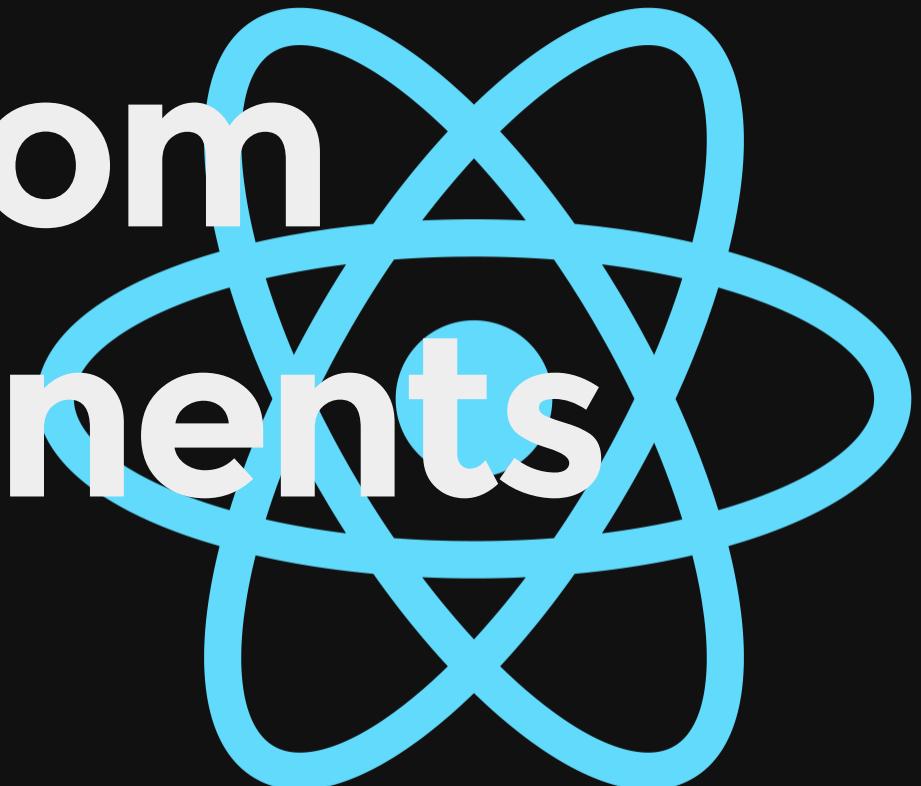


React-Native custom components



Who are you?



Jeremy Grancher

- Front-end developer since 4 years
- iOS developer for a year ¯_(ツ)_/¯
- Enjoying work at Holler
- Loving kittens and Game of Thrones

 <http://twitter.com/jgrancher>

 <http://github.com/jgrancher>

What are you talking about?

- The current state of the React-Native components
- The need of creating a native component
- How to do it
- Lessons learned from creating react-native-sketch

