# REACT NATIVE WITH CLOJURESCRIPT

MICHAEL LANGFORD
@MJ_LANGFORD

# A BIT ABOUT ME:

» Embedded Programming Background (firmware/ robotics/embedded Linux)

» iOS Developer since 2008, some Android off and on

» Clojure in backends/tools off and on since then 2014, looking at using it for Android/Multi Platform devices

# OVERVIEW OF THE PLATFORM STORY

# PRINCIPLES OF MOBILE

» Old Versions of code can exist for long after you want them to

» Cryptographic Signing is an annoying and ever present component

» App Stores set standards for the apps you must adhere to for public apps

# PRINCIPLES OF MOBILE (CONT)

» Smaller surface spaces than computers, so consistency with UI norms is essential

» Fingers are far far fatter than mouse pointers: use different controls than web

» Prepare for maintenance cycle related to new major OS version release

# PRINCIPLES OF MOBILE (CONT)

» Authentication not always needed...local data doesn't need auth

» An app is far more available and personal than a website, ever available, many sensors

» Simulator/Emulators are not the same thing as running on a device

# YOU CAN JUST WRITE APPS ON A SIMULATOR...BUT WHERE'S THE USE OF THAT?

# TO USE IT ON YOUR PHONE, YOU NEED TO DO A FEW THINGS (PLATFORM SPECIFIC)

» Create a developer account

» Provision your phone

# TO PUT IT ON A STORE

» Android: Make a signing key, never ever lose it

» Apple: Make a signing key, preferably in a separate user and import it into your main account

» Either Platform, sign it, upload it, and fill in screen shots

» For Apple, you wait a week or so, and hope you met all the design and functionality guidelines

# SPEAKING OF GUIDELINES

# DOCUMENTS TO COMPLY WITH ON IOS

Human Interface Guide https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/

App Review Guidelines: https://developer.apple.com/app-store/review/guidelines/

# GUIDELINES FOR ANDROID

Material Guide for Android: http://developer.android.com/design/material/index.html

Google Play Store Guidelines: https://play.google.com/about/developer-content-policy.html

# ON TO THE SPECIFICS

# REACT NATIVE: BY FACEBOOK!

» RN is a Facebook project

» “= Android 4.1 (API 16) and >= iOS 7.0

» Allows native modules (Objective-C or Java and some 3rd parties have done Swift) if RN doesn't have a piece of functionality you need

» Requires you to deal with "NPM and JS quality" toolchains

# WHAT DOES RN DO DIFFERENT FROM NATIVE?

» React-style composable UIs

» Declarative UI instead of stateful imperative

» No GUI UI builders

» Flexbox instead of AutoLayout(iOS)/
Layouts(Android)

# HOW IS IT DIFFERENT THAN USING A WEBVIEW (CORDOVA, ETC)?

» Native components, the same components as native code does

» Multiple threads

» Far better performance

# * USES JAVASCRIPTCORE TO RUN THE APPS, NO WEBVIEW REQUIRED

# TOOLS

# STILL REQUIRES THE FULL TOOLCHAIN FOR THE PLATFORM

» iOS requires Xcode, Mac OS X

» Android requires Android Studio still

# PREPARING FOR FROM THE COMMAND LINE (OS X VERSION)

```
brew install ant
brew install maven
brew install gradle
brew install android-sdk
brew install android-ndk
```

(https://gist.github.com/patrickhammond/4ddbe49a67e5eb1b9c03)

# PREPARING FOR IOS FROM THE COMMAND LINE

```
xcode-select --install
```

Push some buttons

# PREPARING THE NODE TOOLS TO GET THE CLOJURESCRIPT ENVIRONMENT

```
» brew install node

» npm install -g natal

or

» npm install -g re-natal
```

# CREATE THE APP (PICK ONE!, THIS CAN BE SLOW)

- Om.Now: natal init WhateverYouCallIt
- Reagent: re-natal init WhateverYouCallIt
- Om.Next: natal init --interface om-next WhateverYouCallIt
- Some of these auto-launch the iOS simulator when done

# OPEN UP THE CORE.CLJS FILE AND START EDITING

TIME TO EXAMINE ONE OF THEM. I'M GOING TO GO FOR OM, AS THE PRINCIPLE CARRIES THROUGH: IT'S THE WEB FRAMEWORK WITH DIFFERENT FUNCTIONS TO GENERATE ELEMENTS

# OM (AKA, OM.NOW)

om/root function takes the top level view, the app state atop, and I honestly don't know what the target map is for in the context of React Native.

```
(om/root main-view app-state {:target 1})
```

# OM/IRENDER PROTOCOL AND THE RENDER FUNCTION

with-error-view: This is a wrapper class that catches errors and displays them in the simulator

view: This specifies a View, the basic building block of items that display on the screen. The first param is a map of attributes, and the rest of the params are other controls

# THE OTHER "CONTROLS" SPECIFIED

text: This specifies basic static text. Same prototype, first param map of attributes, mostly formatting, but other notables, onLayout for animations/becoming viewable, testID for calling this item up in tests.

image: This specifies a image, and can summon things from the web instead of local (not suggested for most things, iOS apps often work offline).

# THE TOUCH "CONTROL" SPECIFIED

touchable-highlight: This specifies a button. I don't know why they don't say "button", but whatever. You can also pass :onPress in the map param

# THEY'RE JUST CLOJURESCRIPT FUNCTIONS THOUGH...SO YOU CAN COMPOSE

```
(image
 {:source
   {:uri "https://raw.githubusercontent.com/cljsinfo/
logo.cljs/master/cljs.png"}
 :style {:width 80 :height 80
 :marginBottom 0}})

 ;could be just

 (image-from-web)
```

# LAYOUTS ARE ACCOMPLISHED WITH THE :FLEXDIRECTION KEY

```
(view {:style {:flexDirection "column"}} (another-
view) (another-view))

(view {:style {:flexDirection "row"}} (another-view)
(another-view))
```

# LAYOUTS ARE ALSO ACCOMPLISHED ALSO WITH THE :JUSTIFYCONTENT KEY

```
(view {:style {:justifyContent "flex-start"}}
(another-view) (another-view))

(view {:style {:justifyContent "space-around"}}
(another-view) (another-view))

(view {:style {:justifyContent "center"}} (another-
view) (another-view))
```

# LAYOUTS CAN ALSO BE ABSOLUTE

```
(view {:style {:position "absolute" :top 10 :left
10 :backgroundColor "#ff0000" :height 20 :width
300}})

;This puts a box at 10,10 on a screen
```

# THERE IS A REPL

» Run natal repl

» In another terminal, run natal launch (and wait)

» Import your app namespace

» Change the app state at will

# THERE IS LIVE RELOAD TOO

» Part of the launch sequence

» Exiting the simulator wrong can mess up the live reload

# GO FORTH AND FILL YOUR POCKETS WITH APPS

# SUGGESTED RESOURCES

Clojurians.slack.com #cljsrn

Github pages of the tools (inline)

Mike Fikes Blog: http://blog.fikesfarm.com

Boot for React Native: https://github.com/mjmeintjes/boot-react-native

React Native website: https://facebook.github.io/react-native/docs/getting-started.html

# DEMO TIME

# THANKS

@MJ_LANGFORD
MICHAEL.LANGFORD@ROWDYLABS.COM