Swift on the Server

@tombuildsstuff

Tom Harvey

- Senior Infrastructure Engineer at Ve London
- @tombuildsstuff on 
- Primarily infrastructure & backend services - also iOS
- We're hiring!

Agenda

- Introduction / Crash-Course in Swift
- Building API's in Swift
- State of Play
- Deploying Swift to Production
- The Future

Introduction to Swift

WWDC 2014

Swift announced at WWDC

Swift

- A modern language designed for the future

Swift

- A modern language designed for the future
- **`Objective-C without the C`**

Swift

- A modern language designed for the future
- **`Objective-C without the C`**
- Designed as a language at every level from Systems Programming to Mobile development

Swift

- A modern language designed for the future
- **`Objective-C without the C`**
- Designed as a language at every level from Systems Programming to Mobile development
- Emphasis on Learning

Swift

- A modern language designed for the future
- **`Objective-C without the C`**
- Designed as a language at every level from Systems Programming to Mobile development
- Emphasis on Learning:
  - Interactive Playgrounds

Swift

- A modern language designed for the future
- **`Objective-C without the C`**
- Designed as a language at every level from Systems Programming to Mobile development
- Emphasis on Learning:
  - Interactive Playgrounds
  - Language Spec as an iBook

Swift

- A modern language designed for the future
- **`Objective-C without the C`**
- Designed as a language at every level from Systems Programming to Mobile development
- Emphasis on Learning:
  - Interactive Playgrounds
  - Language Spec as an iBook
  - REPL

Interactive Playgrounds

Presenting the Scene in the Timeline

SpriteKit content is presented in an `SKView` object. The view runs simulations and renders the content. All content is represented in an `SKScene` object, which is the root node for all nodes in a tree of `SKNode` objects. In this scene, you'll add nodes and create your game's content.

The scene was loaded from a SpriteKit scene file, which we created in Xcode's SpriteKit Level Designer. All resources, including scenes and image assets, have been embedded in the playground bundle and are available for us to use. You can add your own images in the bundle, too. Just right-click on the Balloons.playground file in the Finder and choose Show Package Contents.

```
let sceneView = SKView(frame: CGRect(x: 0, y: 0, width: 850, height: 638))
let scene = SKScene(fileNamed: "GameScene")
scene.scaleMode = .AspectFill
sceneView.presentScene(scene)

XCPShowView(identifier: "Balloons", view: sceneView)
```

EXPERIMENT

Because gravity is defined by the scene's physics world (its `physicsWorld` property), in the the world of SpriteKit laws of physics can be altered!

Turn the gravity upside down by changing the `scene.physicsWorld.gravity` vector. Hint: Invert the sign of the vector's `dy` component.

To actually see the content of a scene in a playground, we use the XCPlayground function `XCPShowView`. This function renders the view live in the timeline editor so that the game is visible. From this point on, every change you make will be rendered live and be visible in the timeline editor.

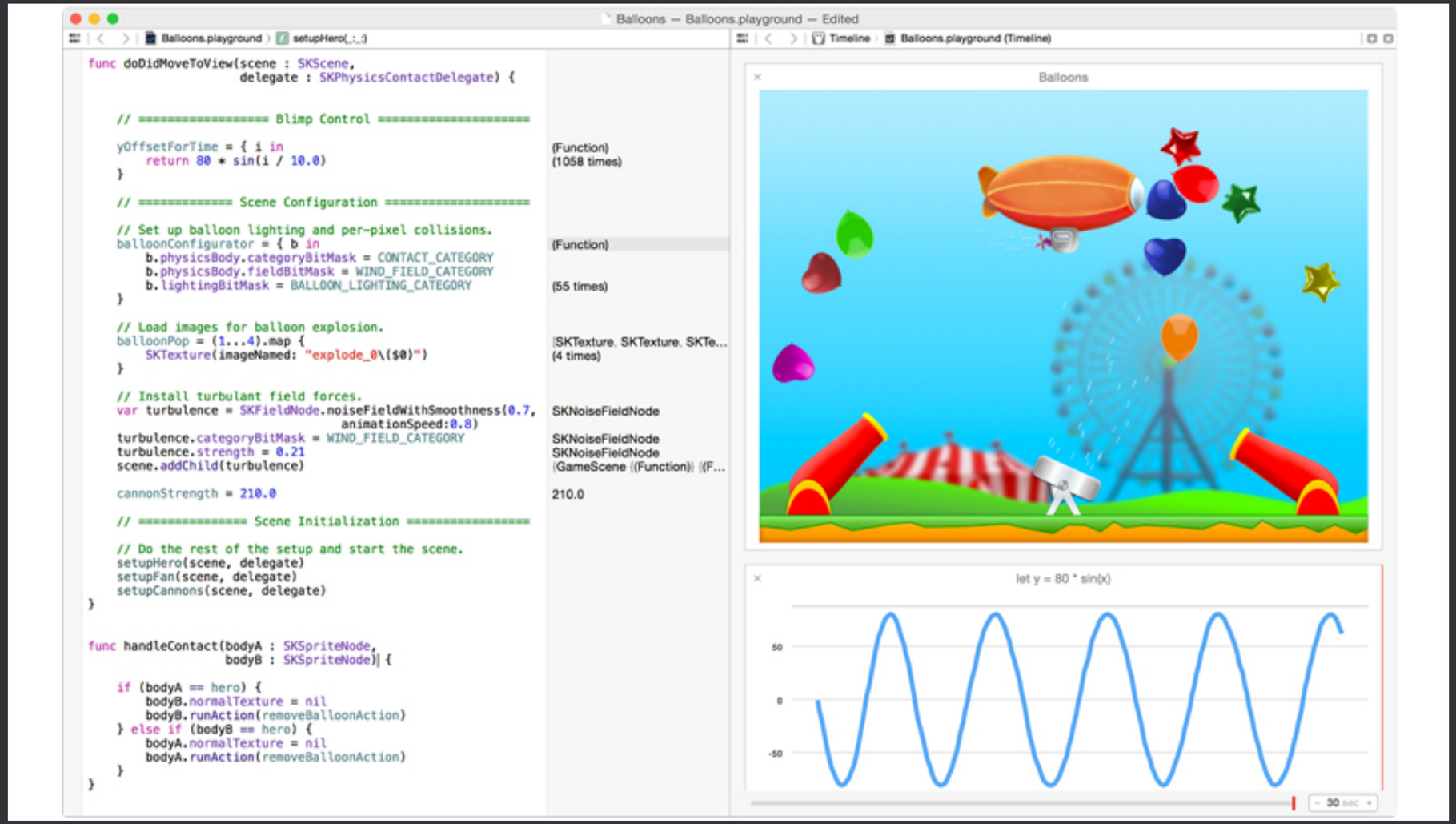
Firing the Cannons

When the cannons fire, let's add a balloon and move it across the scene. The balloons are sprite nodes, and we'll give each balloon a texture with a random element from our collection of balloon images.

Here, we use the `map` function of Swift arrays to create an array of `SKTexture` objects from an array of image names. With our array of textures, we can simply generate a random index within its range and then create a sprite node with the texture at that index.

```
let images = [
    "blue", "heart-blue", "star-blue",
    "green", "star-green", "heart-pink",
    "heart-red", "orange", "red",
    "star-gold", "star-pink", "star-red",
    "yellow"
]
```

Interactive Playgrounds



Introduction to Swift

The Early Days

Introduction to Swift

The Early Days: SourceKit

SourceKitService
Crashed
Crashlog generated in
~/Library/Logs/
DiagnosticReports

Editor functionality
temporarily limited.

Introduction to Swift

The Early Days

- Public Beta: brought lots of feedback + breaking changes

Introduction to Swift

The Early Days

- Public Beta: brought lots of feedback + breaking changes
- 1.0 in September 2014

Sep 9, 2014

Swift Has Reached 1.0

On June 2, 2014 at WWDC, the Swift team finally showed you what we had been working on for years. That was a big day with lots of excitement, for us and for developers around the world. Today, we've reached the second giant milestone:

Swift version 1.0 is now GM.

You can now submit your apps that use Swift to the App Store. Whether your app uses Swift for a small feature or a complete application, now is the time to share your app with the world. It's your turn to excite everyone with your new creations.

Swift for OS X

Today is the GM date for Swift on iOS. We have one more GM date to go for Mac. Swift for OS X currently requires the SDK for OS X Yosemite, and when Yosemite ships later this fall, Swift will also be GM on the Mac. In the meantime, you can keep developing your Mac apps with Swift by downloading the beta of Xcode 6.1.

The Road Ahead

You'll notice we're using the word "GM", not "final". That's because Swift will continue to advance with new features, improved performance, and refined syntax. In fact, you can expect a few improvements to come in Xcode 6.1 in time for the Yosemite launch. Because your apps today embed a version of the Swift GM runtime, they will continue to run well into the future.

The Road Ahead

You'll notice we're using the word "GM", not "final". That's because Swift will continue to advance with new features, improved performance, and refined syntax. In fact, you can expect a few improvements to come in Xcode 6.1 in time for the Yosemite launch. Because your apps today embed a version of the Swift GM runtime, they will continue to run well into the future.

Introduction to Swift

The Early Days

- Public Beta: brought lots of feedback + breaking changes
- 1.0 in September 2014
- The Swift Runtime shipped inside apps - enabling breaking changes

Introduction to Swift

The Early Days

- Public Beta: brought lots of feedback + breaking changes
- 1.0 in September 2014
- The Swift Runtime shipped inside apps - enabling breaking changes
- 2.0 brought Guard Statements, Exceptions + Error Handling

Open Source

In addition to new features, the big news is that Apple will be making Swift open source later this year. We are all incredibly excited about this, and look forward to giving you a lot more information as the open source release gets nearer. Here is what we can tell you so far:

- Swift source code will be released under an OSI-approved permissive license.
- Contributions from the community will be accepted — and encouraged.
- At launch we intend to contribute ports for OS X, iOS, and Linux.
- Source code will include the Swift compiler and standard library.
- We think it would be amazing for Swift to be on all your favorite platforms.

We are excited about the opportunities an open source Swift creates for our industry. Baked-in safety features combined with excellent speed mean it has the chance to dramatically improve software versus using C-based languages. Swift is packed with modern features, it's fun to write, and we believe it will get used in a lot of places. Together, we have an exciting road ahead.

Swift

- ABOUT SWIFT
- BLOG
- DOWNLOAD
- GETTING STARTED
- DOCUMENTATION
- MIGRATING TO SWIFT 3
- SOURCE CODE
- COMMUNITY
- CONTRIBUTING
- CONTINUOUS INTEGRATION

PROJECTS

- COMPILER AND STANDARD LIBRARY
- PACKAGE MANAGER
- CORE LIBRARIES
- REPL AND DEBUGGER
- XCODE PLAYGROUND SUPPORT

Welcome to Swift.org

Swift is now open source!

We are excited by this new chapter in the story of Swift. After Apple unveiled the Swift programming language, it quickly became one of the fastest growing languages in history. Swift makes it easy to write software that is incredibly fast and safe by design. Now that Swift is open source, you can help make the best general purpose programming language available everywhere.

For students, learning Swift has been a great introduction to modern programming concepts and best practices. And because it is now open, their Swift skills will be able to be applied to an even broader range of platforms, from mobile devices to the desktop to the cloud.

Welcome to the Swift community. Together we are working to build a better programming language for everyone.