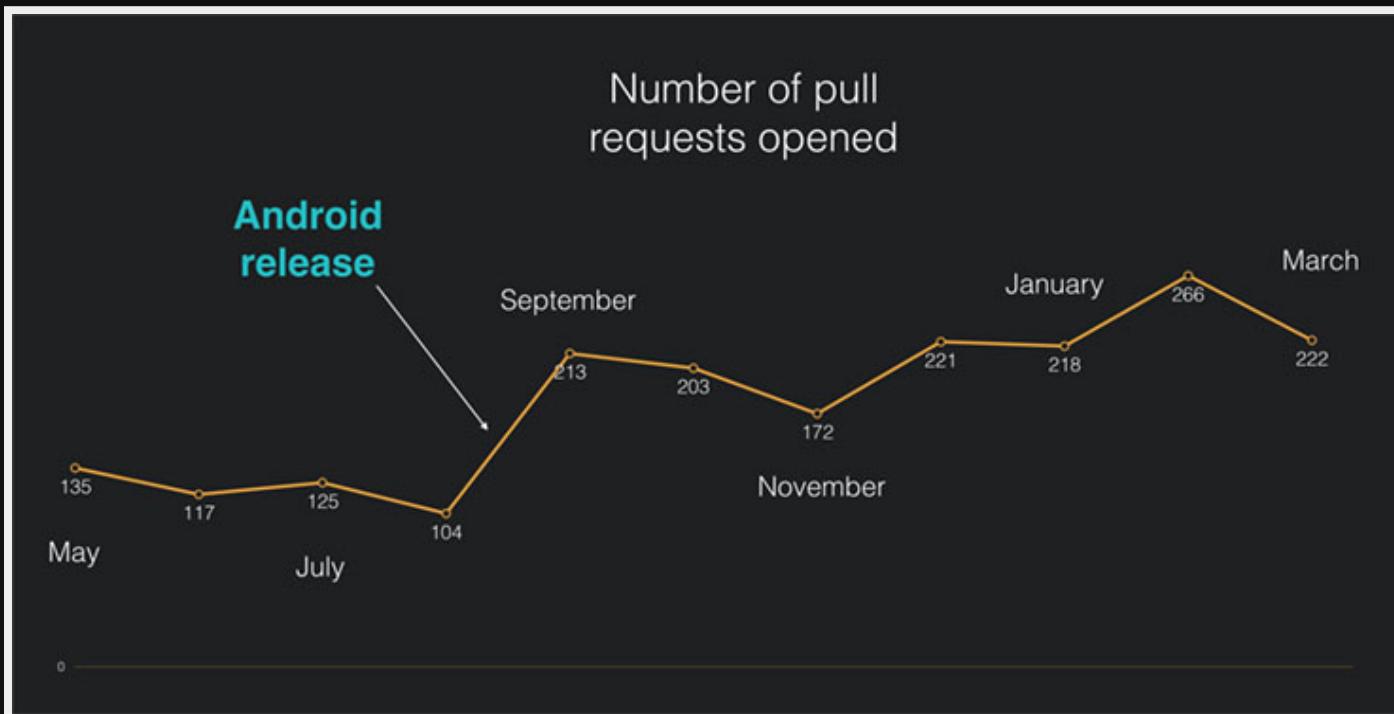
The {this.state.components}

Good pun, eh?



The growth

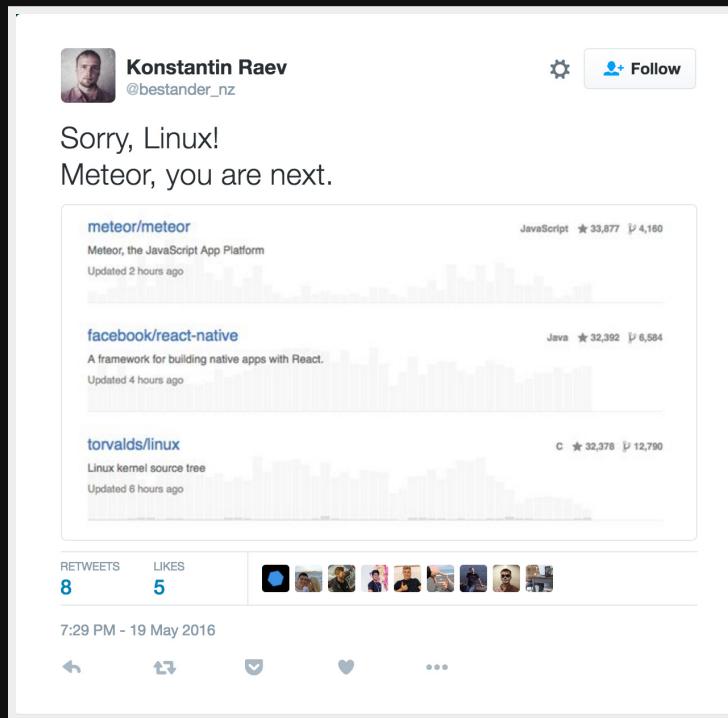
Rapid adoption...



Pull requests / month over the year

The growth

...and still ongoing



The spread doesn't slow down



The growth

The core components are awesome...

