

# When To Go Native Over Javascript

*Harry Tormey*

What's this talk about?

# What's this talk about?

- Tooling
- Third Party Libraries
- Writing Native Code

# Tooling

# Expo

# What is Expo?

*“Tools that enable developers to build and share truly native apps that work across both iOS and Android.”*

# How does Native Code affect expo?

*“Standard Expo projects don’t support custom native code, including third-party libraries which require custom native components. In an Expo project, you only write pure JS.”*

# CodePush

# What is CodePush?

*“A cloud service that enables React Native developers to deploy mobile app updates directly to their users’ devices.”*

# How does Native Code affect CodePush?

*“Any product changes which touch native code (e.g. modifying your AppDelegate.m/MainActivity.java file, adding a new plugin) cannot be distributed via CodePush, and therefore, must be updated via the appropriate store(s)..”*

# Third Party Libraries

# Navigation

# Which Navigation library should I use?

- React Navigation
- React Native Navigation
- React Native Router Flux
- React Router Native
- Native Navigation

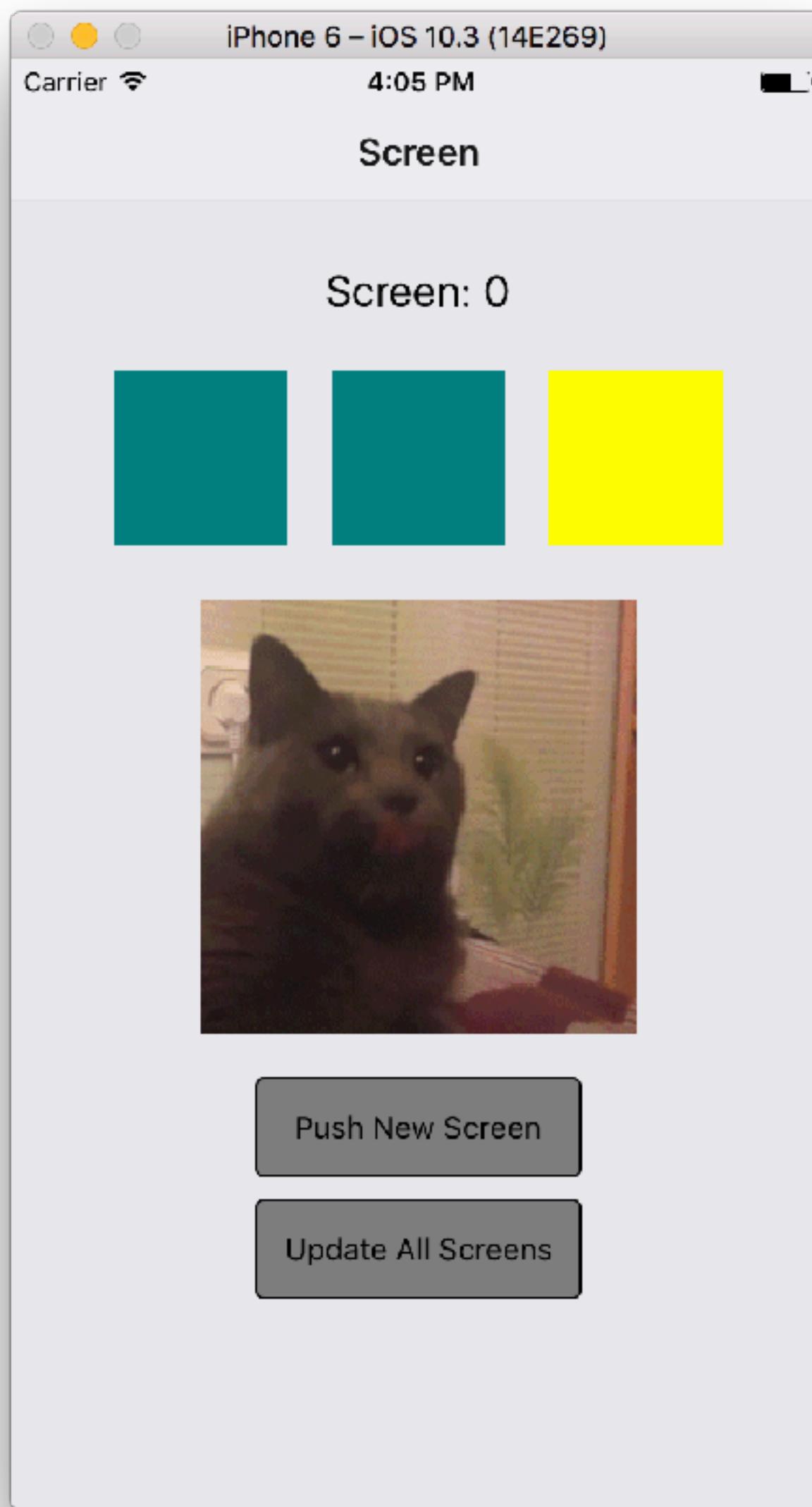
# Native

- React Native Navigation
- Native Navigation

# Javascript

- React Navigation
- React Native Router Flux
- React Router Native

# Example App

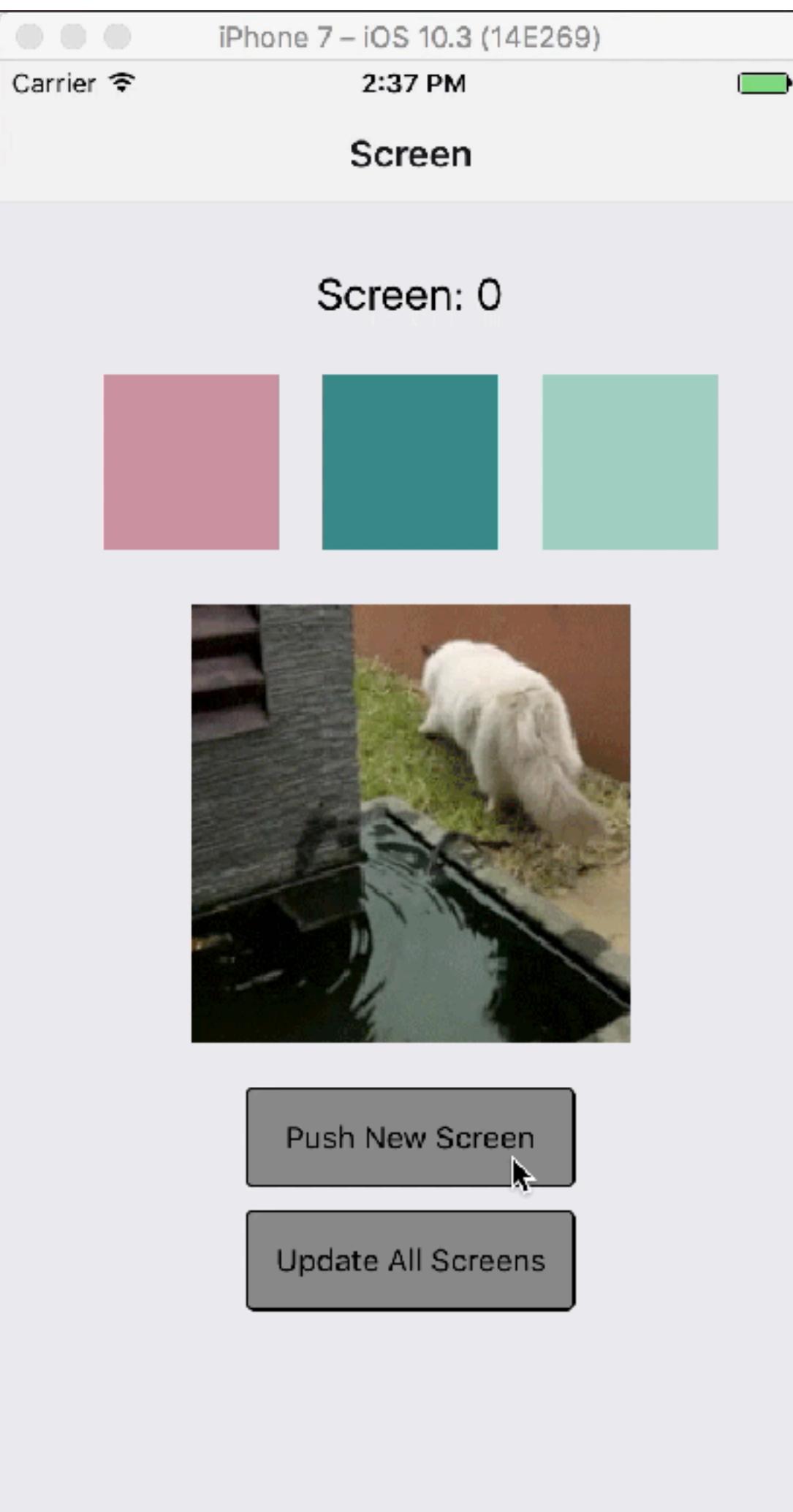


# Example App

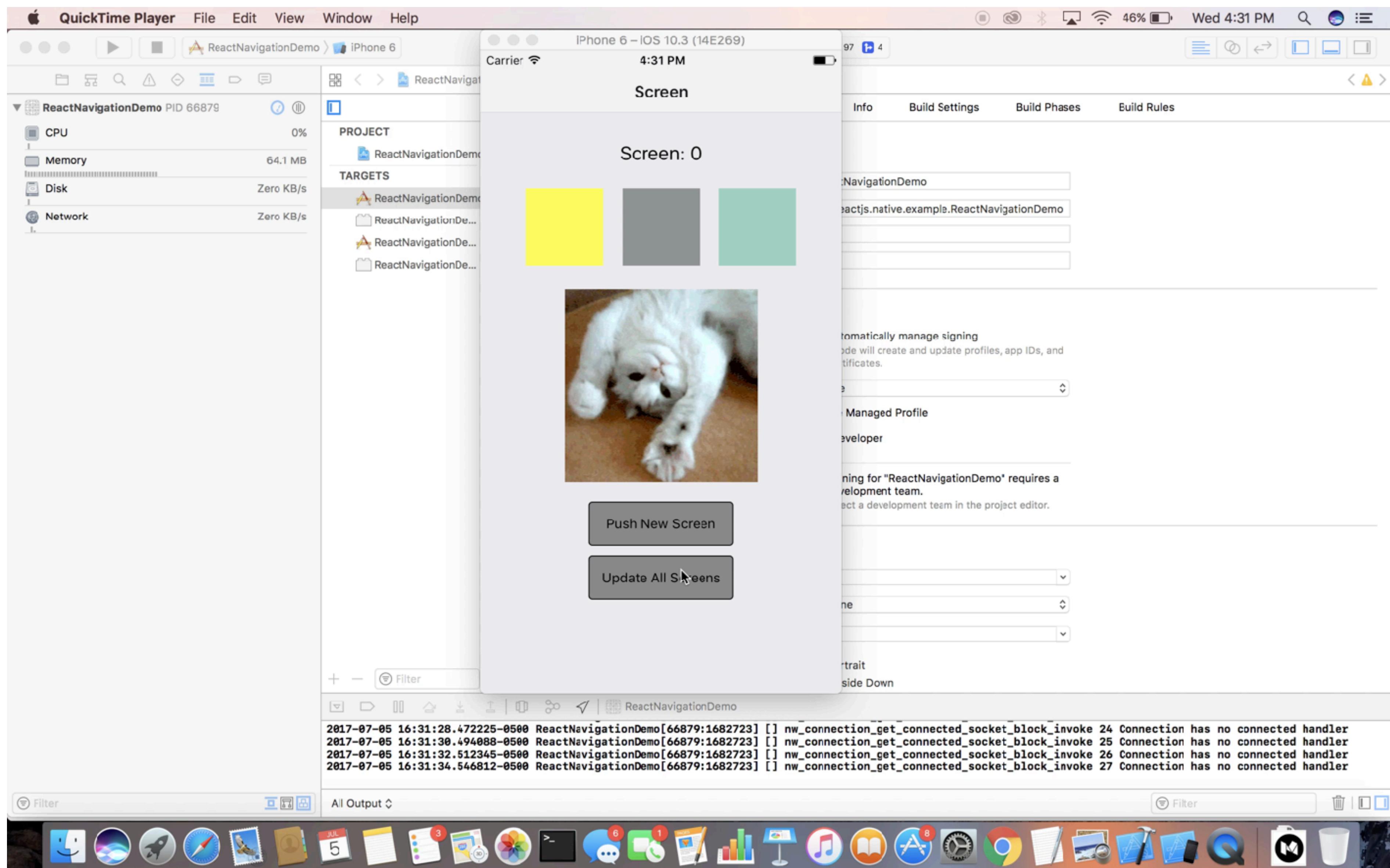
- React Navigation:
  - <https://github.com/hgale/ReactNavigationDemo>
- React Native Navigation:
  - <https://github.com/hgale/ReactNativeNavigationDemo>

