

RubyMotion

A brave new world for iOS development

Michael Erasmus

About Me

About Me

Ruby, JavaScript, Web

About Me

Ruby, JavaScript, Web

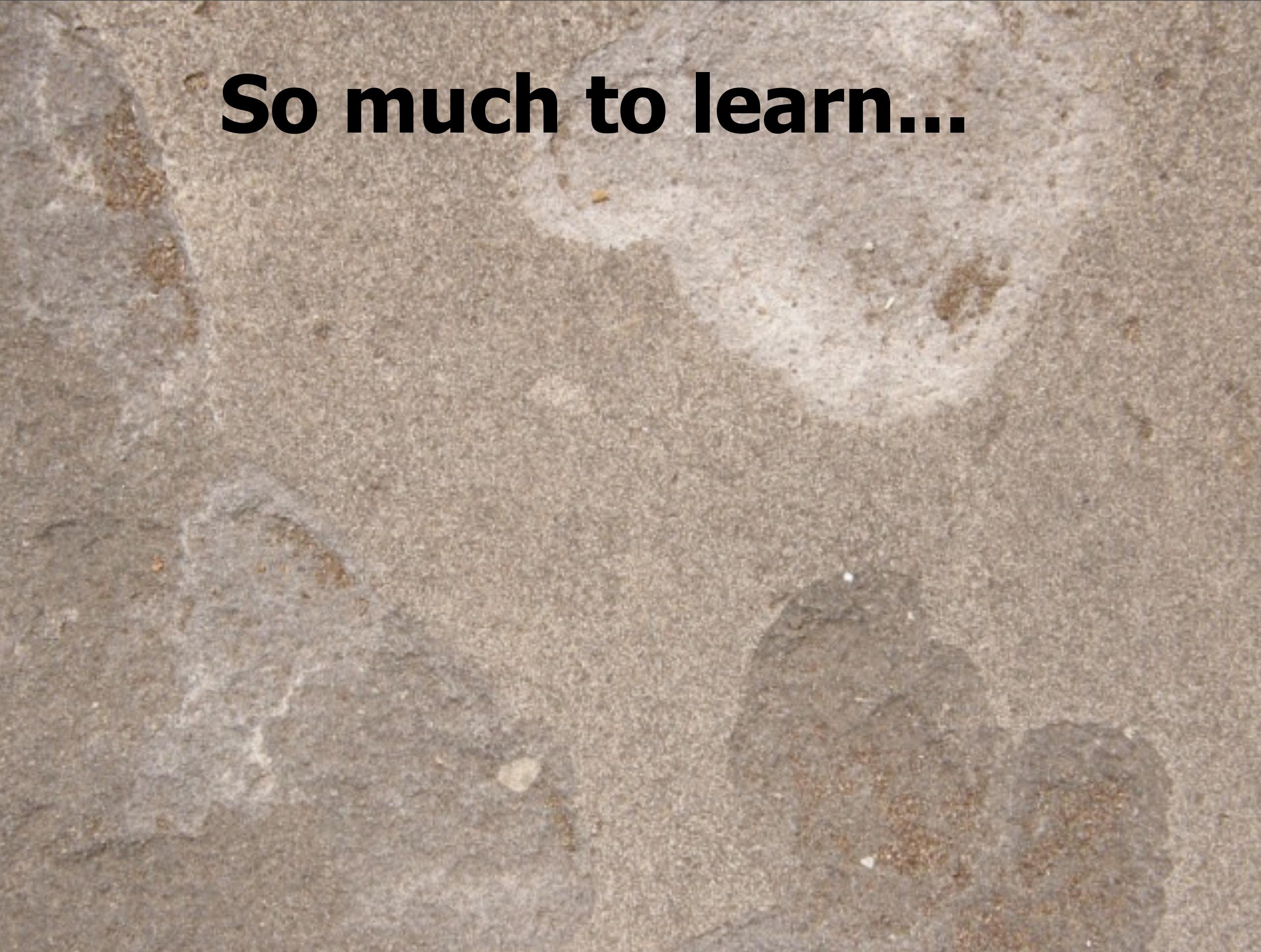
“Backend Guy”

About Me

Ruby, JavaScript, Web

“Backend Guy”

Hey, let's write iPhone apps!



