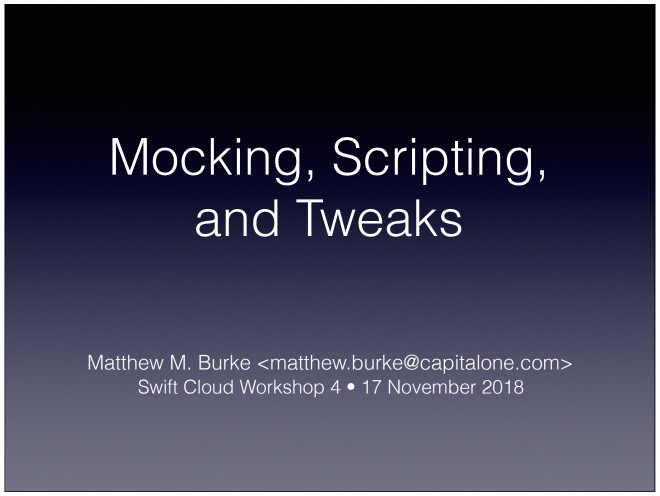
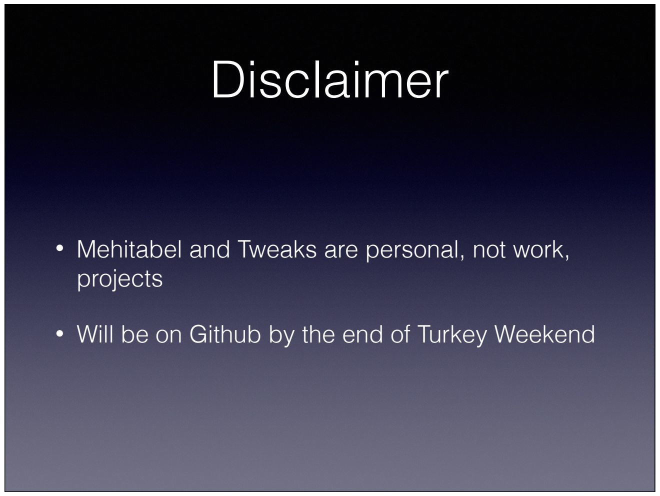


Mocking, Scripting, and Tweaks

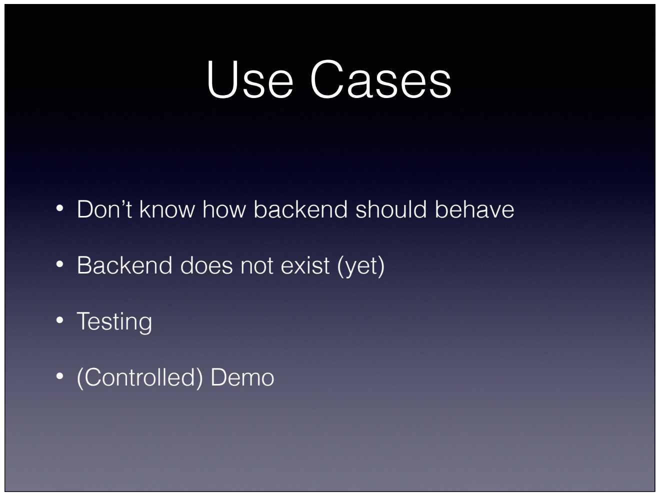
Matthew M. Burke <matthew.burke@capitalone.com>
Swift Cloud Workshop 4 • 17 November 2018



Disclaimer

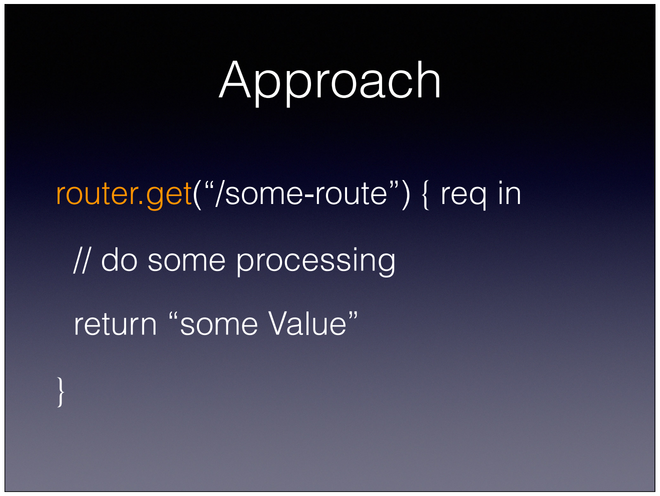
- Mehitabel and Tweaks are personal, not work, projects
- Will be on Github by the end of Turkey Weekend

Although I have given a sales pitch for both at work.
Tweaks is being pursued, but with a fairly different approach.