**What does it do?**

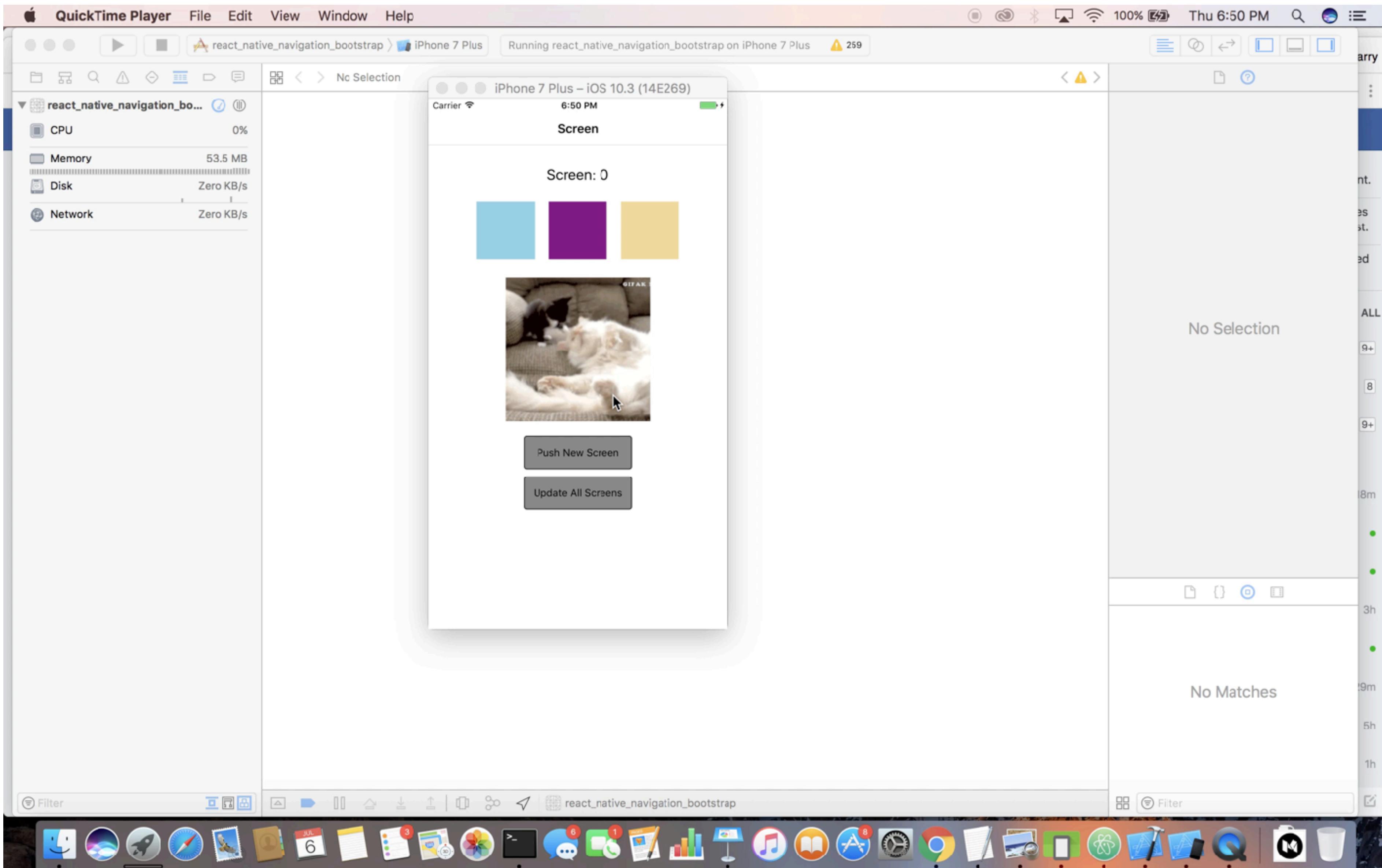
# Example App



# React Navigation



# React Native Navigation



**Why does this matter?**

# Accessibility

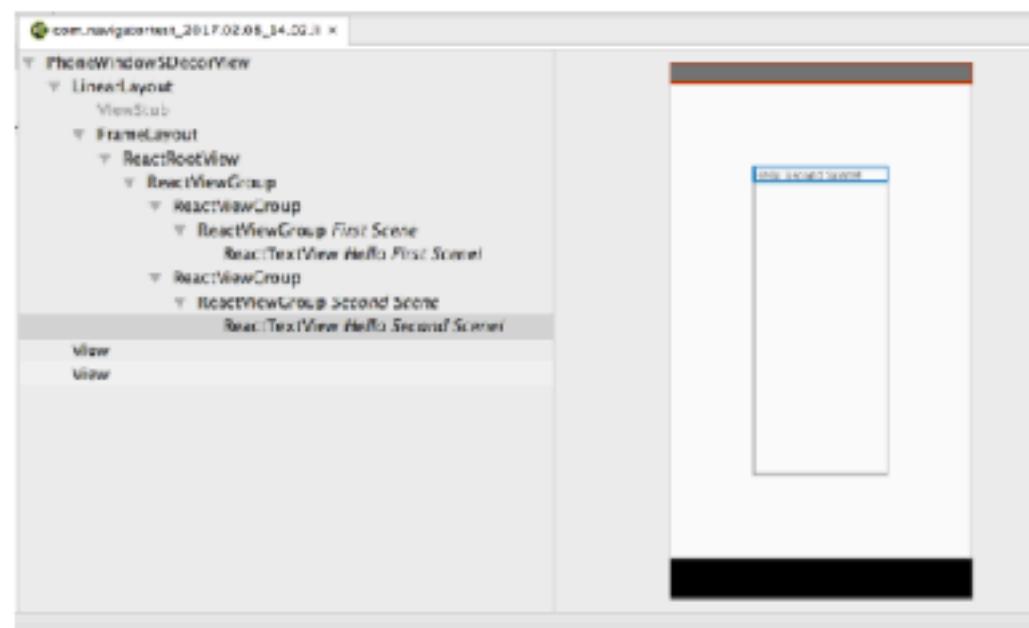
## 1 Navigator breaks Android screen reader [accessibility]



Erica Cooksey

When a new scene is pushed onto the Navigator, the previous scene is still present in the view tree. Thus, if a visually-impaired user enables screen reading through their device accessibility settings, views from both the previous and the current scene are read interleaved to the user. As an aside, rendering all components on the backstack creates significant GPU overdraw issues.

Attached is a screenshot of the "Additional Scenes" sample code in Android's layout inspector. The only change that I made was adding `accessibilityLabel`s to each scene ("First Scene" and "Second Scene", respectively).



