# CONTRIBUTING TO OPEN SOURCE SWIFT

JESSE SQUIRES

# IN THIS TALK

- What are all the different parts of Swift?
- What does each part do?
- LLVM demystified
- What skills do you need?
- Contributing process
- Tips for getting accepted
- Why should you contribute?

corelibs-libdispatch

swift-llbuild

corelibs-xctest

swift-llvm

swift-lldb

swift

swift-clang

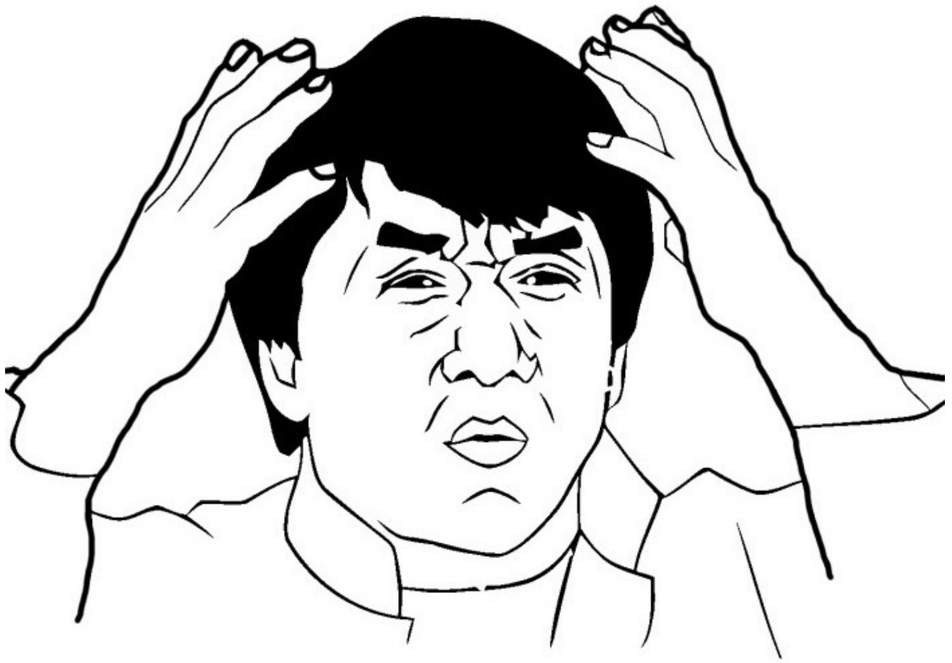
swift-evolution

corelibs-foundation

stdlib

swift-package-manager

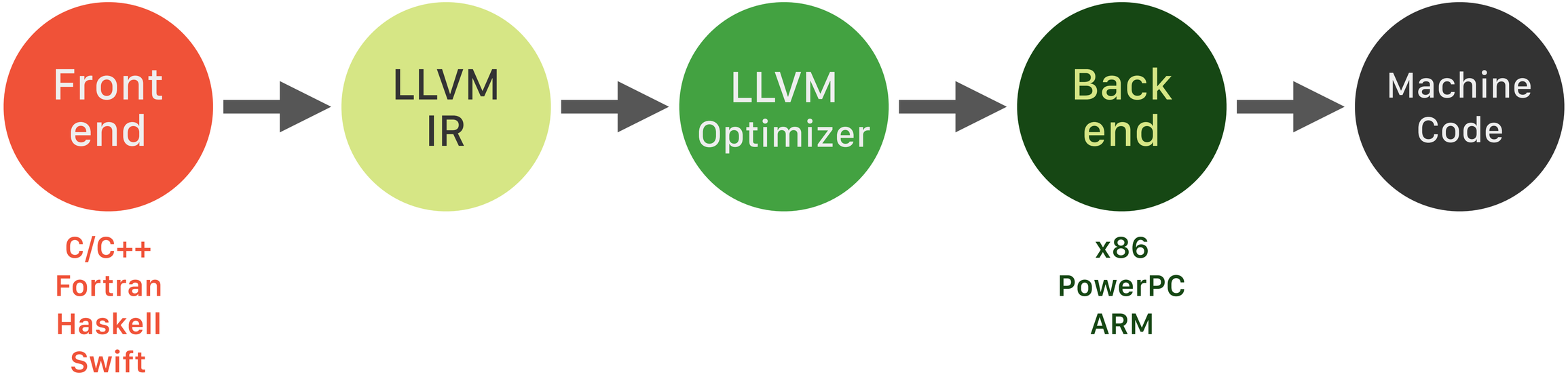
# WHERE DO YOU START?