- ActivityIndicatorIOS
- DatePickerIOS
- DrawerLayoutAndroid
- Image
- ListView
- MapView
- Modal
- **Navigator**
- **NavigatorIOS**
- PickerIOS
- Picker
- ProgressBarAndroid
- ProgressViewIOS
- RefreshControl
- ScrollView
- SegmentedControlIOS
- Slider
- SliderIOS
- StatusBar
- Switch
- TabBarIOS
- TabBarIOS.Item
- Text
- TextInput
- ToolbarAndroid
- TouchableHighlight
- TouchableNativeFeedback
- TouchableOpacity
- TouchableWithoutFeedback
- View
- ViewPagerAndroid
- WebView

The Growth

...and still evolving

NavigationExperimental

Summary:
A new API to unify internal navigation. Also addresses a highly-rated community 'pain':
<https://productpains.com/post/react-native/better-navigator-api-and-docs/>

Offers the following improvements:

- Redux-style navigation logic is easy to reason about
- Navigation state can be easily saved and restored through refreshes
- Declarative navigation views can be implemented in native or JS
- Animations and gestures are isolated and now use the Animated library

public

Reviewed By: hedgerwang

Differential Revision: D2798048

fb-gh-sync-id: 88027ef9ead8a80afa38354252bc377455cc6dbb

ψ master (#1) ⚡ v0.27.0-rc2 ... 0.22.0-rc

 ericvicenti committed with facebook-github-bot-9 on Feb 6 1 parent 2f73ad0 commit a3085464f6ea36fc6b53cd0c711c048ffb1516f9

(↗ °□°) ↗ _ ━ ━)

The Growth



Some explanation:

- The first commit code
- On the docs: Navigator comparison
- First look: React-Native Navigator Experimental



The Growth

New features and enhancements

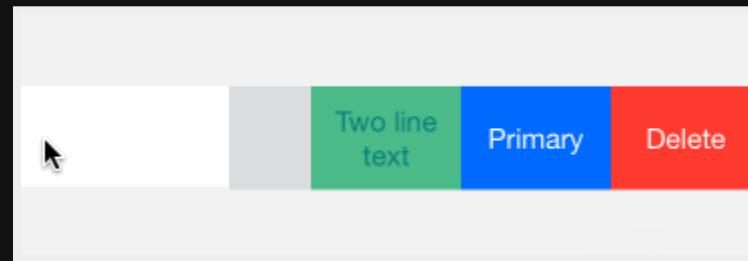
- Animated: Implement toJSON method of Animated ([17f0807](#)) - [@wusuopu](#)
- Make RefreshControl properly controlled by JS ([9aa37e8](#)) - [@janicduplessis](#)
- Add SwipeableListView ([052cd7e](#) [9b1a3c7](#) [2a7f6ae](#)) - [@fred2028](#)
- Expose a new prop `transition` for scene renderer. ([55c3086](#)) - [@hedgerwang](#)
- Add support for custom files in BugReporting ([610cfdc](#)) - [@lexs](#)
- Add "Open file" button to elements inspector ([f203c5d](#)) - [@frantic](#)
- Add `transform-react-jsx-source` to react-native preset ([858643d](#)) - [@frantic](#)
- Fork NavigationAnimatedView to NavigationTransitioner ([7db7f78](#)) - [@hedgerwang](#)
- Merge rnpm into react-native ([149d0b9](#)) - [@grabbou](#)

RNPM though.



The Growth

Experimental swipeable list view?



Capture from [react-native-swipeable](#)