Use Cases

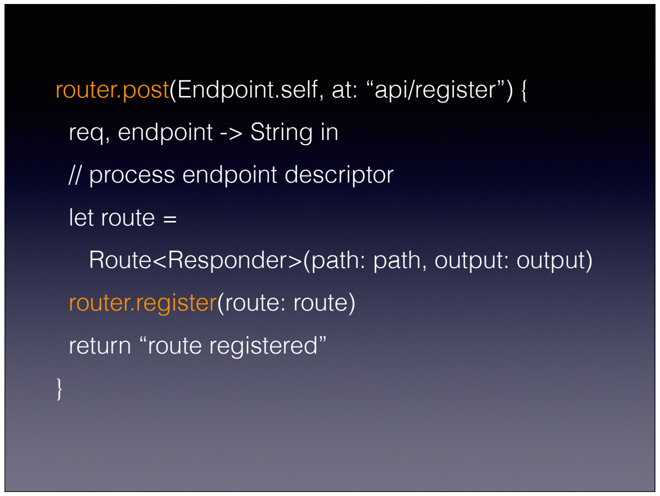
- Don’t know how backend should behave
- Backend does not exist (yet)
- Testing
- (Controlled) Demo

Note this gives you an approach to testing with, potentially, a cleaner separation of project and test infrastructure
Can give you more realistic networking
Also an extremely light-weight, and dev-controlled QA environment


Approach

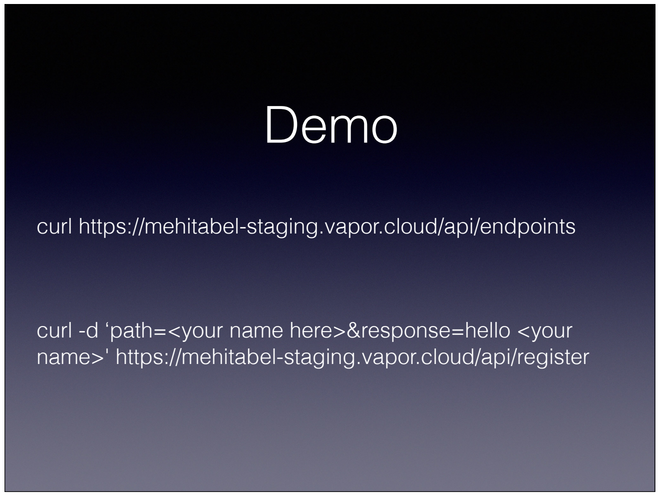
```
router.get(“/some-route”) { req in
  // do some processing
  return “some Value”
}
```

Most web frameworks (not just Swift) involve some sort of DSL-ish way of specifying URL path-handler pairs.
Key point is that we invoke a method to create the route-handler pair. So why not do that at “run-time”?



```
router.post(Endpoint.self, at: "api/register") {
  req, endpoint -> String in
  // process endpoint descriptor
  let route =
     Route<Responder>(path: path, output: output)
  router.register(route: route)
  return "route registered"
}
```

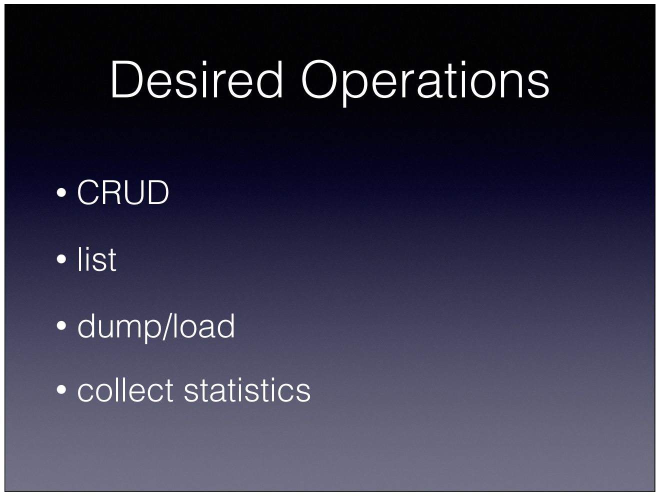
In Vapor, Endpoint implements Content (similar to Codable) and we automatically get conversion from either form body or JSON payload and others (protobuf …)



Demo

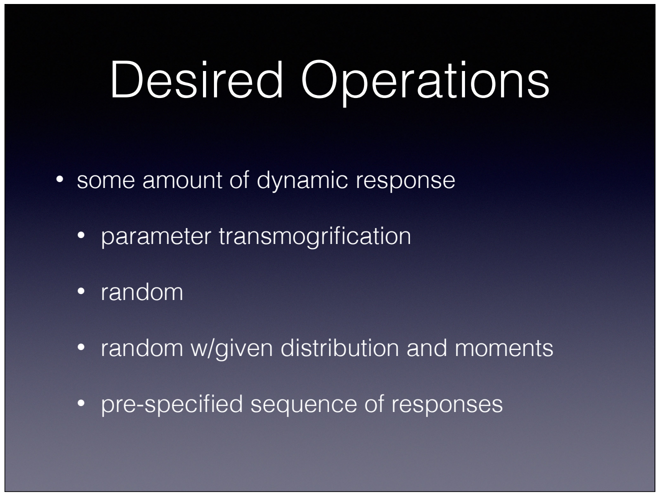
```
curl https://mehitabel-staging.vapor.cloud/api/endpoints

curl -d 'path=<your name here>&response=hello <your
name>' https://mehitabel-staging.vapor.cloud/api/register
```