– The Swift Team

Introduction to Swift

Open Source Swift

Introduction to Swift

Open Source Swift

- Opened in December 2015

Introduction to Swift

Open Source Swift

- Opened in December 2015
- 2.2 was the first version developed in the Open

Introduction to Swift

Open Source Swift

- Opened in December 2015
- 2.2 was the first version developed in the Open
- Open-Source Process - Mailing List & Swift Evolution

Introduction to Swift

Open Source Swift

- Opened in December 2015
- 2.2 was the first version developed in the Open
- Open-Source Process - Mailing List & Swift Evolution
- Working Groups

Introduction to Swift

Open Source Swift

- Opened in December 2015
- 2.2 was the first version developed in the Open
- Open-Source Process - Mailing List & Swift Evolution
- Working Groups
- IBM at WWDC


IBM

Introduction to Swift

Open Source Swift

- Opened in December 2015
- 2.2 was the first version developed in the Open
- Open-Source Process - Mailing List & Swift Evolution
- Working Groups
- IBM at WWDC
- Ported to Linux/Windows/Android/Calculators

A Crash-Course in Swift

A Crash-Course in Swift

Variables

Mutable

```
var foo : String = "something"
foo = "another thing"
```

Immutable

```
let foo : String = "Bar"
```

A Crash-Course in Swift

Type Inference

Mutable

```
var foo = "something"
foo = "another thing"
```

Immutable

```
let foo = "Bar"
```

A Crash-Course in Swift

Methods

```
func foo() -> Bar {
   ...
}
```

A Crash-Course in Swift

Guard Statements

```
func findByName(name: String) -> Person? {
  return nil
}

func pick(name: String?) -> Thing? {
  guard let person = self.findByName(name) else {
    return nil
  }
  ...
}
```

A Crash-Course in Swift

Closures

```
func executeMethod(numberOfTimes: Int, action: () -> ()) {
    for _ in 1...numberOfTimes {
        action()
    }
}

executeMethod(numberOfTimes: 3) { () in
    print("Hello World!")
}
```

Swift Package Manager

Swift Package Manager

- Part of Swift 3

Swift Package Manager

- Part of Swift 3
- Early Days

Swift Package Manager

- Part of Swift 3
- Early Days
- Works based on Git Tag’s - major / minors - e.g. 1.0.0 or 0.1.0

Swift Package Manager

- Part of Swift 3
- Early Days
- Works based on Git Tag’s - major / minors - e.g. 1.0.0 or 0.1.0
- Can Generate Xcode Projects

Swift Package Manager

```
swift package init
```

Swift Package Manager

```
import PackageDescription

let package = Package(
    name: "hello-world",
    dependencies: [
        .Package(url: "https://github.com/vapor/vapor.git",
                 majorVersion: 1,
                 minor: 1)
    ]
)
```

Swift Package Manager

```
swift package fetch
```

Swift Package Manager

```
swift package fetch

swift package update
```

Swift Package Manager

```
swift package fetch

swift package update

swift package generate-xcodeproj
```

Building API's in Swift

Building API's in Swift

- Working with Swift

Building API's in Swift

- Working with Swift
- Web Frameworks

Building API's in Swift

- Working with Swift
- Web Frameworks
- Hello World

Building API's in Swift

- Working with Swift
- Web Frameworks
- Hello World
- Deploying to Heroku

Building API's in Swift

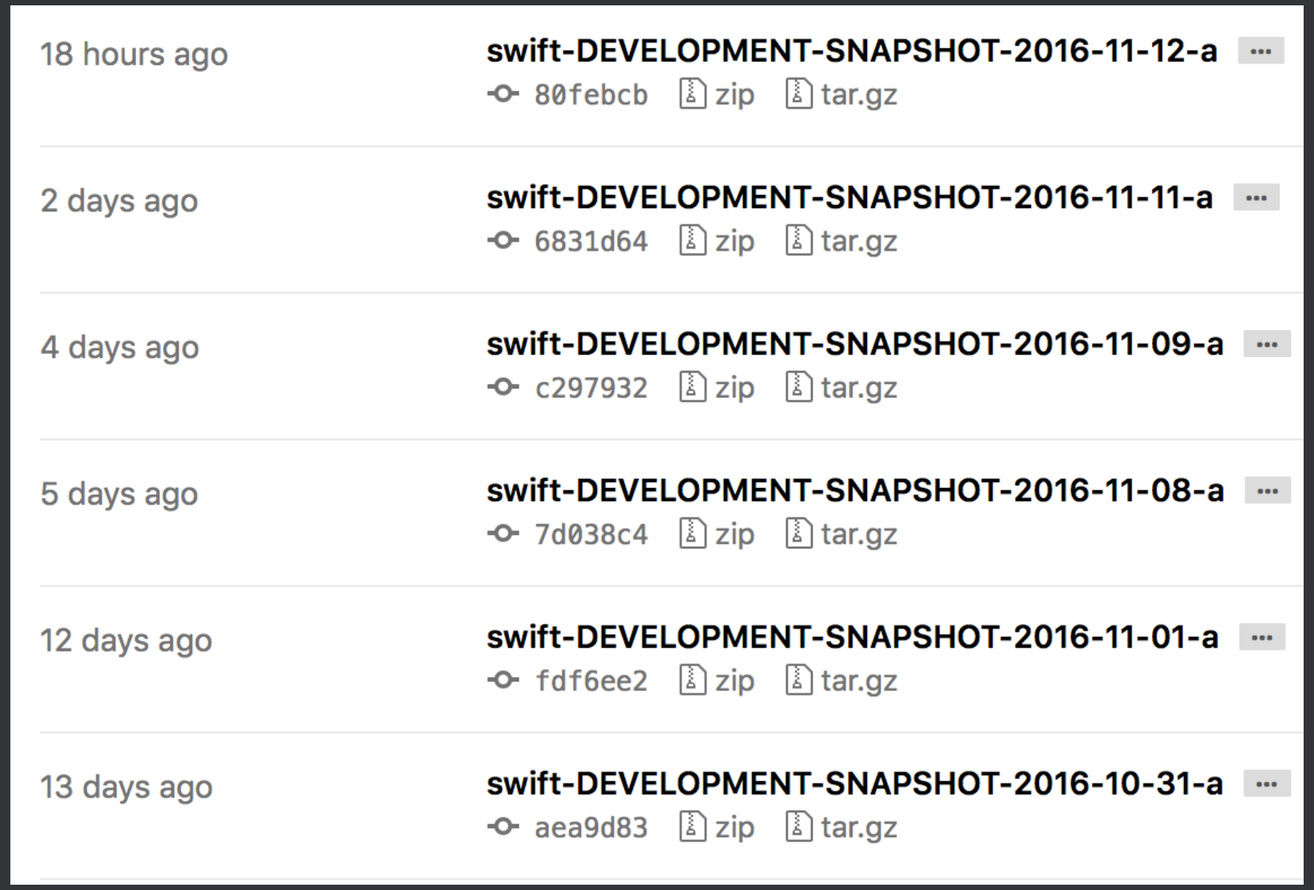
- Working with Swift
- Web Frameworks
- Hello World
- Deploying to Heroku
- Database fun-time

Building API's in Swift

Working with Swift

Building API's in Swift

Working with Swift



Building API's in Swift

Working with Swift: SwiftEnv

Building API's in Swift

Working with Swift: SwiftEnv

```
swiftenv install <version>
```

Building API's in Swift

Working with Swift: SwiftEnv

```
swiftenv install <version>
```

```
swiftenv install 3.0.1
```

Building API's in Swift

Working with Swift: SwiftEnv

```
swiftenv install <version>
```

```
swiftenv global <version>
```

```
swiftenv local <version>
```

Building API's in Swift

Major Web Frameworks

Building API's in Swift

Vapor

Building API's in Swift

Vapor

- Droplet

Building API's in Swift

Vapor

- Droplet
- Routing

Building API's in Swift

Vapor

- Droplet
- Routing
- Handlers

Building API's in Swift

Vapor

- Droplet
- Routing
- Handlers
- Middleware

Building API's in Swift

Vapor: Toolkit

Building API's in Swift

Vapor: Toolkit

- New Project: `vapor new <project>`
- Build / Run: `vapor build` / `vapor run`
- Clean: `vapor clean`

Building API's in Swift

Vapor: Toolkit

- New Project: `vapor new <project>`
- Build / Run: `vapor build` / `vapor run`
- Clean: `vapor clean`
- Xcode: `vapor xcode`
- Docker: `vapor docker (init|build|run|enter)`
- Heroku: `vapor heroku (init|push)`

Building API's in Swift

Vapor: Hello World

```
import Vapor

let drop = Droplet()

drop.get { _ in
    return "Hello World"
}

drop.run()
```

Building API's in Swift

Vapor: Returning JSON

```
import JSON
import Vapor

let drop = Droplet()

drop.get("/hi") { _ in
    return try JSON(node: [
        "hello": "world"
    ])
}

drop.run()
```

Building API's in Swift

Vapor: Returning a Templated HTML page

```
import JSON
import Vapor

let drop = Droplet()

drop.get("/city") { request in
    guard let city = request.query?["city"]?.string else {
        return Response(status: .badRequest)
    }
    return try drop.view.make("towns", [
        "city": city
    ])
}

drop.run()
```