# Accessibility

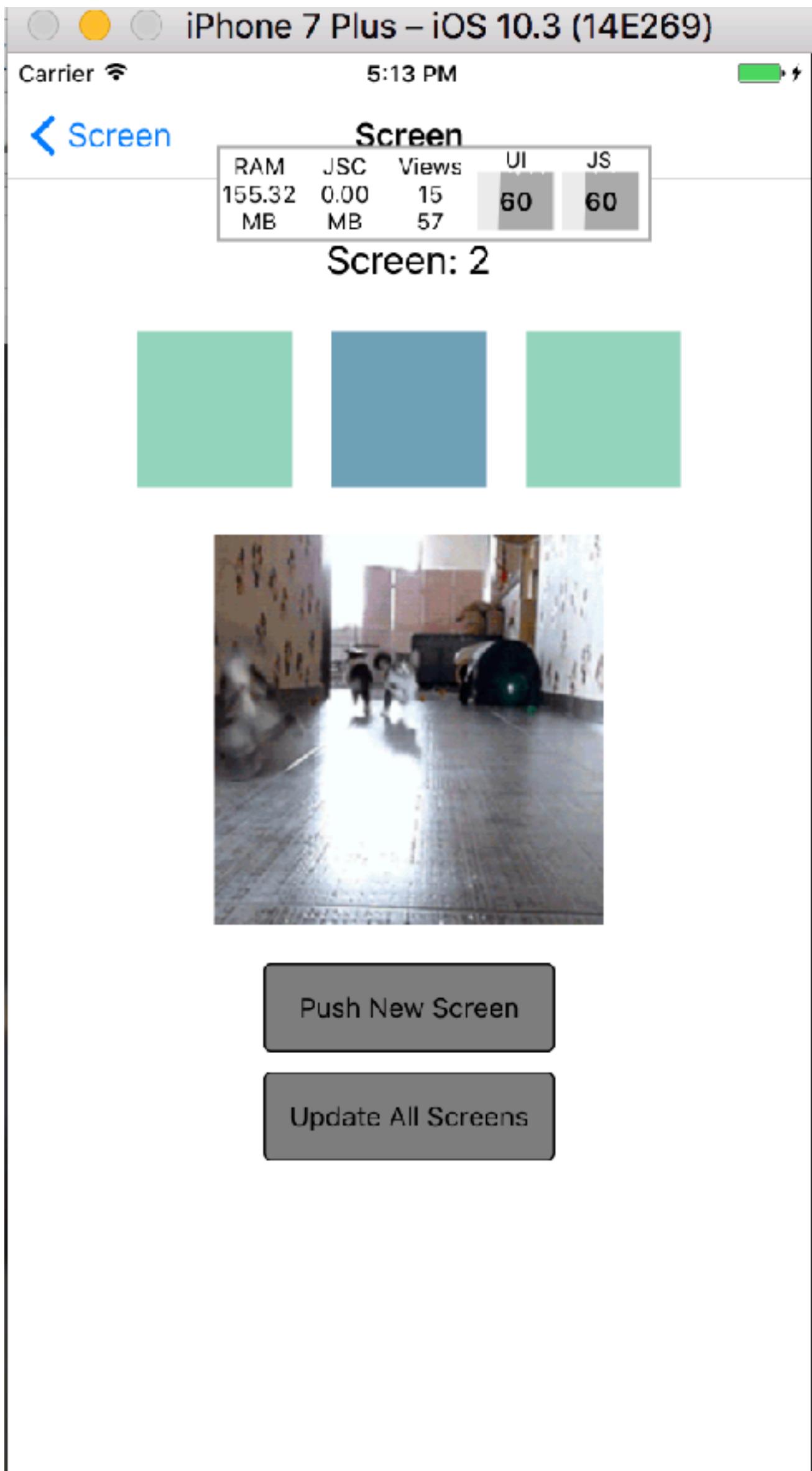
- <https://react-native.canny.io/feature-requests/p/navigator-breaks-android-screen-reader-accessibility>

