Swifter.jl

wookyoung noh
https://github.com/wookay
JuliaCon 2016

Hello everyone, I’m wookyoung.
I’d like to introduce a package, in 8 minutes

* intro
* architecture
* tools for demo
* demo
* summary

it is my first talk at JuliaCon.
I’m supposed to do a demo.
and I wish to you enjoy this presentation :)

intro

here’s an intro.

motivation

* iOS app developer

* habits of yak shaving
  - libcat : Objective-C, Ruby (2011)
  - AppConsole, Swifter.jl : Swift, Julia (2016)

I’m an iOS app developer.
I had made a tool about 5 years ago, to develop iOS apps as more interactive way.
in this year, I began to make that kind of tool using Julia and Swift.

tool for iOS app development

* Xcode
  - debugging (lldb, swift REPL)
  - prototyping (swift playgrounds)
  - UI testing

let’s look around the tool for iOS app development.
certainly, there’s the Xcode.
probably, in this room, the most of the audiences are not familiar with this,
Xcode has breeding iOS app developers.
It provides the way of debugging, prototyping, and UI testing.

dynamic languages
of tool development for iOS app

* Python
  - Chisel for lldb extension
* Ruby
  - fastlane
* Julia

additionally, there are dynamic languages for the tool development.

ideal ways or
realistic capable shovelings

* ideal ways
 - SwiftCall.jl
 - like PyCall.jl, RCall.jl, JavaCall.jl, Cxx.jl

* realistic capable shovelings
 - continuous small improvements

Swifter.jl, this is a package for ios app development in julia.
but it does not provide the ability, to directly call with swift.
what about to make SwiftCall.jl. that is the ideal way, like PyCall, RCall, JavaCall, and Cxx.jl.
but I prefer to concentrate on my interests. so I’m shoveling on
that with continuous small improvements.

architecture

if so what is it made of?

Julia

* general-purpose language

Julia is a general-purpose language.

Swift

* general-purpose language

Swift is a general-purpose language.

an RPC server
written in Swift
with reflection

an RPC client
written in Julia
with meta-programming

I’ve written both sides of, RPC server and client.
The RPC server, has written in swift with reflection.
and for the client side, as in julia with meta-programming.
that’s inevitable two languages problem!

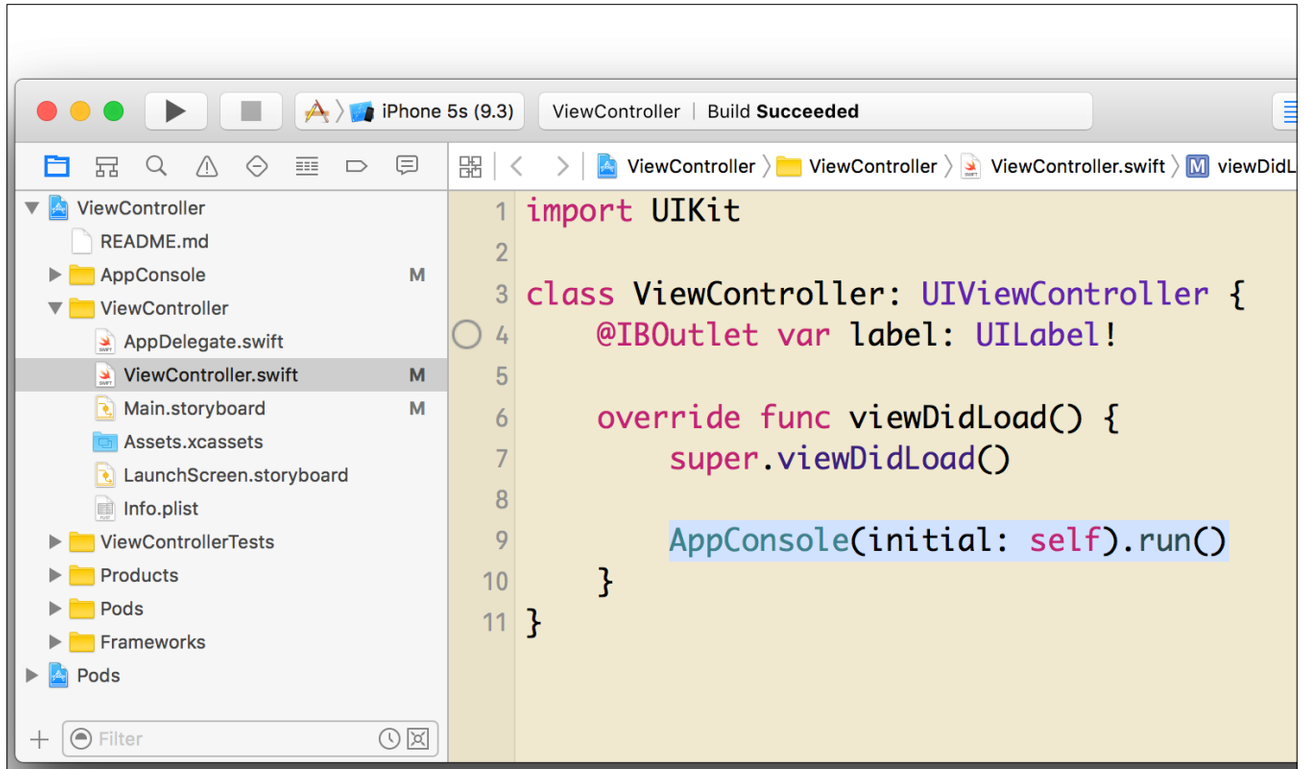
an RPC server
written in Swift
with reflection :
AppConsole