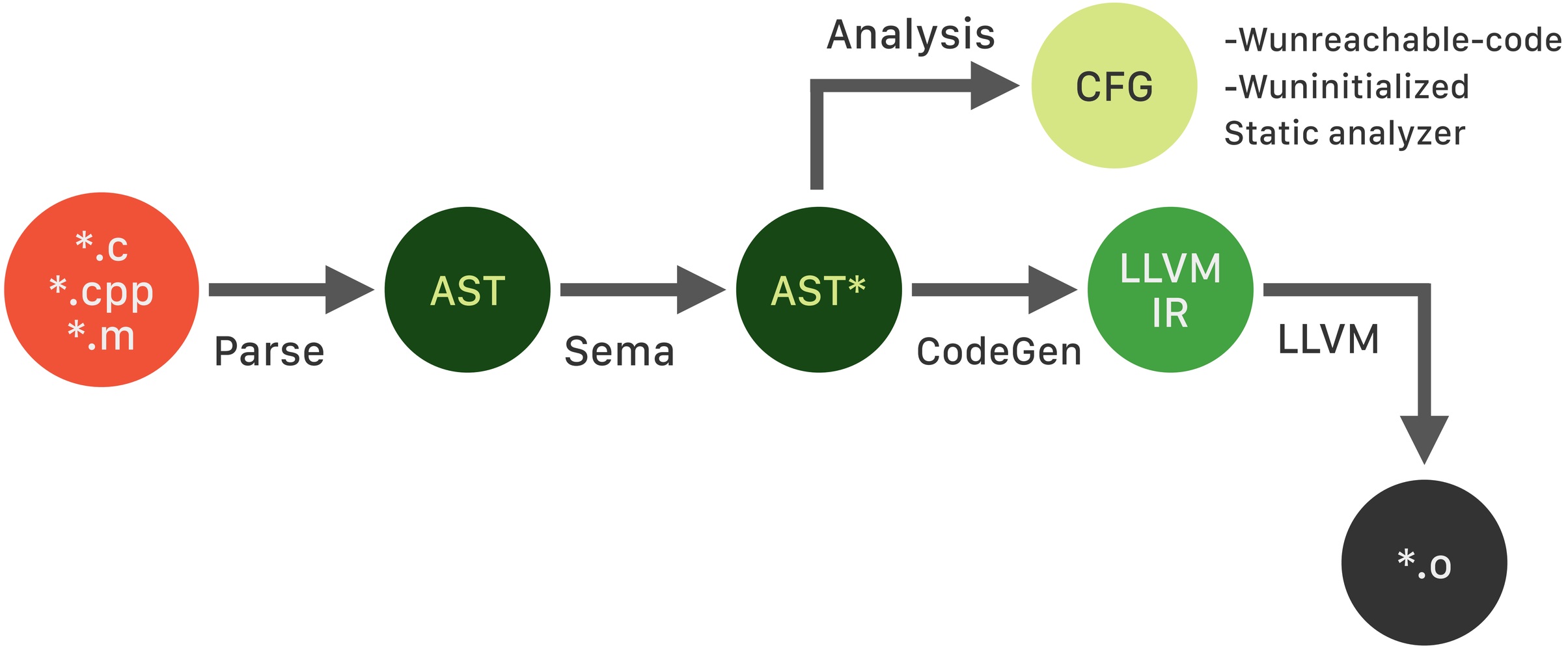
# HOW DOES IT ALL WORK TOGETHER?