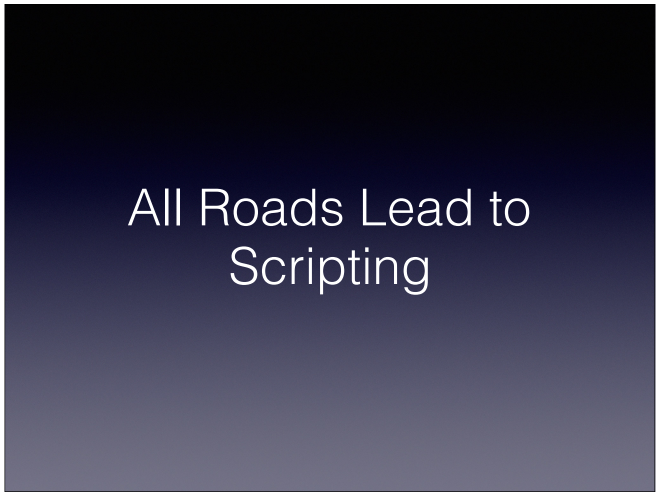
Desired Operations

- CRUD
- list
- dump/load
- collect statistics

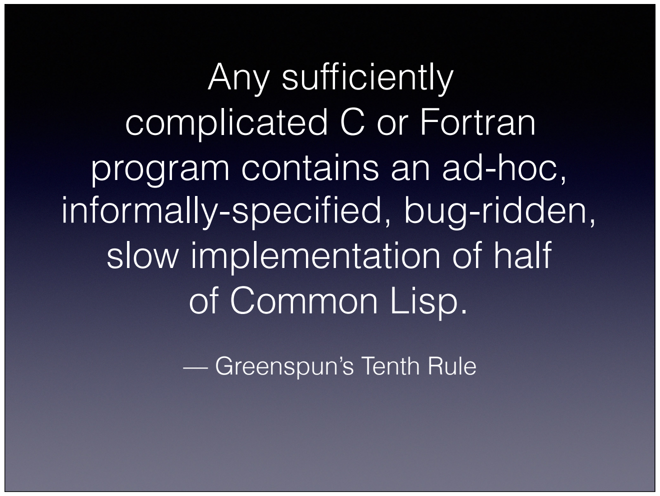


Desired Operations

- some amount of dynamic response
  - parameter transmogrification
  - random
  - random w/given distribution and moments
  - pre-specified sequence of responses



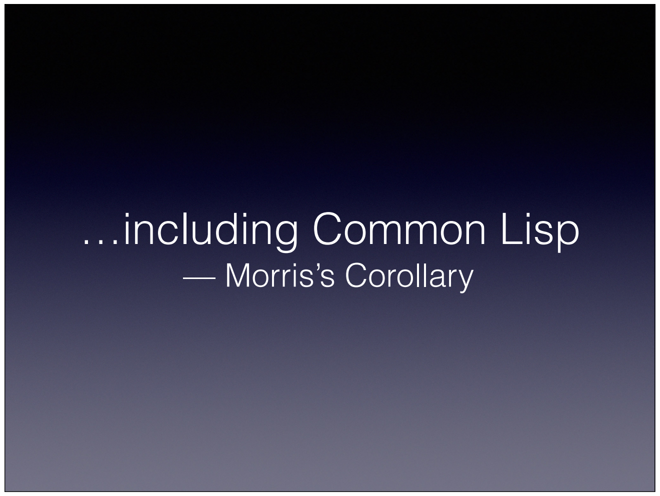
All Roads Lead to Scripting

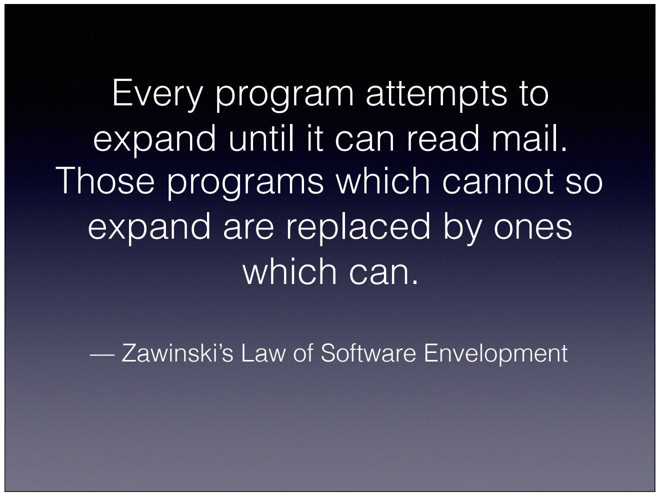


Any sufficiently complicated C or Fortran program contains an ad-hoc, informally-specified, bug-ridden, slow implementation of half of Common Lisp.