an RPC client
written in Julia
with meta-programming :
Swifter.jl

these are two packages. AppConsole and Swifter.jl.

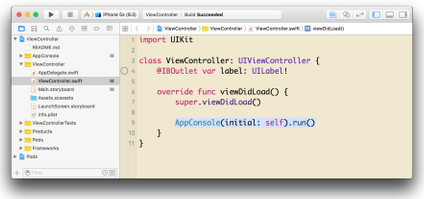
tools for demo

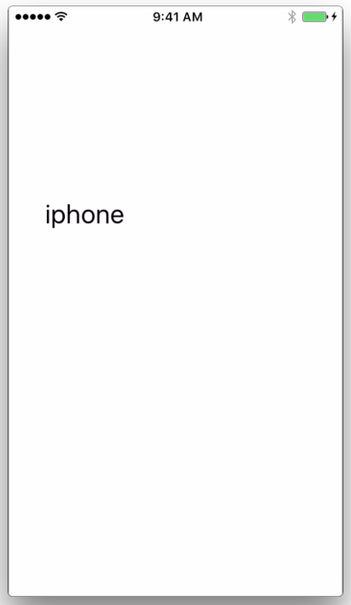
here are some tools to help us for demo (PLAY)



Xcode IDE

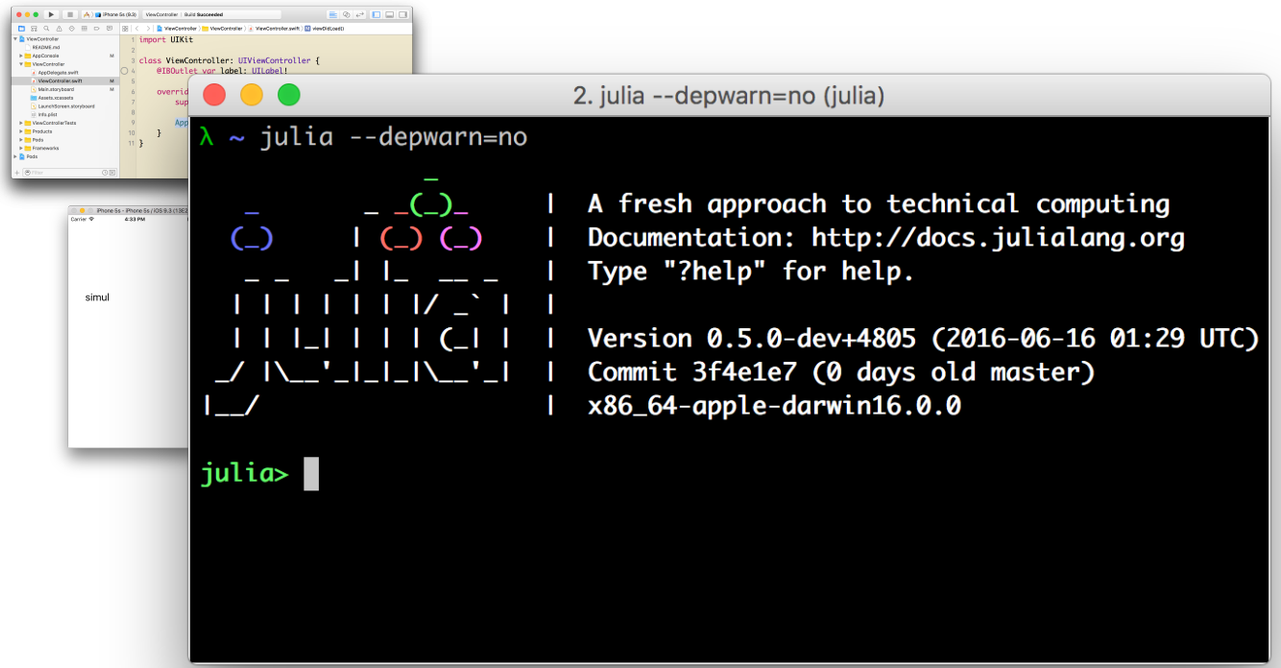
Xcode IDE





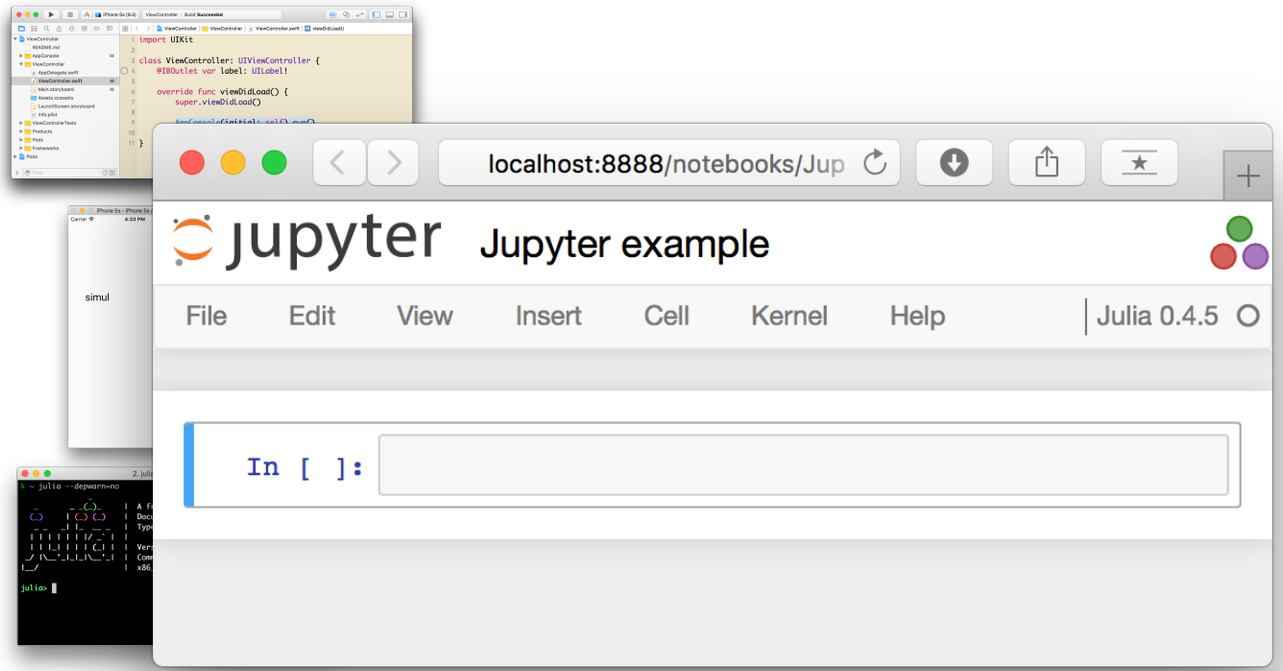
devices: simulator & iphone

simulator and iphone …



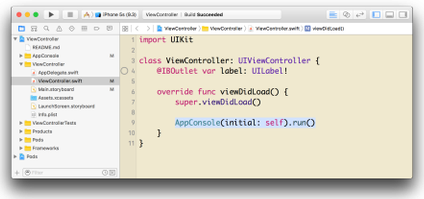
terminal

terminal for julia REPL. …



jupyter notebook

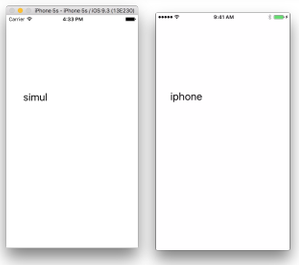
and jupyter notebook …



* Xcode version 7.3.1
* Swift version 2.2

- to run an iOS application
- to run an RPC server
  inside of that

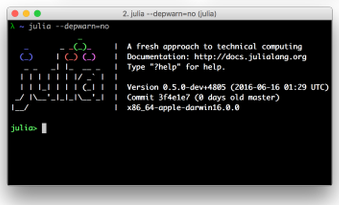
there’s the sample project to run an iOS app.
it's used to run an RPC server, during for the running app.



* simulator
* iphone

- actual running devices

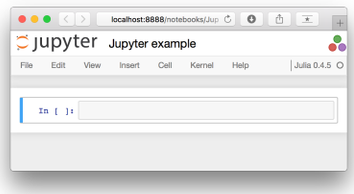