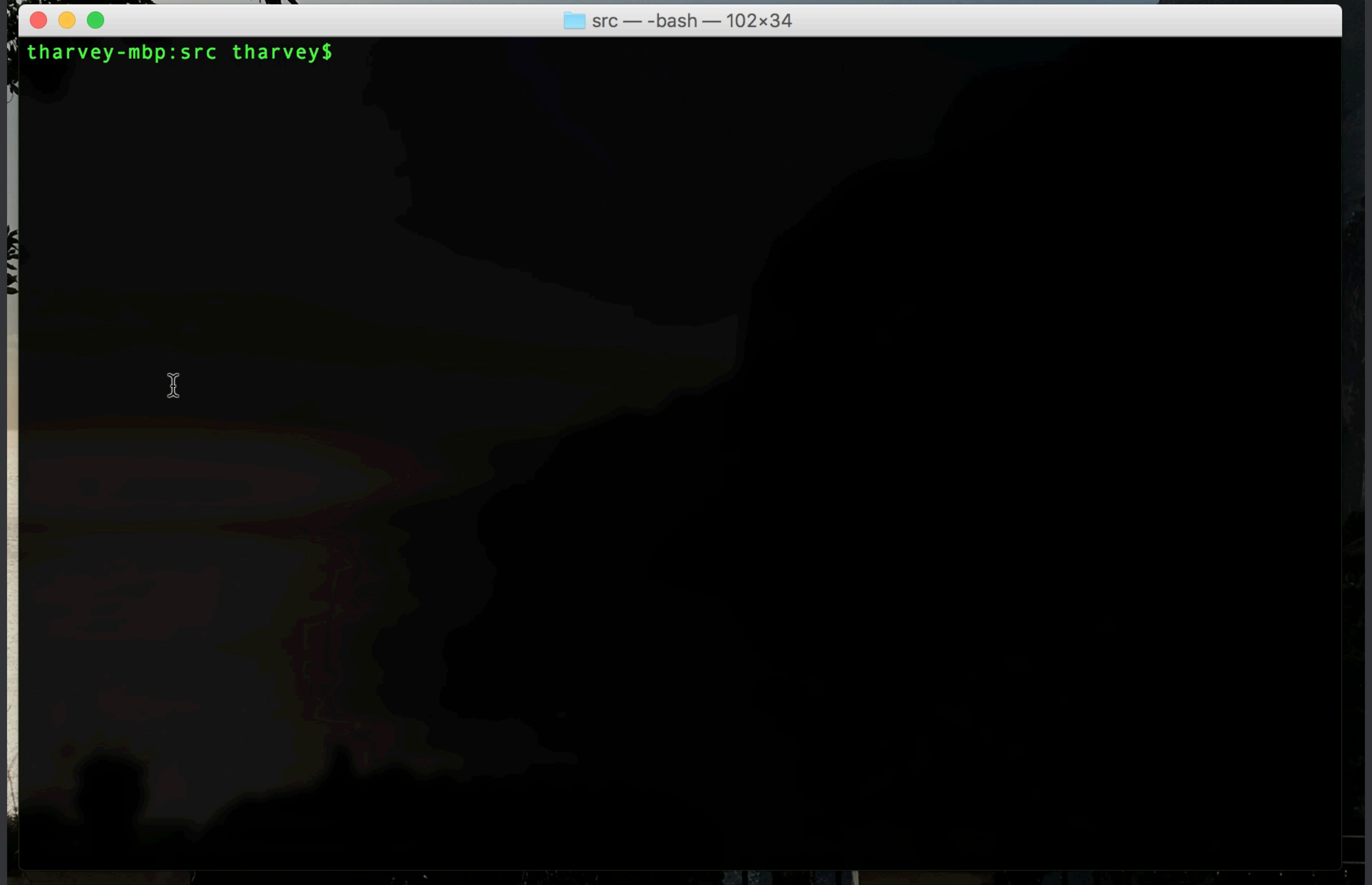
Building API's in Swift

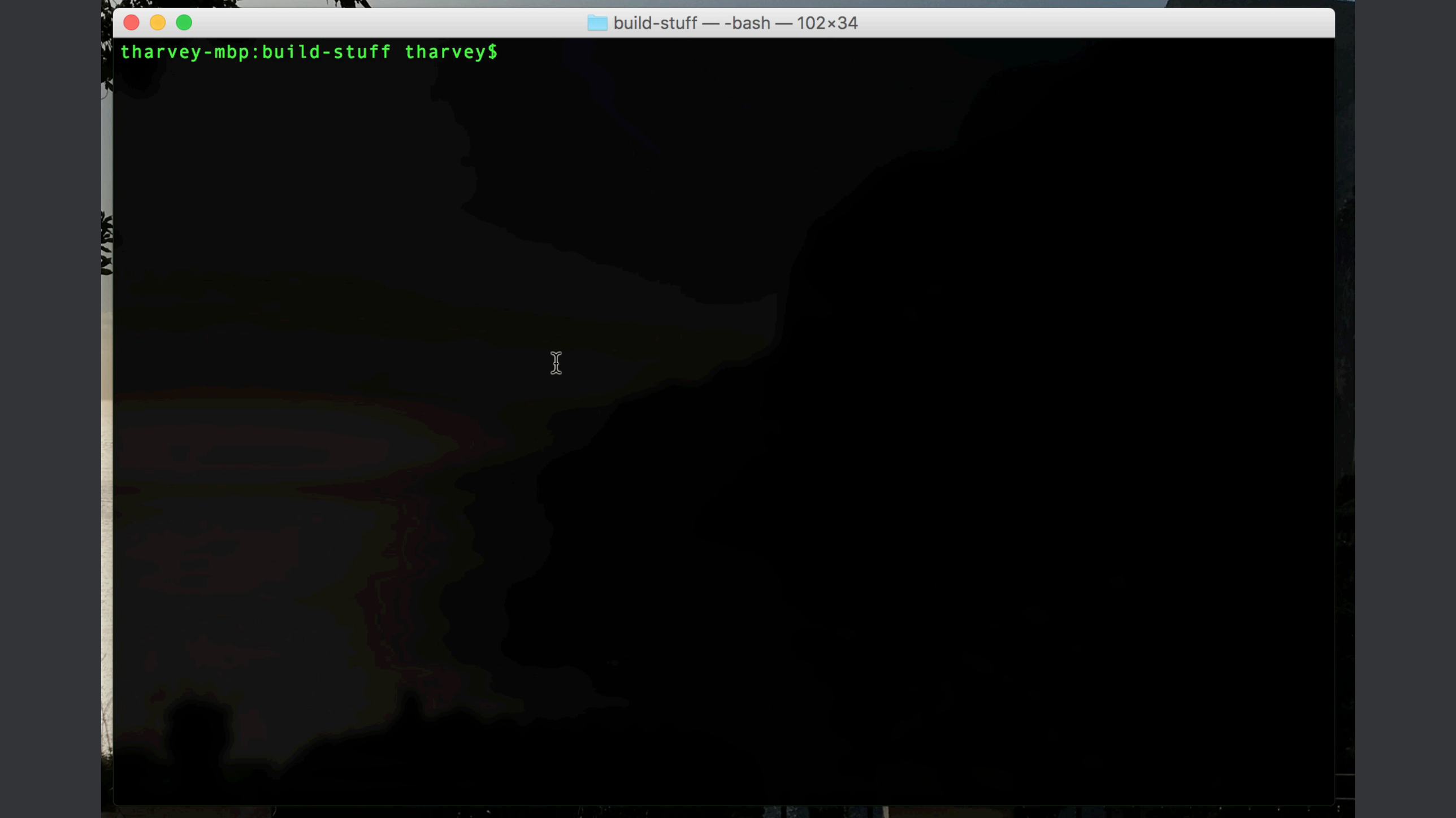
Vapor: Templating Language (Leaf)