— Greenspun's Tenth Rule

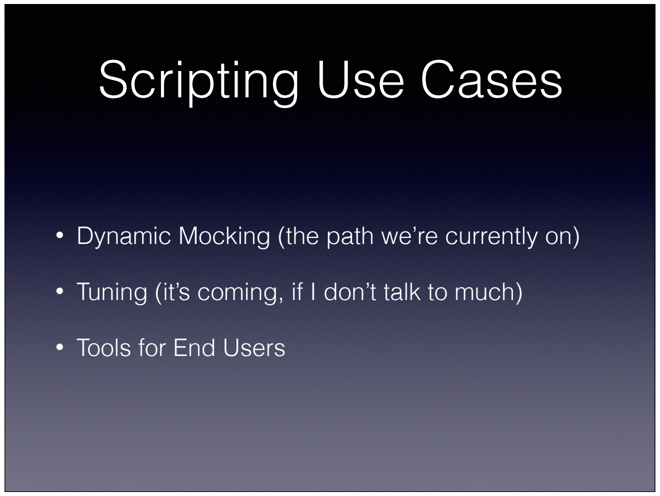
you may have heard of this, but have you heard of the corollary


…including Common Lisp
— Morris's Corollary

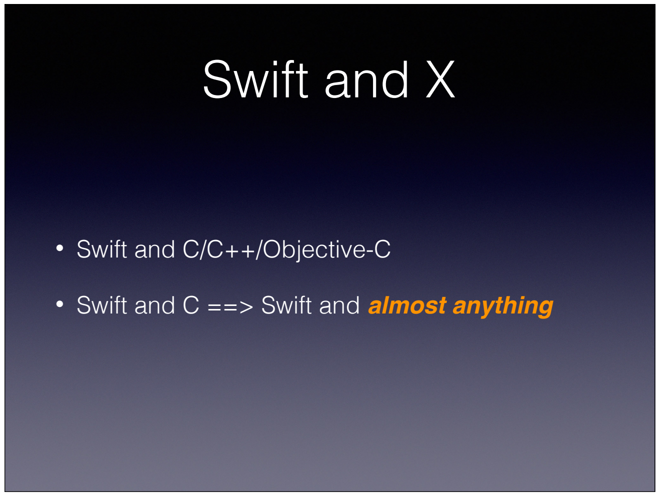

Every program attempts to expand until it can read mail. Those programs which cannot so expand are replaced by ones which can.

— Zawinski's Law of Software Envelopment

and now we're really starting to go off topic…

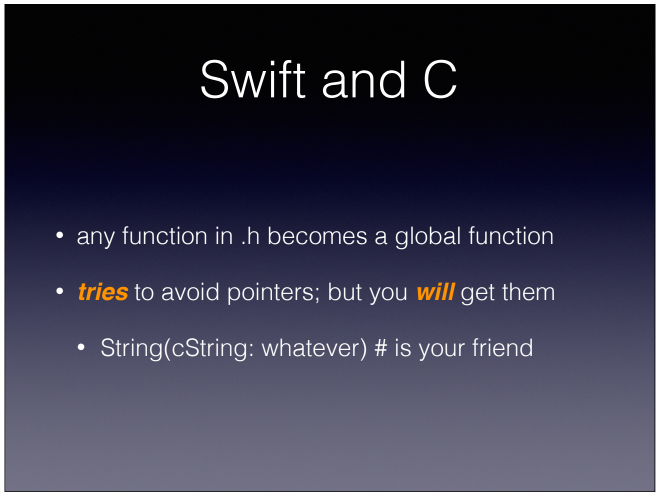

Scripting Use Cases

- Dynamic Mocking (the path we're currently on)
- Tuning (it's coming, if I don't talk to much)
- Tools for End Users

