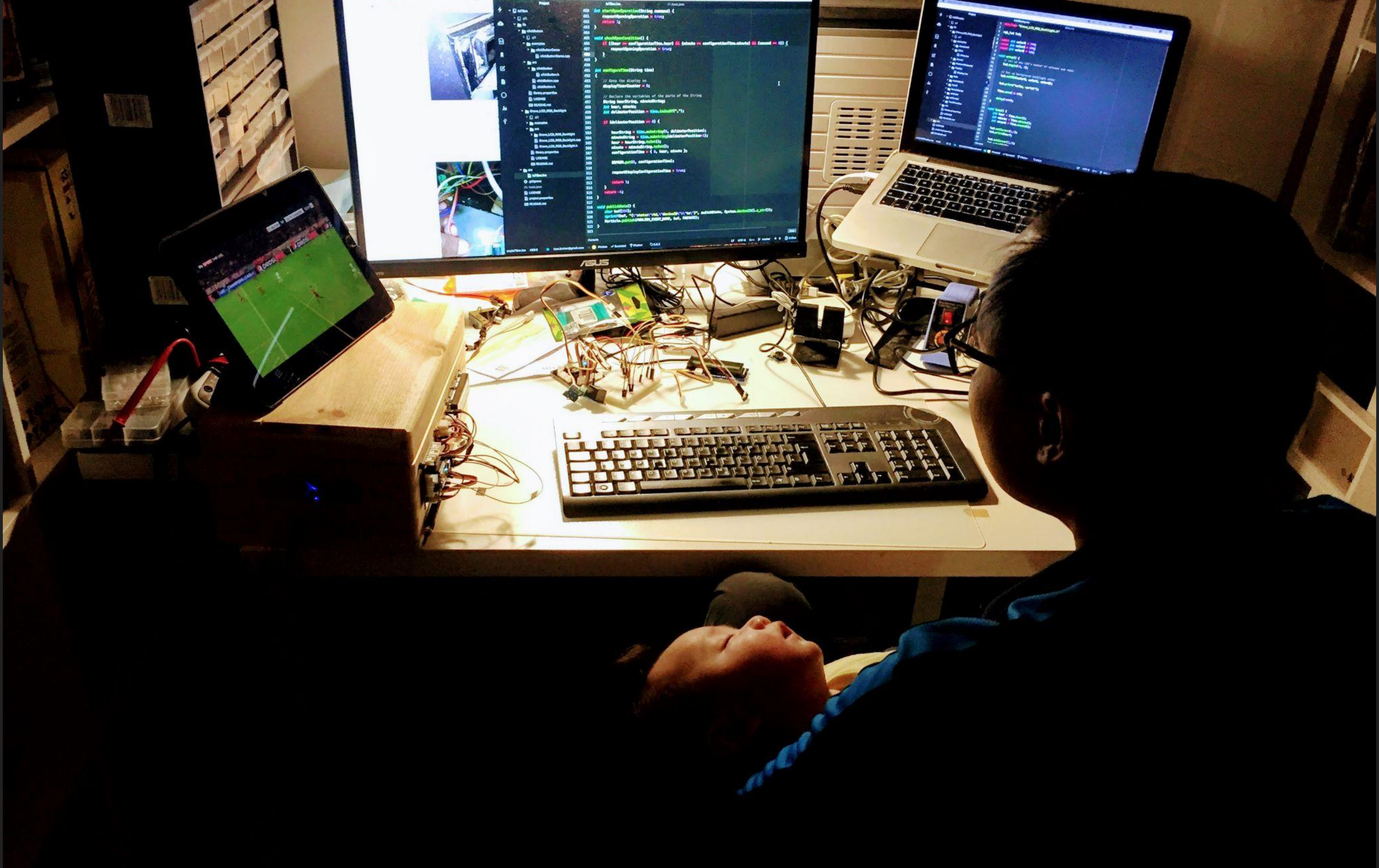
Swift beyond Apple Platforms

An Tran - https://antranapp.com - @antranapp

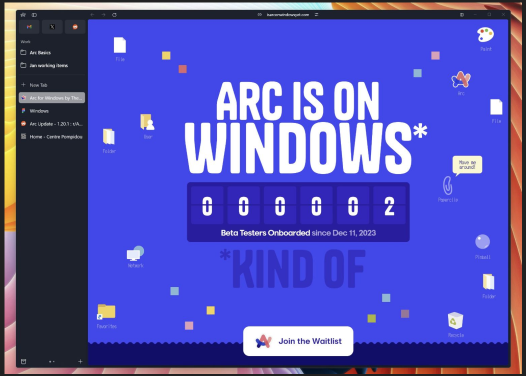


Swift on Apple Platforms

macOS, iOS, tvOS, watchOS, visionOS

Swift is a general-purpose programming language that's approachable for newcomers and powerful for experts.