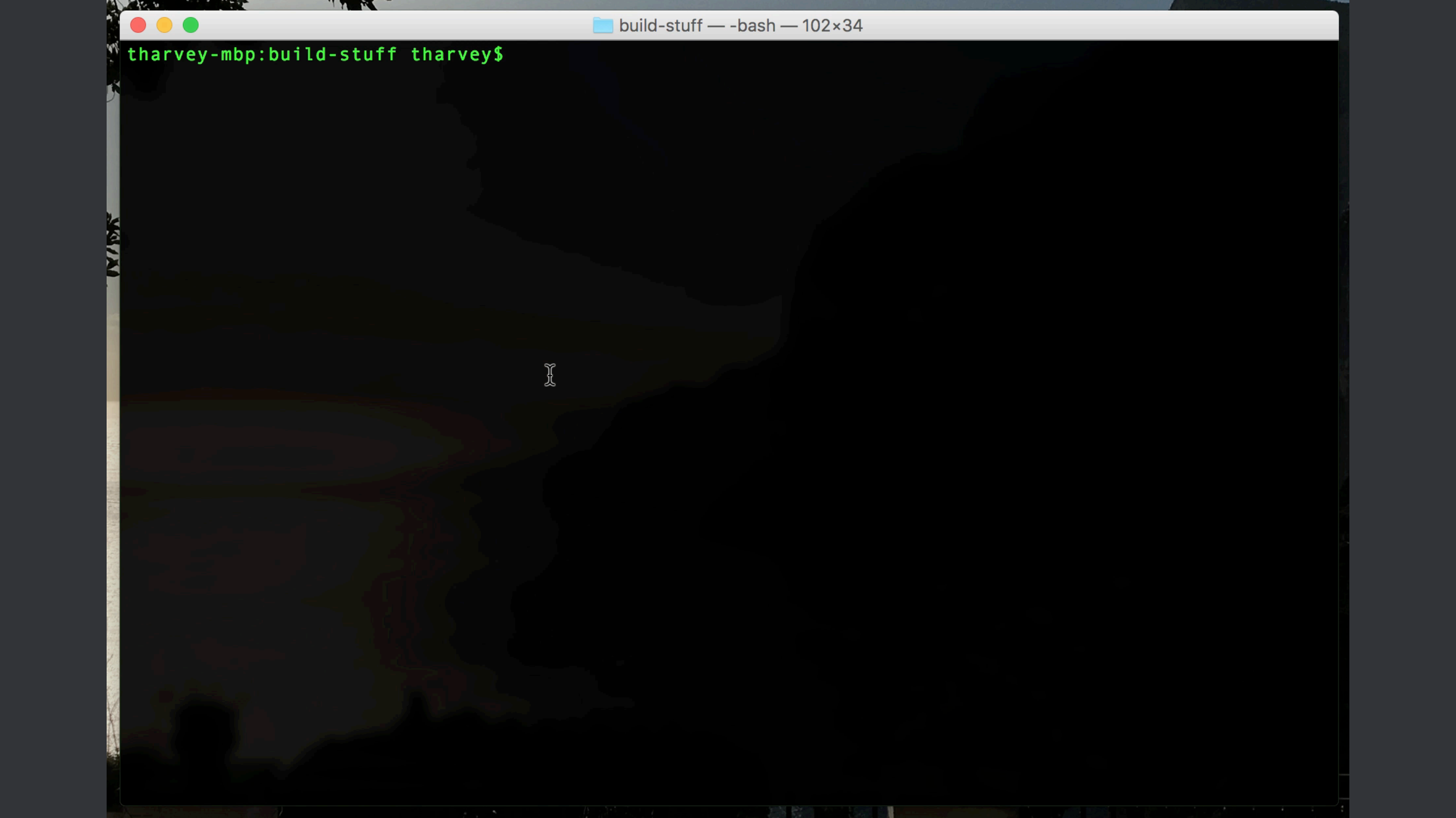
```
<h1>The City is!</h1>
#{city}
```

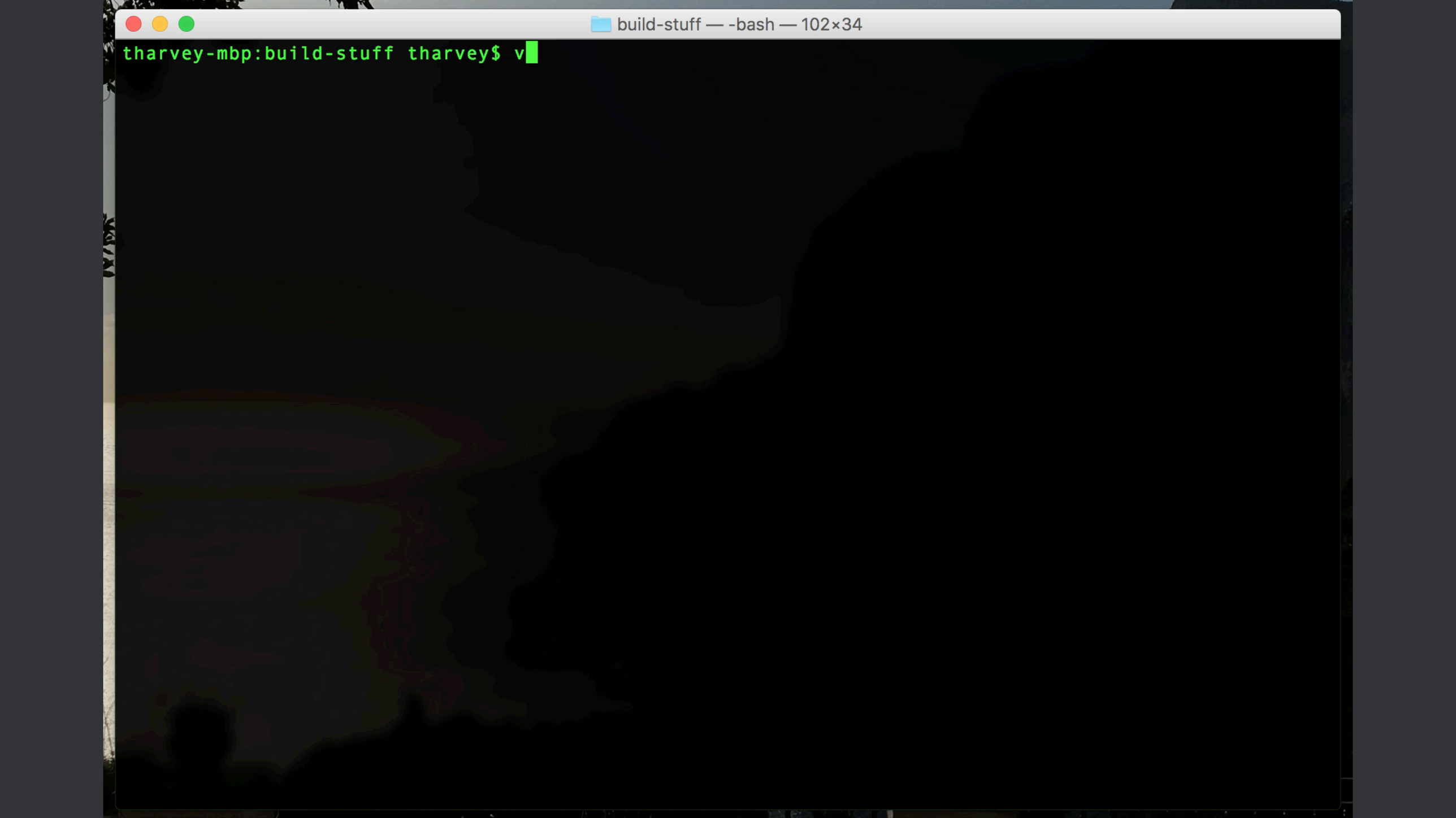
Building API's in Swift

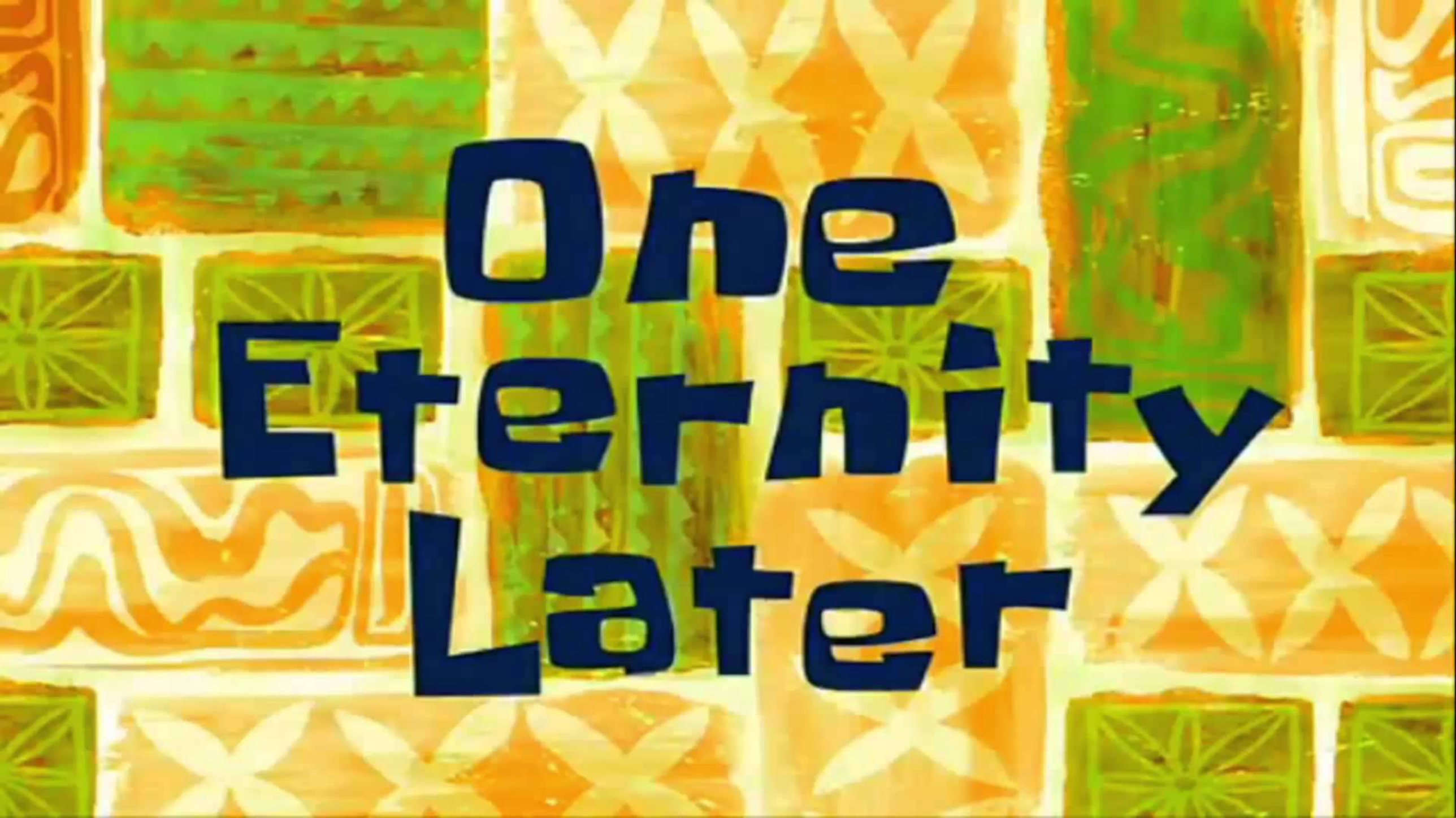
Demo

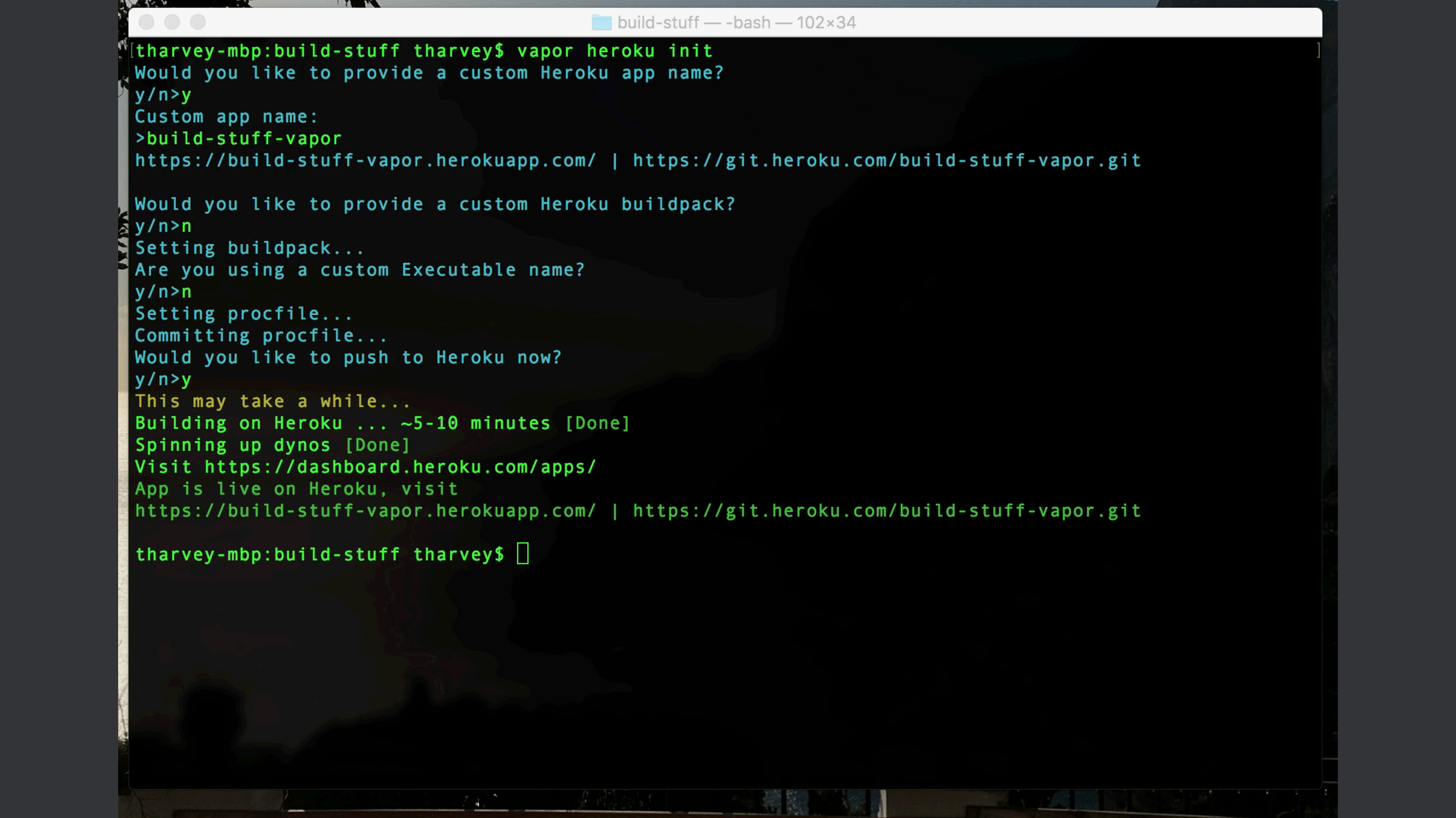

src — -bash — 102×34
tharvey-mbp:src tharvey$

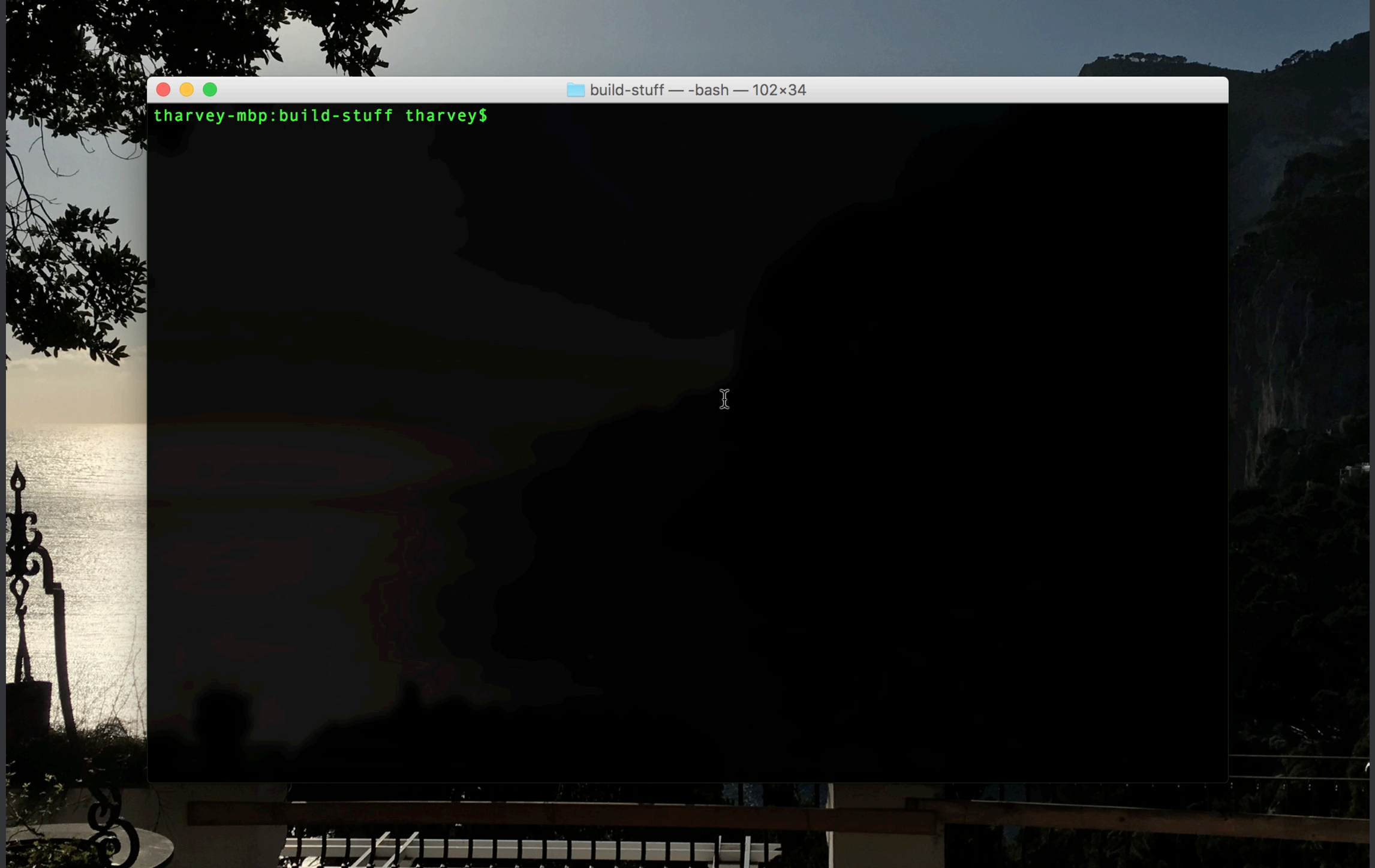

build-stuff — -bash — 102×34
tharvey-mbp:build-stuff tharvey$


build-stuff — -bash — 102×34
tharvey-mbp:build-stuff tharvey$


build-stuff — -bash — 102×34
tharvey-mbp:build-stuff tharvey$ v


One
Eternity
Later


build-stuff — -bash — 102×34
tharvey-mbp:build-stuff tharvey$ vapor heroku init
Would you like to provide a custom Heroku app name?
y/n>y
Custom app name:
>build-stuff-vapor
https://build-stuff-vapor.herokuapp.com/ | https://git.heroku.com/build-stuff-vapor.git
Would you like to provide a custom Heroku buildpack?
y/n>n
Setting buildpack...
Are you using a custom Executable name?
y/n>n
Setting procfile...
Committing procfile...
Would you like to push to Heroku now?
y/n>y
This may take a while...
Building on Heroku ... ~5-10 minutes [Done]
Spinning up dynos [Done]
Visit https://dashboard.heroku.com/apps/
App is live on Heroku, visit
https://build-stuff-vapor.herokuapp.com/ | https://git.heroku.com/build-stuff-vapor.git
tharvey-mbp:build-stuff tharvey$


build-stuff — -bash — 102×34
tharvey-mbp:build-stuff tharvey$

Building API's in Swift

Vapor: Querying PostgreSQL

```
import PostgreSQL

let postgreSQL = PostgreSQL.Database(
    dbname: "example",
    user: "exampleapiuser",
    password: "S0m3TH!nG_Se3rE7"
)
let statement = "SELECT * FROM towns"
let results = try postgreSQL.execute(statement)
```