simulator and iphone, these are actual running devices.



* terminal

- Julia REPL

we need a terminal, to run Julia REPL



* jupyter notebook

- jupyter **interactive** environment
- IJulia

jupyter interactive environment is essential, for Julia programmers.
using IJulia to run the Julia backend.

Demo

Scripting

Download as Julia (.jl) from the jupyter notebook. scripting is a sequence of code cells.
if it does so fast, sleep(0.1) a bit.

more examples

* https://github.com/wookay/Swifter.jl

 - UIControls.ipynb
 - MultipleDevices.ipynb
 - TableViewController.ipynb
 - ViewController.ipynb

to see more examples, go that repository

summary

here's the summary for this presentation

lessons

* continuous small improvements

* Julia applications
 - extending dimensions Array{Idea,N}

it's a lesson for continuous small improvements,
like to keep the compatibility, with the evolution of Julia.
thanks to all of julia commiters, contributers, and communities.
and as you see it, lots of ideas have discovered with Julia,
and have been discussed, solved beautifully with this amazing communities.

References

Julia - http://julialang.org
IJulia.jl - https://github.com/JuliaLang/IJulia.jl
PyCall.jl - https://github.com/stevengj/PyCall.jl
RCall.jl - https://github.com/JuliaStats/RCall.jl
JavaCall.jl - https://github.com/aviks/JavaCall.jl
Cxx.jl - https://github.com/Keno/Cxx.jl

Swift - https://swift.org/
Swift playgrounds -
https://developer.apple.com/swift/playgrounds/
LLDB - http://lldb.llvm.org/
Chisel - https://github.com/facebook/chisel
fastlane - https://fastlane.tools/

Swifter.jl - https://github.com/wookay/Swifter.jl
AppConsole - https://github.com/wookay/AppConsole
libcat - https://github.com/wookay/libcat

swifter - https://github.com/httpswift/swifter
UIView - https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIView_Class/

here are references

q & a

questions?

thank you

thank you very much