OBSCURE SWIFT

Paweł Łopusiński

Glovo

@Losiowaty

Pawel Lopusinski

SIL

Swift Intermediate Language

https://cutt.ly/sil-docs

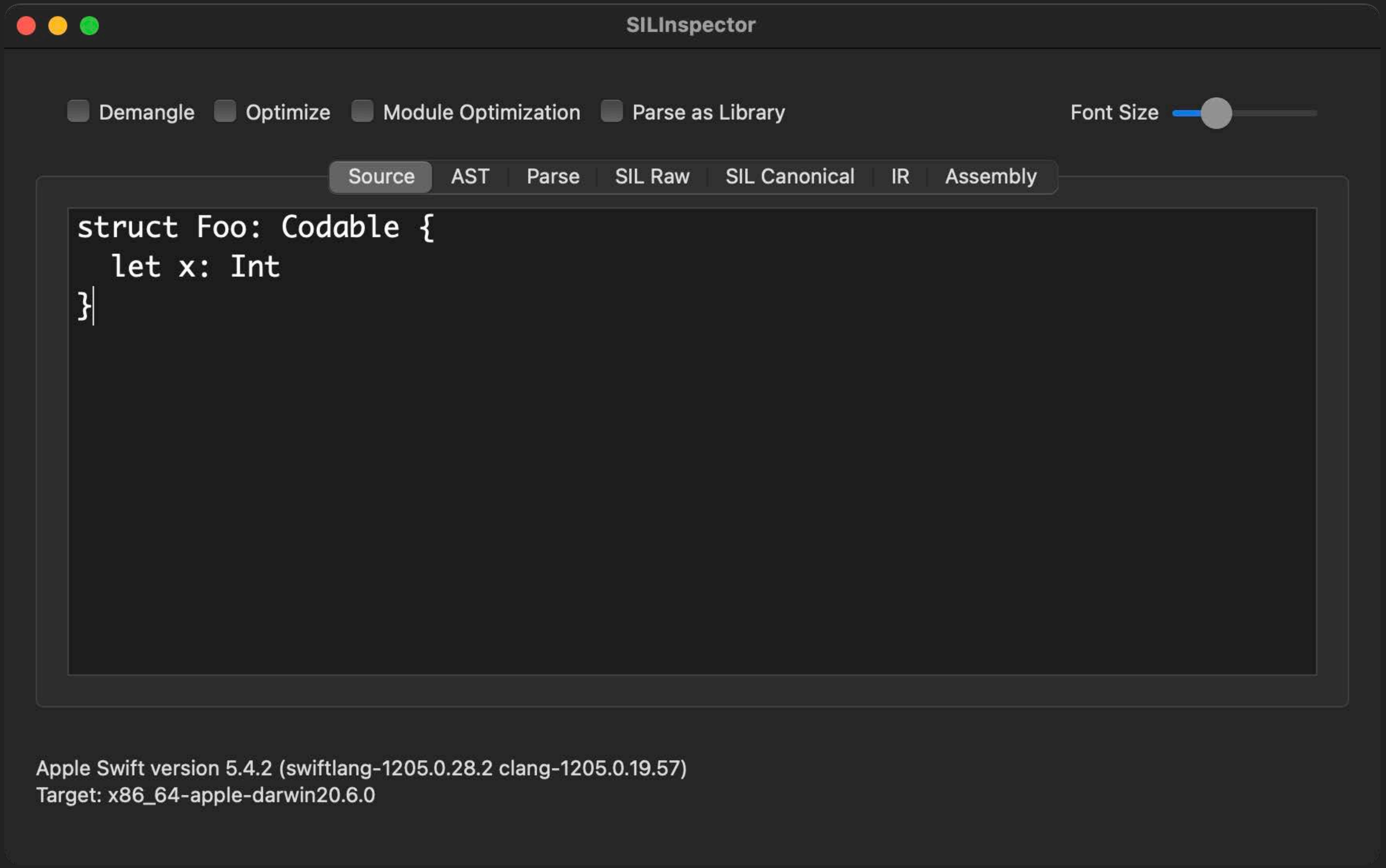
OBSCURE SIL

```
struct Foo: Codable {
  let x: Int
}
```

OBSCURE SIL

```
struct Foo : Decodable & Encodable {
  @_hasStorage let x: Int { get }
  enum CodingKeys : CodingKey {
    case x
    @_implements(Equatable, ==(_:_:)) static func __derived_enum_equals(_ a: Foo.CodingKeys, _ b: Foo.CodingKeys) -> Bool
    func hash(into hasher: inout Hasher)
    init?(stringValue: String)
    init?(intValue: Int)
    var hashValue: Int { get }
    var intValue: Int? { get }
    var stringValue: String { get }
  }
  func encode(to encoder: Encoder) throws
  init(from decoder: Decoder) throws
  init(x: Int)
}
```