So much to learn...

So much to learn...

ios

So much to learn...

ios



Objective-C

So much to learn...

ios



Uhm.. this is awkward

The screenshot shows the Xcode interface with the Adium project open. The left sidebar displays the project structure under the 'Adium' group, including targets like Adium, Adium.pch, Adium.h, Adium.m, and various utility and framework groups. The main editor area shows the source code for 'AllAutoScrollView.m'. The code is a subclass of NSScrollView, implementing methods for initialization and scroll handling. It includes comments about auto-scrolling behavior and focus ring management. The right sidebar contains the 'Identity and Type' panel, which shows the file is an Objective-C source file named 'AllAutoScrollView.m' located at '/Users/dennda/Downloads/adium-76b5e0220340/Frameworks/Utilities/AllAutoScrollView.m'. The 'Target Membership' section has 'AllUtilities.framework' checked. The bottom right corner shows the 'File Template Library' with categories for Objective-C class, protocol, and test case class.

```
/*
 * Adium is the legal property of its developers, whose names are listed in the copyright file included
 * with this source distribution.
 *
 * This program is free software; you can redistribute it and/or modify it under the terms of the GNU
 * General Public License as published by the Free Software Foundation; either version 2 of the License,
 * or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even
 * the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General
 * Public License for more details.
 *
 * You should have received a copy of the GNU General Public License along with this program; if not,