**Whats the difference?**

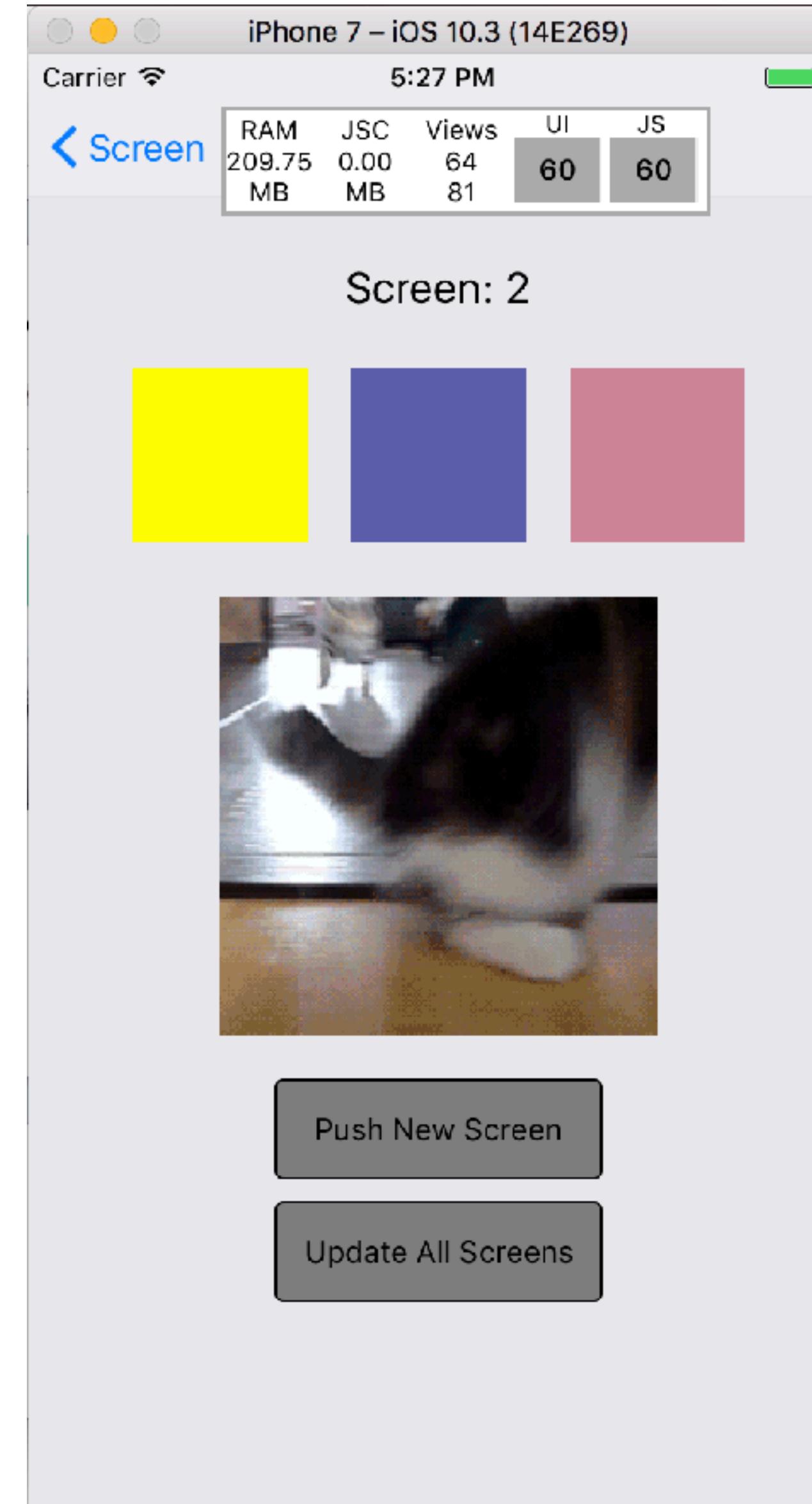
# Native Feel

# Performance

# React Native Navigation



# React Navigation



# Native Libraries Pros

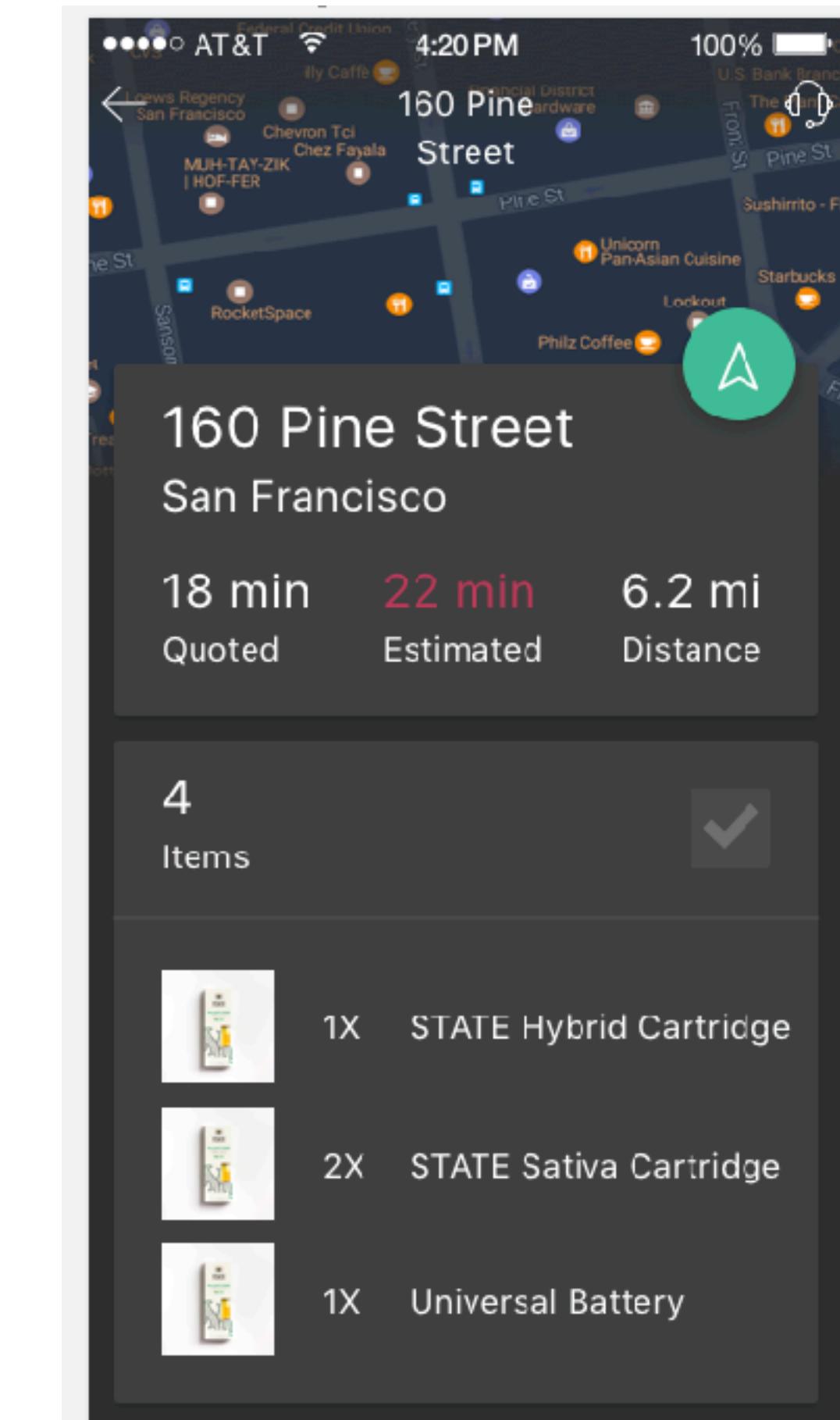
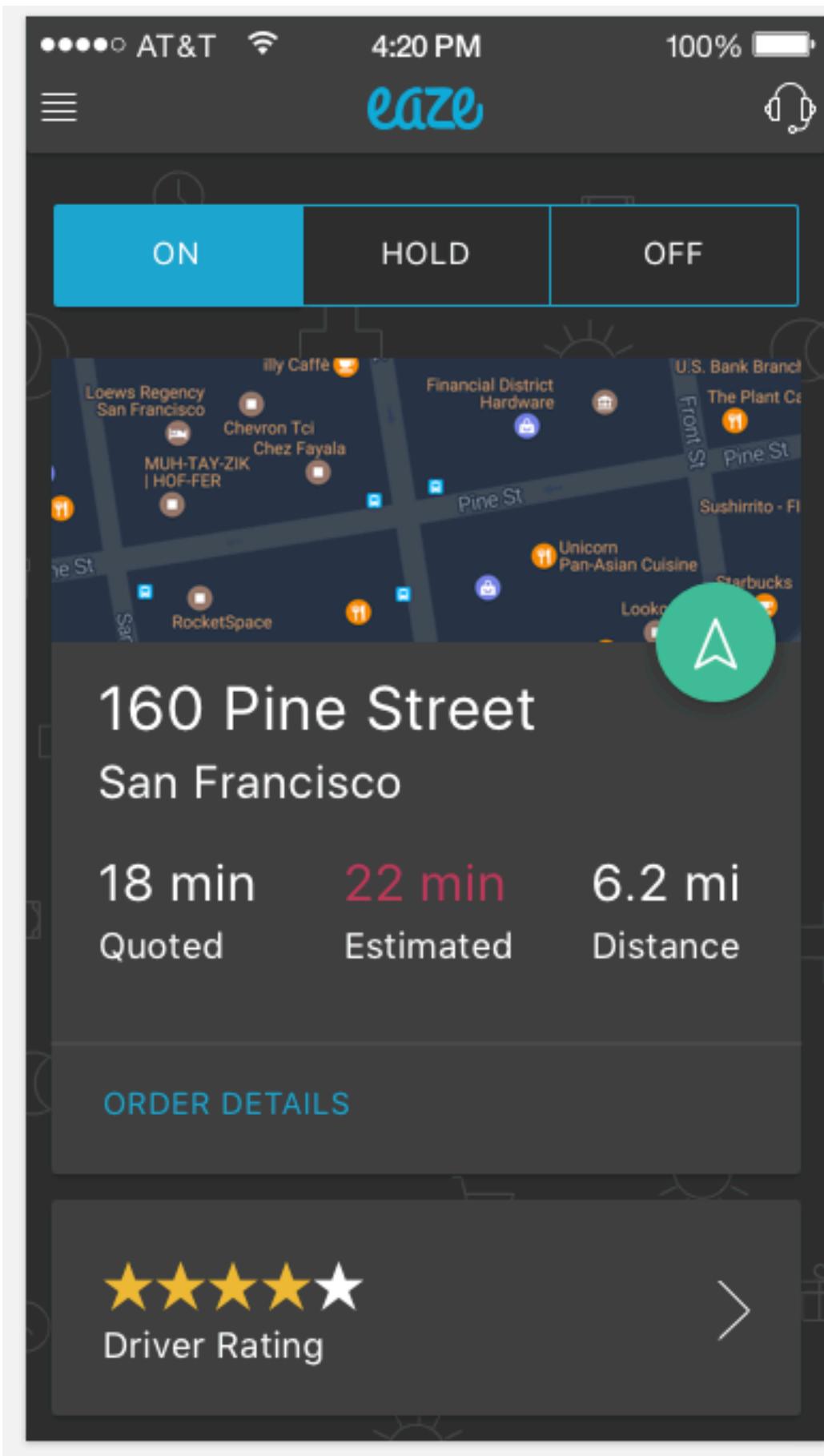
- Performance
- Feels/Behaves “Natively”