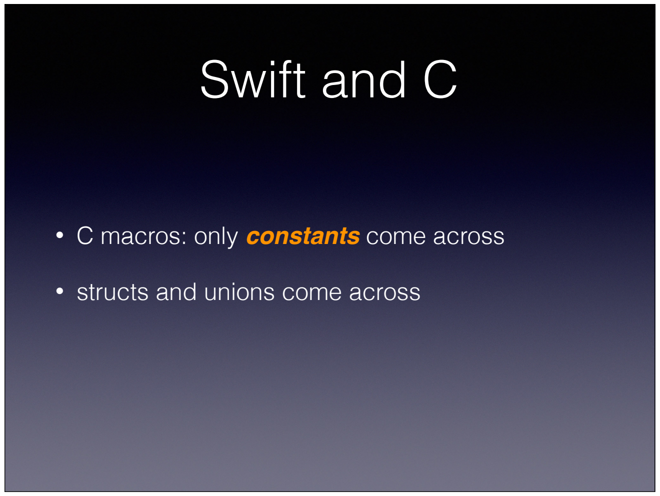

Swift and X

- Swift and C/C++/Objective-C
- Swift and C ==> Swift and ***almost anything***

as Tom said, easy bridging into C

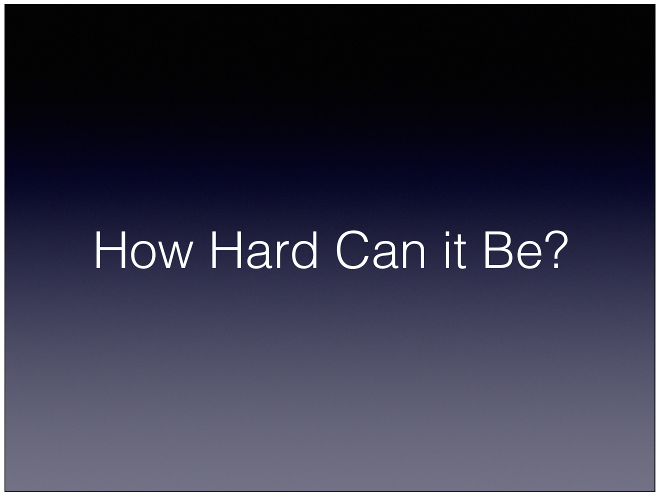

Swift and C

- any function in .h becomes a global function
- ***tries*** to avoid pointers; but you ***will*** get them
  - String(cString: whatever) # is your friend

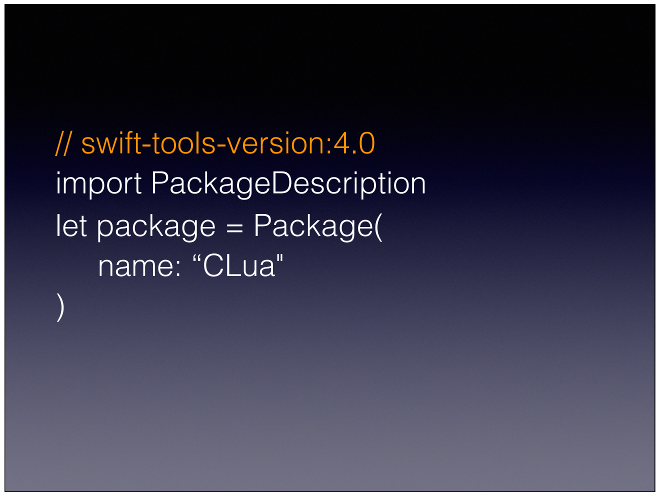

Swift and C

- C macros: only ***constants*** come across
- structs and unions come across

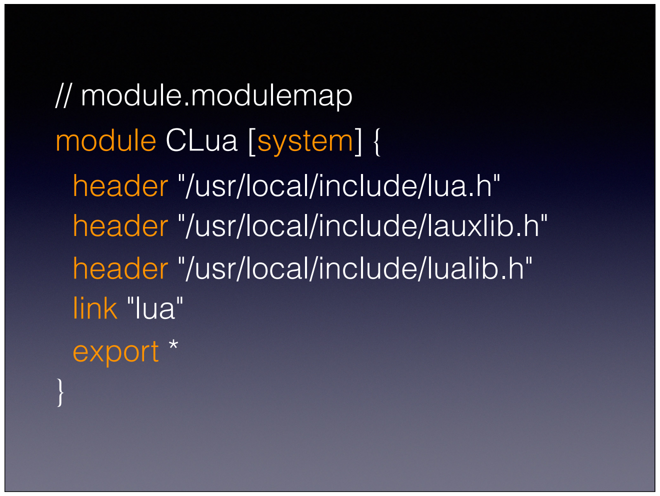
Careful with unions. Explain.



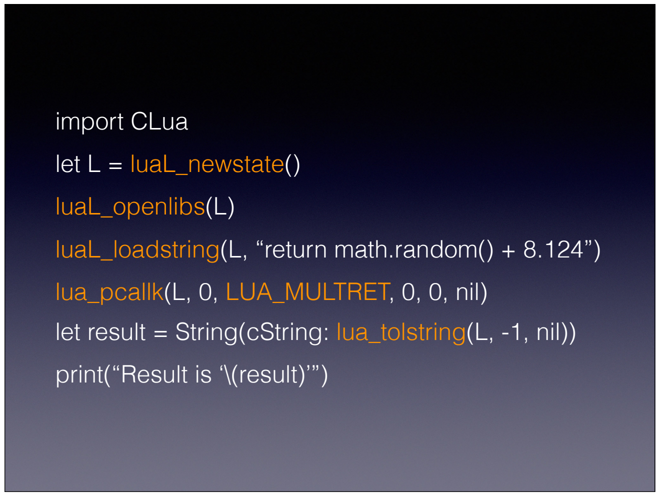
How Hard Can it Be?



```
// swift-tools-version:4.0
import PackageDescription
let package = Package(
    name: “CLua"
)
```