The Growth

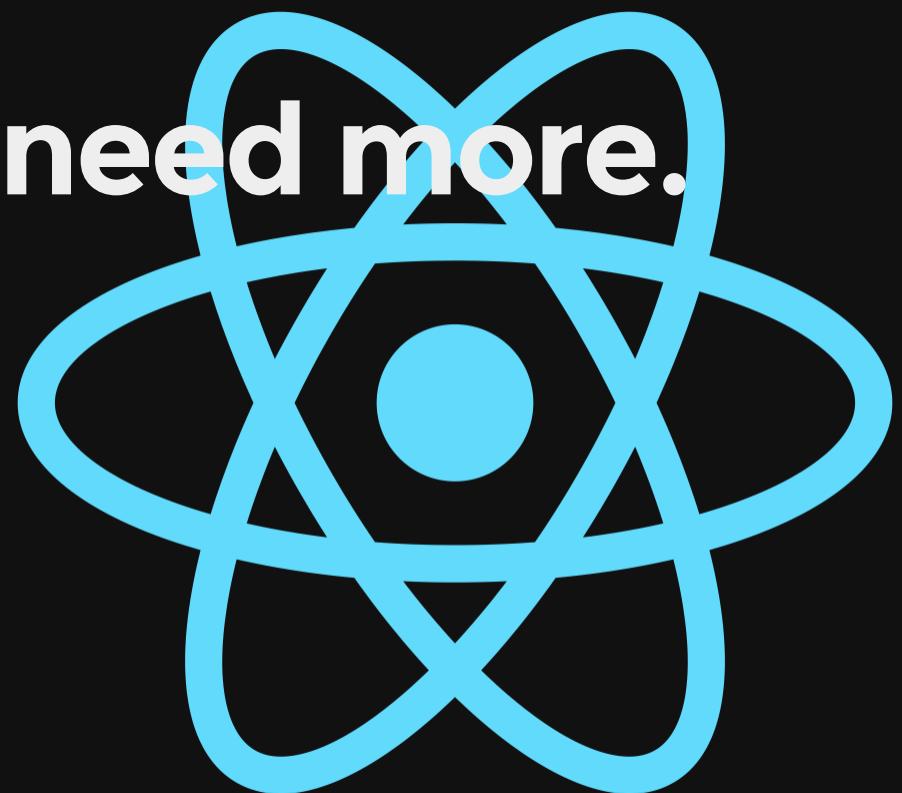
5 days ago	latest	...
	23b1ed4	zip tar.gz
5 days ago	v0.26.2	...
	23b1ed4	zip tar.gz Notes
7 days ago	v0.26.1	...
	8714c9b	zip tar.gz Notes
9 days ago	v0.27.0-rc	...
	0f9e31c	zip tar.gz Notes
10 days ago	v0.26.0	...
	7b54b8e	zip tar.gz Notes
25 days ago	v0.25.1	...
	3336fb5	zip tar.gz Notes
25 days ago	v0.25.0	...
	418a0b2	zip tar.gz Notes

*React Native is following a 2-week train release.
Every two weeks, a Release Candidate (rc) branch
is created off of master and the previous rc branch
is being officially released.*

So, I got your point. React-Native by itself is great.

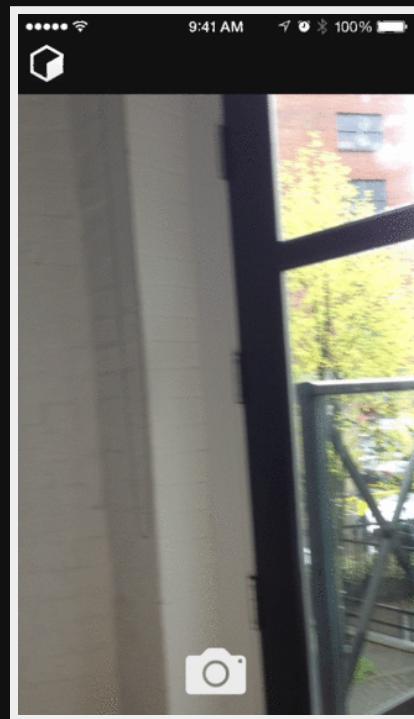
- Solid built-in components list
- Overwhelming community
- Moves and evolves quickly
- Prioritized features requests in Product Pain

But you might need more.



The expansion

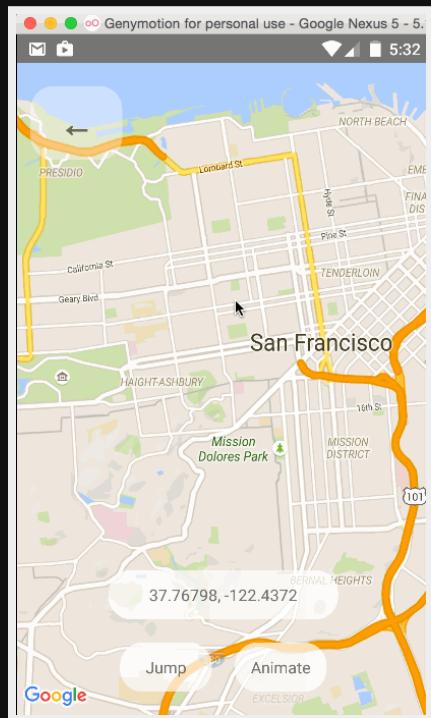
Some important components have been built by the community...