Windows


ARC IS ON WINDOWS*

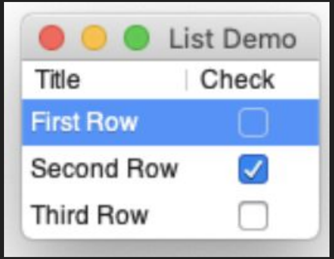
0 0 0 0 0 2

Beta Testers Onboarded since Dec 11, 2023

*KIND OF

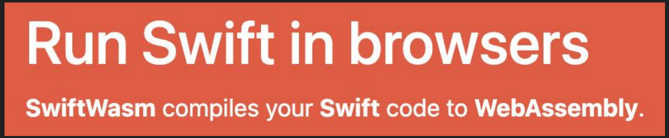
Linux

SwiftGtk


List Demo

Title	Check
First Row	☐
Second Row	☑
Third Row	☐

Browsers


Run Swift in browsers

SwiftWasm compiles your **Swift** code to **WebAssembly**.

Server/Cloud

VAPOR

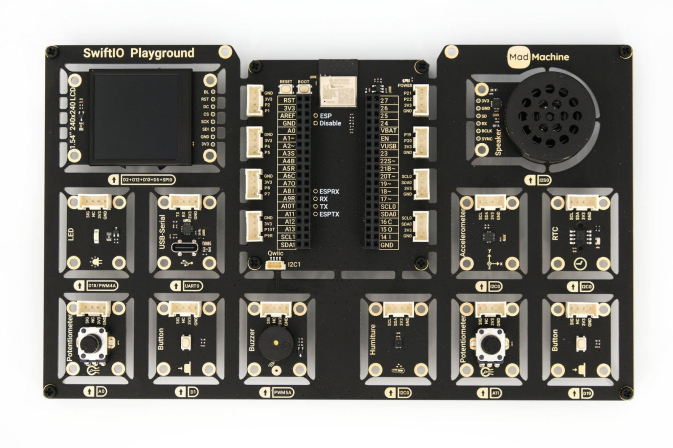
AWS SDK for Swift

Hummingbird

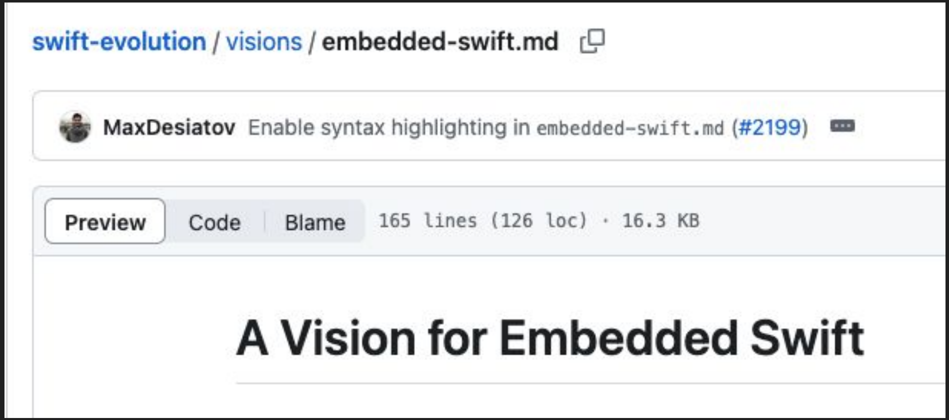
Swift on Embedded Systems

Arduino, Raspberry PI, SwiftIO Playgound

S4A Swift for Arduino



Swift-Arm

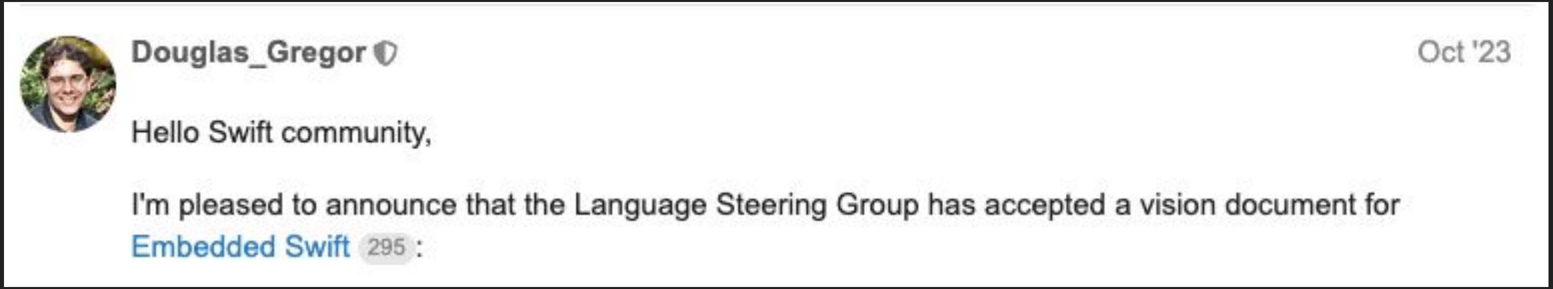


swift-evolution / visions / embedded-swift.md

