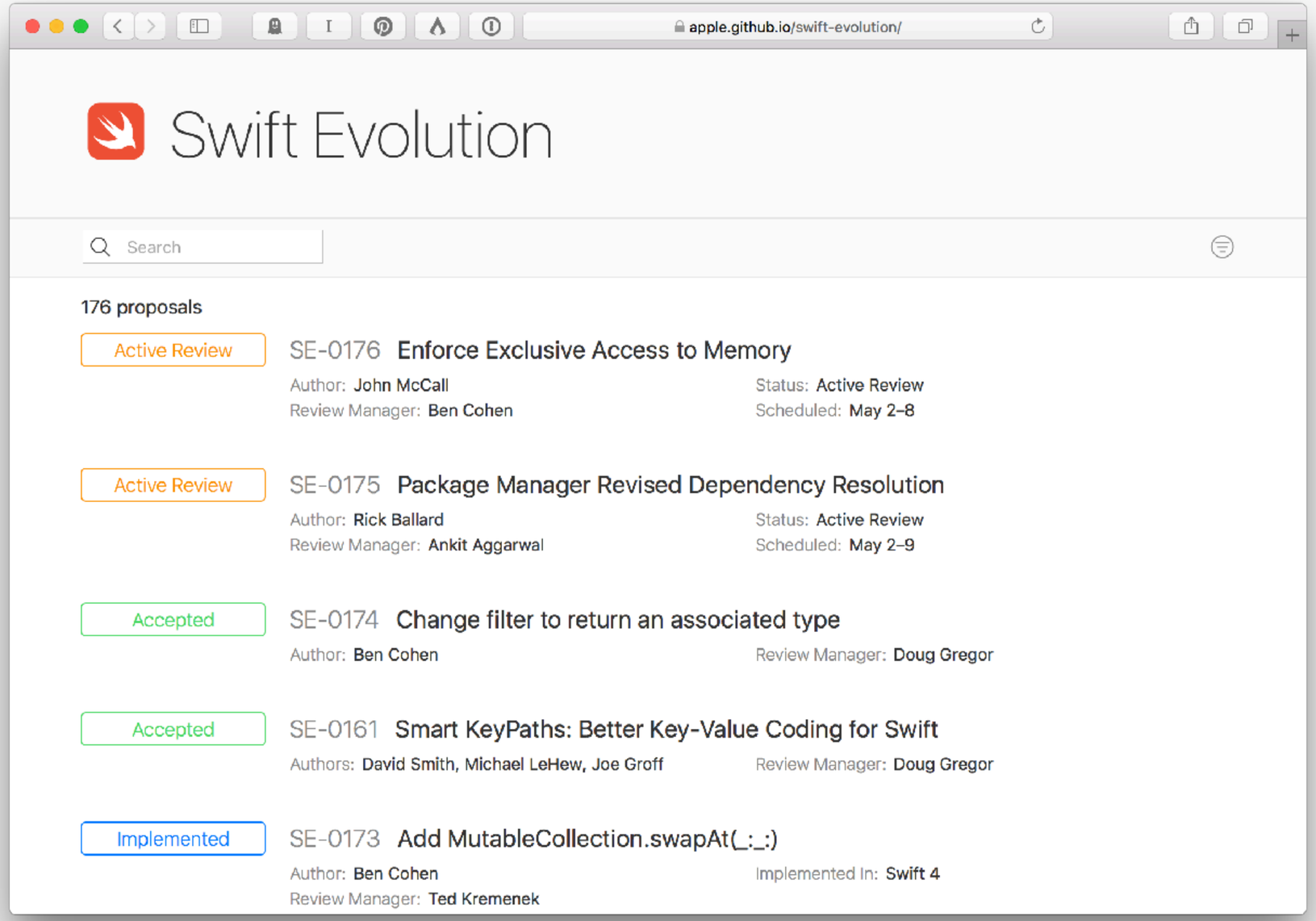
*“To be a truly great community, Swift.org needs to welcome developers from all walks of life, with different backgrounds, and with a wide range of experience. A diverse and friendly community will have more great ideas, more unique perspectives, and produce more great code. We will work diligently to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