<Camera /> component by Loch Wansbrough - ★1130

The expansion

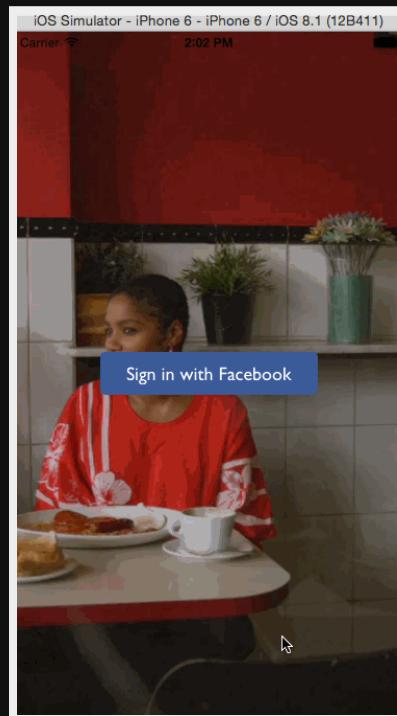
Some important components have been built by the community...



<Maps /> component by Leland Richardson - ★912

The expansion

Some important components have been built by the community...



<Video /> component by Brent Vatne - ★689



The expansion

Some important components have been built by the community...

```
var RNFS = require('react-native-fs');

// Create a path you want to write to
var path = RNFS.DocumentDirectoryPath + '/test.txt';

// Write the file
RNFS.writeFile(path, 'Lorem ipsum dolor sit amet', 'utf8')
  .then((success) => console.log('FILE WRITTEN!'))
  .catch((err) => console.log(err.message));
```

A file-system access by Johannes Lumpe - ★349

The expansion

...and will stay that way.



mkonicek  · Software Engineer at Facebook

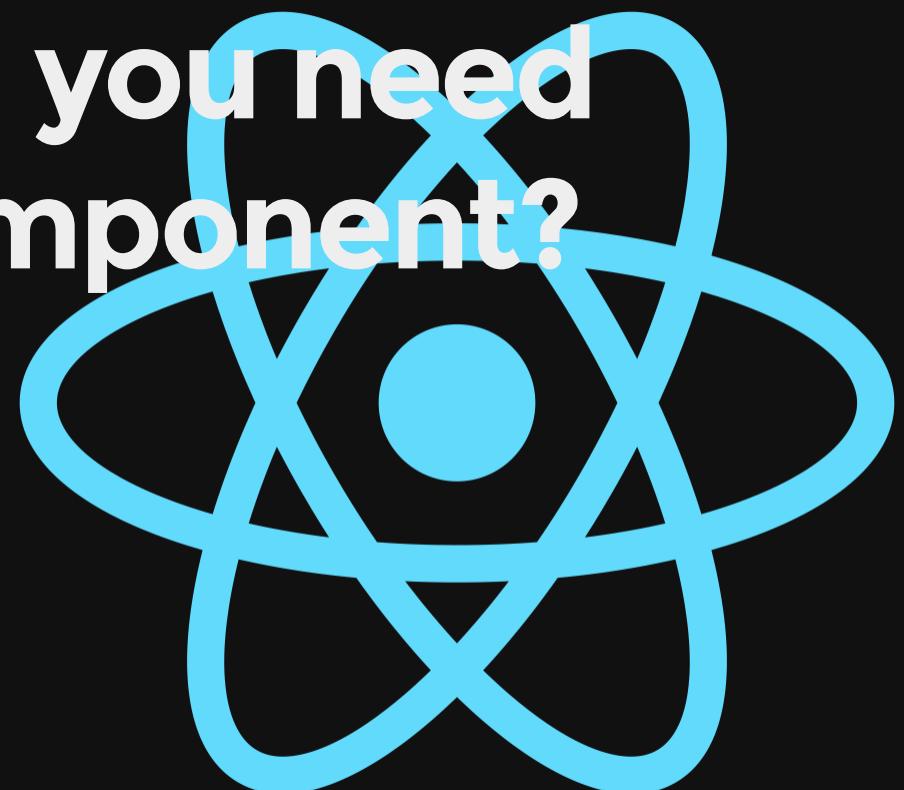
Can't promise we'll be looking into adding <Camera> soon. I think it is great people publish community modules and not everything necessarily needs to be part of the core.

Can you report the bugs you found on <https://github.com/timmh/react-native-camera>?

November 17, 2015

↖_(ツ)_↗

**So, when will you need
a custom component?**





The customisation

When your need is not covered by...

- The core components (UIKit classes: View, Button, TabBar...)
- The community either... (Camera, Maps, ActionSheet, FileSystem...)

Bad luck.

You will have to build the native part first...