OBSCURE SIL



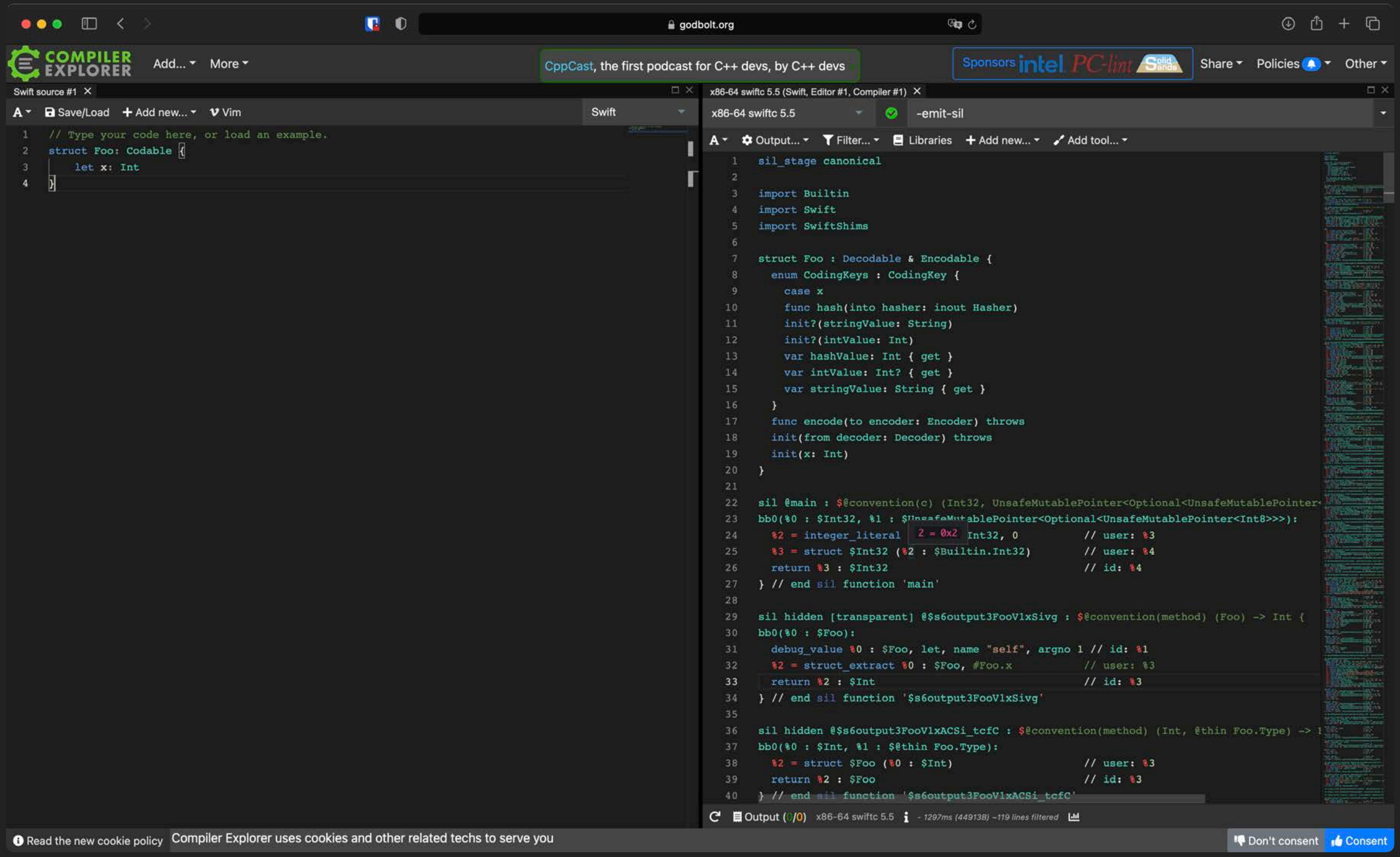
SILInspector

Demangle · Optimize · Module Optimization · Parse as Library · Font Size

Source · AST · Parse · SIL Raw · SIL Canonical · IR · Assembly

```swift
struct Foo: Codable {
  let x: Int
}
```

Apple Swift version 5.4.2 (swiftlang-1205.0.28.2 clang-1205.0.19.57)
Target: x86_64-apple-darwin20.6.0



godbolt.org

CppCast, the first podcast for C++ devs, by C++ devs

Swift source #1

```swift
// Type your code here, or load an example.
struct Foo: Codable {
    let x: Int
}
```

x86-64 swiftc 5.5 (Swift, Editor #1, Compiler #1)

x86-64 swiftc 5.5 · -emit-sil

```
sil_stage canonical

import Builtin
import Swift
import SwiftShims

struct Foo : Decodable & Encodable {
  enum CodingKeys : CodingKey {
    case x
    func hash(into hasher: inout Hasher)
    init?(stringValue: String)
    init?(intValue: Int)
    var hashValue: Int { get }
    var intValue: Int? { get }
    var stringValue: String { get }
  }
  func encode(to encoder: Encoder) throws
  init(from decoder: Decoder) throws
  init(x: Int)
}

sil @main : $@convention(c) (Int32, UnsafeMutablePointer<Optional<UnsafeMutablePointer<
bb0(%0 : $Int32, %1 : $UnsafeMutablePointer<Optional<UnsafeMutablePointer<Int8>>>):
  %2 = integer_literal  2 = 0x2  Int32, 0          // user: %3
  %3 = struct $Int32 (%2 : $Builtin.Int32)        // user: %4
  return %3 : $Int32                              // id: %4
} // end sil function 'main'

sil hidden [transparent] @$s6output3FooV1xSivg : $@convention(method) (Foo) -> Int {
bb0(%0 : $Foo):
  debug_value %0 : $Foo, let, name "self", argno 1 // id: %1
  %2 = struct_extract %0 : $Foo, #Foo.x           // user: %3
  return %2 : $Int                                // id: %3
} // end sil function '$s6output3FooV1xSivg'

sil hidden @$s6output3FooV1xACSi_tcfC : $@convention(method) (Int, @thin Foo.Type) -> F
bb0(%0 : $Int, %1 : $@thin Foo.Type):
  %2 = struct $Foo (%0 : $Int)                    // user: %3
  return %2 : $Foo                                // id: %3
} // end sil function '$s6output3FooV1xACSi_tcfC'
```

Output (0/0) x86-64 swiftc 5.5 - 1297ms (44913B) ~119 lines filtered

Read the new cookie policy · Compiler Explorer uses cookies and other related techs to serve you · Don't consent · Consent

https://github.com/alblue/SILInspector

https://godbolt.org

Apple Swift version 5.5.1 (swiftlang-1300.0.31.4 clang-1300.0.29.6)
Target: x86_64-apple-macosx11.0
Xcode: Version 13.1 (13A1030d)

OBSCURE

Not well-known

Not clearly seen

Relatively unknown

By Merriam-Webster Dictionary : [Obscure | Definition of Obscure by Merriam-Webster]