# WHAT HAPPENS WHEN YOU COMPILE YOUR CODE?

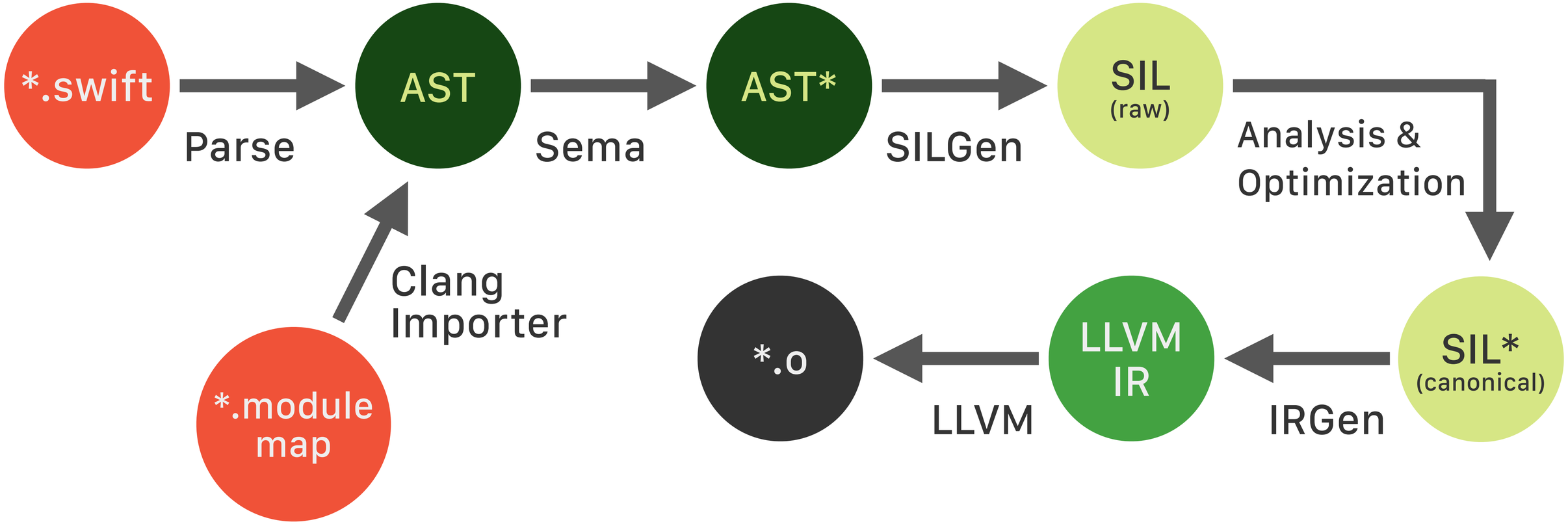
# LLVM COMPILER ARCHITECTURE



# CLANG PIPELINE



# SWIFT PIPELINE



# A BRIEF EXAMPLE

*.swift

```swift
// hello.swift
print("hello swift!")
```

*.swift

```
>swiftc hello.swift
>./hello
hello swift!
```

AST

```
>swiftc -dump-ast hello.swift
```

AST

```
(source_file
    (top_level_code_decl
        (brace_stmt
            (call_expr type='()' location=hello.swift:4:1 range=[hello.swift:4:1 - line:4:21]
nothrow
                (declref_expr type='(Any..., separator: String, terminator: String) -> ()'
location=hello.swift:4:1 range=[hello.swift:4:1 - line:4:1] decl=Swift.
(file).print(_:separator:terminator:) specialized=no)
                (tuple_shuffle_expr implicit type='(Any..., separator: String, terminator:
String)' location=hello.swift:4:7 range=[hello.swift:4:6 - line:4:21] sourceIsScalar
elements=[-2, -1, -1] variadic_sources=[0]
                (paren_expr type='Any' location=hello.swift:4:7 range=[hello.swift:4:6 - line:
4:21]
                (erasure_expr implicit type='Any' location=hello.swift:4:7 range=[hello.swift:
4:7 - line:4:7]
                (call_expr implicit type='String' location=hello.swift:4:7 range=[hello.swift:
4:7 - line:4:7] nothrow
                (constructor_ref_call_expr implicit type='(_builtinStringLiteral: RawPointer,
byteSize: Word, isASCII: Int1) -> String' location=hello.swift:4:7 range=[hello.swift:4:7 -
line:4:7] nothrow
                (declref_expr implicit type='String.Type -> (_builtinStringLiteral:
RawPointer, byteSize: Word, isASCII: Int1) -> String' location=hello.swift:4:7
range=[hello.swift:4:7 - line:4:7] decl=Swift.
(file).String.init(_builtinStringLiteral:byteSize:isASCII:) specialized=no)
                (type_expr implicit type='String.Type' location=hello.swift:4:7
range=[hello.swift:4:7 - line:4:7] typerepr='<<IMPLICIT>>'))
                (string_literal_expr type='(_builtinStringLiteral: Builtin.RawPointer,
byteSize: Builtin.Word, isASCII: Builtin.Int1)' location=hello.swift:4:7 range=[hello.swift:
4:7 - line:4:7] encoding=utf8 value="hello swift!")))))))))
```