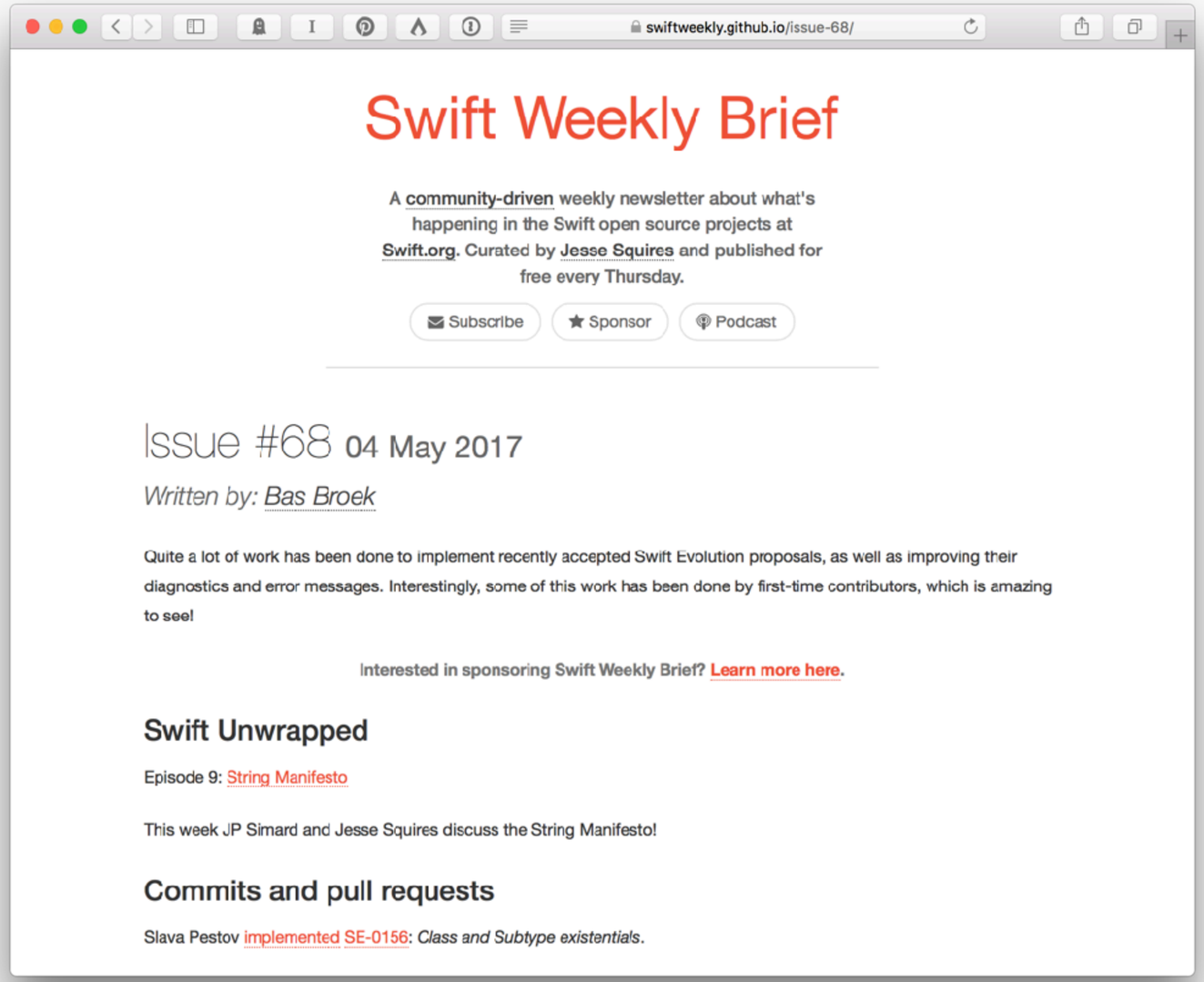

https://lists.swift.org



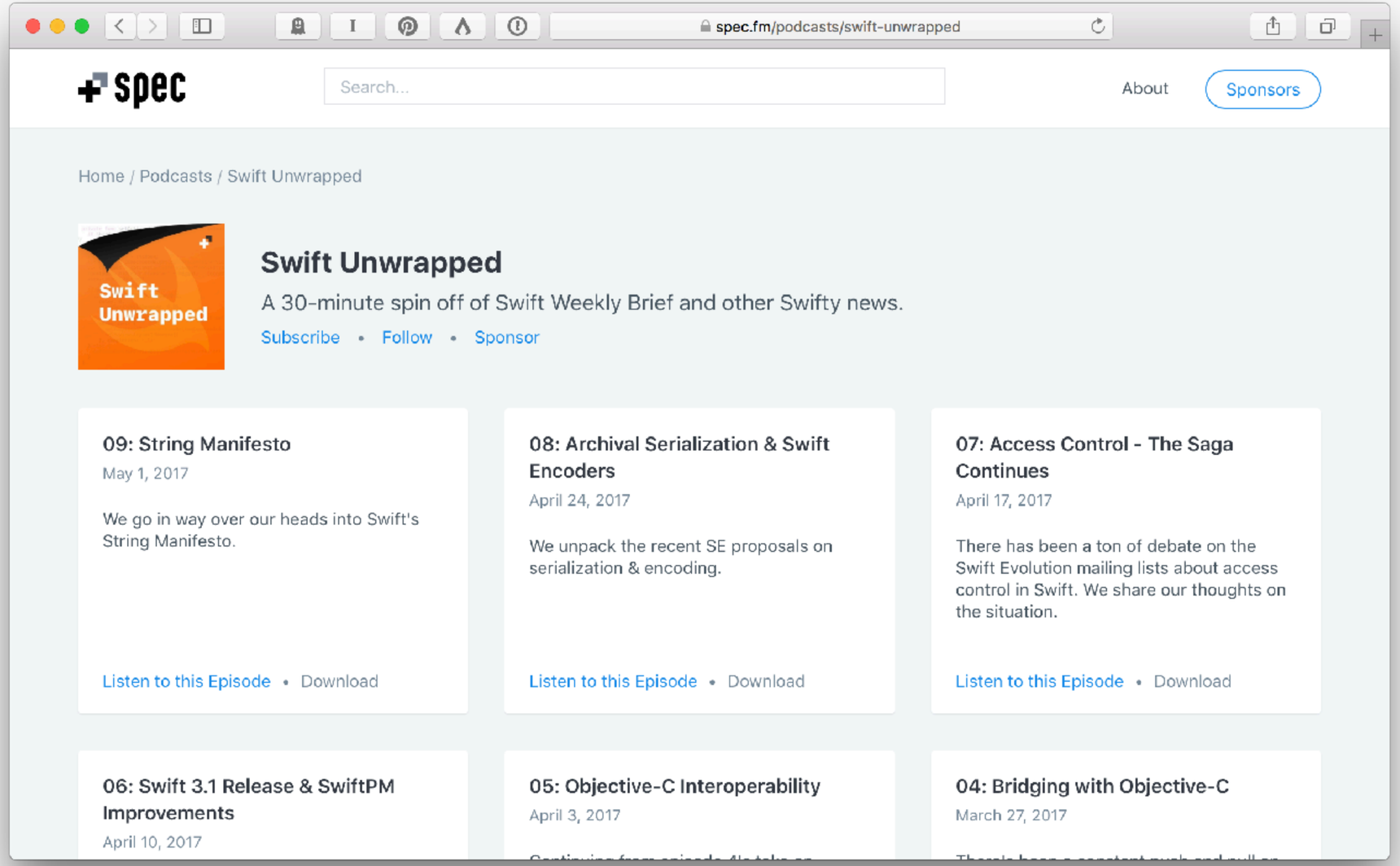

https://stylemac.com/hirundo/




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




https://swiftweekly.github.io




