RubyMotion: Analysis of a Proletarian [programming] Revolution, *circa* Apple iOS

Mobile Development for the Everyman!

What is RubyMotion?

RubyMotion is a revolutionary toolchain for iOS.
It lets you quickly develop and test native iOS applications for iPhone or iPad using Ruby, a simple, concise high-level language made popular by Ruby on Rails.

No more **.h** files, no more **static types**, no more ***bugged-out* Xcode**, no more **@implementation** and **@synthesize**.... whatttttt?

Why RubyMotion?

Complete memory management based on ARC

RubyMotion apps run right on top of **the same core iOS technologies** as apps using Apple's Objective-C language

RubyMotion is based on the MacRuby project, a mature codebase **developed over 5 years at Apple**

You get to call into **the same set of Objective-C APIs** at no cost in size or performance, and new SDK releases are automatically supported.

But the **real reason** why you should use RubyMotion is....

This...

HFAppDelegate.h

```objc
#import <UIKit/UIKit.h>
#import <FBiOSSDK/FacebookSDK.h>

@class HFViewController;

@interface HFAppDelegate : UIResponder<UIApplicationDelegate>

@property (strong, nonatomic) UIWindow *window;
@property (strong, nonatomic) HFViewController *rootViewController;

@end
```

HFAppDelegate.m

```objc
#import "HFAppDelegate.h"
#import "HFViewController.h"
#import <FBiOSSDK/FBProfilePictureView.h>

@implementation HFAppDelegate

@synthesize window = _window;
@synthesize rootViewController = _rootViewController;

- (BOOL)application:(UIApplication *)application
            openURL:(NSURL *)url
  sourceApplication:(NSString *)sourceApplication
         annotation:(id)annotation {

    return [FBSession.activeSession handleOpenURL:url];
}

- (void)applicationWillTerminate:(UIApplication *)application {
    [FBSession.activeSession close];
}

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    [FBProfilePictureView class];

    self.window = [[UIWindow alloc] initWithFrame:[[UIScreen mainScreen] bounds]];

    if ([[UIDevice currentDevice] userInterfaceIdiom] == UIUserInterfaceIdiomPhone) {
        self.rootViewController = [[HFViewController alloc] initWithNibName:@"HFViewController_iPhone" bundle:nil];
    } else {
        self.rootViewController = [[HFViewController alloc] initWithNibName:@"HFViewController_iPad" bundle:nil];
    }
    self.rootViewController.navigationItem.title = @"Hello, Facebook";

    UINavigationController *navigationController =
        [[UINavigationController alloc] initWithRootViewController:self.rootViewController];

    self.window.rootViewController = navigationController;
    [self.window makeKeyAndVisible];

    return YES;
}

@end
```

...could be **this**:

```
class AppDelegate

  def application( application, openURL: url, sourceApplication: sourceApplication, annotation: annotation )
    FBSession.activeSession.handleOpenURL( url )
  end

  def setWindow(new_window)
    @window = new_window

    UIApplication.sharedApplication.setStatusBarHidden( false, animated:false );
  end

  def applicationWillTerminate( application )
    FBSession.activeSession.close()
  end

  def application(application, didFinishLaunchingWithOptions:launchOptions)
    true
  end

  def window
    @window
  end

end
```

(yes, **functionally it's the exact same code**.)

"After years of iOS work, RubyMotion seems like a thousand kittens playing the piano while sliding down a double-rainbow."