AGENDA

Obscure NSObject

Obscure @autoclosure

Obscure default values

Obscure protocol extension

NSOBJECT

OBJECTIVE-C SAYS HI 👋

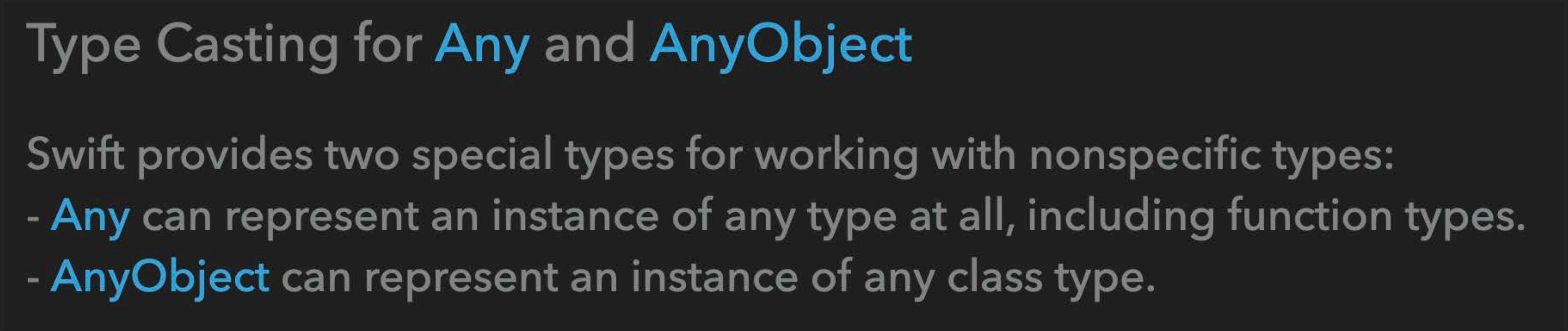
OBSCURE NSOBJECT

	Base Type	“Anything” type
OBJECTIVE-C	NSObject	id*
SWIFT	None	Any AnyObject

* assumes primitive types wrapped in NSNumber / NSValue

OBSCURE NSOBJECT

	Base Type	"Anything" type
OBJECTIVE-C	NSObject	id*
SWIFT	None	Any AnyObject



Type Casting for Any and AnyObject

Swift provides two special types for working with nonspecific types:
- Any can represent an instance of any type at all, including function types.
- AnyObject can represent an instance of any class type.

* assumes primitive types wrapped in NSNumber / NSValue

OBSCURE NSOBJECT

```
let any: Any = 1
```

OBSCURE NSOBJECT

```swift
let any: Any = 1
let anyObject: AnyObject = 1
```

```
Value of type 'Int' expected to be instance of class or class-constrained type
```

OBSCURE NSOBJECT

```
let any: Any = 1
let anyObject: AnyObject = 1 as NSNumber
```

OBSCURE NSOBJECT

```
let any: Any = 1
let anyObject: AnyObject = 1 as NSNumber
anyObject is NSNumber
anyObject is Int
```

OBSCURE NSOBJECT

```swift
let any: Any = 1
let anyObject: AnyObject = 1 as NSNumber
anyObject is NSNumber   // true
anyObject is Int        // true
```

OBSCURE NSOBJECT

```
let any: Any = 1
let anyObject: AnyObject = 1 as NSNumber
anyObject is NSNumber   // true
anyObject is Int        // true
anyObject is Float
```

OBSCURE NSOBJECT

```swift
let any: Any = 1
let anyObject: AnyObject = 1 as NSNumber
anyObject is NSNumber   // true
anyObject is Int        // true
anyObject is Float      // true
```

OBSCURE NSOBJECT

```
let any: Any = 1
let anyObject: AnyObject = 1 as NSNumber
anyObject is NSNumber   // true
anyObject is Int        // true
anyObject is Float      // true
let floatObject: AnyObject = 1.5 as NSNumber
floatObject is Float
floatObject is Int
```

OBSCURE NSOBJECT

```
let any: Any = 1
let anyObject: AnyObject = 1 as NSNumber
anyObject is NSNumber   // true
anyObject is Int        // true
anyObject is Float      // true
let floatObject: AnyObject = 1.5 as NSNumber
floatObject is Float    // true
floatObject is Int      // false
```

When cast to NSNumber:

```
// function_ref NSNumber.init(integerLiteral:)
%16 = function_ref @$sSo8NSNumberC10FoundationE14integerLiteralABSi_tcfC :
      $@convention(method) (Int, @thick NSNumber.Type) -> @owned NSNumber
```

When cast to NSNumber:

```
// function_ref NSNumber.init(integerLiteral:)
%16 = function_ref @$sSo8NSNumberC10FoundationE14integerLiteralABSi_tcfC :
      $@convention(method) (Int, @thick NSNumber.Type) -> @owned NSNumber
```

When cast to NSObject:

```
// function_ref Int._bridgeToObjectiveC()
%6 = function_ref @$sSi10FoundationE19_bridgeToObjectiveCSo8NSNumberCyF :
     $@convention(method) (Int) -> @owned NSNumber
```

When cast to NSNumber:

```
// function_ref NSNumber.init(integerLiteral:)
%16 = function_ref @$sSo8NSNumberC10FoundationE14integerLiteralABSi_tcfC :
      $@convention(method) (Int, @thick NSNumber.Type) -> @owned NSNumber
```

When cast to NSObject:

```
// function_ref Int._bridgeToObjectiveC()
%6 = function_ref @$sSi10FoundationE19_bridgeToObjectiveCSo8NSNumberCyF :
     $@convention(method) (Int) -> @owned NSNumber
```

OBSCURE NSOBJECT

```swift
class FooClass {}
struct FooStruct {}

let anyFooClass: AnyObject = FooClass()
let anyFooStruct: AnyObject = FooStruct()
```

OBSCURE NSOBJECT

```swift
class FooClass {}
struct FooStruct {}

let anyFooClass: AnyObject = FooClass()
let anyFooStruct: AnyObject = FooStruct()
```

```
Value of type 'Int' expected to be instance of class or class-constrained type
```

```
class FooClass {}
struct FooStruct {}

let anyFooClass: AnyObject = FooClass()
let anyFooStruct: AnyObject = FooStruct()
```

```
Value of type 'Int' expected to be instance of class or class-constrained type
Insert ' as AnyObject’
```

FIX

OBSCURE NSOBJECT

```swift
class FooClass {}
struct FooStruct {}

let anyFooClass: AnyObject = FooClass()
let anyFooStruct: AnyObject = FooStruct() as AnyObject
```

OBSCURE NSOBJECT

```swift
class FooClass {}
struct FooStruct {}

let anyFooClass: AnyObject = FooClass()
let anyFooStruct: AnyObject = FooStruct() as AnyObject



anyFooClass  is NSObject
anyFooStruct is NSObject
```