Noooooooo



Catelyn Stark didn't know anything about iOS development.



The customisation

... or...

- You have built a native iOS / Android component that you want to use in JS without reimplementing the logic.

Lucky you.

Because react-native will let you create a bridge for it.



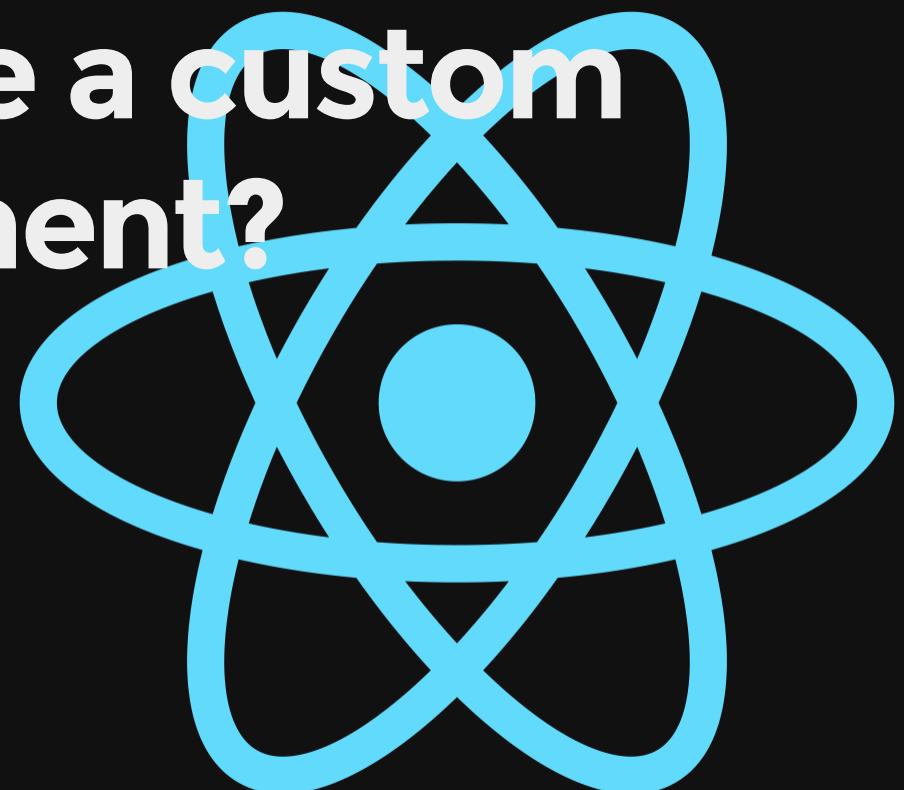


The customisation

A native component *can* be...

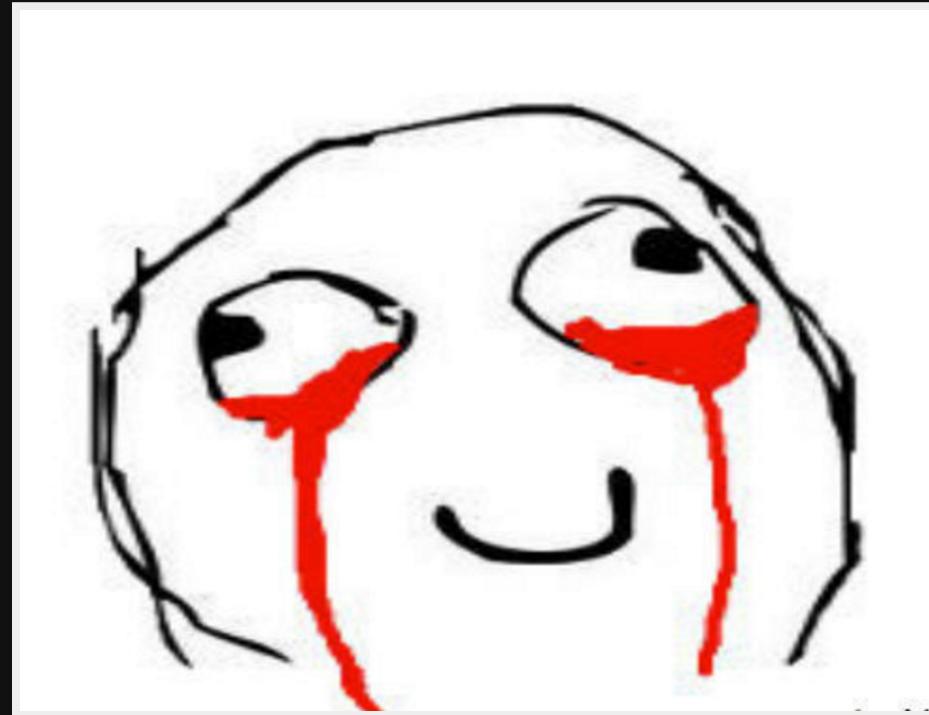
- A "*utility*" that plays with the platform API (ie. FileSystem)
- A "*UI component*" that renders a native view along with it (ie. Maps)