- Johannes Fahrenkrug
Founder, Springenwerk

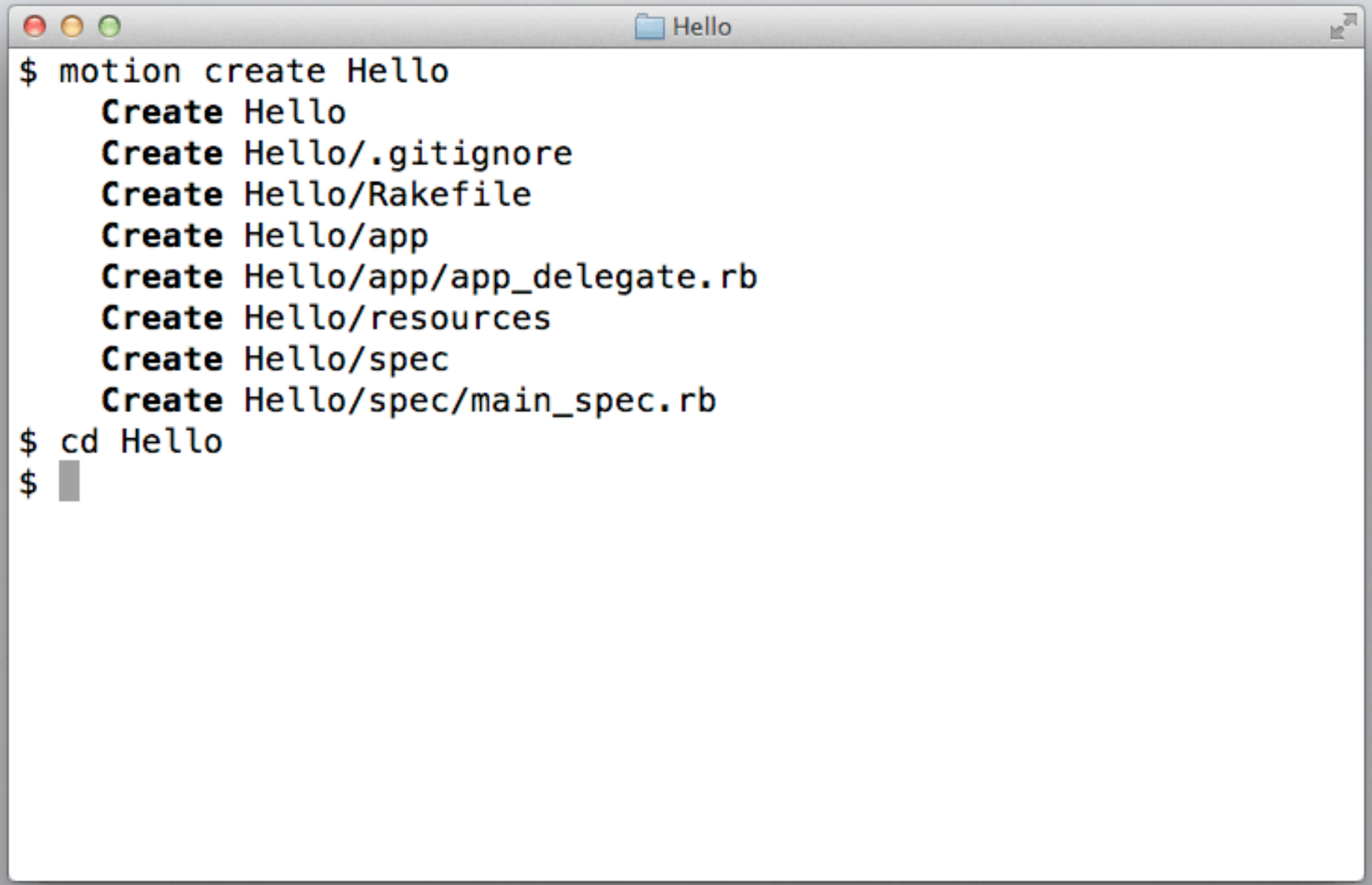
Conceptually, RubyMotion Is A Combination Of Two Major Components

Runtime:

A brand-new implementation of the Ruby language, tightly integrated with the native iOS runtime and optimized for embedded iOS device constraints.

...And Project Management:

A **command-line interface** to create, manage, and interactively develop RubyMotion projects and an interactive console where you can evaluate expressions on the fly.



1. Run your app on your **iOS device**.
2. Submit development builds to **TestFlight**.
3. Generate .ipa archives for **Ad-Hoc distribution**.
4. Prepare a submission for the **App Store**.