# Native Libraries Cons

- Can Require Some Native Knowledge
  - Setup can be tricky
  - Native linking/build errors
  - iOS/Android behave differently

# Native Code

# What does the Driver app do?





VAPORIZERS EDIBLES PREROLLS FLOWERS CONCENTRATES ACCESSORIES



Sativa Hybrid Indica CBD

## Vaporizers



Bloom Farms - Vape Pen Bat..



\$22



Bloom Farms Blueberry Haze

23.3% THC | 50% CBD



\$52



AbsoluteXtracts  
Grape Stomper Solventless

100% Super Critical CO<sub>2</sub> Oil



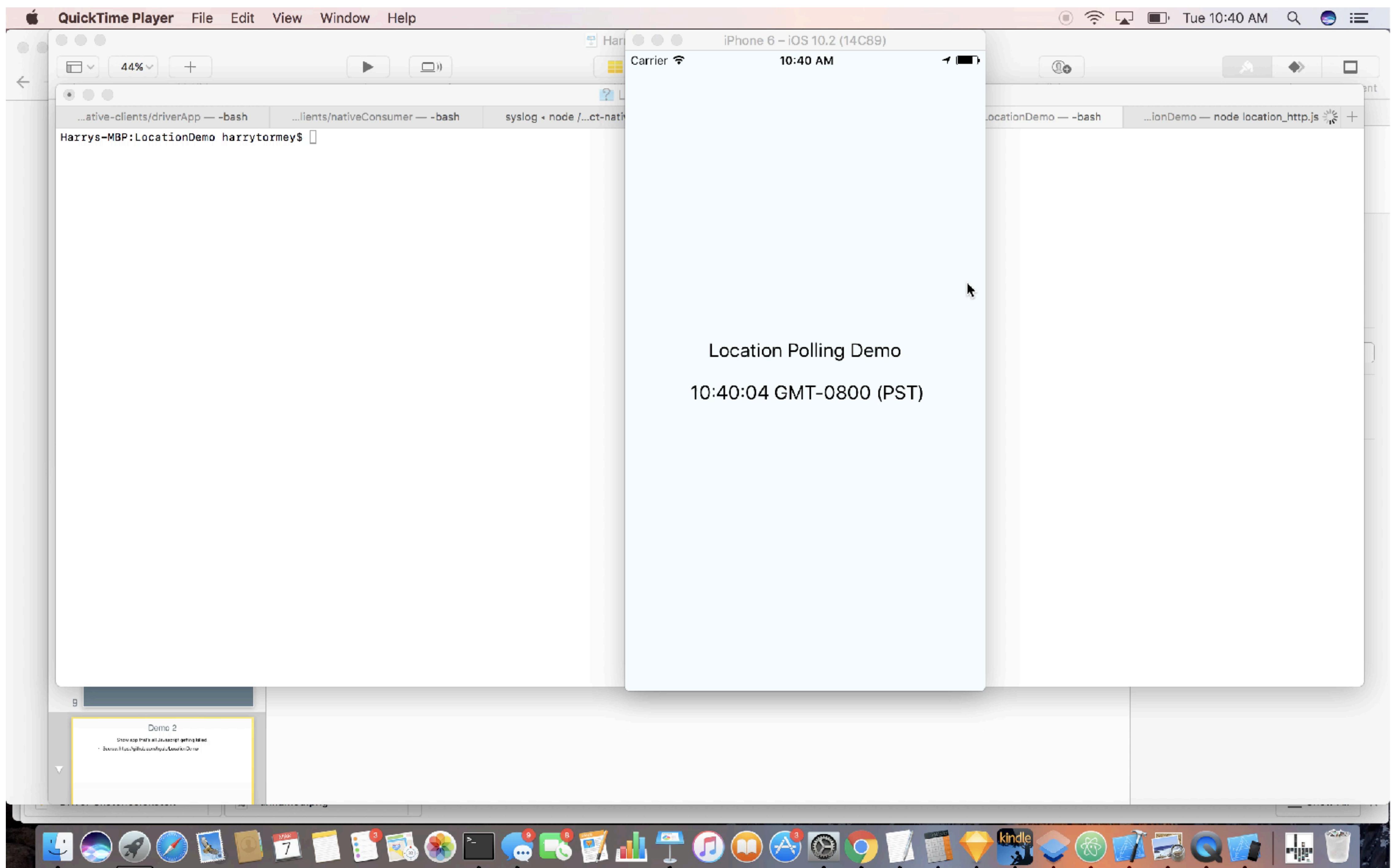
\$45



160 Pine St



# Why not Javascript?



# Demo Source

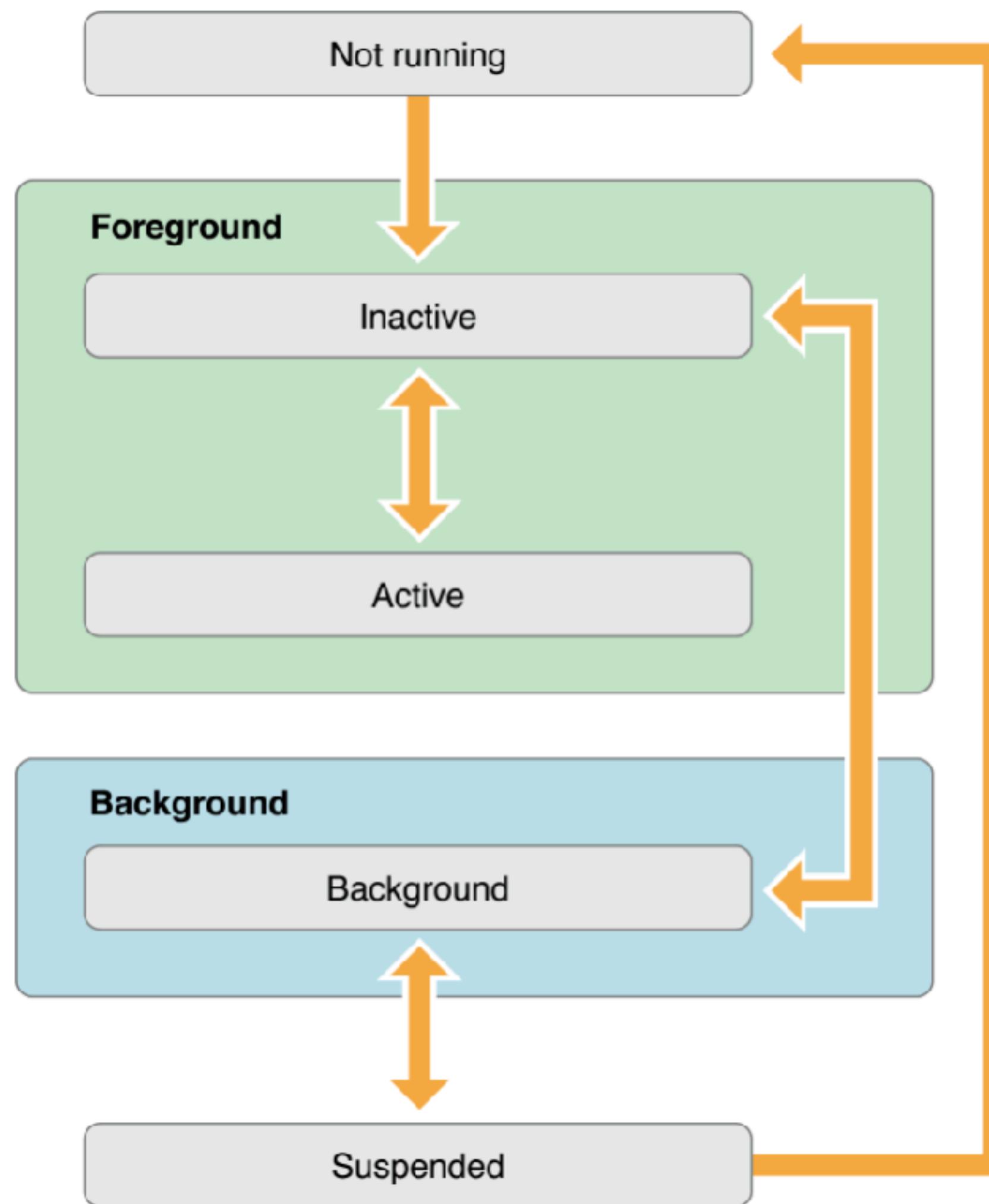
- <https://github.com/hgale/LocationDemo>

# Why not?

```
1 import pole from 'pole'
2
3 this.setState({
4   pollLocation: pole({interval:
5     LOCATION_POLL_FREQUENCY}, (
6     callback) => {
7   let now = new Date()
8   let data = JSON.stringify({'lat': 20,
9     'long': 20, 'time': now })
10  fetch('http://localhost:8080', {
11    method: 'POST',
12    headers: {
13      'Accept': 'application/json',
14      'Content-Type': 'application/json',
15    },
16    body: data
17  })
18  this.updateTime(now)
19  callback()
20 })
21 })
```

# iOS Background Modes

Figure 2–3 State changes in an iOS app



# iOS Background Modes

State	Description
Not running	The app has not been launched or was running but was terminated by the system.
Inactive	The app is running in the foreground but is currently not receiving events. (It may be executing other code though.) An app usually stays in this state only briefly as it transitions to a different state.
Active	The app is running in the foreground and is receiving events. This is the normal mode for foreground apps.
Background	The app is in the background and executing code. Most apps enter this state briefly on their way to being suspended. However, an app that requests extra execution time may remain in this state for a period of time. In addition, an app being launched directly into the background enters this state instead of the inactive state. For information about how to execute code while in the background, see <a href="#">Background Execution</a> .
Suspended	The app is in the background but is not executing code. The system moves apps to this state automatically and does not notify them before doing so. While suspended, an app remains in memory but does not execute any code. When a low-memory condition occurs, the system may purge suspended apps without notice to make more space for the foreground app.