OBSCURE NSOBJECT

```swift
class FooClass {}
struct FooStruct {}

let anyFooClass: AnyObject = FooClass()
let anyFooStruct: AnyObject = FooStruct() as AnyObject



anyFooClass  is NSObject   // false
anyFooStruct is NSObject
```

OBSCURE NSOBJECT

```swift
class FooClass {}
struct FooStruct {}

let anyFooClass: AnyObject = FooClass()
let anyFooStruct: AnyObject = FooStruct() as AnyObject



anyFooClass  is NSObject   // false
anyFooStruct is NSObject   // true
```

OBSCURE NSOBJECT

```
let anyFooStruct: AnyObject = FooStruct() as AnyObject
```

OBSCURE NSOBJECT

```swift
let anyFooStruct: AnyObject = FooStruct() as AnyObject
```

$$\downarrow$$

```
// function_ref _bridgeAnythingToObjectiveC<A>(_:)
%9 = function_ref @$ss27_bridgeAnythingToObjectiveCyyXlxlF :
     $@convention(thin) <τ_0_0> (@in_guaranteed τ_0_0) -> @owned AnyObject
```

```
/// Bridge an arbitrary value to an Objective-C object.

/// - If `T` is a class type, it is always bridged verbatim, the function
///   returns `x`;

/// - otherwise, if `T` conforms to `_ObjectiveCBridgeable`,
///   returns the result of `x._bridgeToObjectiveC()`;

/// - otherwise, we use **boxing** to bring the value into Objective-C.
///   The value is wrapped in an instance of a private Objective-C class
///   that is `id`-compatible and dynamically castable back to the type of
///   the boxed value, but is otherwise opaque.
```

OBSCURE NSOBJECT

```
/// Bridge an arbitrary value to an Objective-C object.

/// - If `T` is a class type, it is always bridged verbatim, the function
///   returns `x`;

/// - otherwise, if `T` conforms to `_ObjectiveCBridgeable`,
///   returns the result of `x._bridgeToObjectiveC()`;

/// - otherwise, we use **boxing** to bring the value into Objective-C.
///   The value is wrapped in an instance of a private Objective-C class
///   that is `id`-compatible and dynamically castable back to the type of
///   the boxed value, but is otherwise opaque.
```

OBSCURE NSOBJECT

```
/// Bridge an arbitrary value to an Objective-C object.

/// - If `T` is a class type, it is always bridged verbatim, the function
///   returns `x`;

/// - otherwise, if `T` conforms to `_ObjectiveCBridgeable`,
///   returns the result of `x._bridgeToObjectiveC()`;

/// - otherwise, we use **boxing** to bring the value into Objective-C.
///   The value is wrapped in an instance of a private Objective-C class
///   that is `id`-compatible and dynamically castable back to the type of
///   the boxed value, but is otherwise opaque.
/// Bridge an arbitrary value to an Objective-C object.

/// - If `T` is a class type, it is always bridged verbatim, the function
///   returns `x`;

/// - otherwise, if `T` conforms to `_ObjectiveCBridgeable`,
///   returns the result of `x._bridgeToObjectiveC()`;

/// - otherwise, we use **boxing** to bring the value into Objective-C.
///   The value is wrapped in an instance of a private Objective-C class
///   that is `id`-compatible and dynamically castable back to the type of
///   the boxed value, but is otherwise opaque.
/// Bridge an arbitrary value to an Objective-C object.

/// - If `T` is a class type, it is always bridged verbatim, the function
///   returns `x`;

/// - otherwise, if `T` conforms to `_ObjectiveCBridgeable`,
///   returns the result of `x._bridgeToObjectiveC()`;

/// - otherwise, we use **boxing** to bring the value into Objective-C.
///   The value is wrapped in an instance of a private Objective-C class
///   that is `id`-compatible and dynamically castable back to the type of
///   the boxed value, but is otherwise opaque.
```

```
@interface __SwiftValue : NSObject <NSCopying>

- (id)copyWithZone:(NSZone *)zone;

@end
```

	Base Type	“Anything” type
OBJECTIVE-C	NSObject	id
SWIFT	None	Any AnyObject

OBSCURE NSOBJECT

	Base Type	“Anything” type
OBJECTIVE-C	NSObject	id
SWIFT	None	Any AnyObject

OBSCURE NSOBJECT

	Base Type	"Anything" type
OBJECTIVE-C	NSObject	id
SWIFT	None	Any AnyObject

$$\begin{tikzcd} \text{id} \arrow[d, bend right, "\text{Before Swift 3}"'] \\ \text{AnyObject} \end{tikzcd}$$

OBSCURE NSOBJECT

	Base Type
OBJECTIVE-C	NSObject
SWIFT	None

“Anything” type

$$\begin{tikzcd}
\text{id} \arrow[d, bend left, "\text{After Swift 3}"] \arrow[dd, bend right, "\text{Before Swift 3}"'] \\
\text{Any} \\
\text{AnyObject}
\end{tikzcd}$$

OBSCURE NSOBJECT

	Base Type	“Anything” type
OBJECTIVE-C	NSObject	
SWIFT	None	

$$\begin{tikzcd}
\text{id} \arrow[dd, bend right=60, "\text{Before Swift 3}"'] \arrow[d, bend left=60, "\text{After Swift 3}"] \\
\text{Any} \\
\text{AnyObject}
\end{tikzcd}$$

```swift
struct FooStruct {}
NSError(domain: "", code: 1, userInfo: ["key": FooStruct()])
```

https://cutt.ly/SE-0116 https://cutt.ly/anyobject

OBSCURE @AUTOCLOSURE

@AUTOCLOSURE

OBSCURE @AUTOCLOSURE

```swift
struct Foo {
  let producer: () -> String
  var value: String { return producer() }

  init(value producer: @escaping @autoclosure () -> String) {
    self.producer = producer
  }
}

var callCounter = 0
func makeString() -> String {
  callCounter += 1
  return "callCounter == \(callCounter)"
}

// let f = Foo(value: String)
let f = Foo(value: makeString())

// 1️⃣ callCounter

f.value

// 2️⃣ callCounter
```

OBSCURE @AUTOCLOSURE

```swift
struct Foo {
  let producer: () -> String
  var value: String { return producer() }

  init(value producer: @escaping @autoclosure () -> String) {
    self.producer = producer
  }
}

var callCounter = 0
func makeString() -> String {
  callCounter += 1
  return "callCounter == \(callCounter)"
}

// let f = Foo(value: String)
let f = Foo(value: makeString())

// 1️⃣ callCounter == 0

f.value

// 2️⃣ callCounter == 1
```

AUTOCLOSURES

An *autoclosure* is a closure that’s automatically created to wrap an expression that’s being passed as an argument to a function. It doesn’t take any arguments, and when it’s called, it returns the value of the expression that’s wrapped inside of it. This syntactic convenience lets you omit braces around a function’s parameter by writing a normal expression instead of an explicit closure.