 * write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
 */

#import "AllAutoScrollView.h"

#define AUTO_SCROLL_CATCH_SIZE 20 //The distance (in pixels) that the scrollview must be within (from the bottom) for auto-scroll to kick in.

@interface AllAutoScrollView : NSScrollView
- (void)_initAutoScrollView;
- (void)documentFrameDidChange:(NSNotification *)notification;
@end

@implementation AllAutoScrollView
/*
 A subclass of NSScrollView that:
 - Automatically scrolls to bottom on new content
 - Shows a focus ring even if the contained view would not normally show one (an NSTextView, for example)
 */
- (id)initWithCoder:(NSCoder *)aDecoder
{
    if ((self = [super initWithCoder:aDecoder])) {
        [self _initAutoScrollView];
    }
    return self;
}

- (id)initWithFrame:(NSRect)frameRect
{
    if ((self = [super initWithFrame:frameRect])) {
        [self _initAutoScrollView];
    }
    return self;
}

- (void)_initAutoScrollView
{
    autoScrollToBottom = NO;
    inAutoScrollToBottom = NO;
    passKeysToDocumentView = NO;

    //Focus ring
    alwaysDrawFocusRingIfFocused = NO;
    lastKey = nil;
    shouldDrawFocusRing = NO;

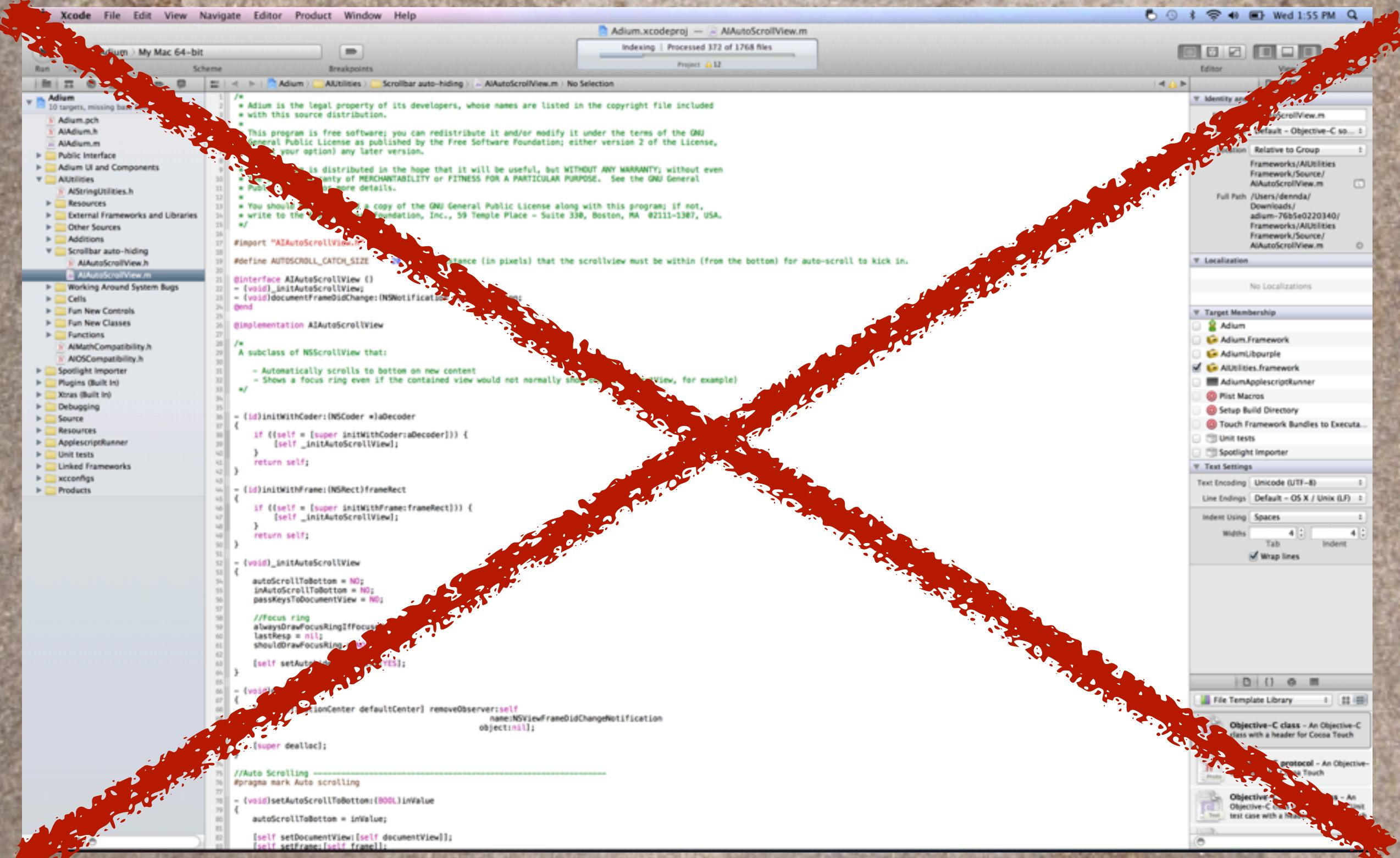
    [self setAutohidesScrollers:YES];
}

- (void)dealloc
{
    [[NSNotificationCenter defaultCenter] removeObserver:self
                                                name:NSViewFrameDidChangeNotification
                                              object:nil];
    [super dealloc];
}

//Auto Scrolling -----
#pragma mark Auto scrolling

- (void)setAutoScrollToBottom:(BOOL)inValue
{
    autoScrollToBottom = inValue;
    [self setDocumentView:[self documentView]];
    [self setFrame:[self frame]];
}
```