SIL
(raw)

```
>swiftc -emit-silgen hello.swift
 | xcrun swift-demangle
```

SIL
(raw)

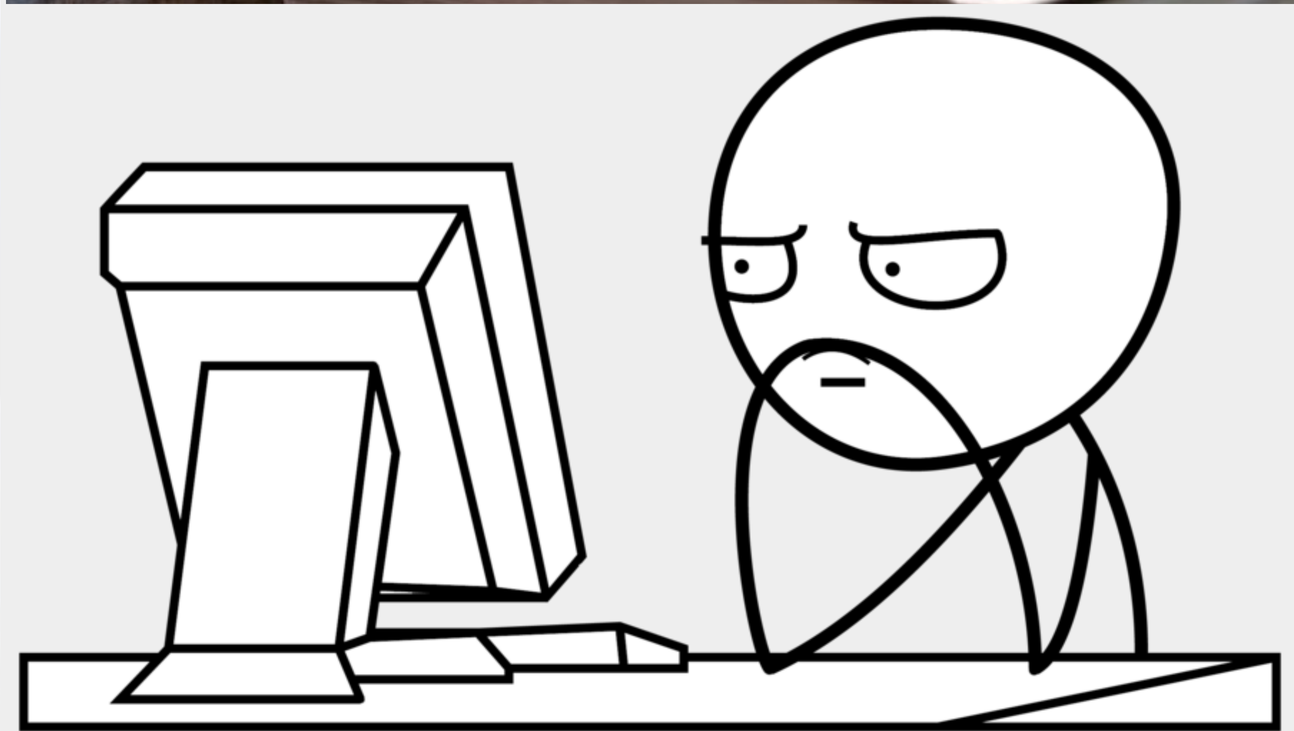
```
sil_stage raw

import Builtin
import Swift
import SwiftShims

// main
sil @main : $@convention(c) (Int32, UnsafeMutablePointer<UnsafeMutablePointer<Int8>>) -> Int32 {
bb0(%0 : $Int32, %1 : $UnsafeMutablePointer<UnsafeMutablePointer<Int8>>):
  // function_ref Swift._didEnterMain (Swift.Int32, argv : Swift.UnsafeMutablePointer<Swift.UnsafeMutablePointer<Swift.Int8>>) -> ()
  %2 = function_ref @Swift._didEnterMain (Swift.Int32, argv : Swift.UnsafeMutablePointer<Swift.UnsafeMutablePointer<Swift.Int8>>) -> () : $@convention(thin) (Int32,
UnsafeMutablePointer<UnsafeMutablePointer<Int8>>) -> () // user: %3
  %3 = apply %2(%0, %1) : $@convention(thin) (Int32, UnsafeMutablePointer<UnsafeMutablePointer<Int8>>) -> ()
  // function_ref Swift.print (Swift.Array<protocol<>>, separator : Swift.String, terminator : Swift.String) -> ()
  %4 = function_ref @Swift.print ([protocol<>], separator : Swift.String, terminator : Swift.String) -> () : $@convention(thin) (@owned Array<protocol<>>, @owned
String, @owned String) -> () // user: %23
  %5 = integer_literal $Builtin.Word, 1           // user: %7
  // function_ref Swift._allocateUninitializedArray <A> (Builtin.Word) -> (Swift.Array<A>, Builtin.RawPointer)
  %6 = function_ref @Swift._allocateUninitializedArray <A> (Builtin.Word) -> ([A], Builtin.RawPointer) : $@convention(thin) <τ_0_0> (Builtin.Word) -> @owned
(Array<τ_0_0>, Builtin.RawPointer) // user: %7
  %7 = apply %6<protocol<>>(%5) : $@convention(thin) <τ_0_0> (Builtin.Word) -> @owned (Array<τ_0_0>, Builtin.RawPointer) // users: %8, %9
  %8 = tuple_extract %7 : $(Array<protocol<>>, Builtin.RawPointer), 0 // user: %23
  %9 = tuple_extract %7 : $(Array<protocol<>>, Builtin.RawPointer), 1 // user: %10
  %10 = pointer_to_address %9 : $Builtin.RawPointer to $*protocol<> // user: %11
  %11 = init_existential_addr %10 : $*protocol<>, $String // user: %18