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**

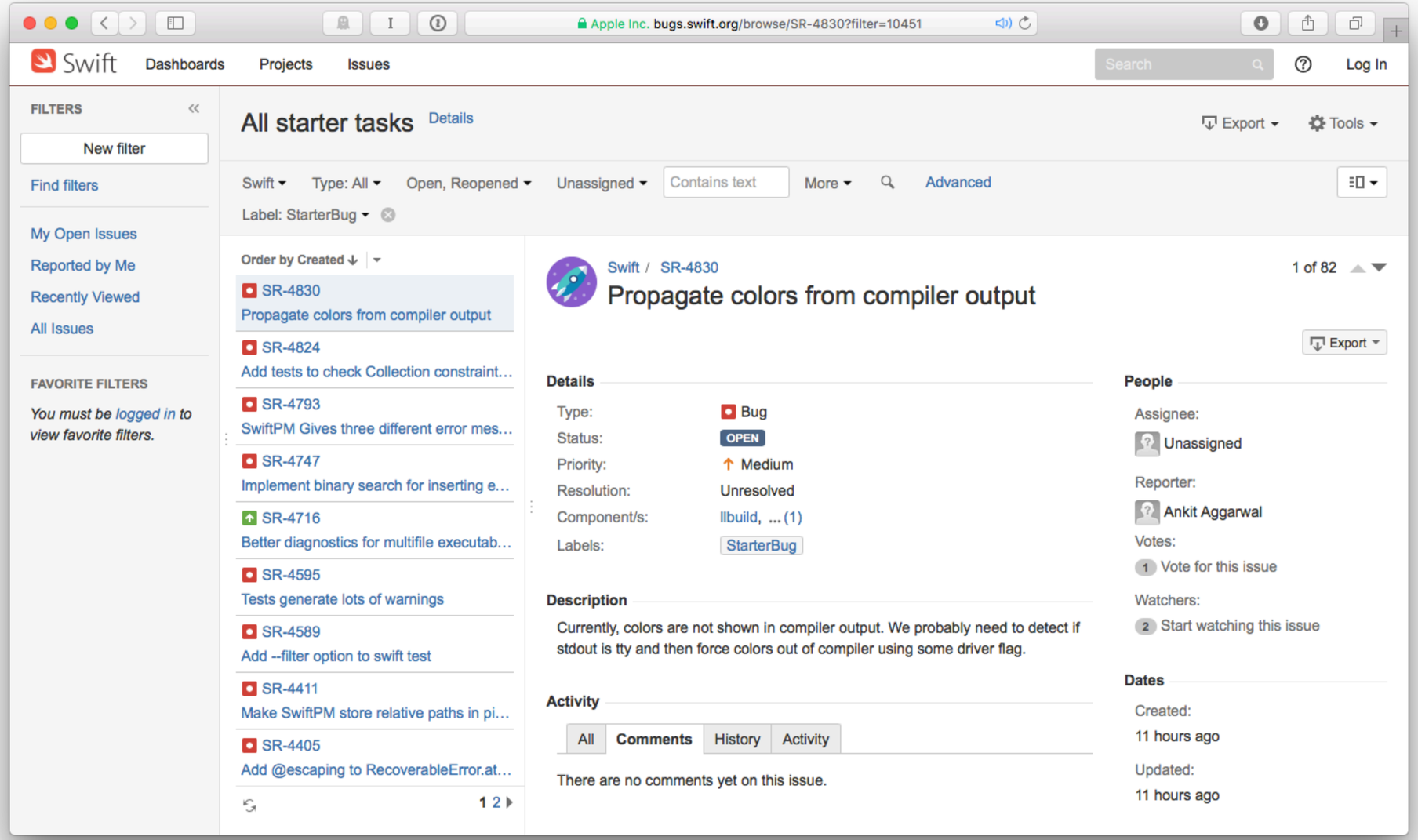
**Brian Gesiak**

@modocache

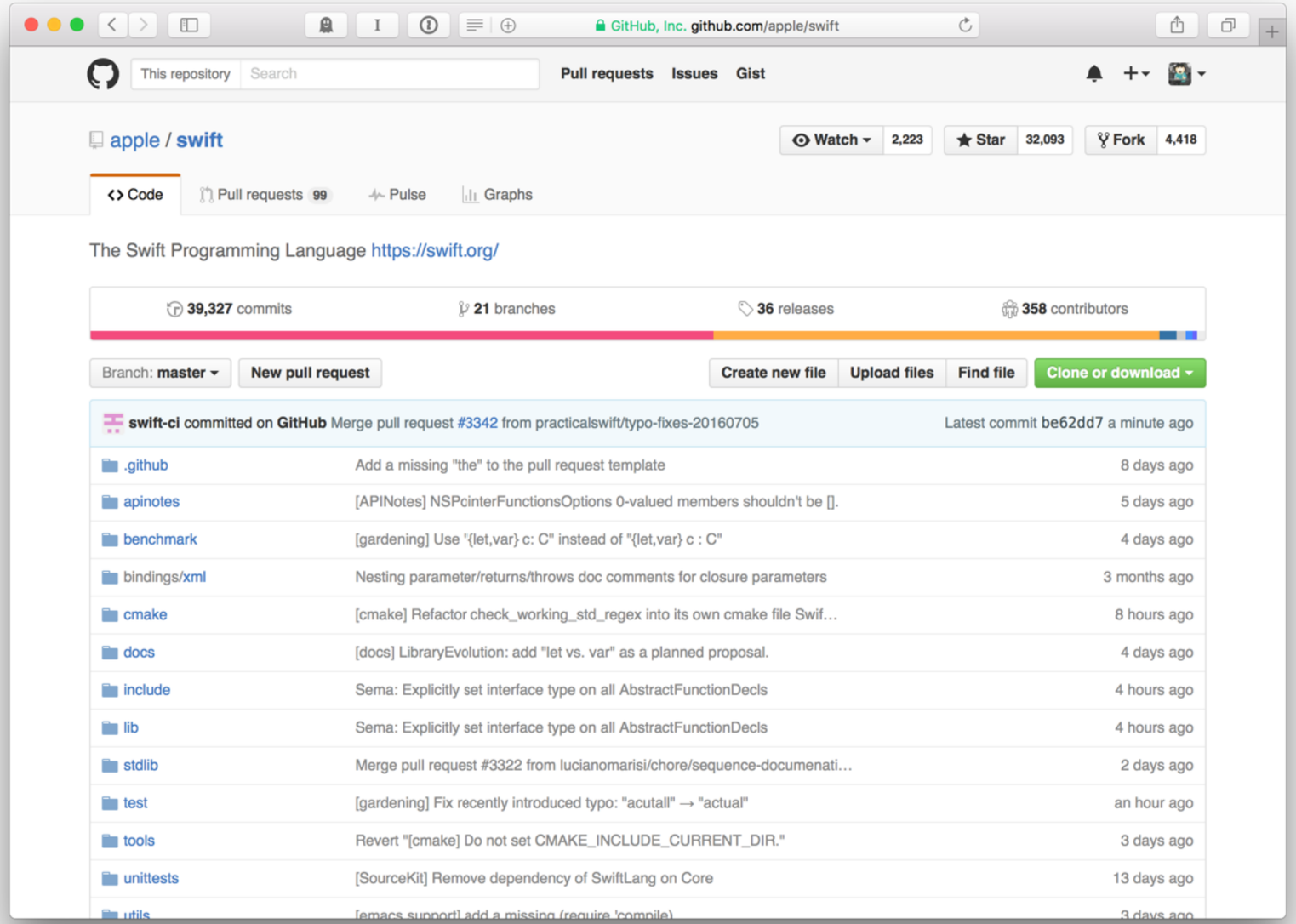
To become one of the top 100 most prolific committers to Swift, you need 15 commits. For LLVM, you'd need 130.