Hello

```
$ rake device
(in /private/tmp/Hello)
     Build ./build/iPhoneOS-5.0-Development
    Deploy ./build/iPhoneOS-5.0-Development/Hello.ipa
$ rake testflight notes='Hi team, this is the first build of our
new app, let me know what you think!'
```

Like the Chrome Developer Tools?

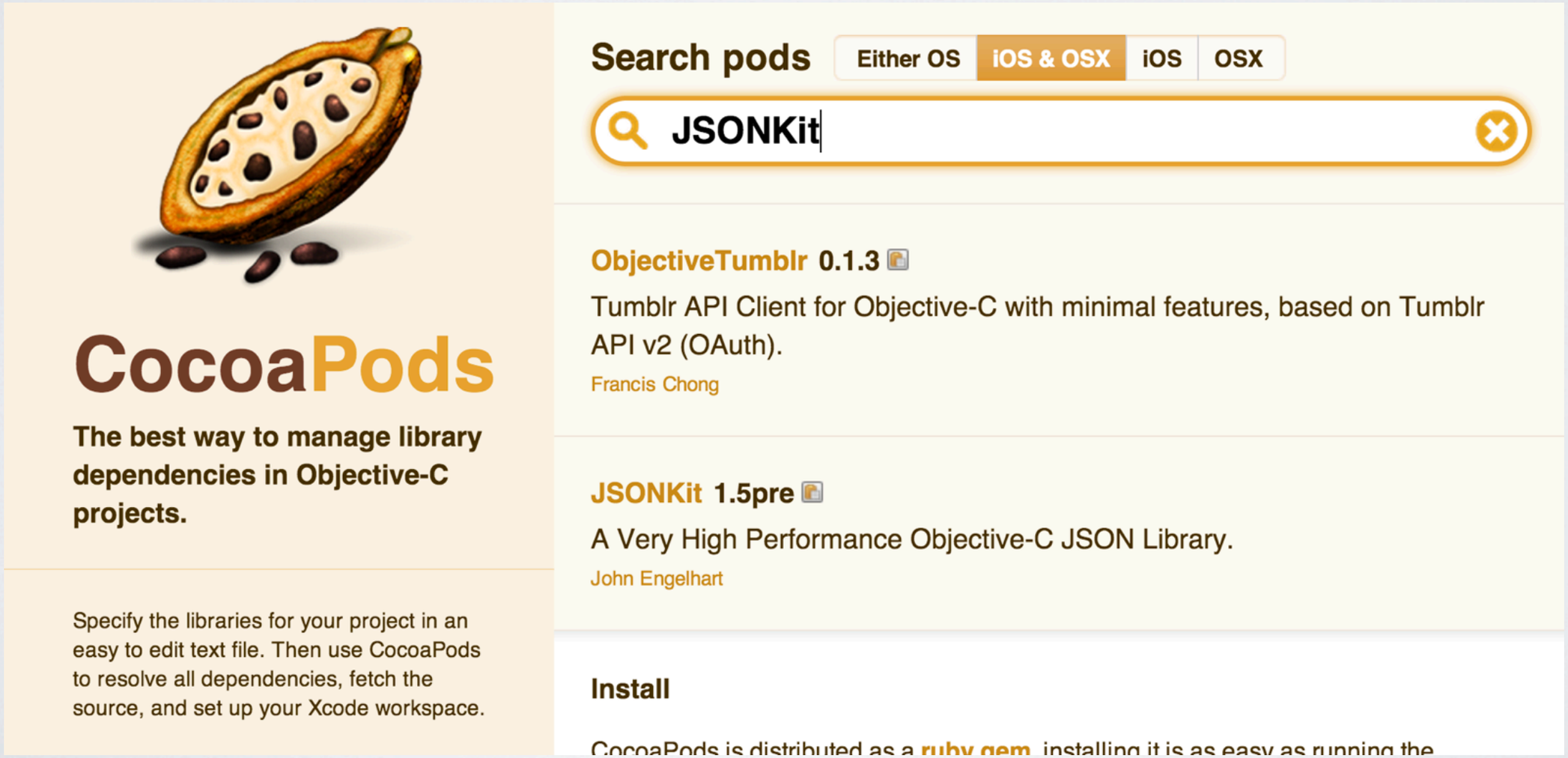
RubyMotion comes with an interactive console that lets you navigate and introspect the iOS API jungle with ease, from the comfort of your terminal.