MaxDesiatov Enable syntax highlighting in `embedded-swift.md` (#2199)

`165 lines (126 loc) · 16.3 KB`

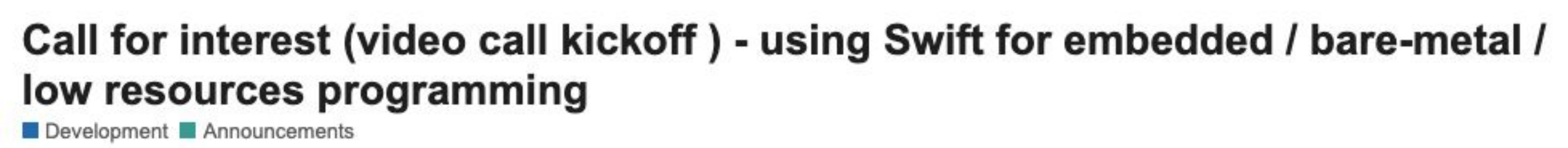
A Vision for Embedded Swift



Douglas_Gregor Oct '23

Hello Swift community,

I'm pleased to announce that the Language Steering Group has accepted a vision document for Embedded Swift 295 :

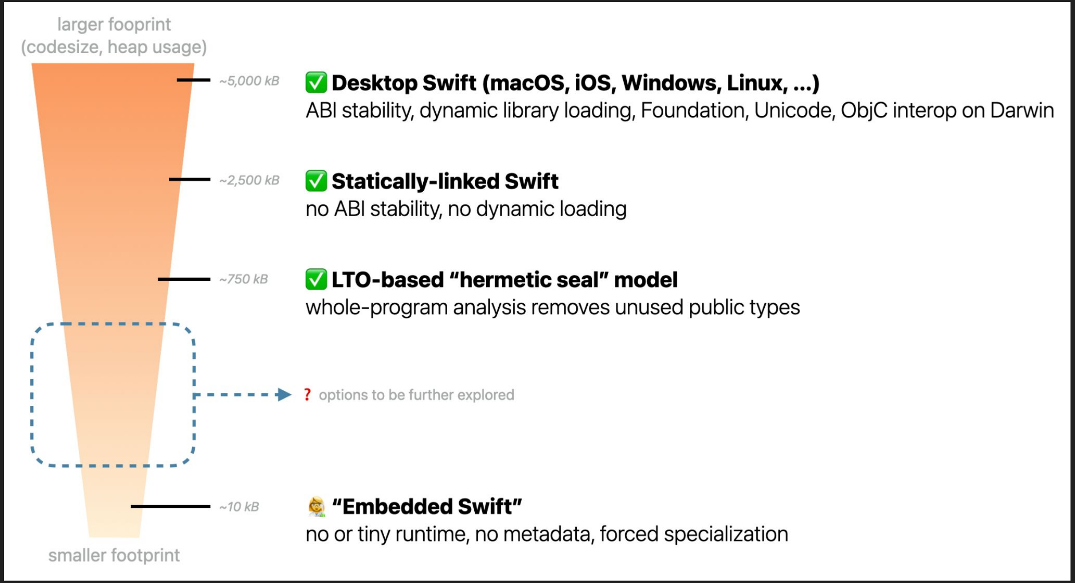


Call for interest (video call kickoff) - using Swift for embedded / bare-metal / low resources programming

🟦 Development 🟩 Announcements

Embedded Swift

- Classes cannot be instantiated,
- Indirect enums cannot be constructed.
- Escaping closures are not allowed.
- Dynamic containers (arrays, dictionaries, sets) and strings not supported
- …..