# Testing

- Test on a Device
- Test with app backgrounded and screen locked
- Test with app backgrounded for ~8 minutes on iOS
- Test 10-20 minutes on Android
- Test with device unplugged and not on wifi

# What about react-native-background-timer?

```
1
2 import BackgroundTimer from 'react-native-background-timer'
3
4 const intervalId = BackgroundTimer.setInterval(() => {
5   let now = new Date()
6   let data = JSON.stringify({ 'lat': 20, 'long': 20, 'time': now })
7   fetch('http://192.168.107.24:8080', {
8     method: 'POST',
9     headers: {
10       'Accept': 'application/json',
11       'Content-Type': 'application/json',
12     },
13     body: data
14   })
15   this.updateTime(now)
16 }, LOCATION_POLL_FREQUENCY)
```

# No

- On iOS Based on Executing Finite-Length Tasks.
- See: [https://github.com/ocetnik/react-native-background-timer/  
blob/master/ios/RNBackgroundTimer.m](https://github.com/ocetnik/react-native-background-timer/blob/master/ios/RNBackgroundTimer.m)
- How much time you get after your app gets backgrounded is determined is not guaranteed. App will be killed after roughly ~3-5 minutes.

# iOS Background Modes

## ▼ **Background Modes**

**ON**

- Modes:
- Audio, AirPlay, and Picture in Picture
  - Location updates
  - Voice over IP
  - Newsstand downloads
  - External accessory communication
  - Uses Bluetooth LE accessories
  - Acts as a Bluetooth LE accessory
  - Background fetch
  - Remote notifications

---

Steps:  Add the Required Background Modes key to your info plist file

---

# iOS Background Modes

## NSLocationAlwaysUsageDescription

`NSLocationAlwaysUsageDescription` (String – iOS) This key lets you describe the reason your app accesses the user's location information at all times. Include this key when your app uses location services in a potentially nonobvious way while running in the foreground or the background. For example, a social app might include this key when it uses location information to track the user's location and display other users that are nearby. In this case, the fact that the app is tracking the user's location might not be readily apparent. The system includes the value of this key in the alert panel displayed to the user when requesting permission to use location services.