Uhm... this is awkward



RubyMotion to the Rescue!



So...what exactly is it?

"a revolutionary **toolchain** that lets you quickly develop and **test native** iOS and OS X applications for **iPhone**, **iPad** and **Mac**, all using the awesome **Ruby** language you know and love."

rubymotion.com



So...what exactly is it?

"a revolutionary **toolchain** that lets you quickly develop and **test native** iOS and OS X applications for **iPhone**, **iPad** and **Mac**, all using the awesome **Ruby** language you know and love."

rubymotion.com



Toolchain

Toolchain

Ruby, Rake, Bundler, RubyGems, irb

Toolchain

Ruby, Rake, Bundler, RubyGems, irb

Vim, Emacs, Sublime Text 2

Toolchain

Ruby, Rake, Bundler, RubyGems, irb

Vim, Emacs, Sublime Text 2

UIKit, AppKit, CocoaPods

Native

Not a “bridge”

Compiled machine language

Performance on par with Obj-C

App store friendly

Ruby

Ruby 1.9 dialect

A couple of syntax tweaks

Some of the standard lib

Multiplatform

iPhone

iPad

Mac OSX

Testing

MacBacon

RSpec like

Focus on testing



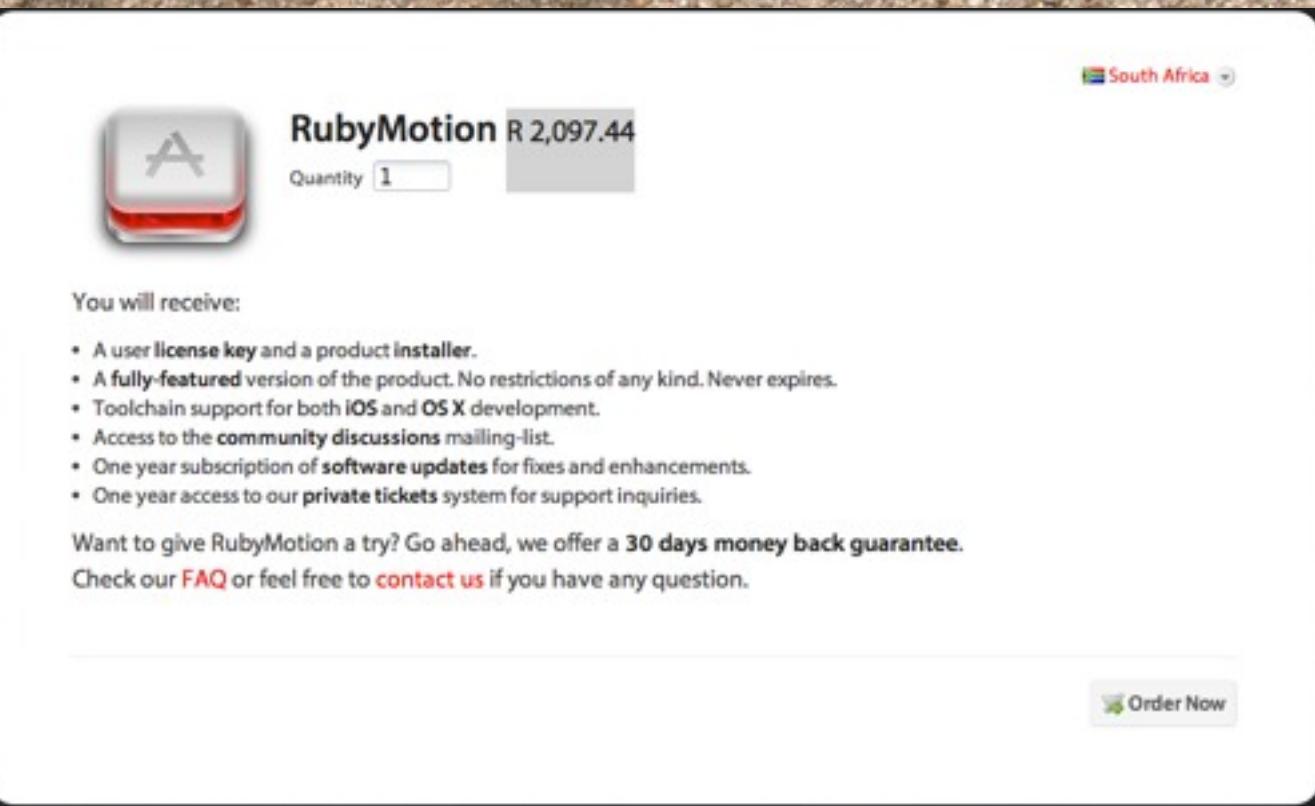
What's the catch?

ROBERT

chan4chan.com

Well...

It's not open source



well...

Some of it's open source

<https://github.com/HipByte/RubyMotion>

Where does it come from?

Where does it come from?

<http://www.hipbyte.com/>



Where does it come from?