  // function_ref Swift.String.init (Swift.String.Type)(_builtinStringLiteral : Builtin.RawPointer, byteSize : Builtin.Word, isASCII : Builtin.Int1) -> Swift.String
  %12 = function_ref @Swift.String.init (Swift.String.Type)(_builtinStringLiteral : Builtin.RawPointer, byteSize : Builtin.Word, isASCII : Builtin.Int1) ->
Swift.String : $@convention(thin) (Builtin.RawPointer, Builtin.Word, Builtin.Int1, @thin String.Type) -> @owned String // user: %17
  %13 = metatype $@thin String.Type               // user: %17
  %14 = string_literal utf8 "hello swift!"        // user: %17
  %15 = integer_literal $Builtin.Word, 12         // user: %17
  %16 = integer_literal $Builtin.Int1, -1         // user: %17
  %17 = apply %12(%14, %15, %16, %13) : $@convention(thin) (Builtin.RawPointer, Builtin.Word, Builtin.Int1, @thin String.Type) -> @owned String // user: %18
  store %17 to %11 : $*String                     // id: %18
  // function_ref Swift.(print (Swift.Array<protocol<>>, separator : Swift.String, terminator : Swift.String) -> ()).(default argument 1)
  %19 = function_ref @Swift.(print ([protocol<>], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 1) : $@convention(thin) () -> @owned
String // user: %20
  %20 = apply %19() : $@convention(thin) () -> @owned String // user: %23
  // function_ref Swift.(print (Swift.Array<protocol<>>, separator : Swift.String, terminator : Swift.String) -> ()).(default argument 2)
  %21 = function_ref @Swift.(print ([protocol<>], separator : Swift.String, terminator : Swift.String) -> ()).(default argument 2) : $@convention(thin) () -> @owned
String // user: %22
  %22 = apply %21() : $@convention(thin) () -> @owned String // user: %23
  %23 = apply %4(%8, %20, %22) : $@convention(thin) (@owned Array<protocol<>>, @owned String, @owned String) -> ()
  %24 = integer_literal $Builtin.Int32, 0         // user: %25
  %25 = struct $Int32 (%24 : $Builtin.Int32)      // user: %26
  return %25 : $Int32                             // id: %26
}
```