AUTOCLOSURES

An *autoclosure* is a closure that's automatically created to wrap an expression that's being passed as an argument to a function. It doesn't take any arguments, and when it's called, it returns the value of the expression that's wrapped inside of it. This syntactic convenience lets you omit braces around a function's parameter by writing a normal expression instead of an explicit closure.

AUTOCLOSURES

An *autoclosure* is a closure that's automatically created to wrap an expression that's being passed as an argument to a function. It doesn't take any arguments, and when it's called, it returns the value of the expression that's wrapped inside of it. This syntactic convenience lets you omit braces around a function's parameter by writing a normal expression instead of an explicit closure.

```
let f = Foo(value: makeString())
```

AUTOCLOSURES

An *autoclosure* is a closure that's automatically created to wrap an expression that's being passed as an argument to a function. It doesn't take any arguments, and when it's called, it returns the value of the expression that's wrapped inside of it. This syntactic convenience lets you omit braces around a function's parameter by writing a normal expression instead of an explicit closure.

```
let f = Foo(value: makeString())
```

$\downarrow$

```
let f = Foo(value: { makeString() })
```

OBSCURE @AUTOCLOSURE

```swift
struct Foo {
  let producer: () -> String
  var value: String { return producer() }

  init(value: @escaping @autoclosure () -> String) {
    self.producer = value
  }
}

var callCounter = 0
func makeString() -> String {
  callCounter += 1
  return "callCounter == \(callCounter)"
}

// let f = Foo(value: String)
let f = Foo(value: makeString())
```

OBSCURE @AUTOCLOSURE

```swift
struct Foo {
  let producer: () -> String
  var value: String { return producer() }

  init(value: @escaping @autoclosure () -> String) {
    self.producer = value
  }
}

var callCounter = 0
func makeString() -> String {
  callCounter += 1
  return "callCounter == \(callCounter)"
}
```

```swift
// let f = Foo(value: String)
let f = Foo(value: makeString())
```

⟺

```swift
let value = makeString()
let f = Foo(value: value)
```

CLOSED SOURCE

```
var callCounter = 0
func makeString() -> String {
  callCounter += 1
  return "callCounter == \(callCounter)"
}

// let f = Foo(value: String)
let f = Foo(value: makeString())
```

```swift
struct ExternalSDK {
  let producer: () -> String
  var value: String { return producer() }

  init(value: @escaping @autoclosure () -> String) {
    self.producer = value
  }
}
```

Framework boundary

```swift
class FooClass {
  var sdk: ExternalSDK?

  init() {
    self.sdk = ExternalSDK(value: makeString())
  }

  func makeString() -> String {
    return "FooClass"
  }
}
```

OBSCURE @AUTOCLOSURE

```
struct ExternalSDK {
  let producer: () -> String
  var value: String { return producer() }

  init(value: @escaping @autoclosure () -> String) {
    self.producer = value
  }
}
```

Framework boundary

```
class FooClass {
  var sdk: ExternalSDK?

  init() {
    self.sdk = ExternalSDK(value: { self.makeString() } )
  }

  func makeString() -> String {
    return "FooClass"
  }
}
```

```
class FooClass {
  var sdk: ExternalSDK?

  init() {
    self.sdk = ExternalSDK(value:
      { [weak self] in
        self?.makeString() ?? ""
      }
    )
  }

  func makeString() -> String {
    return "FooClass"
  }
}
```

OBSCURE @AUTOCLOSURE

```swift
class FooClass {
  var sdk: ExternalSDK?

  init() {
    self.sdk = ExternalSDK(value:
      { [weak self] in
        self?.makeString() ?? ""
      }
    )   Cannot convert value of type '() -> String' to expected argument type 'String'
  }

  func makeString() -> String {
    return "FooClass"
  }
}
```

OBSCURE @AUTOCLOSURE

```
class FooClass {
  var sdk: ExternalSDK?

  init() {
    self.sdk = ExternalSDK(value:
      { [weak self] in
        self?.makeString() ?? ""
      }
    )  Cannot convert value of type '() -> String' to expected argument type 'String'
  }

  func makeString() -> String {
    return "FooClass"
  }
}
```

```
class FooClass {
  var sdk: ExternalSDK?

  init() {
    self.sdk = ExternalSDK(value:
      { [weak self] in
        self?.makeString() ?? ""
      }()
    )
  }

  func makeString() -> String {
    return "FooClass"
  }
}
```

Our closure

```
// closure #1 in implicit closure #1 in FooClass.init()
sil private @$s4main8FooClassCACycfcSSycfu_SSyXEfU_ :
$@convention(thin) (@guaranteed { var @sil_weak Optional<FooClass> }) -> @owned String
```

Our closure

```
// closure #1 in implicit closure #1 in FooClass.init()
sil private @$s4main8FooClassCACycfcSSycfu_SSyXEfU_ :
$@convention(thin) (@guaranteed { var @sil_weak Optional<FooClass> }) -> @owned String
```

Generated autoclosure

```
// implicit closure #1 in FooClass.init()
sil private [transparent] @$s4main8FooClassCACycfcSSycfu_ :
  $@convention(thin) (@guaranteed FooClass) -> @owned String
```

```swift
class FooClass {
  var sdk: ExternalSDK?

  init() {
    let producer = { [weak self] in
      self?.makeString() ?? ""
    }
    self.sdk = ExternalSDK(value: producer())
  }

  func makeString() -> String {
    return "FooClass"
  }
}
```

Our closure

```
// closure #1 in FooClass.init()
sil private @$s4main8FooClassCACycfcSSycfU_ :
$@convention(thin) (@guaranteed { var @sil_weak Optional<FooClass> }) -> @owned String
```