How to create a custom component?



DEMO

The iOS code explained



Creating a custom *utility* component

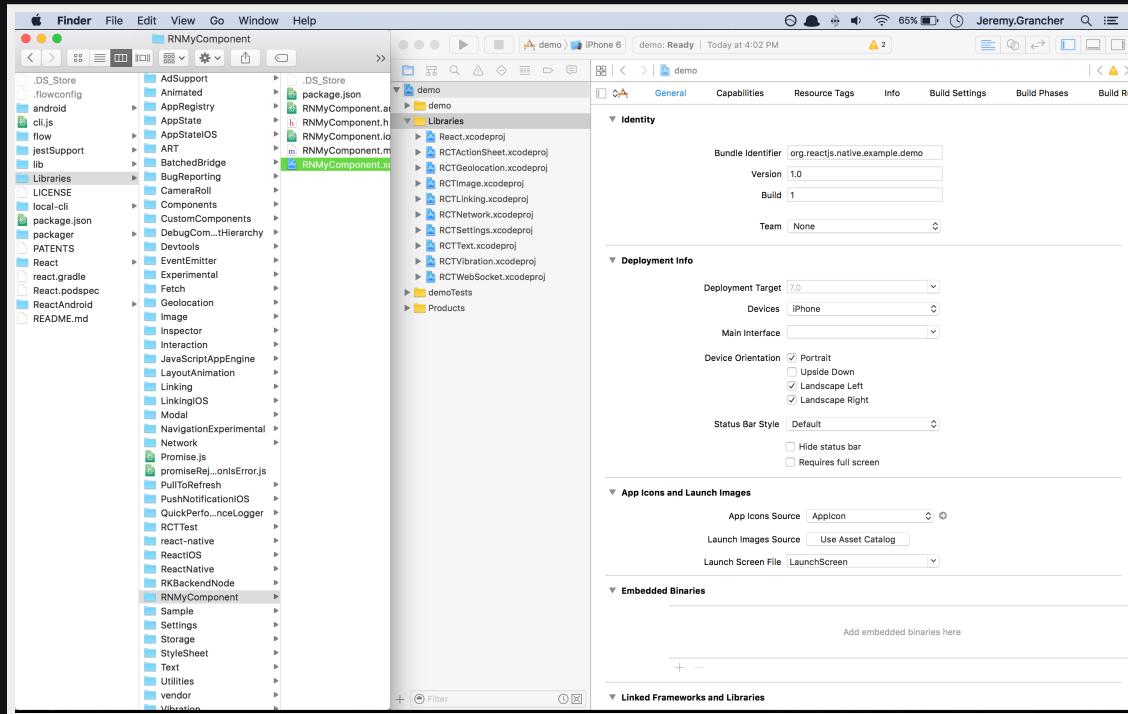
- In the root of your project, use this CLI command to generate your native component:

```
// In your react-native project  
$ react-native new library --name RNMyComponent
```

That will create a sample component here:
.node_modules/react-native/Libraries/RNMyComponent