RubyCocoa -> MacRuby -> RubyMotion

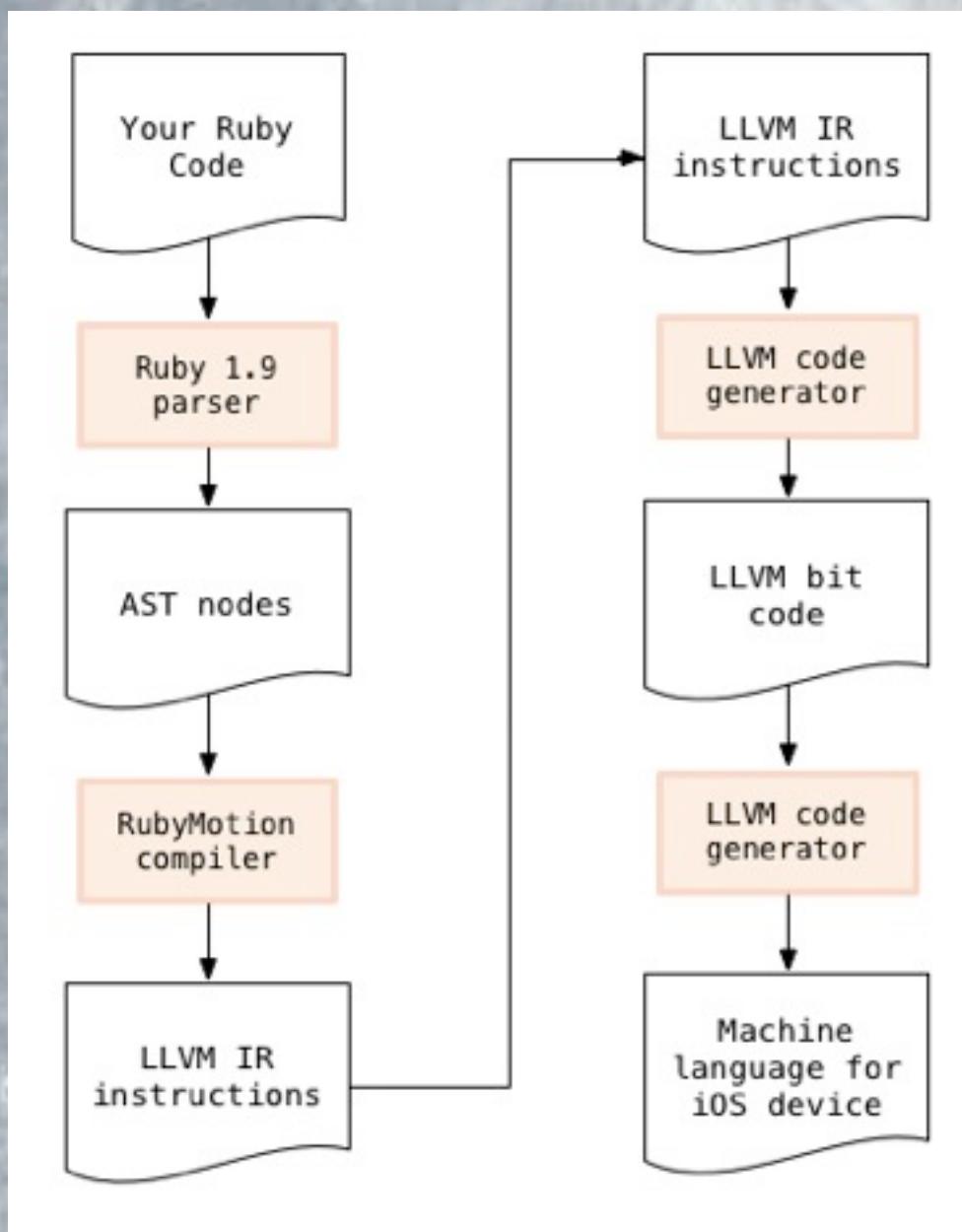
How does it work?

How does it work?

LLVM



How does it work?