Generated autoclosure

```
// implicit closure #1 in FooClass.init()
sil private [transparent] @$s4main8FooClassCACycfcSSycfu_ :
$@convention(thin) (@guaranteed @callee_guaranteed () -> @owned String) -> @owned String
```

OBSCURE @AUTOCLOSURE

❤️ @autoclosure

💔 lacking IDE support

❤️ @autoclosure

💔 lacking IDE support

```
struct ExternalSDK {
  init(value: @escaping @autoclosure () -> String) {}
}

// "visible" interface: init(value: String)

protocol SDKWrapper {
  init(value: String)
}

extension ExternalSDK: SDKWrapper {}
    Type 'ExternalSDK' does not conform to protocol ‘SDKWrapper'
```

❤️ @autoclosure

💔 lacking IDE support

```swift
struct ExternalSDK {
  init(value: @escaping @autoclosure () -> String) {}
}

// "visible" interface: init(value: String)

protocol SDKWrapper {
  init(value: String)
}

extension ExternalSDK: SDKWrapper {}
```

```
Type 'ExternalSDK' does not conform to protocol ‘SDKWrapper'
```

VALUE = DEFAULT

WHAT A (OVER)RIDE! 🏎️

OBSCURE DEFAULT VALUES

```swift
class DatePrinter {
  func printDate(_ date: Date = Date()) {
    print("Date is == \(date)")
  }
}

DatePrinter().printDate()
```

OBSCURE DEFAULT VALUES

```swift
class DatePrinter {
  func printDate(_ date: Date = Date()) {
    print("Date is == \(date)")
  }
}

DatePrinter().printDate() // Date is == 2022-02-17 17:00:00 +0100
```

OBSCURE DEFAULT VALUES

```swift
class DatePrinter {
  func printDate(_ date: Date = Date()) {
    print("Date is == \(date)")
  }
}

DatePrinter().printDate() // Date is == 2022-02-17 17:00:00 +0100

class EpochDatePrinter: DatePrinter {
  override func printDate(_ date: Date = Date(timeIntervalSince1970: 0)) {
    print("Epoch date is == \(date)")
  }
}

EpochDatePrinter().printDate()
```

OBSCURE DEFAULT VALUES

```swift
class DatePrinter {
  func printDate(_ date: Date = Date()) {
    print("Date is == \(date)")
  }
}

DatePrinter().printDate() // Date is == 2022-02-17 17:00:00 +0100

class EpochDatePrinter: DatePrinter {
  override func printDate(_ date: Date = Date(timeIntervalSince1970: 0)) {
    print("Epoch date is == \(date)")
  }
}

EpochDatePrinter().printDate() // Epoch date is == 1970-01-01 00:00:00 +0000
```

OBSCURE DEFAULT VALUES

```
class DatePrinter {
  func printDate(_ date: Date = Date()) {
    print("Date is == \(date)")
  }
}

DatePrinter().printDate() // Date is == 2022-02-17 17:00:00 +0100

class EpochDatePrinter: DatePrinter {
  override func printDate(_ date: Date = Date(timeIntervalSince1970: 0)) {
    print("Epoch date is == \(date)")
  }
}

EpochDatePrinter().printDate() // Epoch date is == 1970-01-01 00:00:00 +0000

let datePrinter: DatePrinter = EpochDatePrinter()
datePrinter.printDate()
```

OBSCURE DEFAULT VALUES

```swift
class DatePrinter {
  func printDate(_ date: Date = Date()) {
    print("Date is == \(date)")
  }
}

DatePrinter().printDate() // Date is == 2022-02-17 17:00:00 +0100

class EpochDatePrinter: DatePrinter {
  override func printDate(_ date: Date = Date(timeIntervalSince1970: 0)) {
    print("Epoch date is == \(date)")
  }
}

EpochDatePrinter().printDate() // Epoch date is == 1970-01-01 00:00:00 +0000

let datePrinter: DatePrinter = EpochDatePrinter()
datePrinter.printDate() // Epoch date is == 2022-02-17 17:00:00 +0100
```

OBSCURE DEFAULT VALUES

```
// DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VF :
    $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()
```

OBSCURE DEFAULT VALUES

```
// DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VF :
    $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()

// default argument 0 of DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
    $@convention(thin) () -> @out Date
```

OBSCURE DEFAULT VALUES

```
// DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VF :
    $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()

// default argument 0 of DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
    $@convention(thin) () -> @out Date

// function_ref default argument 0 of DatePrinter.printDate(_:)
%5 = function_ref @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
     $@convention(thin) () -> @out Date
%6 = alloc_stack $Date
%7 = apply %5(%6) : $@convention(thin) () -> @out Date
%8 = class_method %4 : $DatePrinter, #DatePrinter.printDate :
     (DatePrinter) -> (Date) -> (),
     $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()
%9 = apply %8(%6, %4) :
     $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()
```

OBSCURE DEFAULT VALUES

```
// default argument 0 of DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
    $@convention(thin) () -> @out Date
```

OBSCURE DEFAULT VALUES

```
// default argument 0 of DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
    $@convention(thin) () -> @out Date

// default argument 0 of EpochDatePrinter.printDate(_:)
sil hidden [ossa] @$s4main16EpochDatePrinterC05printC0yy10Foundation0C0VFfA_ :
    $@convention(thin) () -> @out Date
```

OBSCURE DEFAULT VALUES

```
// default argument 0 of DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
    $@convention(thin) () -> @out Date

// default argument 0 of EpochDatePrinter.printDate(_:)
sil hidden [ossa] @$s4main16EpochDatePrinterC05printC0yy10Foundation0C0VFfA_ :
    $@convention(thin) () -> @out Date
```

```
// function_ref default argument 0 of DatePrinter.printDate(_:)
%10 = function_ref @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
      $@convention(thin) () -> @out Date
%11 = alloc_stack $Date
%12 = apply %10(%11) : $@convention(thin) () -> @out Date
%13 = class_method %9 : $DatePrinter, #DatePrinter.printDate :
      (DatePrinter) -> (Date) -> (),
      $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()
%14 = apply %13(%11, %9) :
      $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()
```