Building API's in Swift

Vapor: Querying MySQL

```
import PostgreSQL

let database = try MySQL.Database(
  host: "localhost",
  user: "username",
  password: "securepassword",
  database: "towns")
let result = try database.execute("SELECT name, country FROM towns")
```

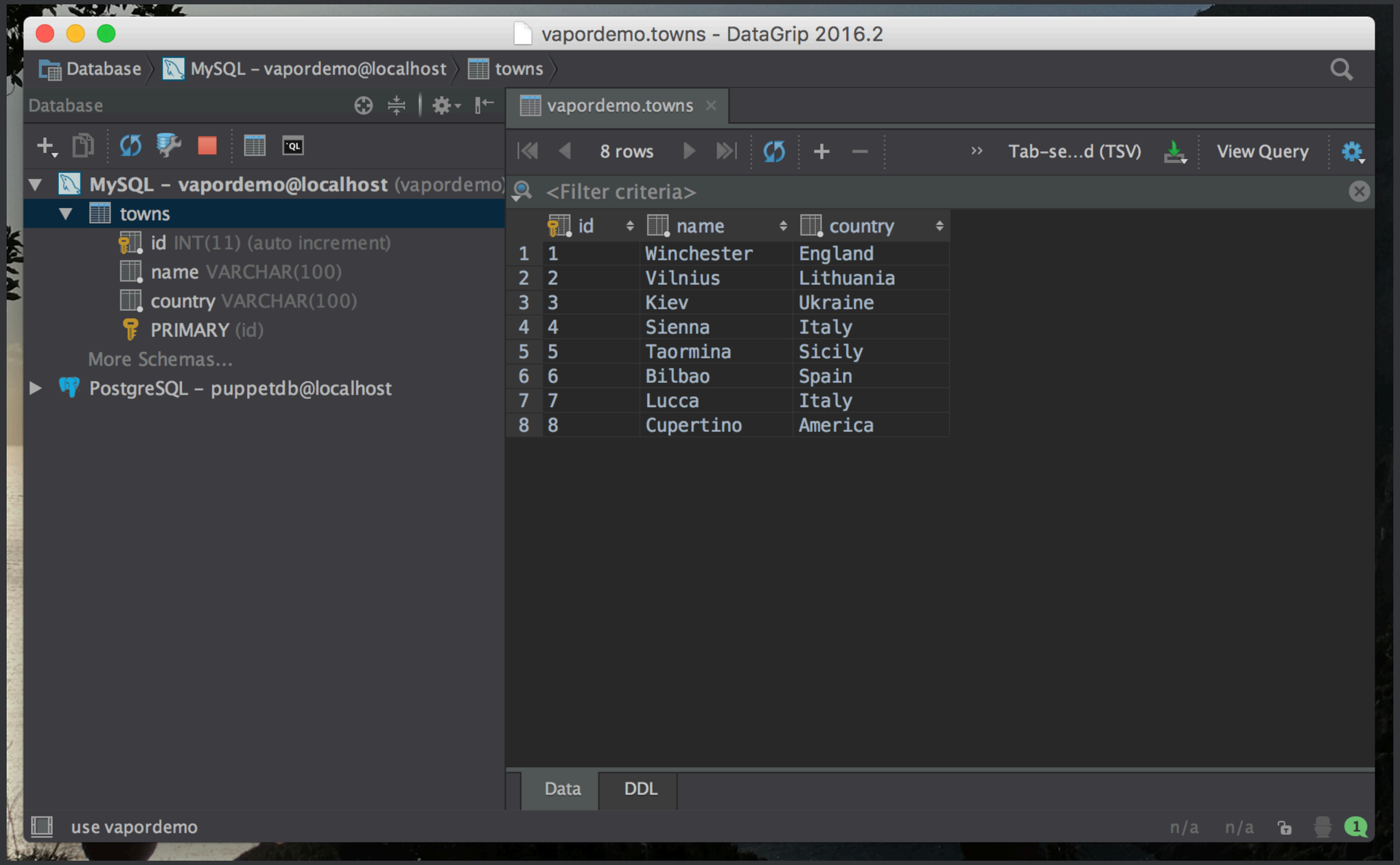
Building API's in Swift

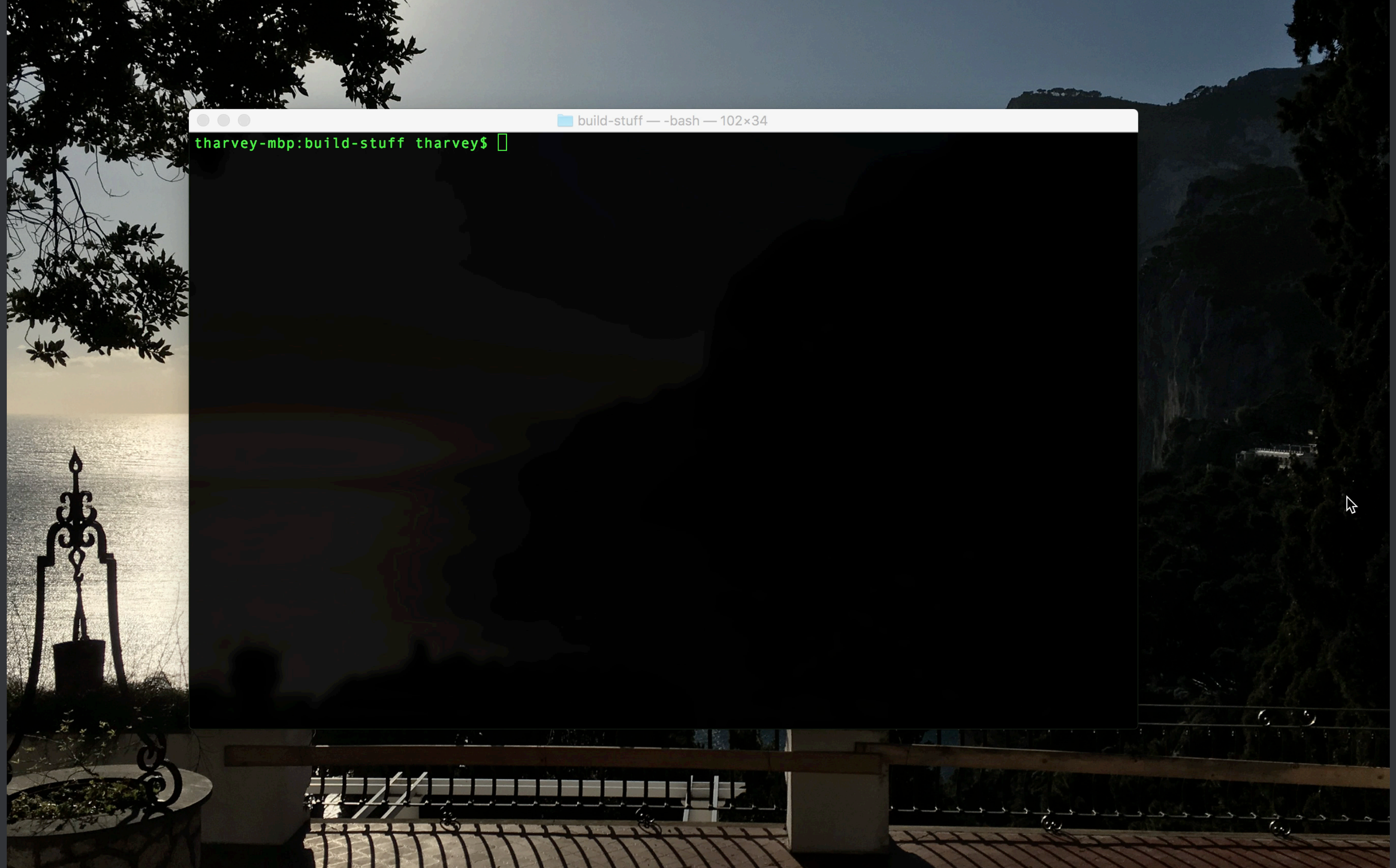
Vapor: Connecting to Redis

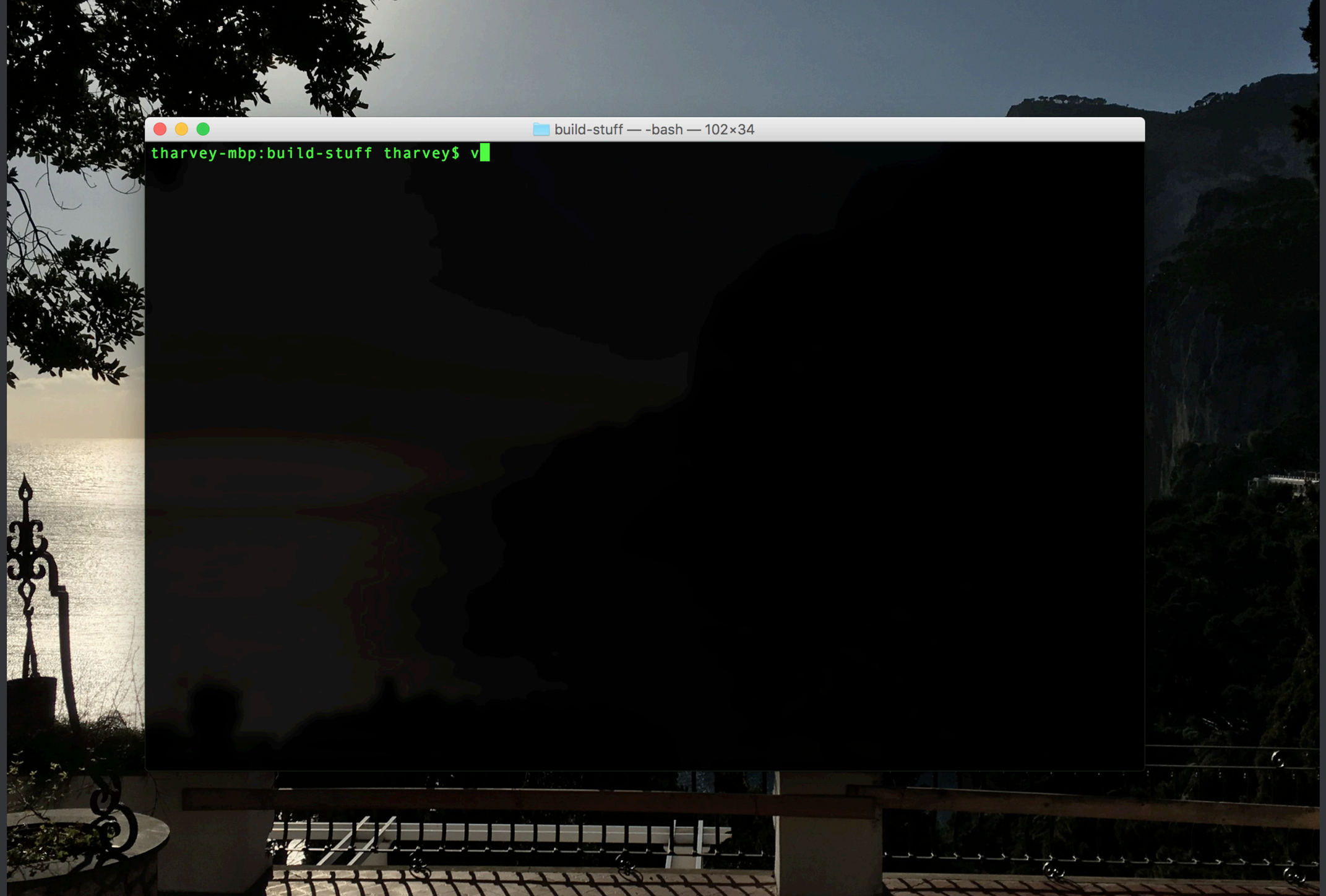
```
let config = RedbirdConfig(address: "127.0.0.1",
                           port: 6379,
                           password: "foopass")
let client = try Redbird(config: config)
let params = ["mykey", "hello_redis"]
let response = try client.command("SET", params: params)
```