LLVM
IR

```
>swiftc -emit-ir hello.swift
```

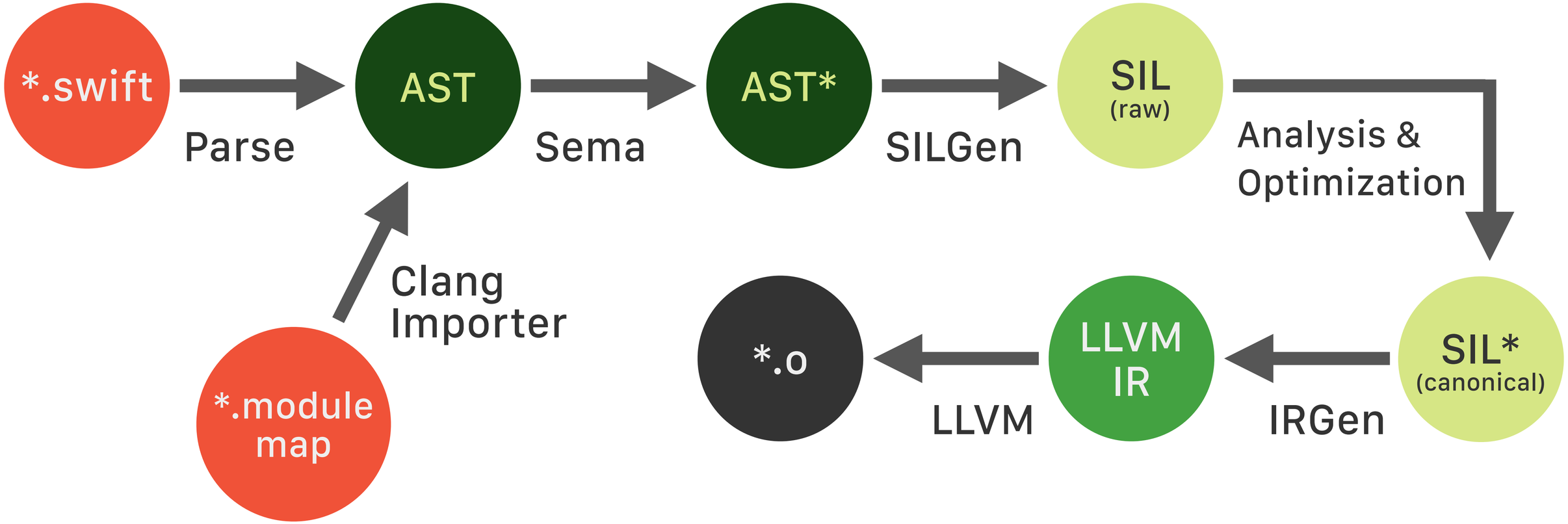
*.o

```
>swiftc -emit-assembly hello.swift
```