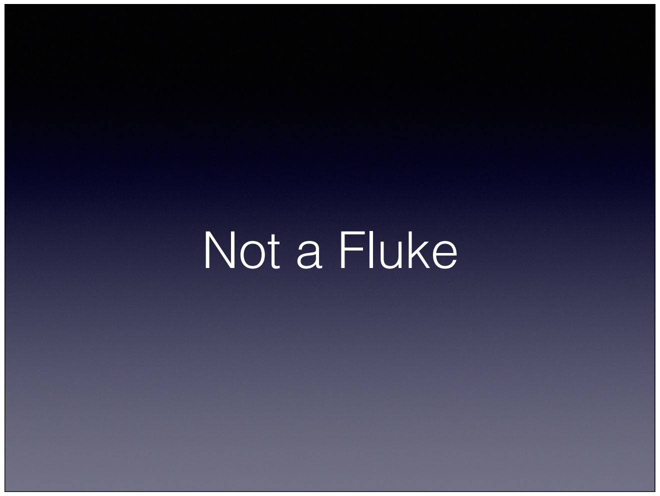


```
// module.modulemap
module CLua [system] {
  header "/usr/local/include/lua.h"
  header "/usr/local/include/lauxlib.h"
  header "/usr/local/include/lualib.h"
  link "lua"
  export *
}
```

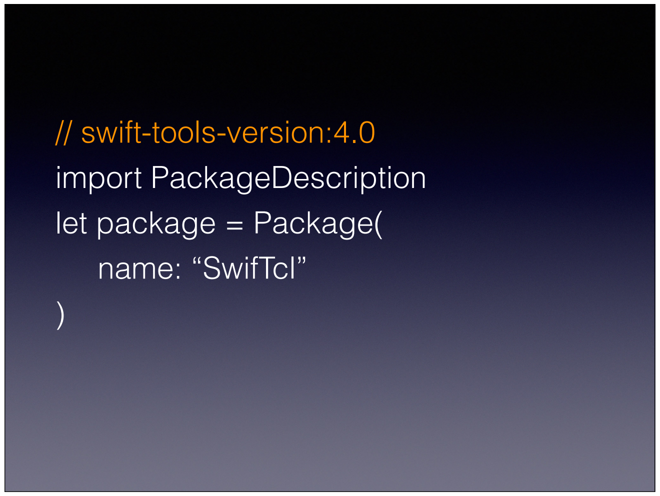


```
import CLua
let L = luaL_newstate()
luaL_openlibs(L)
luaL_loadstring(L, “return math.random() + 8.124”)
lua_pcallk(L, 0, LUA_MULTRET, 0, 0, nil)
let result = String(cString: lua_tolstring(L, -1, nil))
print(“Result is ‘\(result)’”)
```

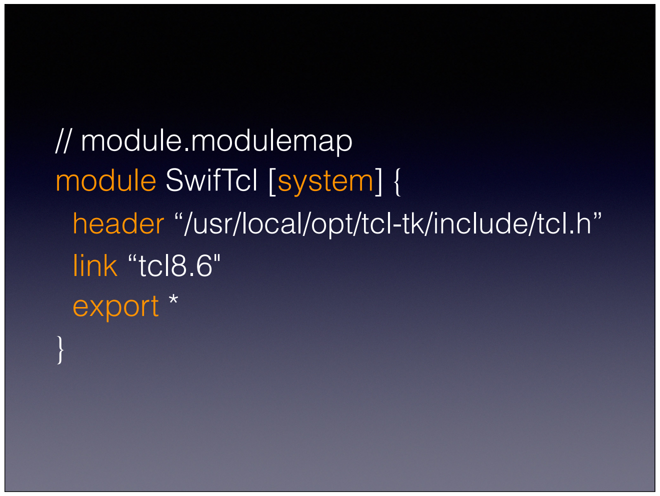
Anybody happen to know what’s special about 8.124?



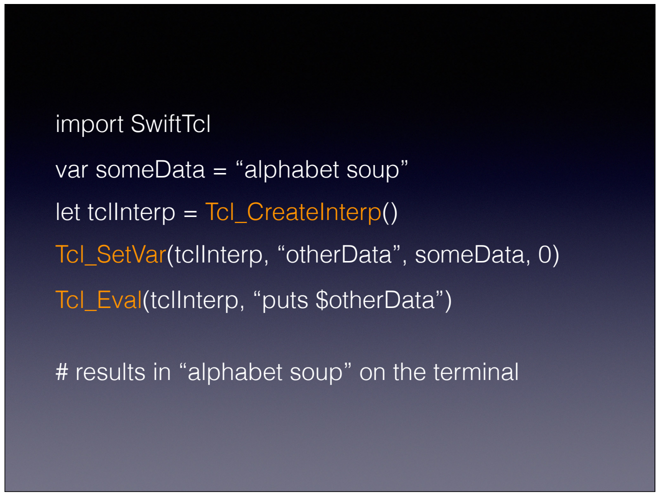
Not a Fluke



```
// swift-tools-version:4.0
import PackageDescription
let package = Package(
    name: “SwifTcl”
)
```