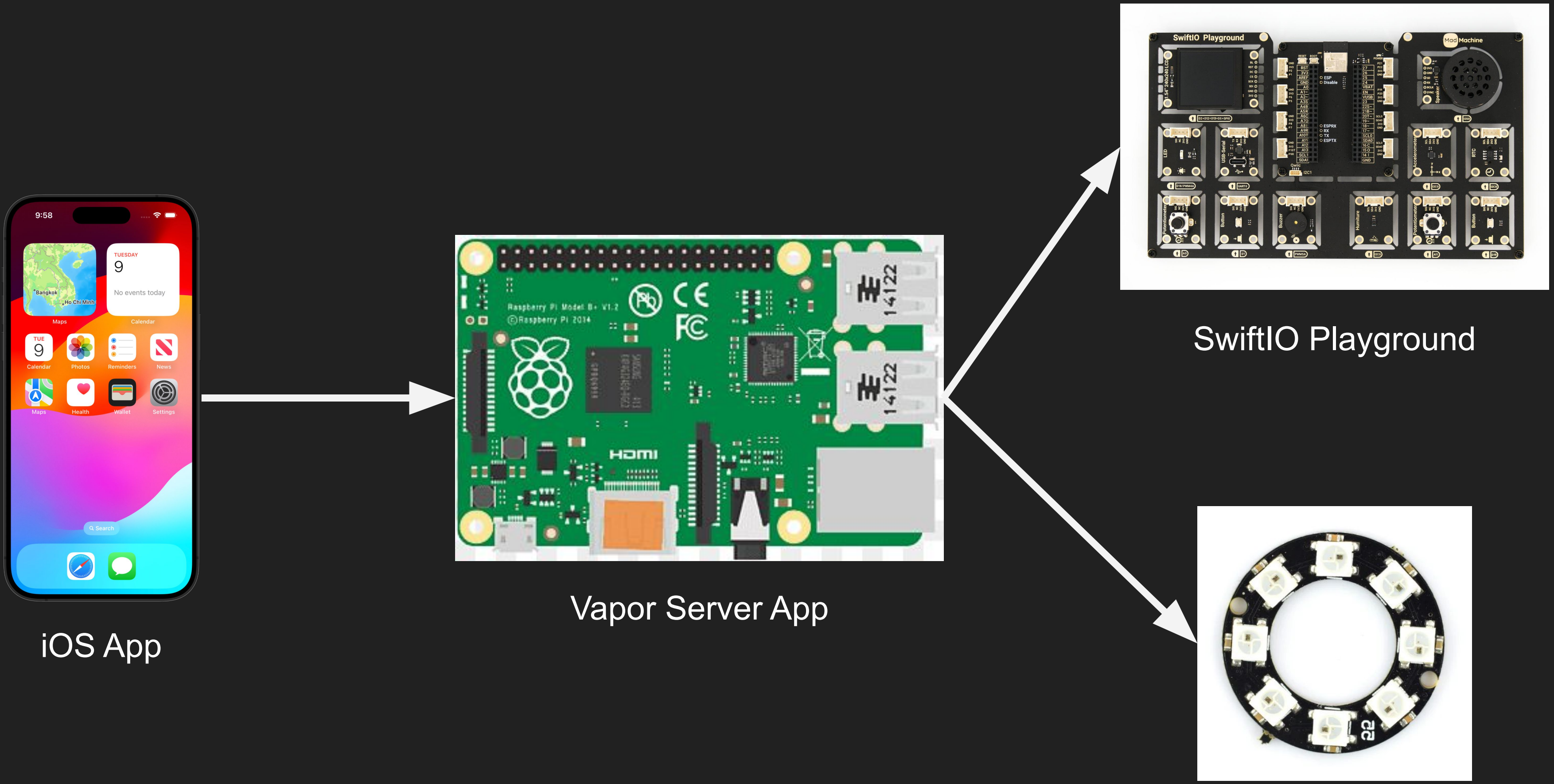
✅ **Desktop Swift (macOS, iOS, Windows, Linux, ...)**
ABI stability, dynamic library loading, Foundation, Unicode, ObjC interop on Darwin