Building API's in Swift

Demo


vapordemo.towns - DataGrip 2016.2
Database
MySQL – vapordemo@localhost
towns
Database
vapordemo.towns
8 rows
Tab-se...d (TSV)
View Query
<Filter criteria>
MySQL – vapordemo@localhost (vapordemo)
towns
id INT(11) (auto increment)
name VARCHAR(100)
country VARCHAR(100)
PRIMARY (id)
More Schemas...
PostgreSQL – puppetdb@localhost
id
name
country
1 1 Winchester England
2 2 Vilnius Lithuania
3 3 Kiev Ukraine
4 4 Sienna Italy
5 5 Taormina Sicily
6 6 Bilbao Spain
7 7 Lucca Italy
8 8 Cupertino America
Data
DDL
use vapordemo
n/a
n/a


build-stuff — -bash — 102×34
tharvey-mbp:build-stuff tharvey$


build-stuff — -bash — 102×34
tharvey-mbp:build-stuff tharvey$ v

Building API's in Swift

Demo: Summary

Building API's in Swift

Demo: Summary

- Vapor Toolkit -> Hello World

Building API's in Swift

Demo: Summary

- Vapor Toolkit -> Hello World
- Endpoint to Return a list of Towns

Building API's in Swift

Demo: Summary

- Vapor Toolkit -> Hello World
- Endpoint to Return a list of Towns
- Deploying that to Heroku

Building API's in Swift

Demo: Summary

- Vapor Toolkit -> Hello World
- Endpoint to Return a list of Towns
- Deploying that to Heroku
- Pulling the list of towns from MySQL

Building API's in Swift

Demo: Summary

- Vapor Toolkit -> Hello World
- Endpoint to Return a list of Towns
- Deploying that to Heroku
- Pulling the list of towns from MySQL
- Inserting values into MySQL

Deploying Swift

Deploying Swift

- Heroku

Deploying Swift

- Heroku
- Docker

Deploying Swift

- Heroku
- Docker
- nginx + Vapor

Deploying Swift

- Heroku
- Docker
- nginx + Vapor
- Vapor via a Supervisor script

Deploying Swift on Heroku

Deploying Swift on Heroku

```
vapor heroku init
```

Deploying Swift on Heroku

```
vapor heroku init
```

```
vapor heroku push
```

Deploying Swift on Heroku

```
vapor heroku init
```

```
vapor heroku push
```

https://build-stuff-vapor.herokuapp.com

Deploying Swift with Docker

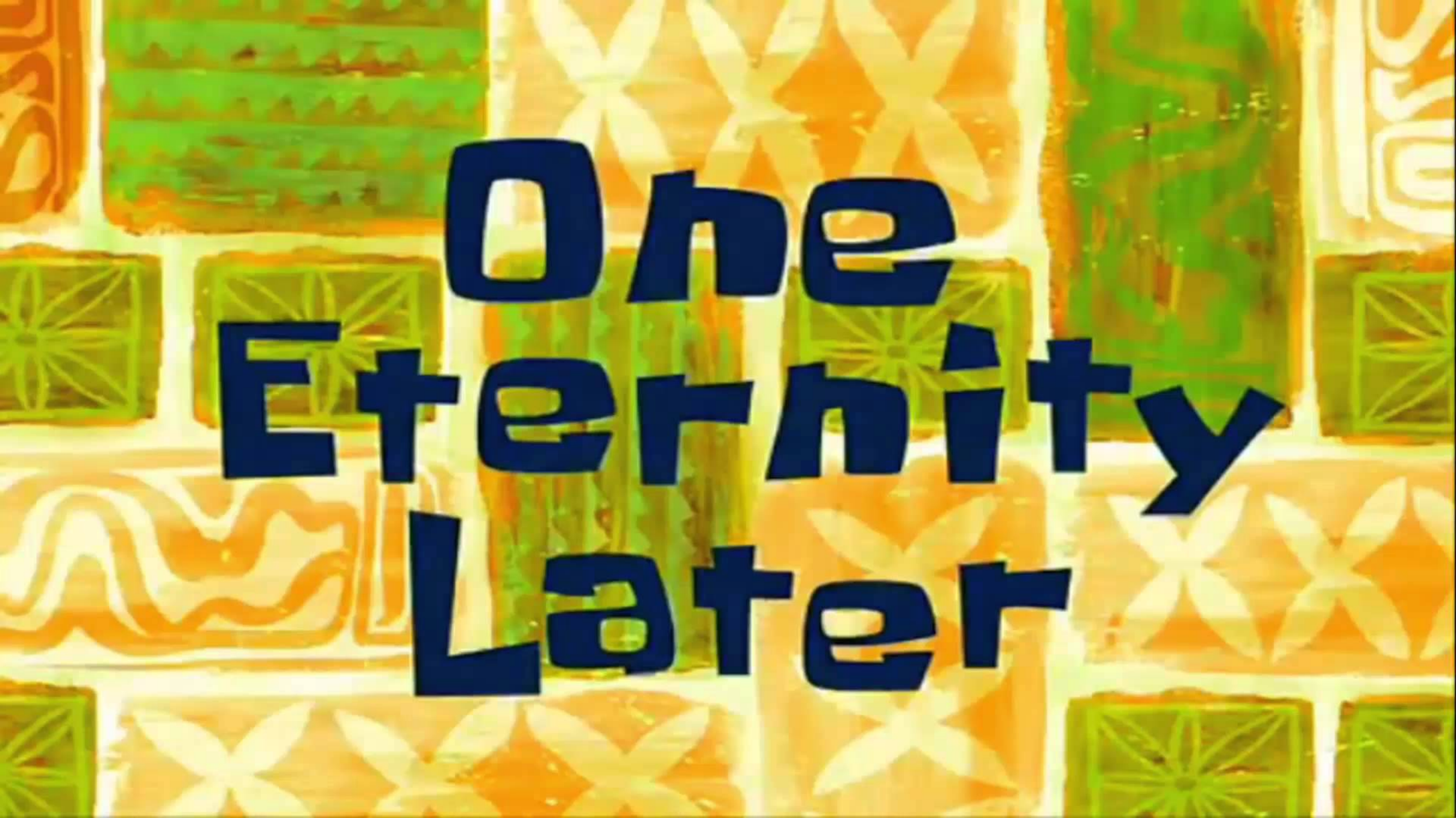
Deploying Swift with Docker

```
vapor docker init
```

Deploying Swift with Docker

```
vapor docker init
```

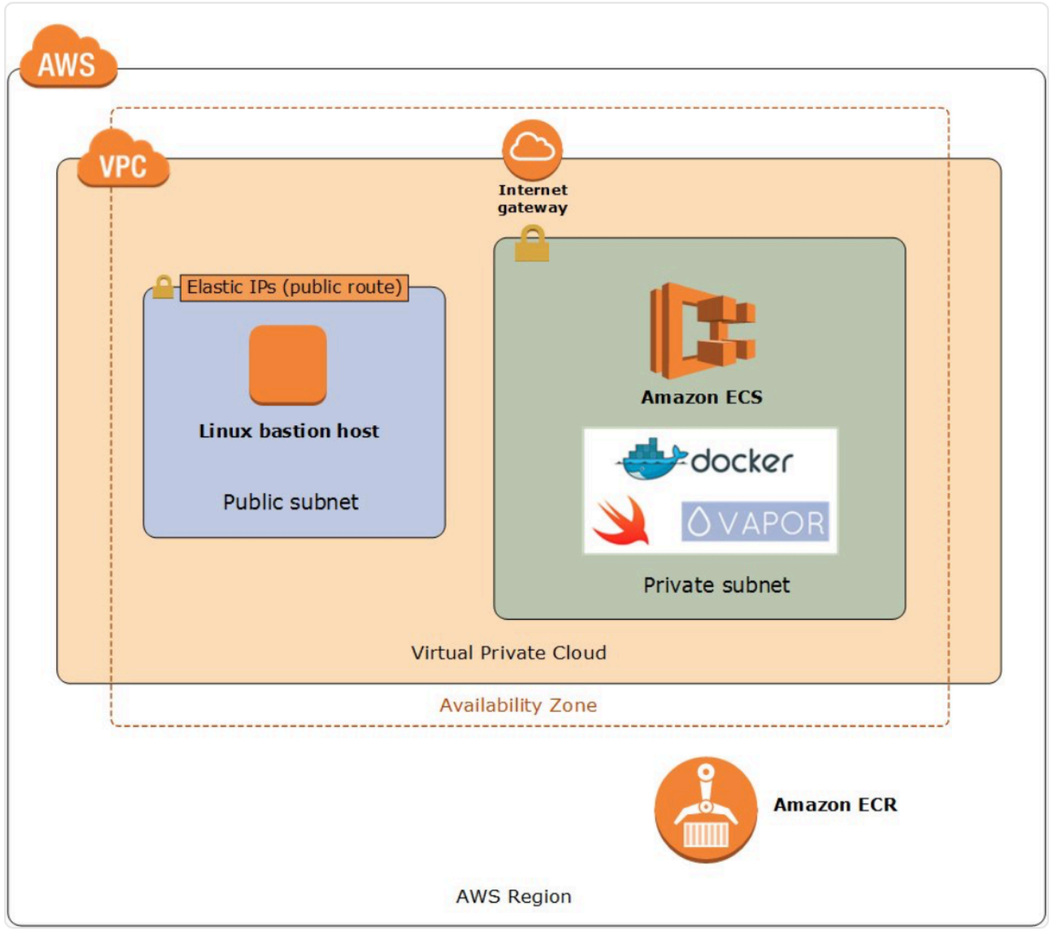
```
vapor docker build
```


One
Eternity
Later

Amazon Web Services
@awscloud

Follow

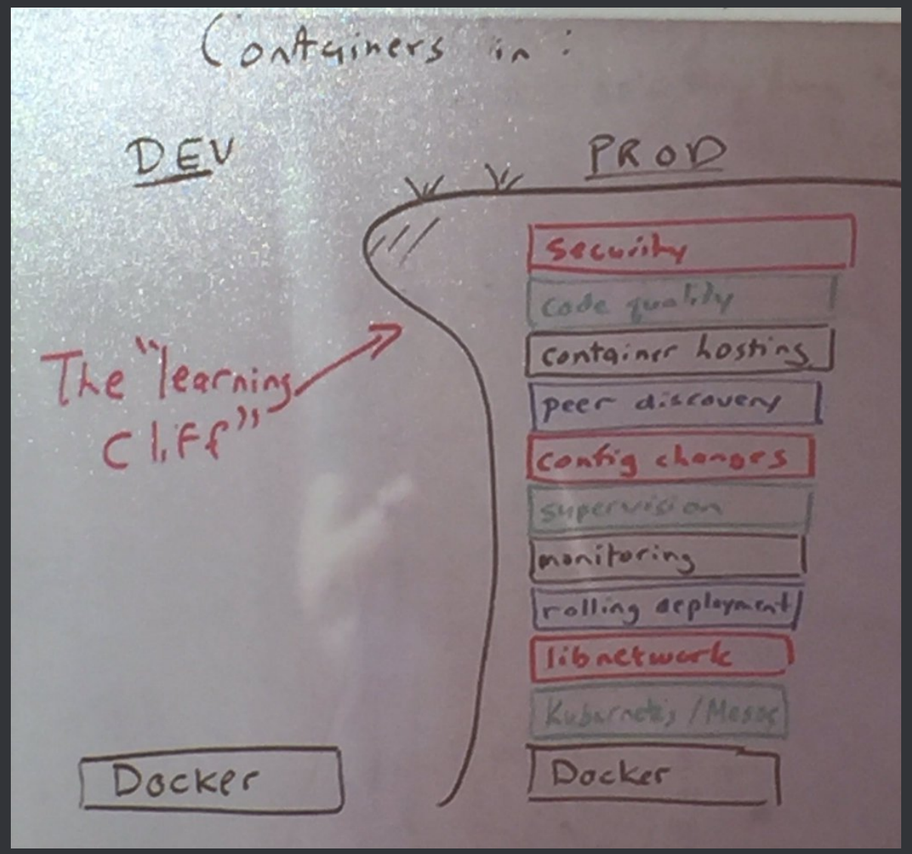
Deploy a Swift web app on AWS in 10 minutes with new Quick Start! amzn.to/2exg2h6 #CloudComputing