The console is connected to your application running in the simulator.
Changes you interactively make are reflected in the app in real time.

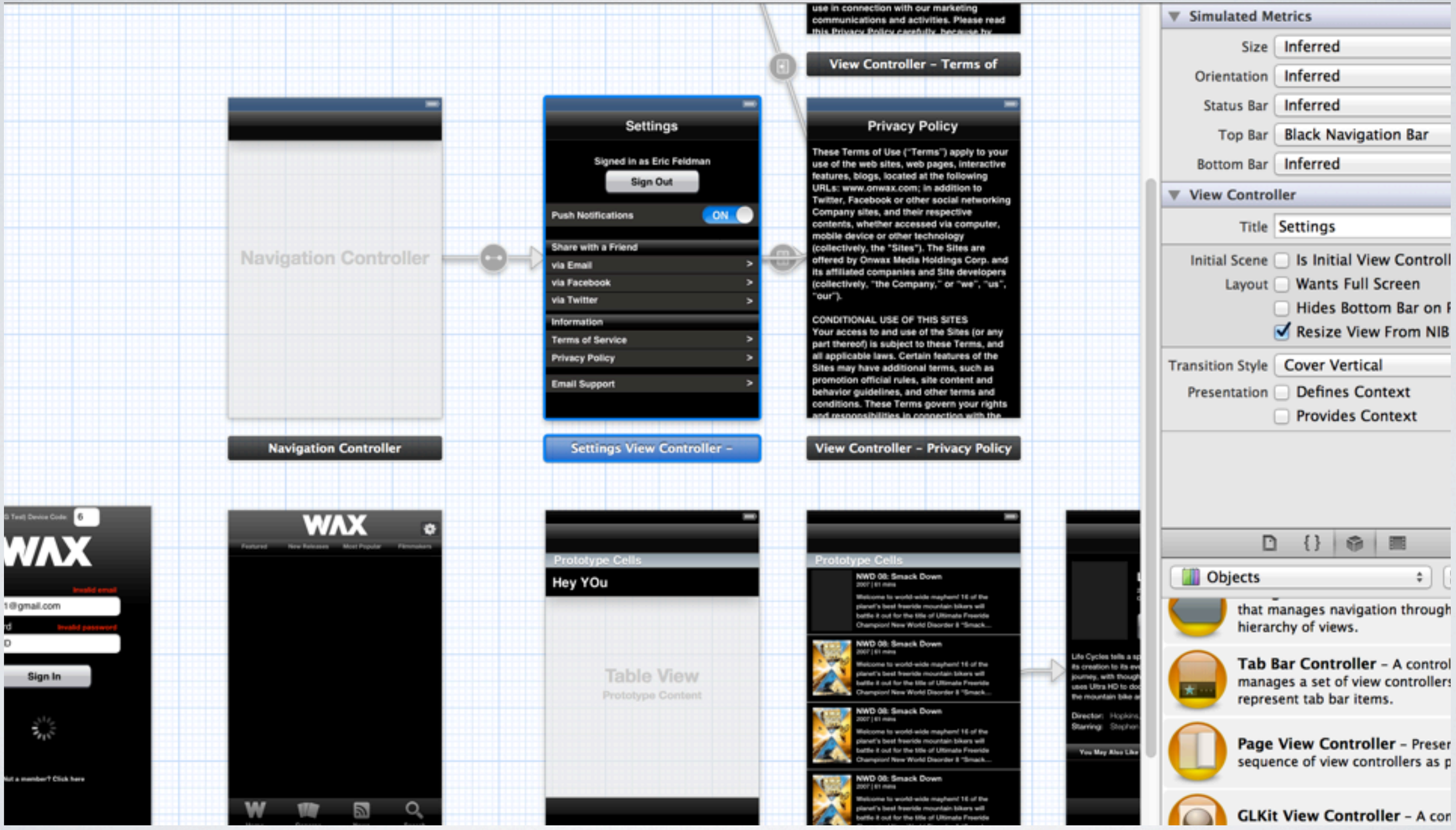
```
(main)> tree
  0: . UIWindow(#d9453e0, [[0.0, 0.0], [768.0, 1024.0]])
  1: +-- UIImageView(#c422fb0, [[0.0, 0.0], [2048.0, 1536.0]])
  2: `-- UILayoutContainerView(#d946280, [[0.0, 0.0], [768.0, 1024.0]])
  3:     `-- UINavigationTransitionView(#c4252e0, [[0.0, 0.0], [1024.0, 768.0]])
  4:         `-- UIViewControllerWrapperView(#c425740, [[0.0, 0.0], [1024.0, 768.0]])
  5:             `-- UIView(#a629d10, [[0.0, 0.0], [1024.0, 768.0]])
  6:                 `-- UIView(#a62a150, [[0.0, 0.0], [1024.0, 768.0]]) stylename: :container
  7:                     `-- UIView(#a62f3e0, [[0.0, 0.0], [1024.0, 768.0]])
  8:                        +-- UIImageView(#a62fb80, [[68.0, 446.0], [781.0, 119.0]]) stylename: :welcome_label
  9:                        +-- UIImageView(#a62fdc0, [[0.0, -385.0], [1024.0, 770.0]]) stylename: :carousel
 10:                        `-- UIButton(#a630030, [[68.0, 624.0], [287.0, 61.0]]) stylename: :done_button
 11:                            `-- UIImageView(#dad8750, [[0.0, 0.0], [287.0, 61.0]])

=> UIWindow(#d9453e0, [[0.0, 0.0], [768.0, 1024.0]])
(main)> adjust 10
=> UIButton(#a630030, [[68.0, 624.0], [287.0, 61.0]]), child of UIView(#a62f3e0) stylename: :done_button
(main)> right 100
```