✅ **Statically-linked Swift**
no ABI stability, no dynamic loading

✅ **LTO-based “hermetic seal” model**
whole-program analysis removes unused public types

❓ options to be further explored

“Embedded Swift”
no or tiny runtime, no metadata, forced specialization



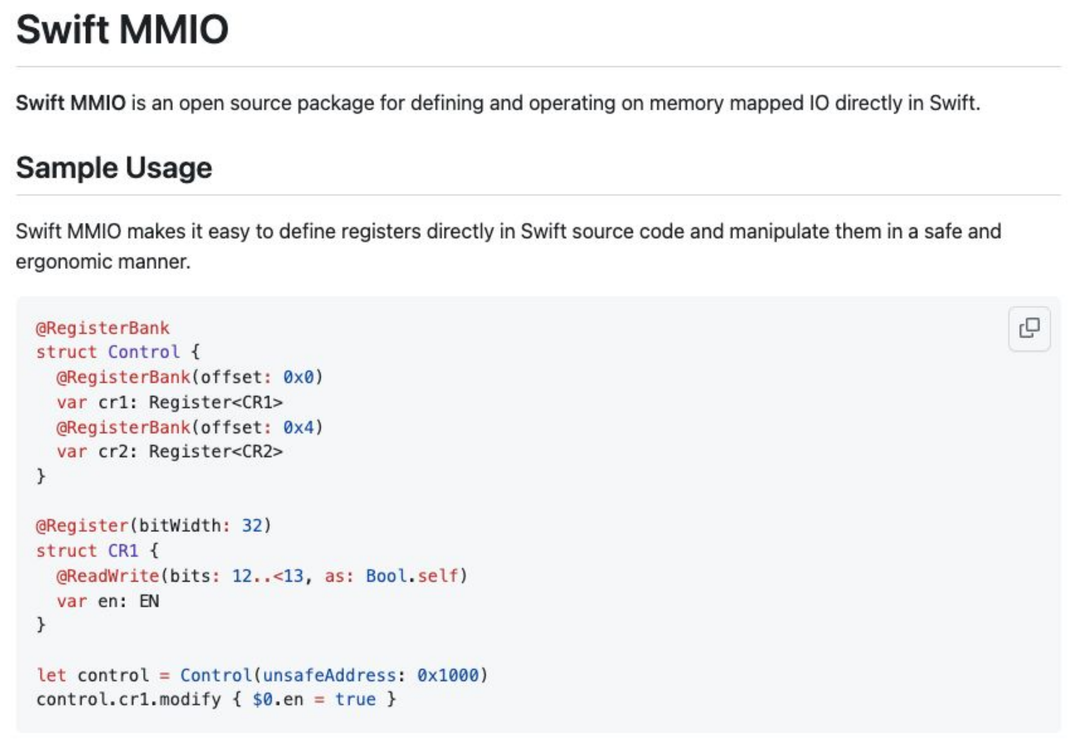
iOS App

Vapor Server App

SwiftIO Playground

Neo Pixel

Swift Memory-Mapped Input/Output (MMIO)



Swift MMIO

Swift MMIO is an open source package for defining and operating on memory mapped IO directly in Swift.

Sample Usage

Swift MMIO makes it easy to define registers directly in Swift source code and manipulate them in a safe and ergonomic manner.

```swift
@RegisterBank
struct Control {
  @RegisterBank(offset: 0x0)
  var cr1: Register<CR1>
  @RegisterBank(offset: 0x4)
  var cr2: Register<CR2>
}

@Register(bitWidth: 32)
struct CR1 {
  @ReadWrite(bits: 12..<13, as: Bool.self)
  var en: EN
}

let control = Control(unsafeAddress: 0x1000)
control.cr1.modify { $0.en = true }
```