| RETWEETS | LIKES |
| --- | --- |
| 4 | 24 |

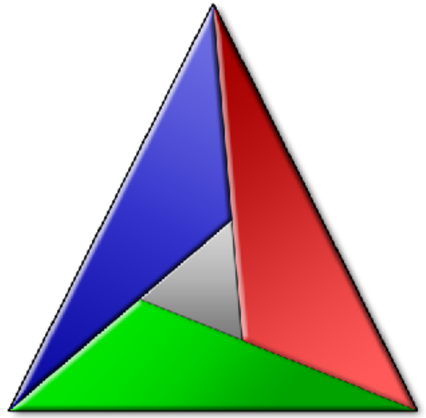
5:56 PM - 6 May 2017




https://bugs.swift.org




https://github.com/apple/swift



CMake

Cross-platform Make

# Ninja

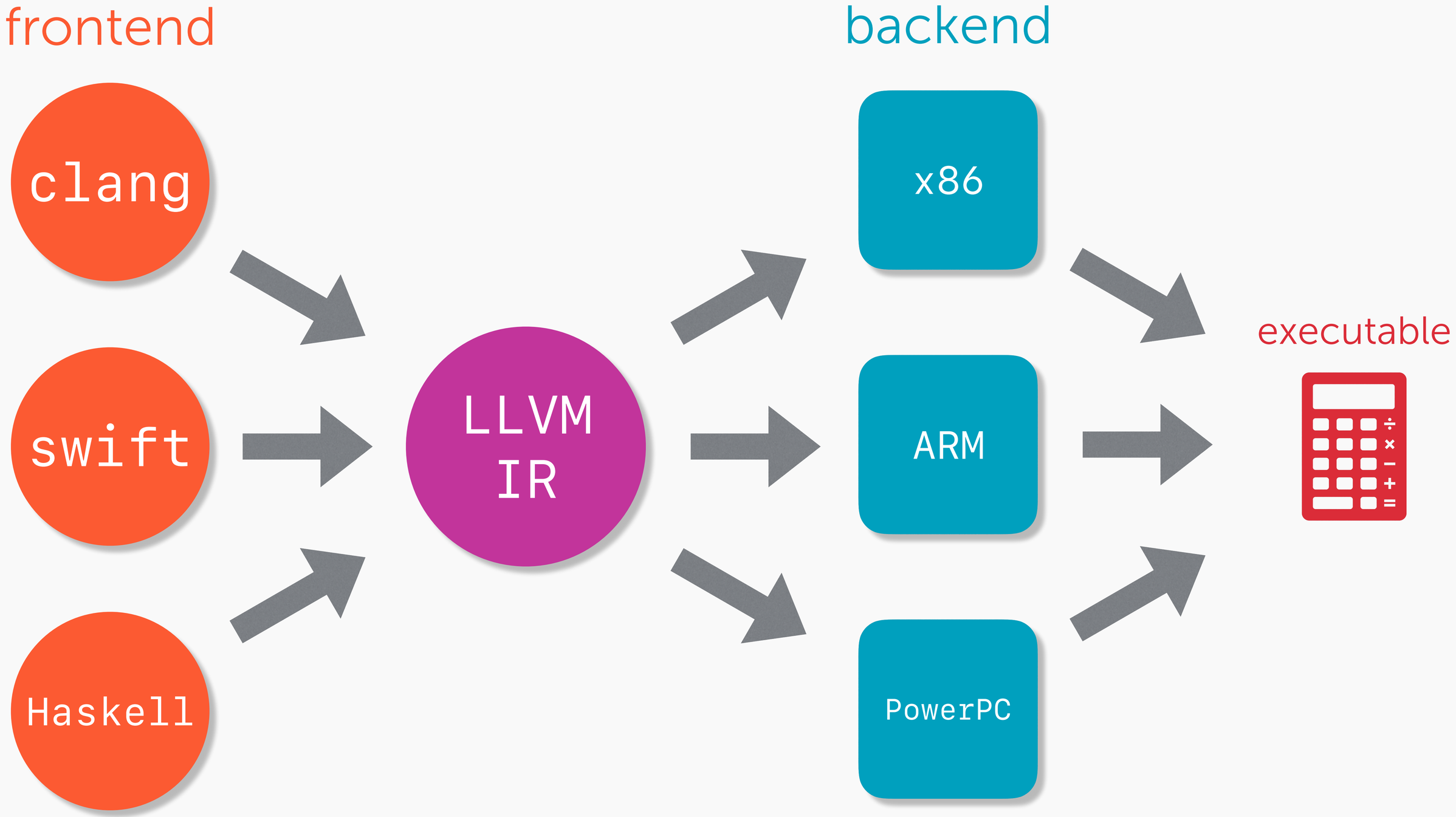
```
brew install cmake ninja
```

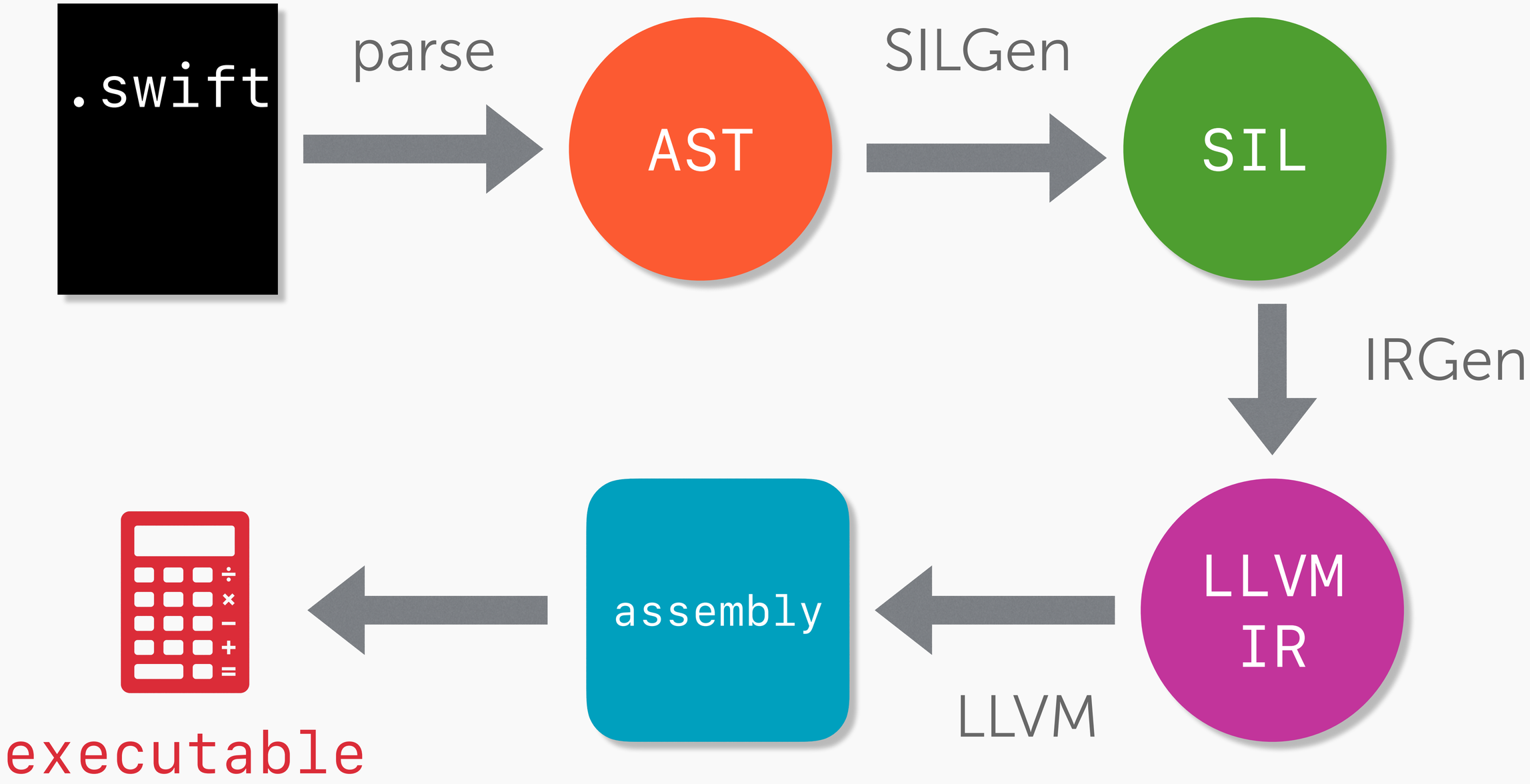
```
mkdir swift-source

cd swift-source

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

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

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

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

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

# LLVM


frontend
backend
clang
swift
Haskell
LLVM
IR
x86
ARM
PowerPC
executable


.swift
parse
AST
SILGen
SIL
IRGen
LLVM IR
LLVM
assembly
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(_builtinIntege
rLiteral:) 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(_builtinInt
egerLiteral:) 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 specia
lized=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 r
          (declref_expr type='(Int, Int) -> Int' locatio
```

# 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.x : 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.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, %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 %4(%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) -> () : $@convention(thin) (@owned Array<Any>, @owned String, @owned String) -> (), loc "add.swift":2:1, scope 1 // user: %29
  %16 = integer_literal $Builtin.Word, 1, loc "add.swift":2:7, 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) <τ_0_0> (Builtin.Word) -> (@owned Array<τ_0_0>, Builtin.RawPointer), loc "add.swift":2:7, scope 1 // user: %18
  %18 = apply %17<Any>(%16) : $@convention(thin) <τ_0_0> (Builtin.Word) -> (@owned Array<τ_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 // users: %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, $Int, loc "add.swift":2:7, scope 1 // user: %24
  %23 = load [trivial] %3 : $*Int, loc "add.swift":2:7, scope 1 // user: %24
  store %23 to [trivial] %22 : $*Int, loc "add.swift":2:7, scope 1 // id: %24
  // function_ref (print([Any], separator : String, terminator : String) -> ()).(default argument 1)
  %25 = function_ref @$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() : $@convention(thin) () -> @owned String, loc "add.swift":2:7, scope 1 // user: %28
  // function_ref (print([Any], separator : String, terminator : String) -> ()).(default argument 2)
  %27 = function_ref @$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: %29
  %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: %31
  %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] [fragile] @$Swift.+ infix (Swift.Int, Swift.Int) -> Swift.Int : $@convention(thin) (Int, Int) -> Int


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


// print([Any], separator : String, terminator : String) -> ()
sil [noinline] [_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) <τ_0_0> (Builtin.Word) -> (@owned Array<τ_0_0>, Builtin.RawPointer)