**Important:** To protect user privacy, an iOS app linked on or after iOS 10.0, and which accesses the user's location information, must statically declare the intent to do so. Include the `NSLocationAlwaysUsageDescription` key in your app's `Info.plist` file and provide a purpose string for this key. If your app attempts to access the user's location information without a corresponding purpose string, your app exits.

This key is required when you use the `requestAlwaysAuthorization` method of the `CLLocationManager` class to request authorization for location services. If this key is not present and you call the `requestAlwaysAuthorization` method, the system ignores your request and prevents your app from using location services.

This key is supported in iOS 8.0 and later.

# iOS Background Modes

The screenshot shows the Xcode interface with the project 'LocationDemo' selected. The main window displays the 'Info.plist' file under the 'LocationDemo' group. The 'Information Property List' section contains various key-value pairs. One key, 'Privacy - Location When In Use Usage Description', has its value set to 'We need your location in order to post it'. This configuration is essential for enabling background modes related to location services.

Key	Type	Value
Localization native development region	String	en
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	org.reactjs.native.example.\$(PRODUCT_NAME:rfc1034identifier)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle creator OS Type code	String	????
Bundle version	String	1
Application requires iPhone environment	Boolean	YES
► App Transport Security Settings	Dictionary	(1 item)
Privacy - Location When In Use Usage Description	String	We need your location in order to post it
► Required background modes	Array	(1 item)
Launch screen interface file base name	String	LaunchScreen
► Required device capabilities	Array	(1 item)
► Supported interface orientations	Array	(3 items)
View controller-based status bar appearance	Boolean	NO

```
1 - (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)
2 launchOptions {
3     self.locationManager = [[CLLocationManager alloc] init];
4     self.locationManager.pausesLocationUpdatesAutomatically = NO;
5     self.locationManager.allowsBackgroundLocationUpdates = YES;
6     self.locationManager.delegate=self;
7     self.locationManager.desiredAccuracy=kCLLocationAccuracyBest;
8     [self.locationManager requestAlwaysAuthorization];
9     return YES;
10 }
11
12 #pragma - CLLocationManagerDelegate
13
14 - (void)locationManager:(CLLocationManager *)manager didChangeAuthorizationStatus:
15 CLAuthorizationStatus)status {
16     if(status==kCLAuthorizationStatusAuthorizedAlways)
17         [self.locationManager startUpdatingLocation];
18 }
19
20 - (void)locationManager:(CLLocationManager *)manager didUpdateLocations:(NSArray *)locations {
21     // Emit event
22 }
23
```

# Does this work now?

```
1 import pole from 'pole'
2
3 this.setState({
4   pollLocation: pole({interval:
5     LOCATION_POLL_FREQUENCY}, (
6     callback) => {
7   let now = new Date()
8   let data = JSON.stringify({'lat': 20,
9     'long': 20, 'time': now })
10  fetch('http://localhost:8080', {
11    method: 'POST',
12    headers: {
13      'Accept': 'application/json',
14      'Content-Type': 'application/json',
15    },
16    body: data
17  })
18  this.updateTime(now)
19  callback()
20 })
21 })
```

# No

React Native Timers don't work in the background

- See Brent Vatne's feature request: <https://react-native.canny.io/feature-requests/p/background-timer-execution>

```

1 RCT_EXPORT_METHOD(startPostingLocationTo:(NSString *)url everyInterval:(NSInteger)interval) {
2     if (!url || url.length == 0 || interval <= 0) {
3         return;
4     }
5
6     self.url = [NSURL URLWithString:url];
7
8     double seconds = interval / 1000.0;
9
10    if (self.pollingTimer) {
11        [self stopPostingLocation];
12    }
13
14    self.pollingTimer = [NSTimer scheduledTimerWithTimeInterval:seconds target:self selector:@selector(postLocation:) userInfo:nil
15                                         repeats:YES];
16 }
17
18 - (NSArray<NSString *> *)supportedEvents {
19     return @[@"LocationUpdated"];
20 }
21
22 RCT_EXPORT_METHOD(stopPostingLocation) {
23     [self.pollingTimer invalidate];
24     self.pollingTimer = nil;
25 }
26
27 - (void)postLocation:(NSTimer *)timer {
28     if (!self.url && !self.currentLocation) {
29         return;
30     }
31
32     NSMutableURLRequest *request = [[NSMutableURLRequest alloc] initWithURL:self.url];
33     [request addValue:@"application/json" forHTTPHeaderField:@"Content-Type"];
34     [request addValue:@"application/json" forHTTPHeaderField:@"Accept"];
35     request.HTTPMethod = @"POST";
36
37     NSDateFormatter *dateFormat = [[NSDateFormatter alloc] init];
38     [dateFormat setDateFormat:@"yyyy-MM-dd'T'HH:mm:ss.SSS"];
39     NSString *dateString = [dateFormat stringFromDate:[NSDate date]];
40
41     NSDictionary *params = @{@"lat":@(self.currentLocation.coordinate.latitude), @"long":@(self.currentLocation.coordinate.longitude),
42                             @"time":dateString};
43     NSError *error = nil;
44     NSData *data = [NSJSONSerialization dataWithJSONObject:params
45                                         options:kNilOptions error:&error];
46     if (!data || error) {
47         return;
48     }
49
50     [request setHTTPBody:data];
51     NSURLSessionDataTask *postDataTask = [self.session dataTaskWithRequest:request
52                                         completionHandler:^(NSData *data, NSURLResponse *response, NSError *requestError)
53                                         {
54         NSHTTPURLResponse *httpResponse = (NSHTTPURLResponse *) response;
55         if (requestError || [httpResponse statusCode] != 200) {
56             return;
57         }
58         [self sendEventWithName:@"LocationUpdated" body:params];
59     }];
60     [postDataTask resume];
61 }

```

# Things To Note

- 90% of the DriverApp is Javascript
- Code Push doesn't support Native code
  - <https://github.com/Microsoft/react-native-code-push>

*Note: Any product changes which touch **native** code (e.g. modifying your `AppDelegate.m` / `MainActivity.java` file, adding a new plugin) cannot be distributed via CodePush, and therefore, must be updated via the appropriate store(s).*

eaze

# Contact Me:

Email: [harry.n.gale@gmail.com](mailto:harry.n.gale@gmail.com)

Medium: <http://launchdrawer.com/>

Twitter: [@htormey](https://twitter.com/htormey)