Take code like this:

```
- (void)viewDidLoad {
    [super viewDidLoad];
    responseData = [(NSMutableData *)responseData retain];
    NSURLRequest *request = [NSURLRequest requestWithURL:[NSURL URLWithString:@"http://www.example.com/api.json"]];
    [[NSURLConnection alloc] initWithRequest:request delegate:self];
}

- (void)connection:(NSURLConnection *)connection didReceiveResponse:(NSURLResponse *)response {
    [responseData setLength:0];
}

- (void)connection:(NSURLConnection *)connection didReceiveData:(NSData *)data {
    [responseData appendData:data];
}

- (void)connection:(NSURLConnection *)connection didFailWithError:(NSError *)error {
    label.text = [NSString stringWithFormat:@"Connection failed: %@", [error description]];
}

- (void)connectionDidFinishLoading:(NSURLConnection *)connection {
    [connection release];

    NSString *responseString = [(NSString *)responseData initWithData:responseData encoding:NSUTF8StringEncoding];
    [responseData release];

    NSError *error;
    SBJSON *json = [[SBJSON alloc] autorelease];
    NSArray *myArray = [json objectWithString:responseString error:&error];
    [responseString release];

    if (myArray == nil)
        label.text = [NSString stringWithFormat:@"JSON parsing failed: %@", [error localizedDescription]];
    else {
        NSMutableString *text = [NSMutableString stringWithString:@"Array Data:\n"];
        for (int i = 0; i < [myArray count]; i++)
            [text appendFormat:@"%@\n", [myArray objectAtIndex:i]];

        label.text = text;
    }
}
```

Write it like this:

```
require 'net/http'
require 'json'

url = "http://www.example.com/api.json"
response = Net::HTTP.get_response(URI.parse(url))

if response.code.to_i == 200
  json = JSON.parse(response.body)
  puts json.inspect
else
  puts "An Error Occurred (#{response.code}): #{response.body}"
end
```

Demo

Is it worth it?



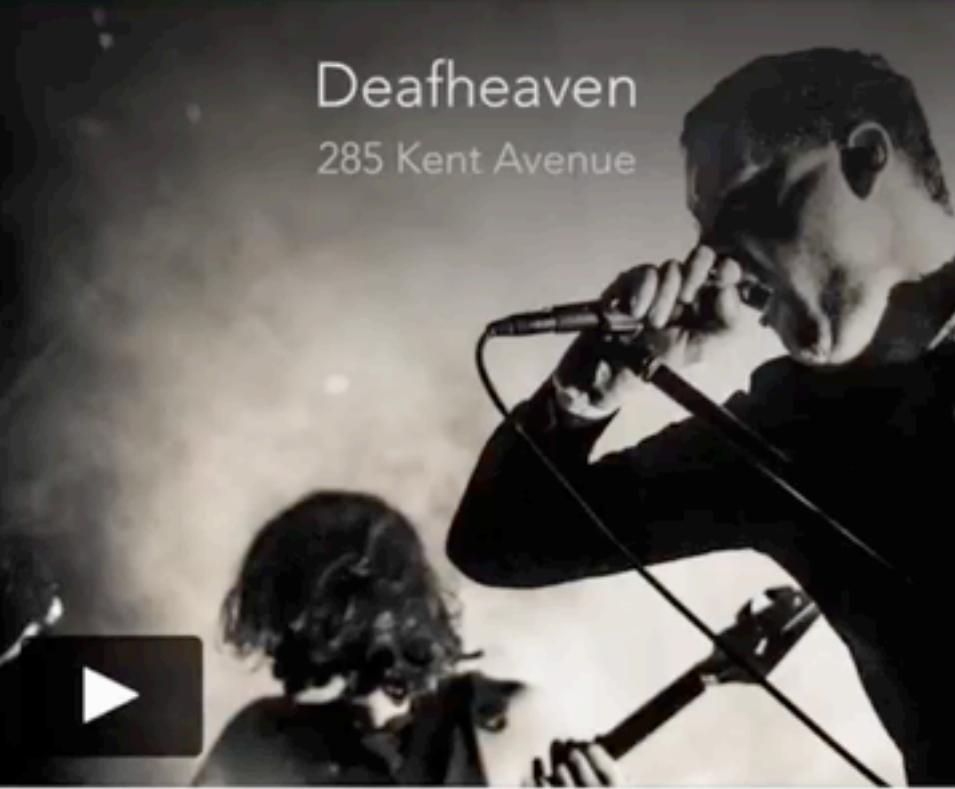
I immediately regret this decision.