// (print([Any], separator : String, terminator : String) -> ()).(default argument 1)
sil [noinline] [_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 [noinline] [_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
```

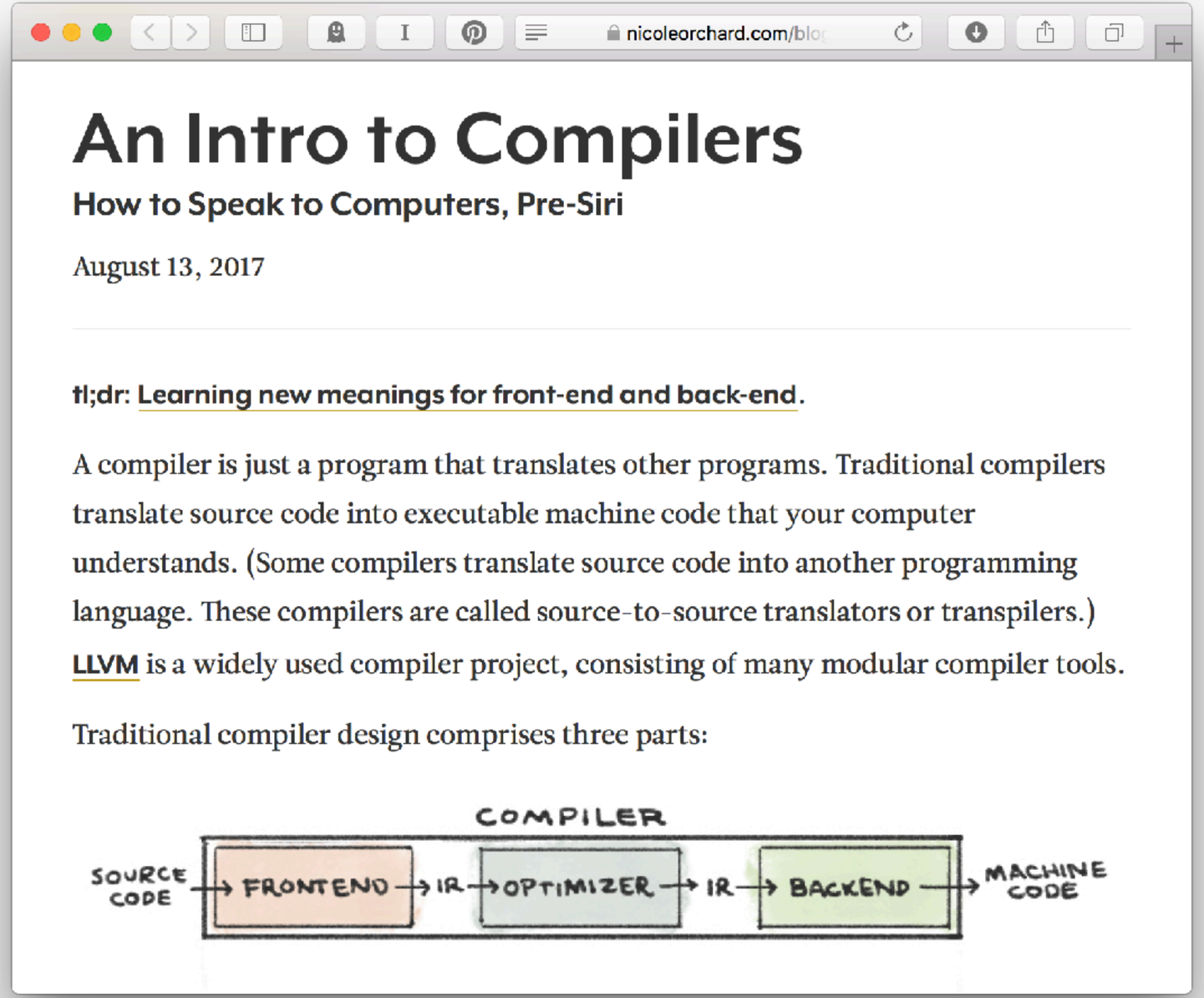
# swiftc -emit-sil add.swift | xcrun swift-demangle

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

```
➜ swift-test 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_odr 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_odr 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 <A> (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, i64) #1

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

; Function Attrs: nounwind readnone
define linkonce_odr hidden %swift.type* @type metadata accessor for Any() #2 {
entry:
  %protocols = alloca [0 x %swift.protocol*], align 8
  %0 = load %swift.type*, %swift.type** @lazy cache variable for type metadata for Any, align 8
  %1 = icmp eq %swift.type* %0, null
  br i1 %1, label %cacheIsNull, label %cont

cacheIsNull:                                      ; preds = %entry
  %2 = bitcast [0 x %swift.protocol*]* %protocols to i8*
  call void @llvm.lifetime.start(i64 0, i8* %2)
  %3 = bitcast [0 x %swift.protocol*]* %protocols to %swift.protocol**
  %4 = call %swift.type* @swift_rt_swift_getExistentialTypeMetadata(i64 0, %swift.protocol** %3) #6
  %5 = bitcast %swift.protocol** %3 to i8*
  call void @llvm.lifetime.end(i64 0, i8* %5)
  store atomic %swift.type* %4, %swift.type** @lazy cache variable for type metadata for Any release, align 8
  br label %cont

cont:                                             ; preds = %cacheIsNull, %entry
  %6 = phi %swift.type* [ %0, %entry ], [ %4, %cacheIsNull ]
  ret %swift.type* %6
}

; Function Attrs: argmemonly nounwind
declare void @llvm.lifetime.start(i64, i8* nocapture) #3

; Function Attrs: noinline nounwind
define linkonce_odr hidden %swift.type* @swift_rt_swift_getExistentialTypeMetadata(i64, %swift.protocol**) #4 {
entry:
  %load = load %swift.type* (i64, %swift.protocol**)*, %swift.type* (i64, %swift.protocol**)** @_swift_getExistentialTypeMetadata
  %2 = tail call %swift.type* %load(i64 %0, %swift.protocol** %1)
  ret %swift.type* %2
}

; Function Attrs: argmemonly nounwind
declare void @llvm.lifetime.end(i64, i8* nocapture) #3

; Function Attrs: noinline
declare { i64, i64, i64 } @Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 1)() #1

; Function Attrs: noinline
declare { i64, i64, i64 } @Swift.(print ([Any], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 2)() #1

attributes #0 = { "no-frame-pointer-elim"="true" "no-frame-pointer-elim-non-leaf" "target-cpu"="core2" "target-features"="+ssse3,+cx16,+fxsr,+mmx,+x87,+sse
attributes #1 = { noinline "no-frame-pointer-elim"="true" "no-frame-pointer-elim-non-leaf" "target-cpu"="core2" "target-features"="+ssse3,+cx16,+fxsr,+mmx,
attributes #2 = { nounwind readnone "no-frame-pointer-elim"="true" "no-frame-pointer-elim-non-leaf" "target-cpu"="core2" "target-features"="+ssse3,+cx16,+f
attributes #3 = { argmemonly nounwind }
attributes #4 = { noinline nounwind }
attributes #5 = { nounwind readnone }
attributes #6 = { nounwind }