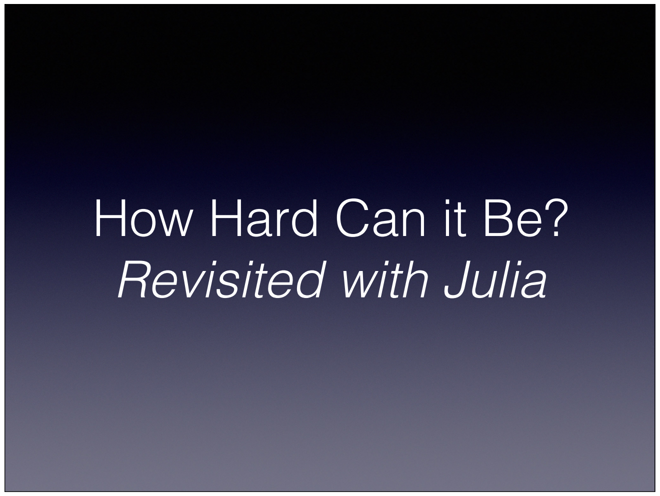


```
// module.modulemap
module SwifTcl [system] {
  header “/usr/local/opt/tcl-tk/include/tcl.h”
  link “tcl8.6"
  export *
}
```



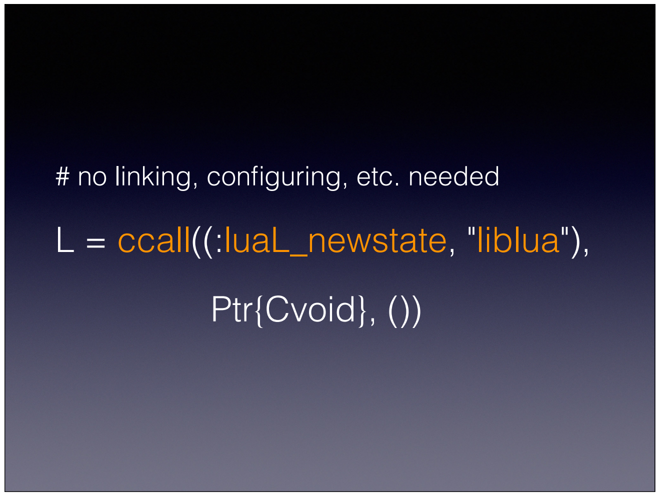
```
import SwiftTcl
var someData = “alphabet soup”
let tclInterp = Tcl_CreateInterp()
Tcl_SetVar(tclInterp, “otherData”, someData, 0)
Tcl_Eval(tclInterp, “puts $otherData”)

# results in “alphabet soup” on the terminal
```

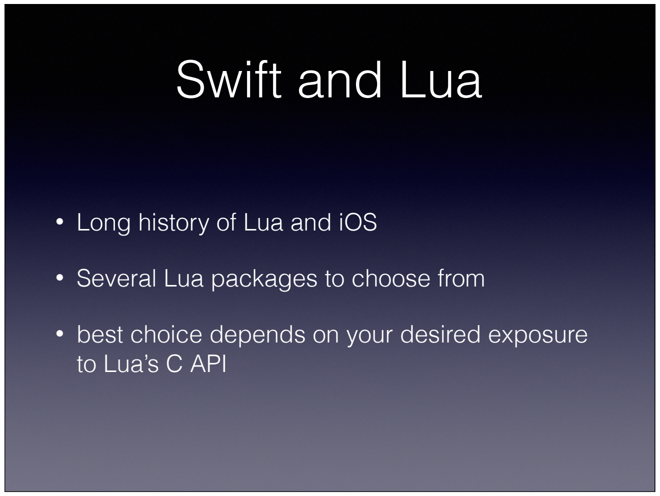


How Hard Can it Be?

Revisited with Julia

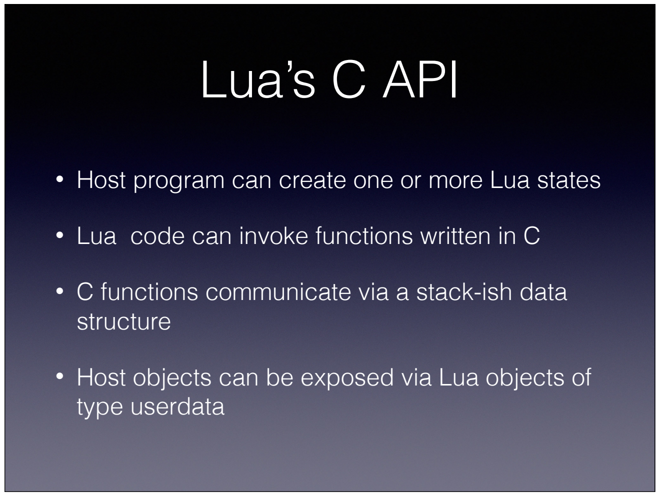


```julia
# no linking, configuring, etc. needed
L = ccall((:luaL_newstate, "liblua"),
          Ptr{Cvoid}, ())
```


