Working with Binary Data in Swift

JP Simard

@

Realm

Realm is a mobile database hundreds of millions of people rely on

Objective-C | Swift | Android

Enter your email to join our community newsletter | Subscribe

Sign up for our community newsletter to hear about Realm tutorials, events, tips & more!

Google · amazon · hipmunk · Pinterest · ebay · Budweiser

AVIS · SAP · HYATT · BBC · intel · intuit

L'ORÉAL · adidas · Alibaba · IBM · GoPro

cisco · Walmart · NIKKEI · SoftBank · Virgin · Homeland Security

Options

- → `NSData`
- → `[UInt8]`
- → `withUnsafePointer()`
- → ???

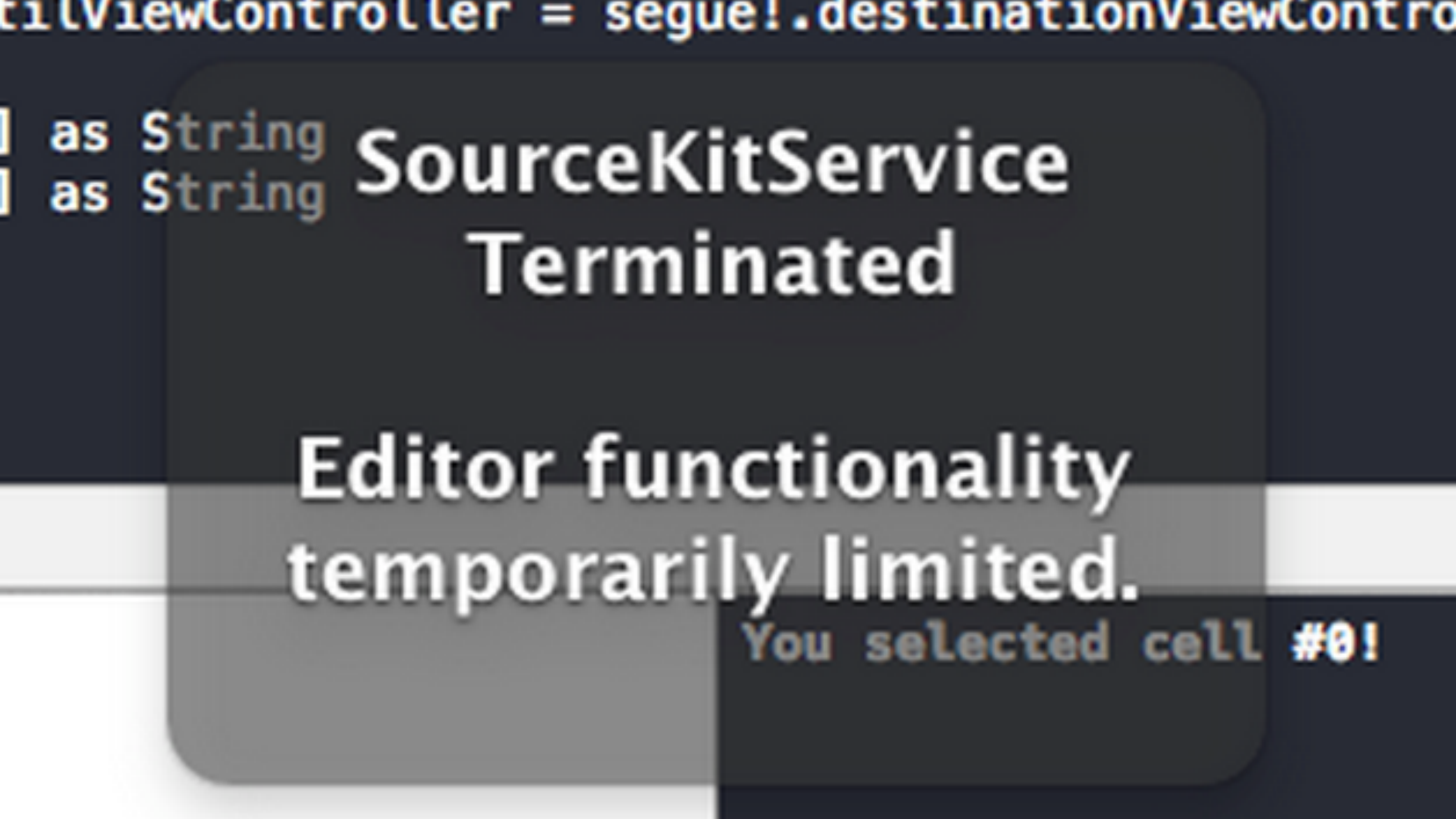
Layout-Aligned Structs

```swift
func encode<T>(var value: T) -> NSData {
    return withUnsafePointer(&value) { p in
        NSData(bytes: p, length: sizeofValue(value))
    }
}

func decode<T>(data: NSData) -> T {
    let pointer = UnsafeMutablePointer<T>.alloc(sizeof(T))
    data.getBytes(pointer, length: sizeof(T))
    return pointer.move()
}
```

```swift
enum Either<T> {
    case Left(T)
    case Right(T)
}

let value = Either.Left("Swift Summit")
let data = encode(value)
data // => <NSData>
let decoded: Either<String> = decode(data)
decoded // => Either.Left("Swift Summit")
```