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

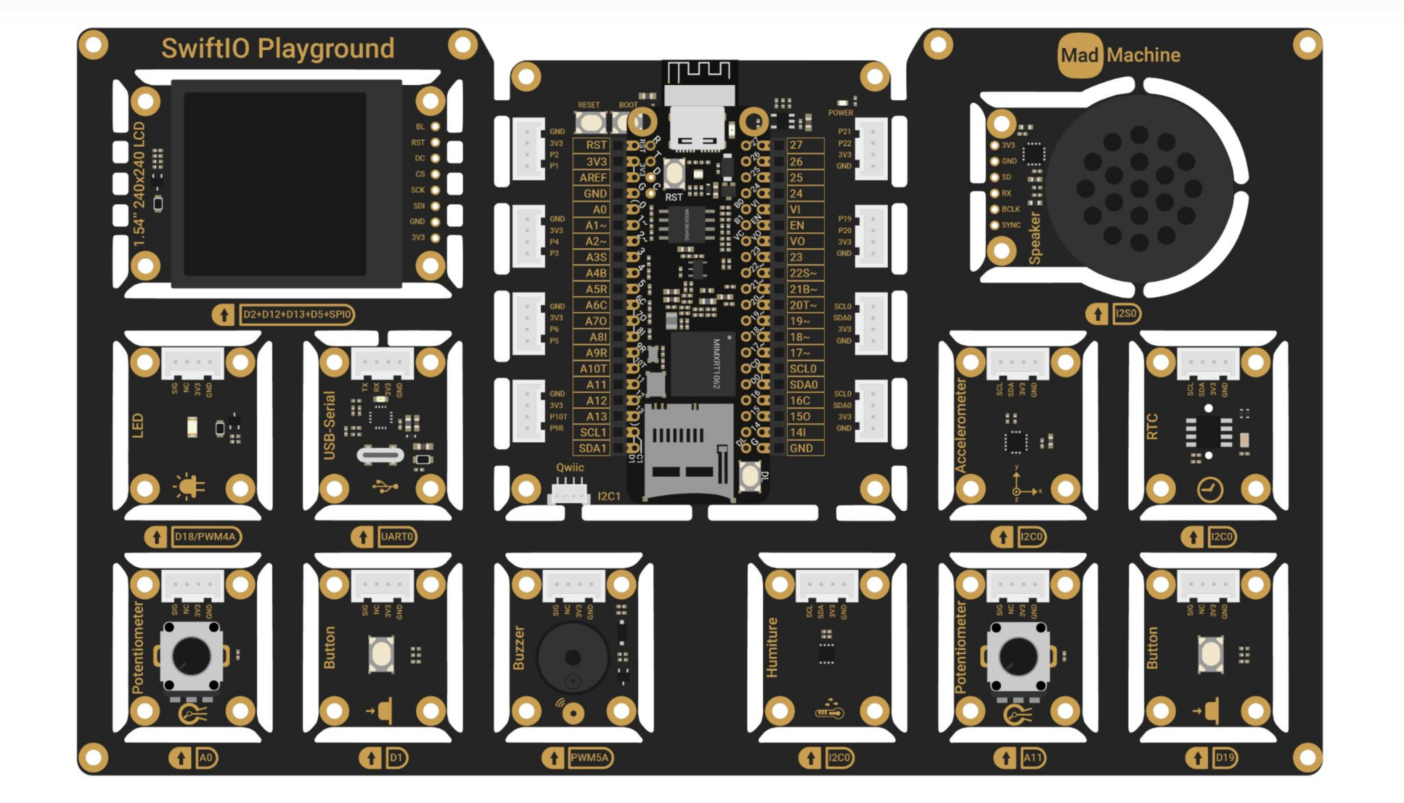
```swift
import SwiftIO
import MadBoard
import ST7789

let uart = UART(Id.UART0)

// Initialize the SPI pin and the digital pins for the LCD.
let bl = DigitalOut(Id.D2)
let rst = DigitalOut(Id.D12)
let dc = DigitalOut(Id.D13)
let cs = DigitalOut(Id.D5)
let spi = SPI(Id.SPI0, speed: 30_000_000)

// Initialize the LCD using the pins above. Rotate the screen to keep the original at the upper left.
let screen = ST7789(spi: spi, cs: cs, dc: dc, rst: rst, bl: bl, rotation: .angle90)

// Store some color values for easier reference later.
let red   = UInt16(0xF800).byteSwapped
let green = UInt16(0x07E0).byteSwapped
let blue = UInt16(0x001F).byteSwapped

while true {
    // Check if there is any message from computer.
    let count = uart.checkBufferReceived()

    if count > 0 {
        // Read data from UART buffer.
        var buffer = [UInt8](repeating: 0, count: count)
        uart.read(into: &buffer)

        // Decode the data since the text from computer is sent in UTF8 format.
        let command = String(decoding: buffer, as: UTF8.self)
        // Connect the port on your micro board in serial monitor to see printed message
        print(command)

        // Set digital output according to the command from computer.
        switch command  {
        case "0": screen.clearScreen(red)
        case "1": screen.clearScreen(green)
        case "2": screen.clearScreen(blue)
        default: break
        }
    }

    sleep(ms: 10)
}
```

