WHAT'S NEW IN SWIFT 5.2

FEDERICO ZANETELLO

★★★★★ fivestars.blog • @zntfdr

XCODE 11.4 🎉

- iOS/macOS/tvOS/watchOS universal apps 🚀
- View debugger enhancements 👀
- Xcode Previews canvas enhancements 🖼️
- simctl improvements 🖥️
- Xcode Simulators enhancements 📲
- All the Swift 5.2 goodies 🏎️
- XCTest improvements 🤖

ALL THE SWIFT 5.2 GOODIES

SWIFT EVOLUTION

SE-0253

Callable values of user-defined nominal types

Call functions directly from the instance name

```swift
struct Incrementer {
    var value: Int

    //                 ↓ new in Swift 5.2
    mutating func callAsFunction(add number: Int) {
        value += number
    }
}

var inc = Incrementer(value: 5)
inc(add: 1) // inc.value = 6
inc(add: 4) // inc.value = 10
```

SE-0249

Key Path Expressions as Functions

Even more uses of keypaths!

```
struct User {
    let email: String
    let isAdmin: Bool
}

let users: [User] = [...]

//                                  ↓ new in Swift 5.2
let allEmails: [String] = users.map(\.email)
let adminsOnly: [User] = users.filter(\.isAdmin)
```

SWIFT REPORT

SR-6118

mechanism to hand through #file/#line in subscripts

Subscripts default arguments!

```swift
struct Subscriptable {
  //                       ↓ new in Swift 5.2
  subscript(x: Int, y: Int = 0) -> Int {
    ...
  }
}

let s = Subscriptable()
print(s[2])
```

SR-4206

Override checking does not properly enforce requirements

No more generic restrictions in overrides

```
protocol P {}

class Base {
  func foo<T>(arg: T) {}
}

class Derived: Base {
  //                   ↓ error in Swift 5.2
  override func foo<T: P>(arg: T) {}
}
```

SR-11429

Don't look through CoerceExprs in markDirectCallee

as operator can now be used to disambiguate a call to a function with argument labels.

```swift
func foo(x: Int) {}
func foo(x: UInt) {}

(foo as (Int) -> Void)(5)  // Calls foo(x: Int)
(foo as (UInt) -> Void)(5) // Calls foo(x: UInt)

// Swift <5.2: Error: Ambiguous reference to member 'foo(x:)'
```

SR-2189

Nested function with local default value crashes

local functions default arguments can capture outer scope values

```swift
func outer(x: Int) -> (Int, Int) {
  var x = 5
  //                 ↓ works in Swift 5.2
  func inner(y: Int = x) -> Int {
    return y
  }

  return (inner(), inner(y: 0))
}
```

SR-2790

Reject UnsafePointer initialization via implicit pointer conversion

Safer temporary pointers management with new warnings

```
struct S {
  var ptr: UnsafePointer<Int8>
}

func foo() {
  var i: Int8 = 0
  let ptr = UnsafePointer(&i) // dangling pointer

  let s1 = S(ptr: [1, 2, 3]) // argument should be a pointer that outlives the call

  let s2 = S(ptr: "hello") // argument should be a pointer that outlives the call
}
```

SR-11841

Lazy filter runs in unexpected order

Lazy filters now run the other (right) way

```
let evens = (1...10).lazy
    .filter { $0.isMultiple(of: 2) }
    .filter { print($0); return true }
_ = evens.count

// Swift <5.2: Prints 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 on separate lines
// Swift 5.2: Prints 2, 4, 6, 8, 10 on separate lines
```

LINKS

Slides:
github.com/zntfdr/talks

Resources:
github.com/apple/swift-evolution
github.com/apple/swift
forums.swift.org

Slides Style:
jessesquires.com • @jesse_squires

WHAT'S NEW IN SWIFT 5.2

FEDERICO ZANETELLO

★★★★★ fivestars.blog • @zntfdr