OBSCURE DEFAULT VALUES

```
// default argument 0 of DatePrinter.printDate(_:)
sil hidden [ossa] @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
    $@convention(thin) () -> @out Date

// default argument 0 of EpochDatePrinter.printDate(_:)
sil hidden [ossa] @$s4main16EpochDatePrinterC05printC0yy10Foundation0C0VFfA_ :
    $@convention(thin) () -> @out Date


%7 = upcast %6 : $EpochDatePrinter to $DatePrinter
store %7 to [init] %3 : $*DatePrinter
%9 = load_borrow %3 : $*DatePrinter
// function_ref default argument 0 of DatePrinter.printDate(_:)
%10 = function_ref @$s4main11DatePrinterC05printB0yy10Foundation0B0VFfA_ :
      $@convention(thin) () -> @out Date
%11 = alloc_stack $Date
%12 = apply %10(%11) : $@convention(thin) () -> @out Date
%13 = class_method %9 : $DatePrinter, #DatePrinter.printDate :
      (DatePrinter) -> (Date) -> (),
      $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()
%14 = apply %13(%11, %9) :
      $@convention(method) (@in_guaranteed Date, @guaranteed DatePrinter) -> ()
```

OBSCURE DEFAULT VALUES

```
open class DatePrinter {
  open fun printDate(date: LocalDate = LocalDate.now()) {
    print("Date is " + date)
  }
}

class EpochDatePrinter: DatePrinter() {
  override fun printDate(date: LocalDate = LocalDate.of(1970, 1, 1) {
    print("Epoch date is " + date)
  }
}
```

Kotlin

OBSCURE DEFAULT VALUES

```
open class DatePrinter {
  open fun printDate(date: LocalDate = LocalDate.now()) {
    print("Date is " + date)
  }
}

class EpochDatePrinter: DatePrinter() {
  override fun printDate(date: LocalDate = LocalDate.of(1970, 1, 1) {
    print("Epoch date is " + date)
  }
}
```

Kotlin

```
error: an overriding function is not allowed to specify default values for its
       parameters
```

OBSCURE DEFAULT VALUES

DISALLOW?

IMPROVE?

OBSCURE DEFAULT VALUES

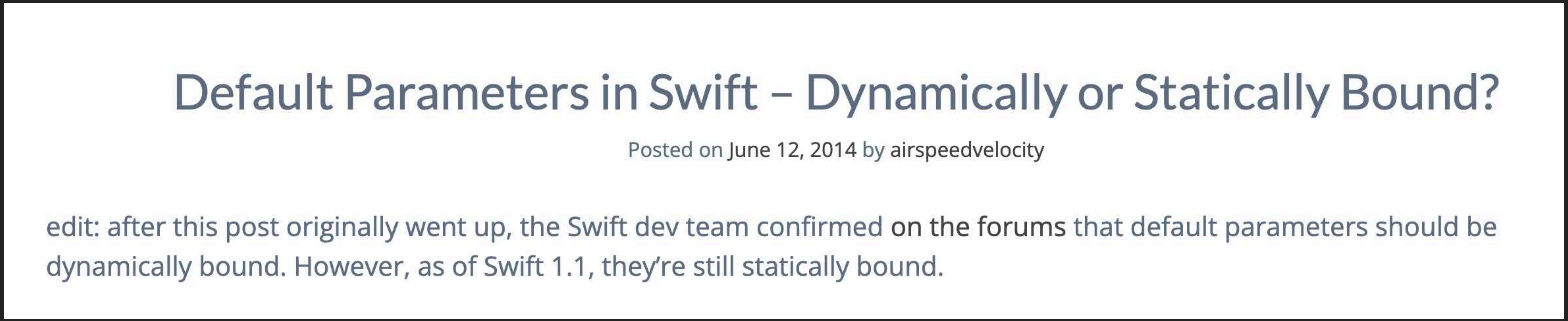
DISALLOW?

IMPROVE?

OBSCURE DEFAULT VALUES

DISALLOW?

IMPROVE?



Default Parameters in Swift – Dynamically or Statically Bound?

Posted on June 12, 2014 by airspeedvelocity

edit: after this post originally went up, the Swift dev team confirmed on the forums that default parameters should be dynamically bound. However, as of Swift 1.1, they're still statically bound.

https://cutt.ly/default-values

EXTENSIONS

F O C U S

OBSCURE PROTOCOL EXTENSION

```
struct Overloaded {
  func foo() -> Int { 1 }
  func foo() -> String { "1" }
  func foo(_ x: Int) -> Int { x }



}
```

OBSCURE PROTOCOL EXTENSION

```swift
struct Overloaded {
  func foo() -> Int { 1 }
  func foo() -> String { "1" }
  func foo(_ x: Int) -> Int { x }

  var fooProperty: Float = 1
  var fooProperty: Bool = true
}
```

Invalid redeclaration of 'fooProperty'

OBSCURE PROTOCOL EXTENSION

```swift
protocol FooProtocol {
  func foo()
}

extension FooProtocol {
  func foo() {
    print("FooProtocol.foo()")
  }
}

struct Foo: FooProtocol {



}
```

OBSCURE PROTOCOL EXTENSION

```
protocol FooProtocol {
  func foo()
}

extension FooProtocol {
  func foo() {
    print("FooProtocol.foo()")
  }
}

struct Foo: FooProtocol {
  func foo() {
    print("Foo.foo()")
  }
}
```

OBSCURE PROTOCOL EXTENSION

```swift
func printer(_ string: String) { print(string) }

protocol ValueProvider { var value: String? { get } }

extension ValueProvider {
  var value: String? { nil }
}

struct Foo: ValueProvider {
  var value = "foo"
}

let f = Foo()
printer(f.value)
```

OBSCURE PROTOCOL EXTENSION

```
func printer(_ string: String) { print(string) }

protocol ValueProvider { var value: String? { get } }

extension ValueProvider {
  var value: String? { nil }
}

struct Foo: ValueProvider {
  var value = "foo"
}

let f = Foo()
printer(f.value)

let string: String = f.value                        // "foo"
let optionalString: String? = f.value               //  nil
```

OBSCURE PROTOCOL EXTENSION

```swift
func printer(_ string: String) { print(string) }

protocol ValueProvider { var value: String? { get } }

extension ValueProvider {
  var value: String? { nil }
}

struct Foo: ValueProvider {
  var value = "foo"
}

let f = Foo()
printer(f.value)

let string: String = f.value                        // "foo"
let optionalString: String? = f.value               //  nil

let direct = f.value                                // String
let viaProtocol = (f as ValueProvider).value        // Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```swift
struct Foo: ValueProvider {}
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```swift
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

~~// Foo.value.getter~~

OBSCURE PROTOCOL EXTENSION