!llvm.module.flags = !{!0, !1, !2, !3, !4, !5, !9, !10}

!0 = !{i32 1, !"Objective-C Version", i32 2}
!1 = !{i32 1, !"Objective-C Image Info Version", i32 0}
!2 = !{i32 1, !"Objective-C Image Info Section", !"__DATA, __objc_imageinfo, regular, no_dead_strip"}
!3 = !{i32 4, !"Objective-C Garbage Collection", i32 1024}
!4 = !{i32 1, !"Objective-C Class Properties", i32 64}
!5 = !{i32 6, !"Linker Options", !6}
!6 = !{!7, !8}
!7 = !{!"-lswiftCore"}
!8 = !{!"-lobjc"}
!9 = !{i32 1, !"PIC Level", i32 2}
!10 = !{i32 1, !"Swift Version", i32 4}
```

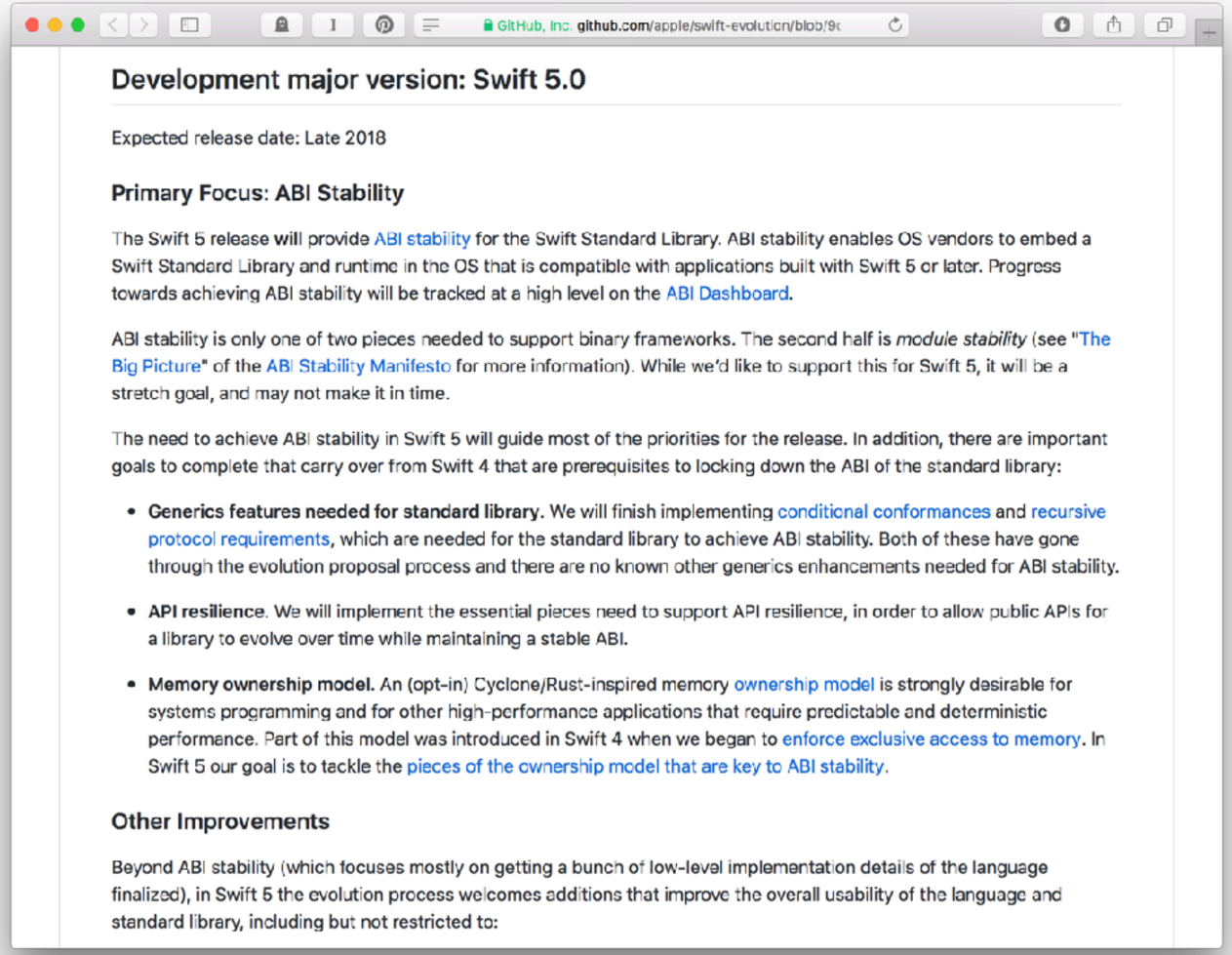
# swiftc -emit-assembly add.swift | xcrun swift-demangle