Objective-C 3rd Party Library Support!

CocoaPods was originally designed to be integrated in Objective-C Xcode projects, but RM worked with the awesome CocoaPods author to make it possible to be used in RubyMotion projects.



And... Interface Builder!



But The *Real* Reason I Love RubyMotion Is...

The Community DSLs!

(...Domain Specific Languages)

"Have you ever had one of those 'ugh, I never want to spend any time in this piece of code again' moments?

Ruby provides many **metaprogramming** features that allow you to build Domain Specific Languages; pieces of syntax that allow you to solve a specific problem in a concise and simple fashion, wrapping around the more complex implementational details."

Ruby, As A Dynamic Language, Is ***Extremely*** Flexible

For example, **addition** is performed with the plus (**+**) operator. But, if you'd rather use the **readable word plus**, you could add such a method to Ruby's built-in Numeric class.

```ruby
class Numeric
  def plus(x)
    self.+(x)
  end
end

y = 5.plus 6
# y is now equal to 11
```

Objective-C Methods Are **Verbose, LOW-level** And Long Method Names Are ***Not Fun***...

```
NSString *filtered = [unfiltered stringByReplacingOccurrencesOfString:@"verbose"
                                                           withString: @"explicit"
                                                              options: NSCaseInsensitiveSearch
                                                                range: NSMakeRange(0, [unfiltered length])];
```

(...Yuk!)