```swift
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

```
f.value
// function_ref ValueProvider.value.getter
%13 = function_ref @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <τ_0_0 where τ_0_0 : ValueProvider> (@in_guaranteed τ_0_0)
        -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

```
f.value
// function_ref ValueProvider.value.getter
%13 = function_ref @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <τ_0_0 where τ_0_0 : ValueProvider> (@in_guaranteed τ_0_0)
        -> @owned Optional<String>
```

```
  (f as ValueProvider).value
%23 = open_existential_addr immutable_access %19 :
      $*ValueProvider to $*@opened("7D5182E8-3F52-11EC-BDBA-ACDE48001122") (ValueProvider)

%26 = witness_method $@opened("7D5182E8-3F52-11EC-BDBA-ACDE48001122") (ValueProvider),
#ValueProvider.value!getter :
      <Self where Self : ValueProvider> (Self) -> () -> String?, %23 :
      $*@opened("7D5182E8-3F52-11EC-BDBA-ACDE48001122") (ValueProvider) :
      $@convention(witness_method: ValueProvider) <τ_0_0 where τ_0_0 : ValueProvider>
          (@in_guaranteed τ_0_0) -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```swift
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

```
sil_witness_table hidden Foo: ValueProvider module main {
  method #ValueProvider.value!getter:
    <Self where Self : ValueProvider> (Self) -> () -> String? :
      @$s4main3FooVAA13ValueProviderA2aDP5valueSSSgvgTW
}

// protocol witness for ValueProvider.value.getter in conformance Foo
sil private [transparent] [thunk] @$s4main3FooVAA13ValueProviderA2aDP5valueSSSgvgTW :
  $@convention(witness_method: ValueProvider) (@in_guaranteed Foo)
    -> @owned Optional<String> {

bb0(%0 : $*Foo):
  // function_ref ValueProvider.value.getter
  %1 = function_ref @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <τ_0_0 where τ_0_0 : ValueProvider> (@in_guaranteed τ_0_0)
      -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

```
sil_witness_table hidden Foo: ValueProvider module main {
  method #ValueProvider.value!getter:
    <Self where Self : ValueProvider> (Self) -> () -> String? :
      @$s4main3FooVAA13ValueProviderA2aDP5valueSSSgvgTW
}

// protocol witness for ValueProvider.value.getter in conformance Foo
sil private [transparent] [thunk] @$s4main3FooVAA13ValueProviderA2aDP5valueSSSgvgTW :
  $@convention(witness_method: ValueProvider) (@in_guaranteed Foo)
    -> @owned Optional<String> {

bb0(%0 : $*Foo):
  // function_ref ValueProvider.value.getter
  %1 = function_ref @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <τ_0_0 where τ_0_0 : ValueProvider> (@in_guaranteed τ_0_0)
      -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```swift
struct Foo: ValueProvider {
  var value: String? = "foo"
}
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {
  var value: String? = "foo"
}

// Foo.value.getter
sil hidden [transparent] @$s4main3FooV5valueSSSgvg :
    $@convention(method) (@guaranteed Foo) -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {
  var value: String? = "foo"
}

// Foo.value.getter
sil hidden [transparent] @$s4main3FooV5valueSSSgvg :
    $@convention(method) (@guaranteed Foo) -> @owned Optional<String>

f.value
%10 = struct_element_addr %3 : $*Foo, #Foo.value
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {
  var value: String? = "foo"
}

// Foo.value.getter
sil hidden [transparent] @$s4main3FooV5valueSSSgvg :
    $@convention(method) (@guaranteed Foo) -> @owned Optional<String>

f.value
%10 = struct_element_addr %3 : $*Foo, #Foo.value


// protocol witness for ValueProvider.value.getter in conformance Foo
sil private [transparent] [thunk] @$s4main3FooVAA13ValueProviderA2aDP5valueSSSgvgTW :
  $@convention(witness_method: ValueProvider) (@in_guaranteed Foo)
    -> @owned Optional<String> {

bb0(%0 : $*Foo):
  %1 = load %0 : $*Foo
  // function_ref Foo.value.getter
  %2 = function_ref @$s4main3FooV5valueSSSgvg :
      $@convention(method) (@guaranteed Foo) -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```swift
struct Foo: ValueProvider {
  var value = "foo"
}
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {
  var value = "foo"
}


// Foo.value.getter
sil hidden [transparent] @$s4main3FooV5valueSSSgvg :
    $@convention(method) (@guaranteed Foo) -> @owned String
```

OBSCURE PROTOCOL EXTENSION

```
struct Foo: ValueProvider {
  var value = "foo"
}
```

```
// ValueProvider.value.getter
sil hidden @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <Self where Self : ValueProvider> (@in_guaranteed Self)
        -> @owned Optional<String>
```

```
// Foo.value.getter
sil hidden [transparent] @$s4main3FooV5valueSSSgvg :
    $@convention(method) (@guaranteed Foo) -> @owned String


// protocol witness for ValueProvider.value.getter in conformance Foo
sil private [transparent] [thunk] @$s4main3FooVAA13ValueProviderA2aDP5valueSSSgvgTW :
  $@convention(witness_method: ValueProvider) (@in_guaranteed Foo)
    -> @owned Optional<String> {

bb0(%0 : $*Foo):
  // function_ref ValueProvider.value.getter
  %1 = function_ref @$s4main13ValueProviderPAAE5valueSSSgvg :
    $@convention(method) <τ_0_0 where τ_0_0 : ValueProvider> (@in_guaranteed τ_0_0)
      -> @owned Optional<String>
```

OBSCURE PROTOCOL EXTENSION

```swift
extension Foo: ValueProvider {
  var value: String { "" }
}
```

OBSCURE PROTOCOL EXTENSION

```swift
extension Foo: ValueProvider {
  var value: String { "" }
}
```

```
Property 'value' nearly matches defaulted requirement 'value' of protocol ‘ValueProvider'
```

https://cutt.ly/protocol-gist https://cutt.ly/extension-forum

THANK YOU!

@Losiowaty

Pawel Lopusinski