RETWEETS 69 LIKES 82

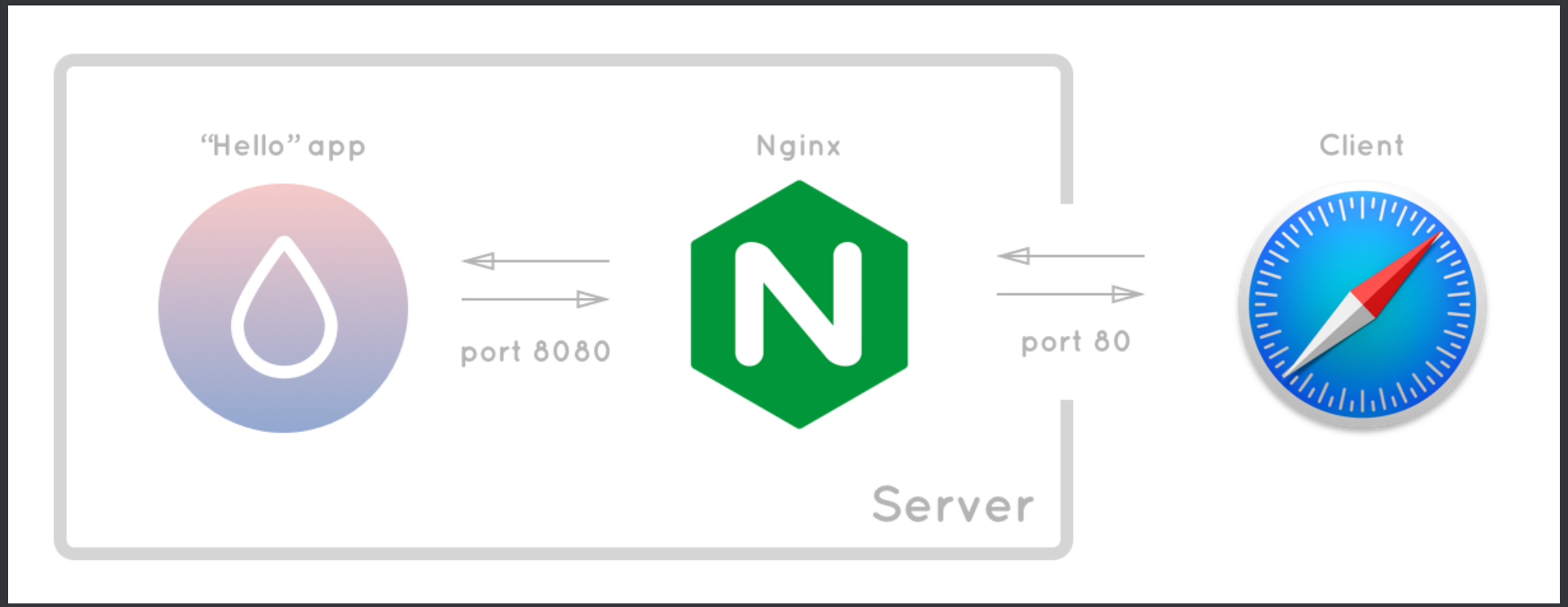
2:16 pm - 1 Nov 2016

Deploying Swift with Docker



https://twitter.com/mfdii/status/697532387240996864

Deploying Swift with nginx + Vapor



https://vapor.github.io/documentation/deploy/nginx.html

Deploying Swift

- Fundamentally no different to any other language

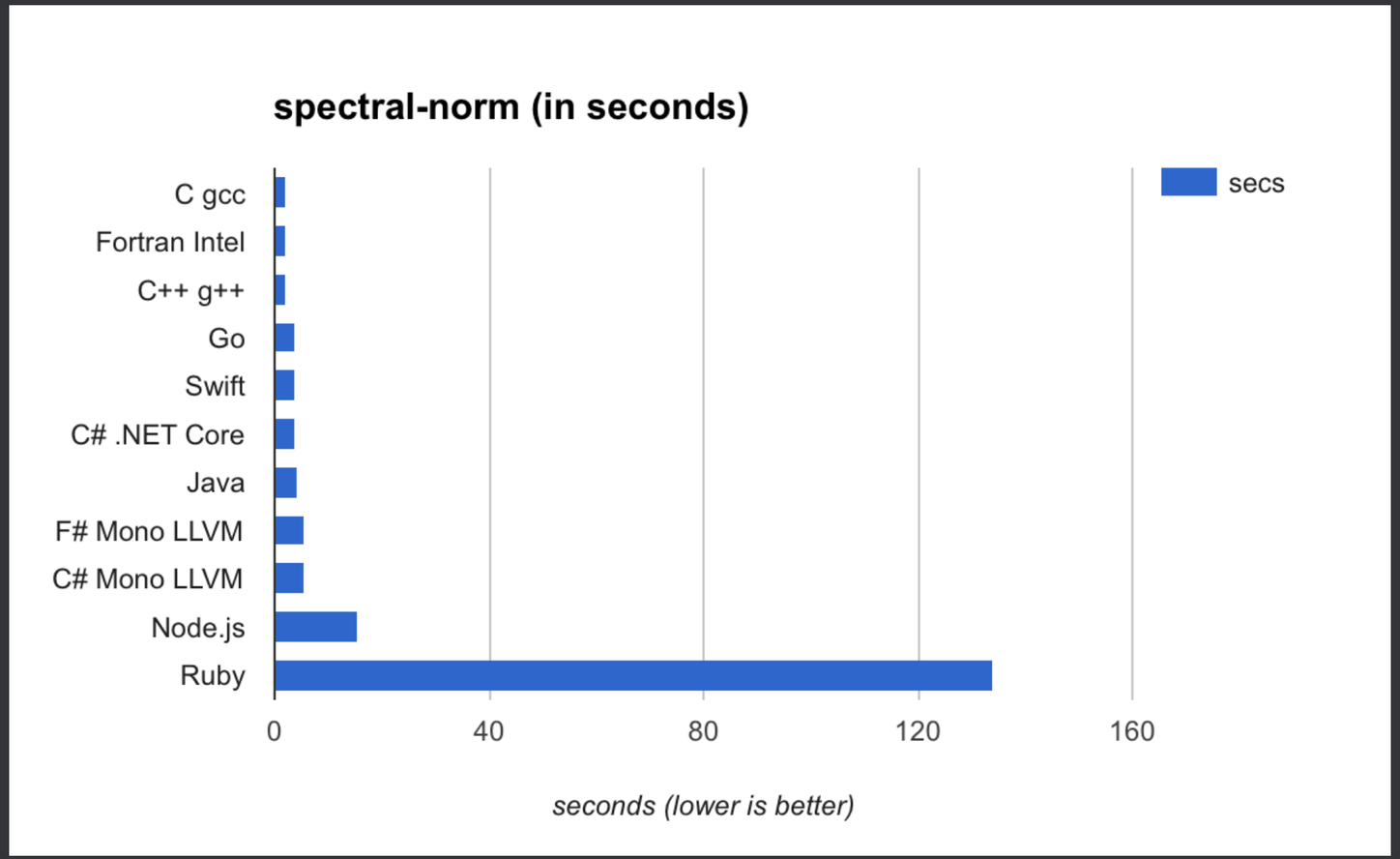
State of Play

State of Play

Performance

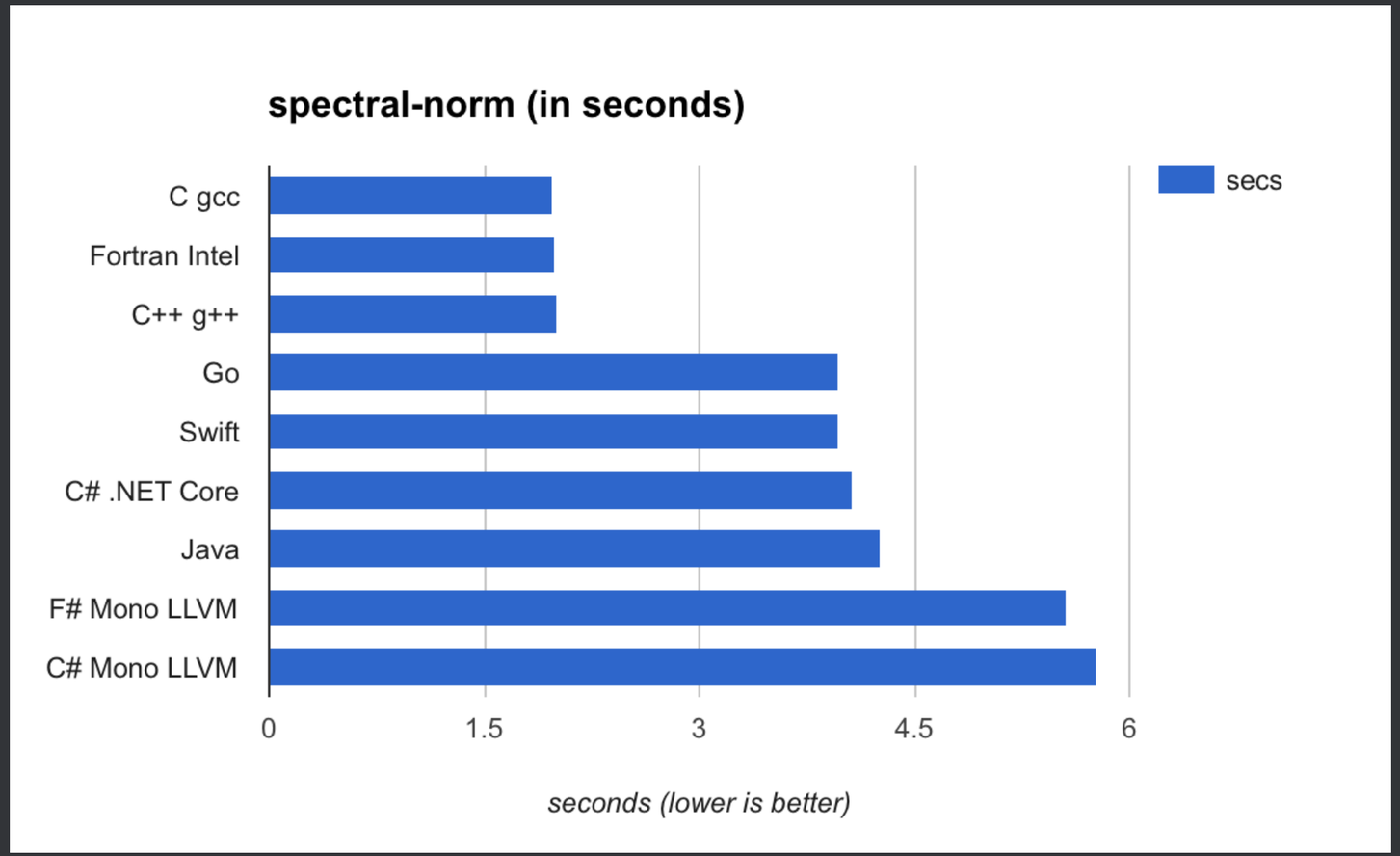
State of Play

Swift is pretty Swift



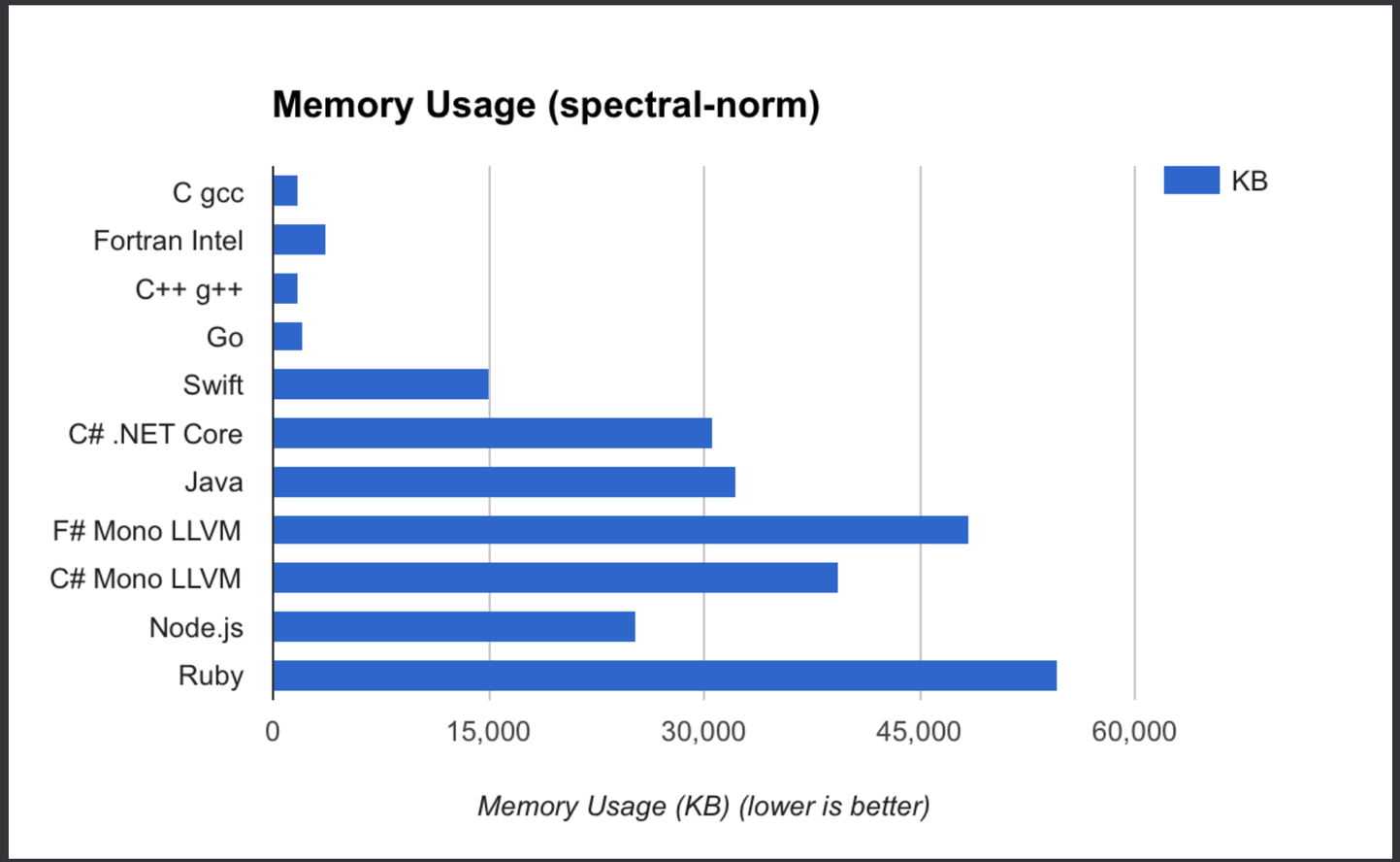
State of Play

Swift is pretty Swift



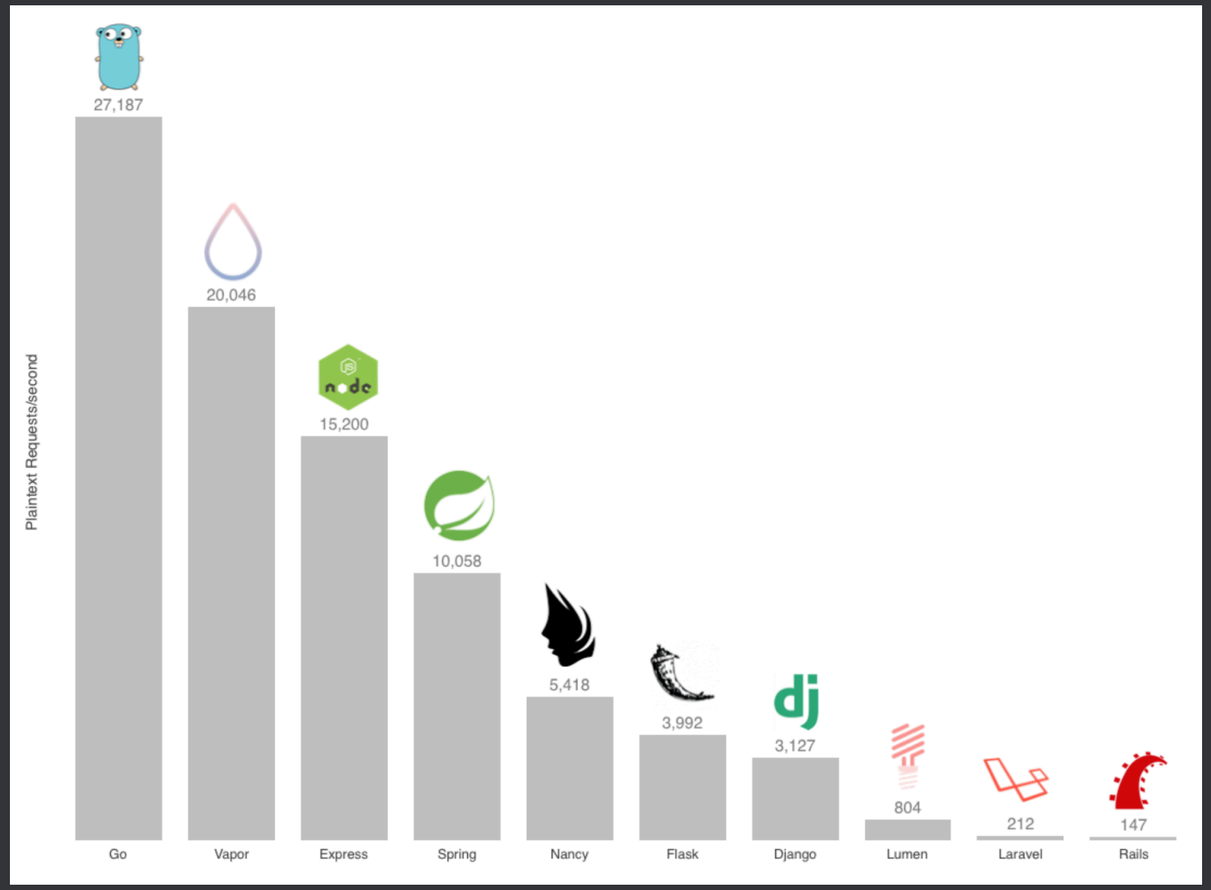
State of Play

Low Memory Footprint



State of Play

Requests per Second



State of Play

The awesome parts

- Swift is a young language

State of Play

The awesome parts

- Swift is a young language
  - **You** can influence it via Swift Evolution (apple/swift-evolution)
  - Swift API's Working Group on **swift.org** - just launched

State of Play

The awesome parts

- Swift is a young language
  - **You** can influence it via Swift Evolution (apple/swift-evolution)
  - Swift API's Working Group on **swift.org** - just launched
- Active community adding/suggesting features

State of Play

The awesome parts

- Swift is a young language
  - **You** can influence it via Swift Evolution (apple/swift-evolution)
  - Swift API's Working Group on **swift.org** - just launched
- Active community adding/suggesting features
- IDE Support is 👍 - XCode / AppCode/CLion / Atom / Vim etc

State of Play

The *interesting* parts

- Swift is a young language
  - Source Compatibility between versions
  - SwiftEnv makes it better

Abizer Nasir
@abizern

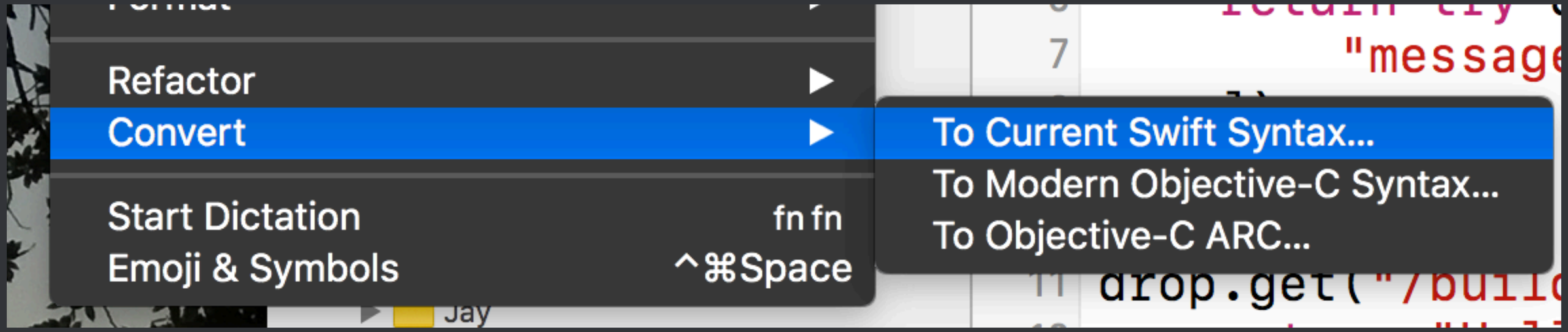
Follow