<http://jukely.com>

9:41 AM

THURSDAY, JUL 4

Deafheaven
285 Kent Avenue



\$12 | 10:45 PM |  PURCHASE

Recommended to 3 friends and 4 more



+2

BAND TOGETHER

Matching your synced genres

<http://thetempleapp.com/>

The image shows two smartphones side-by-side, both displaying the "temple" app interface. The left phone shows a "Progress" screen with a large blue "0" indicating a streak of 10+ days in a row. Below this is a bar chart for "months" showing activity levels for Jan, Feb, Mar, Apr, and May. The right phone shows a "Today" screen with sections for "Fitness" (Quick, Light, Moderate, Intense), "Fluids" (Half Glass, Glass, Bottle, Large Bottle), and "Fuel" (Snack, Small Meal, Full Meal, Large Meal). Each section has a circular entry button with a number (e.g., 3, 2, 1) and a plus sign. Progress bars at the top of each section indicate completion levels (e.g., 113% for Fluids, 65% for Fuel).

Temple

Hydration, Food, Fitness
and More

Download on the App Store

The health and fitness tracker so easy and fun you'll actually use it.

One tap entry and a cool, fresh design.
Record your daily activities quickly and get back to your life

Get fit, hydrate more, drop weight, etc.

[Features](#) [Screenshots](#)

177 [Like](#)

37signals Basecamp





Some awesome gems

Bubblewrap

<http://bubblewrap.io>

```
> App.documents_path  
# "/Users/mattetti/Library/Application Support/iPhone Simulator/5.0/Applications/EEC6454E-1816-451.  
> App.resources_path  
# "/Users/mattetti/Library/Application Support/iPhone Simulator/5.0/Applications/EEC6454E-1816-451.  
> App.name  
# "testSuite"  
> App.identifier  
# "io.bubblewrap.testSuite"  
> App.alert("BubbleWrap is awesome!")  
# creates and shows an alert message.  
> App.run_after(0.5) { p "It's #{Time.now}" }  
# Runs the block after 0.5 seconds.  
> App.open_url("http://matt.aimonetti.net")  
# Opens the url using the device's browser. (accepts a string url or an instance of `NSURL`.  
> App::Persistence['channels'] # application specific persistence storage  
# ['NBC', 'ABC', 'Fox', 'CBS', 'PBS']  
> App::Persistence['channels'] = ['TF1', 'France 2', 'France 3']  
# ['TF1', 'France 2', 'France 3']
```

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

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

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

```
def viewWillAppear(animated)
  @foreground_observer = App.notification_center.observe UIApplicationWillEnterForegroundNotification { |notification|
    loadAndRefresh
  }
end

@reload_observer = App.notification_center.observe ReloadNotification do |notification|
  loadAndRefresh
end
end

def viewWillDisappear(animated)
  App.notification_center.unobserve @foreground_observer
  App.notification_center.unobserve @reload_observer
end

def reload
  App.notification_center.post ReloadNotification
end
```

Sugarcube

<https://github.com/rubymotion/sugarcube>

Promotion

<https://github.com/clearsightstudio/ProMotion>

```
class AppDelegate < PM::Delegate
  def on_load(app, options)
    open RootScreen.new(nav_bar: true)
  end
end

class RootScreen < PM::Screen
  title "Root Screen"

  def push_new_screen
    open NewScreen
  end
end

class NewScreen < PM::TableScreen
  title "Table Screen"

  def table_data
    [
      {
        cells: [
          { title: "About this app", action: :tapped_about },
          { title: "Log out", action: :log_out }
        ]
      }
    ]
  end
end
```

Walt

<https://github.com/clayallsopp/Walt>

```
@view = UIView.alloc.initWithFrame(...)

Walt.animate(
  assets: [
    id: "logo",
    position: [100, 0],
    size: [110, 40],
    url: "http://bit.ly/S98Ta5"
  ],
  animations: [
    duration: 2,
    operations: [
      move: "logo",
      to: 150,
      axis: :y
    ],
    after: {
      duration: 2,
      operations: [
        rotate: "logo",
        to: 360
      ]
    }
  ],
  in: @view
)
```

Joybox

<https://joybox.io>

Questions?

@michael_erasmus

motioncasts.tv