Links

- SwiftLED Project 🥳
- Swift on embedded systems
- Introducing Swift MMIO
- Swift for Arduino
- SwiftIO Playground board
- Swift ARM
- Introduce embedded development using Swift
- Swift Forums: embedded

One More Thing …

A Free SwiftIO Playground (thanks to madmachine.io 🎉)



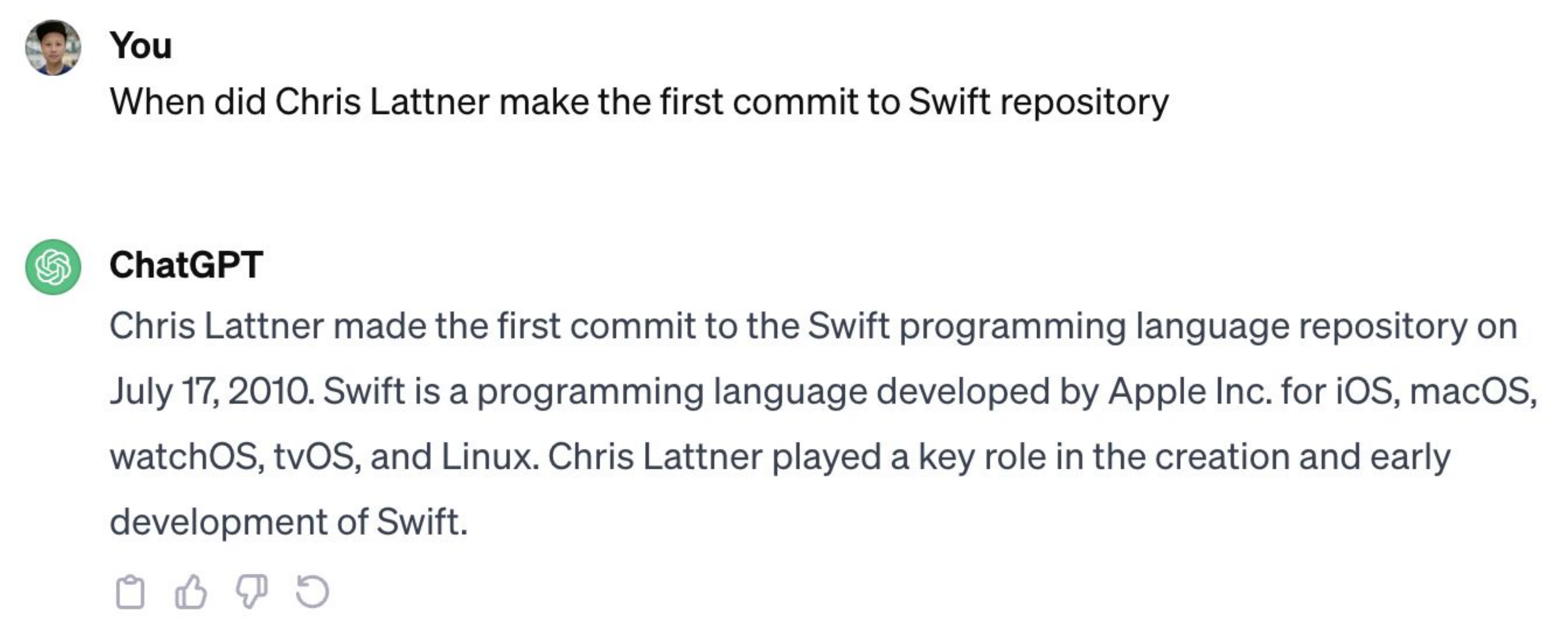
… QUIZ TIME …

Get ready on Discord!

Are you ready?

When (dd/mm/yy) did Chris Lattner make the first commit to the Swift programming language repository?

Solution!



You

When did Chris Lattner make the first commit to Swift repository

ChatGPT

Chris Lattner made the first commit to the Swift programming language repository on July 17, 2010. Swift is a programming language developed by Apple Inc. for iOS, macOS, watchOS, tvOS, and Linux. Chris Lattner played a key role in the creation and early development of Swift.

Thank you

An Tran - https://antranapp.com - @antranapp