Swift 2.3 to 3.0 migration after lunch. Wish me luck.

LIKES
6

4:37 am - 26 Sep 2016

6


Refactor
Convert
Start Dictation
fn fn
Emoji & Symbols
^⌘Space
To Current Swift Syntax...
To Modern Objective-C Syntax...
To Objective-C ARC...
Jay
7
"message
drop.get("/

State of Play

The *interesting* parts

- Swift is a young language
  - Source Compatibility between versions
  - SwiftEnv makes it better

State of Play

The *interesting* parts

- Swift is a young language
- The lack of reflection is painful (JSON)
  - Using the ORM gets around some of this, but not really

State of Play

The *interesting* parts

- Swift is a young language
- The lack of reflection is painful (JSON)
- Early days for Tooling
  - Native Build Server Integration
  - Libraries for some things, not others (e.g. Postgres/MySQL/Redis/RabbitMQ)

Things to Watch

Things to Watch

- Swift 4 will fix the pain points (e.g. Reflection)

Things to Watch

- Swift 4 will fix the pain points (e.g. Reflection)
- API Working Group on **swift.org**

Things to Watch

- Swift 4 will fix the pain points (e.g. Reflection)
- API Working Group on **swift.org**
- Build Tooling & Libraries

Things to Watch

- Swift 4 will fix the pain points (e.g. Reflection)
- API Working Group on **swift.org**
- Build Tooling & Libraries
- Swift as a Scripting Language

Thanks

Enjoy the Conference

@tombuildsstuff

tombuildsstuff/swift-on-the-server

Thanks

Links

- Vapor - **https://vapor.codes**
- Swift Evolution - **apple/swift-evolution**
- Swift API's Working Group - **swift.org**
- Vapor/Swift on AWS - **http://bit.ly/2eX1JBS**
- End-to-End app dev with Swift - **http://bit.ly/2fdr4rX**
- Intro to Swift on the Server - **http://bit.ly/2gdT03c**