```
➜ 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_def_cfa_offset 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
```




https://nicoleorchard.com/blog/compilers

# Swift 5 👀



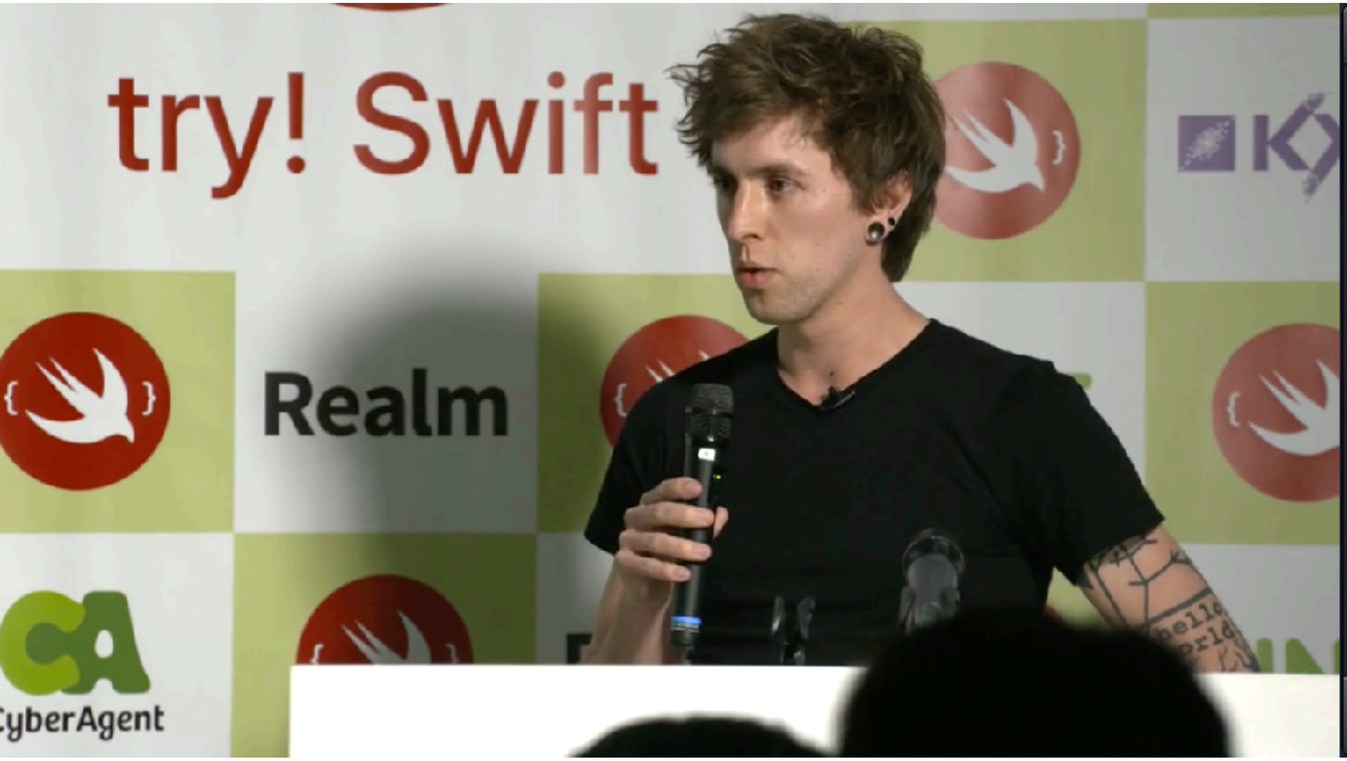

https://github.com/apple/swift-evolution

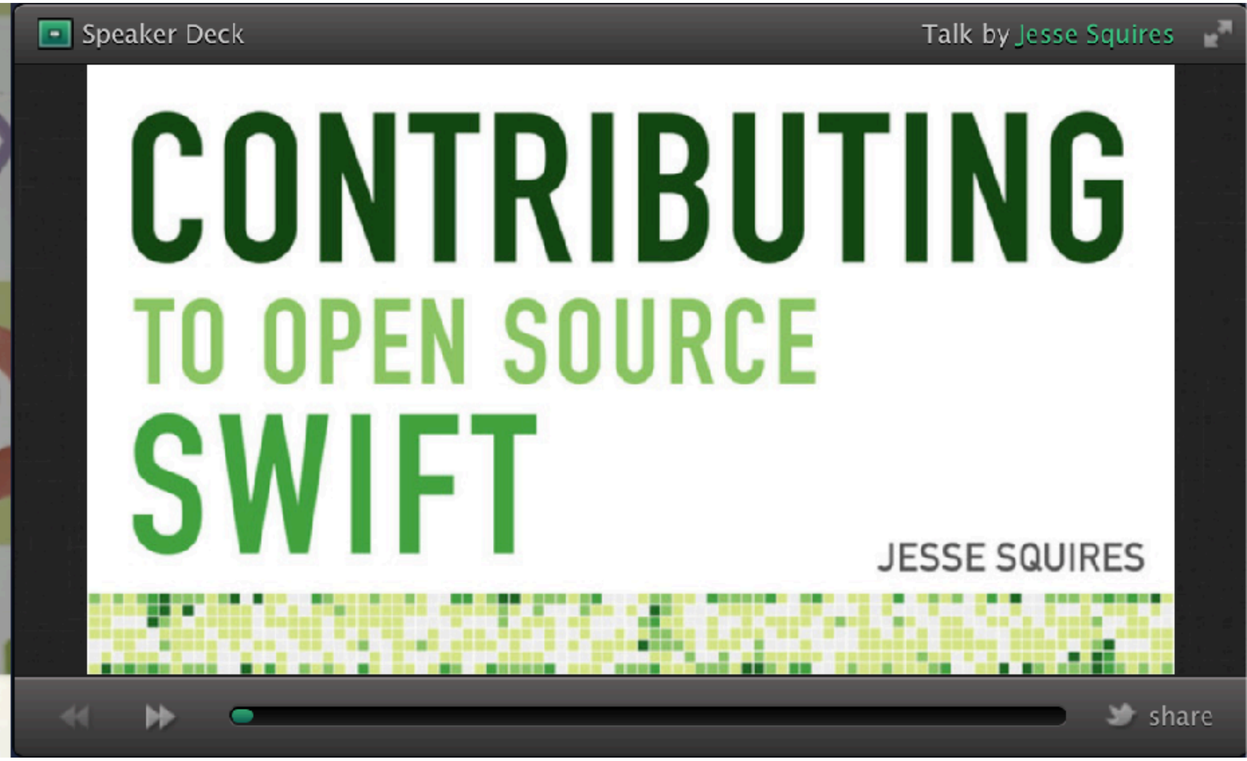
# Swift 5 👀

- 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

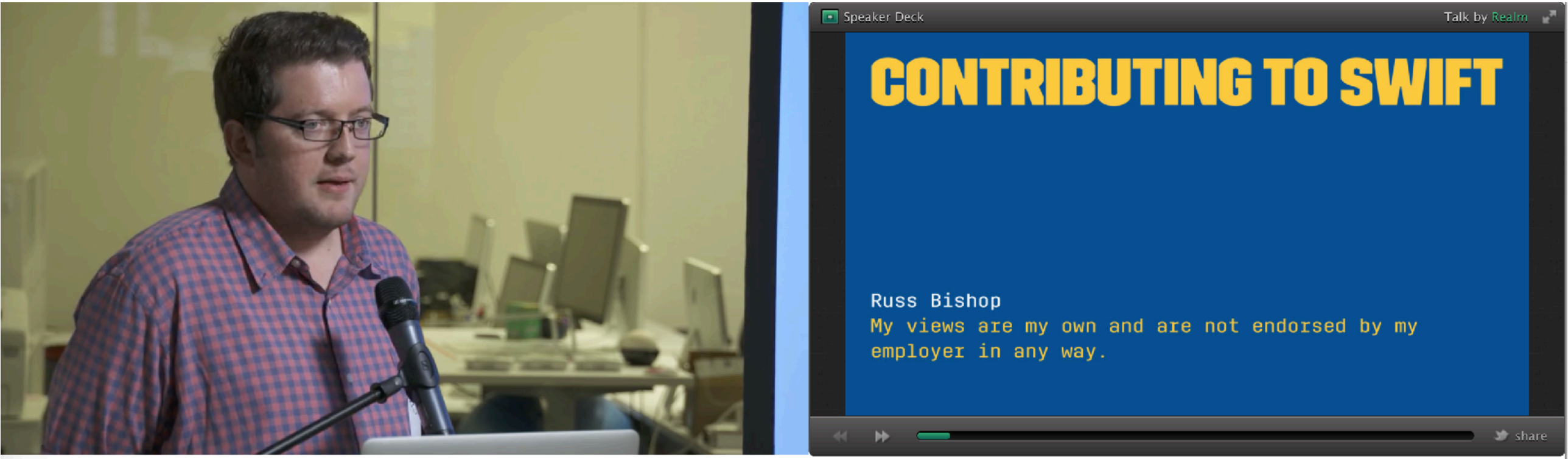

https://www.youtube.com/watch?v=GnT2ZeHVJe4




# Jesse Squires - Contributing to Open Source Swift

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




# 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