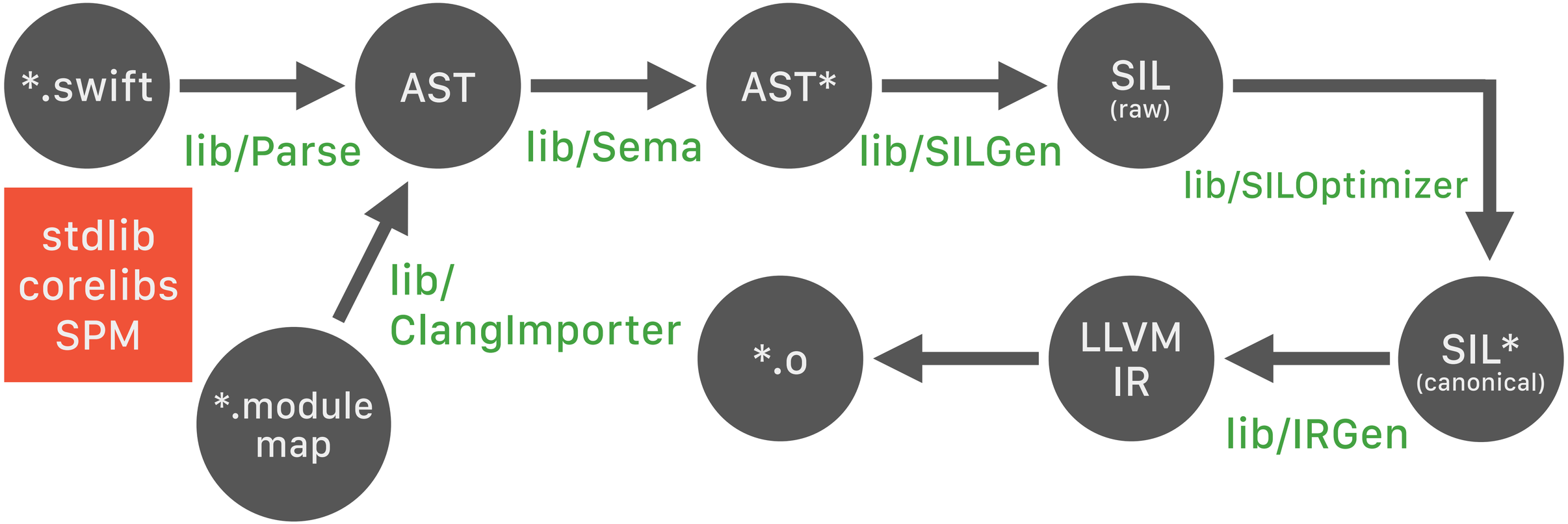
# PROJECTS AND REPOSITORIES

# SWIFT PIPELINE



# github / apple / swift

swift-llvm
swift-clang
swift-lldb




stdlib
corelibs
SPM

# SWIFT CORE

## Compiler

- Hard
- C++
- High

## stdlib

- Medium
- Swift*
- Medium

## SourceKit

- Hard
- C++
- Low

# INFRA

- LLVM Clones with Swift-specific changes
- Very Hard
- C++
- Low
- Upstream changes not Swift-specific
- Governed by LLVM dev policies, license, etc.

swift-llvm

swift-clang

swift-lldb

# PACKAGE MANAGER

swift package manager

- Medium
- Swift
- High

swift-llbuild

- Hard
- C++
- Low

# CORE LIBS

**Foundation**

- Easy
- Swift, C
- High

**XCTest**

- Easy
- Swift
- High

**libdispatch (GCD)**

- Hard
- C
- Low

# EVOLUTION

Proposals

Development Schedule

Release Schedule

# CONTRIBUTING.md

✨ Do it right ✨

swift.org/contributing

mailing lists

typos

documentation

tests

bug fixes

# contribute

code clean up

translations

proposals

new features

# CONTRIBUTING.md

## #ProTips

- Be kind, thoughtful
- Ask for help
- Follow best practices
- Rejected? Don’t stop contributing!
- Core team knows the big picture
- CODE_OWNERS.txt
- No one is an *expert* in *everything*!

# AWESOME CONTRIBUTORS


practicalswift
**@practicalswift**
compiler crashers

Brian Gesiak
**@modocache**
corelibs-xctest

Erica Sadun
**@ericasadun**
proposals

Greg Titus
**@gregtitus**
parser, AST, sema

Making small improvements is the way everyone starts getting involved!

Chris Lattner

# why contribute?

# SWIFT IS MORE THAN A PROGRAMMING LANGUAGE

# SWIFT IS A COMMUNITY

# DON'T BE AFRAID

```
try! contribute()
```

# THANKS!

JESSESQUIRES.COM • @JESSE_SQUIRES • @SWIFTLYBRIEF