Swift and Lua

- Long history of Lua and iOS
- Several Lua packages to choose from
- best choice depends on your desired exposure to Lua's C API

Mention Codea, love2d, Corona, and roll-your-own


Lua's C API

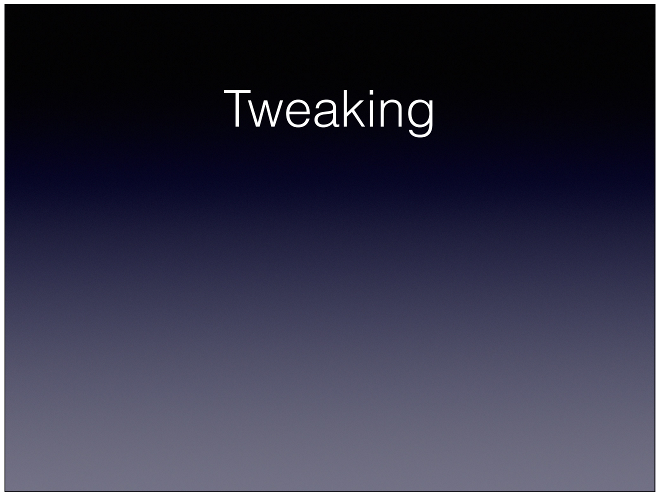
- Host program can create one or more Lua states
- Lua code can invoke functions written in C
- C functions communicate via a stack-ish data structure
- Host objects can be exposed via Lua objects of type userdata

Lua Userdata and Metatables

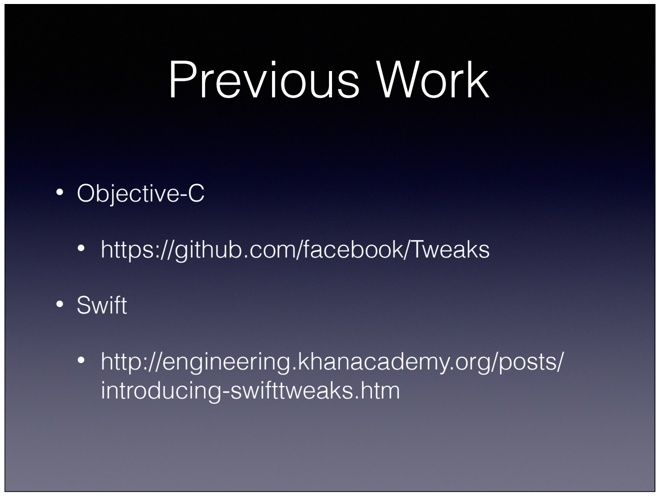
- Userdata (*full and light*) - host memory block manipulated by Lua
- rawget(getmetatable(o) or {}, "__ev")
- Indexing a table is an event.

Lua Swift Packages

- https://github.com/profburke/clua
- https://github.com/DavidSkrundz/CLua
- https://github.com/sdegutis/lua4swift


Tweaking

An easy way to fine-tune, and adjust parameters for iOS apps in development.
As opposed to Cydia tweaks for Jailbroken devices.


Previous Work

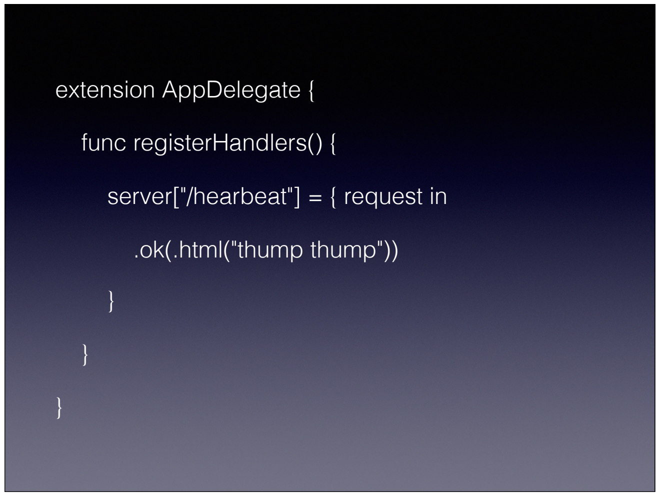
- Objective-C
  - https://github.com/facebook/Tweaks
- Swift
  - http://engineering.khanacademy.org/posts/introducing-swifttweaks.htm

Mehitabel and Tweaks

You've got chocolate in my peanut butter.

- Can be used remotely or on-device
- Proxy devices HTTP through the on-device webserver

```
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
   var window: UIWindow?
   let server = TweaksServer()
   func application(_ application: UIApplication, didFinish… {
      registerHandlers()
      try? server.start()
      return true
   }
}
```



```
extension AppDelegate {
    func registerHandlers() {
        server["/hearbeat"] = { request in
            .ok(.html("thump thump"))
        }
    }
}
```