$ jazzy♪♫

Soulful docs for Swift & Objective-C



```
tilViewController = segue!.destinationViewContro

] as String
] as String
```

SourceKitService Terminated

Editor functionality temporarily limited.

```
You selected cell #0!
```

Get SourceKit syntax map

```
struct A {
    subscript(index: Int) -> () {
        return ()
    }
}
```

Result

```
00 00 00 00 00 00 00 00  30 00 00 00 00 00 00 00

40 4c 81 01 01 00 00 00  00 00 00 00 0c 00 00 00

78 4c 81 01 01 00 00 00  07 00 00 00 14 00 00 00

b0 4c 81 01 01 00 00 00  12 00 00 00 1e 00 00 00

00
```

Result

```
00 00 00 00 00 00 00 00  30 00 00 00 00 00 00 00
--------16 bytes-------  --------16 bytes-------
40 4c 81 01 01 00 00 00  00 00 00 00 0c 00 00 00
--------16 bytes-------  --------16 bytes-------
78 4c 81 01 01 00 00 00  07 00 00 00 14 00 00 00
--------16 bytes-------  --------16 bytes-------
b0 4c 81 01 01 00 00 00  12 00 00 00 1e 00 00 00
--------16 bytes-------  --------16 bytes-------
00
```

```swift
struct SyntaxToken {
    let type: String
    let offset: Int
    let length: Int
}
```

Strideable

```
tokens = 16.stride(through: numberOfTokens * 16, by: 16).map { parserOffset in    .






}
```

```
tokens = 16.stride(through: numberOfTokens * 16, by: 16).map { parserOffset in
    var uid = UInt64(0), offset = 0, length = 0
    data.getBytes(&uid, range: NSRange(location: parserOffset, length: 8))
    data.getBytes(&offset, range: NSRange(location: 8 + parserOffset, length: 4))
    data.getBytes(&length, range: NSRange(location: 12 + parserOffset, length: 4))




}
```

```swift
tokens = 16.stride(through: numberOfTokens * 16, by: 16).map { parserOffset in
    var uid = UInt64(0), offset = 0, length = 0
    data.getBytes(&uid, range: NSRange(location: parserOffset, length: 8))
    data.getBytes(&offset, range: NSRange(location: 8 + parserOffset, length: 4))
    data.getBytes(&length, range: NSRange(location: 12 + parserOffset, length: 4))

    return SyntaxToken(
        type: stringForSourceKitUID(uid) ?? "unknown",
        offset: offset,
        length: length >> 1
    )
}
```

Collection of Bytes

- Making our own
- Conforming to `ExtensibleCollectionType`
- What `Index` type should we use? `Int`?

Just end up with `[UInt8]`

Links

→ SourceKittenFramework SyntaxMap

→ Convert structs and enums to NSData

→ robnapier.net/nsdata

→ Simon Lewis on parsing OLE/COM

→ github.com/realm/jazzy

→ realm.io

try! ask(...)

```swift
struct Question {
    let value: String
    let canJPAnswer: Bool
}
func ask<S: SequenceType where S.Generator.Element == Question>(questions: S) throws {
    // excercise for attendees
}
```