...Which Is Where RubyMotion **DSLs** Come In:

Objective-C:

```
[[UIApplication sharedApplication] openURL:[NSURL URLWithString:urlString]];
```

Ruby:

```
UIApplication.sharedApplication.openURL( NSURL.URLWithString( url ))
```

SugarCube DSL:

```
"http://www.pop.us".nsurl.open()
```

MY FAVORITE DSL'S

Teacup

https://github.com/rubymotion/teacup

Using teacup, you can easily create and style layouts while keeping your code dry. The goal is to offer a rubyesque (well, actually a rubymotion-esque) way to create interfaces programmatically.

```
Teacup::Stylesheet.new :thankyou_styles do

  style :placeholder,
    image: UIImage.imageNamed( 'placeholder-3.png' ),
    width: 1024,
    height: 768


  style :welcome_label,
    image: UIImage.imageNamed( 'label-thankyou.png' ),
    x: 63,
    y: 69,
    width: 600,
    height: 250


  style :image_container,
    image: UIImage.imageNamed( 'bg-image-thankyou.png' ),
    x: 0,
    y: 384,
    width: 1024,
    height: 386

end
```

Sugarcube

https://github.com/rubymotion/sugarcube

These extensions hope to make development in RubyMotion more enjoyable by tacking "UI" methods onto the base classes (String, Fixnum, Numeric).

```
image.scale_to [37, 37]
image.rounded  # default: 5 pt radius
image.rounded(10)

image.in_rect([[10, 10], [100, 100]])  # get part

image.darken  # => good for "pressed" buttons
image.darken(brightness: -0.5, saturation: -0.2)

image.rotate(:left)
image.rotate(:right)
image.rotate(:flip)  # 180° - if you have a bette
image.rotate(45.degrees)
```

```
:center.uialignment  # => UITextAlign
:upside_down.uiorientation  # => UIDe
:rounded.uibuttontype  # => UIButtonT
:highlighted.uicontrolstate  # => UIC
:touch.uicontrolevent  # => UIControl
:change.uicontrolevent  # => UIContro
:all.uicontrolevent  # => UIControlEv
:blue.uicolor  # UIColor.blueColor
```

```
# and the completed argument
view.fade_out { |completed|
  view.removeFromSuperview
}
```

```
# UIFont from name
"my_font".uifont # => UIFont.fontWithName("my_font", size:UIFont.systemFontSize)
"my_font".uifont(20) # => UIFont.fontWithName("my_font", size:20)
```

BubbleWrap

https://github.com/rubymotion/bubblewrap

Cocoa wrappers and helpers for RubyMotion (Ruby for iOS) - Making Cocoa APIs more Ruby like, one API at a time.

```
> Device.iphone?
# true
> Device.ipad?
# false
> Device.front_camera?
# true
> Device.rear_camera?
# true
> Device.orientation
# :portrait
> Device.simulator?
```

```
> "matt_aimonetti".camelize
=> "MattAimonetti"
> "MattAimonetti".underscore
=> "matt_aimonetti"
```

```
# Uses the front camera
BW::Device.camera.front.picture(media_types: [:movie, :image]) do |result|
  image_view = UIImageView.alloc.initWithImage(result[:original_image])
end
```

```
BW::HTTP.get("https://api.github.com/users/mattetti") do |response|
  p response.body.to_str
end
```

```
> EM.schedule { puts Thread.current.object_id }
146027920
=> nil
> EM.schedule_on_main { puts Thread.current.object_id }
112222480
=> nil
```

DEMO TIME!

POP SIGN-IN APP

MORE INFO:

RubyMotion: http://www.rubymotion.com/

The best tutorial... ever: http://rubymotion-tutorial.com/

Teacup: https://github.com/rubymotion/teacup

Sugarcube: https://github.com/rubymotion/sugarcube

BubbleWrap: https://github.com/rubymotion/BubbleWrap

StoryBoard Example: https://github.com/damassi/RubyMotion-Storyboard-Example

Facebook SDK: https://github.com/damassi/Motion-FacebookSDK

Urban Airship Integration: https://github.com/damassi/RubyMotion-UrbanAirship

THANKS!