Creating a custom *utility* component

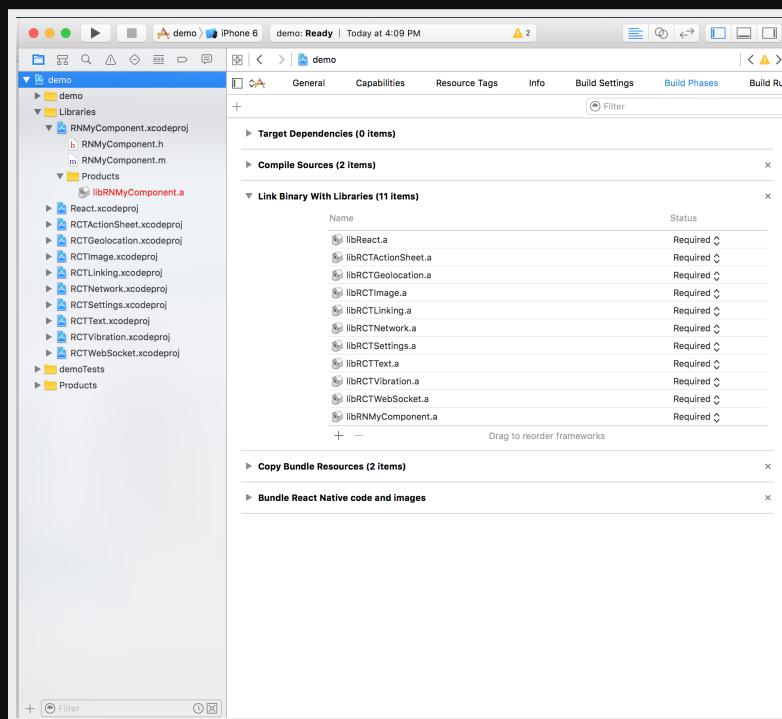
Then, you need link it to your application:



Drag and drop your component's .xcodeproj file into 'Libraries'

Creating a custom *utility* component

Finally, add your static library to the libraries linked to the binary.



Drag and drop your component's .a file into 'Build Phases'

Creating a custom *utility* component

The important things:

- The class has to extend the `<RCTBridgeModule>` protocol:

```
// MyComponentManager.h

#import "RCTBridgeModule.h"

@interface MyComponentManager : NSObject <RCTBridgeModule>
@end
```

That will *automagically* allow this class to access the React world.

Creating a custom *utility* component

The important things:

- The class has to call the RCT_EXPORT_MODULE() macro:

```
// MyComponentManager.m

@implementation MyComponentManager

RCT_EXPORT_MODULE(); // Default module name: Same as class name, without 'Manager'

@end
```

That will allow you to access, in JS, the module named
`<MyComponent />`

Creating a custom *utility* component

The important things:

- The class has to call RCT_EXPORT_METHOD() macro if you want to call a method from your JS code.

```
// MyComponentManager.m

RCT_EXPORT_METHOD(doSomethingWithThatString:(NSString *)string)
{
    RCTLogInfo(@"Pretending to do something: %@", string);
}
```

That will allow you to do, in JS:
MyComponent.doSomethingWithThatString('Hello')

Creating a custom *utility* component

We can (*we have to!*) finally wrap our module in a React component.

```
// MyComponent.ios.js
var NativeComponent = require('NativeModules').MyComponent;

var MyComponent = {
  doSomethingWithThatString: function(string) {
    // You can do a check here, pass a default parameter, etc...
    NativeComponent.doSomethingWithThatString(string);
  }
};

module.exports = MyComponent;
```

Finally, in *your* code:
MyComponent.doSomethingWithThatString('Love React')

Creating a custom *utility* component

A bit more...

- RCTConvert is your friend.

```
#import "RCTConvert.h"
// ...
RCT_EXPORT_METHOD(doSomethingWithThatColor:(NSString *)hexaColor)
{
    UIColor *color = [RCTConvert UIColor:hexaColor];
}
```

That will allow you to do, in JS:
MyComponent.doSomethingWithThatColor('#123456')

Creating a custom *utility* component

A bit more...

- The return type of bridge methods is always *void*.
- If you need to get some data from a native method, you'll have to use Promises, as the bridge is asynchronous.

```
RCT_EXPORT_METHOD(findSomethingAsynchronous,
                  resolver:(RCTPromiseResolveBlock)resolve
                  rejecter:(RCTPromiseRejectBlock)reject)
{
    NSArray *results = ...
    if (results) { resolve(results); }
    else {
        NSError *error = ...
        reject(@"no_results", @"There were no results", error);
    }
}
```

That will allow you to do, in JS:

```
MyComponent.findSomethingAsynchronous().then((r) =>
    console.log(r))
```

Creating a custom *utility* component

Your module can also..

- Specify which thread its methods should be run on
- Export constants to JS
- Send events to JS

```
#import "RCTBridge.h"
#import "RCTEventDispatcher.h"

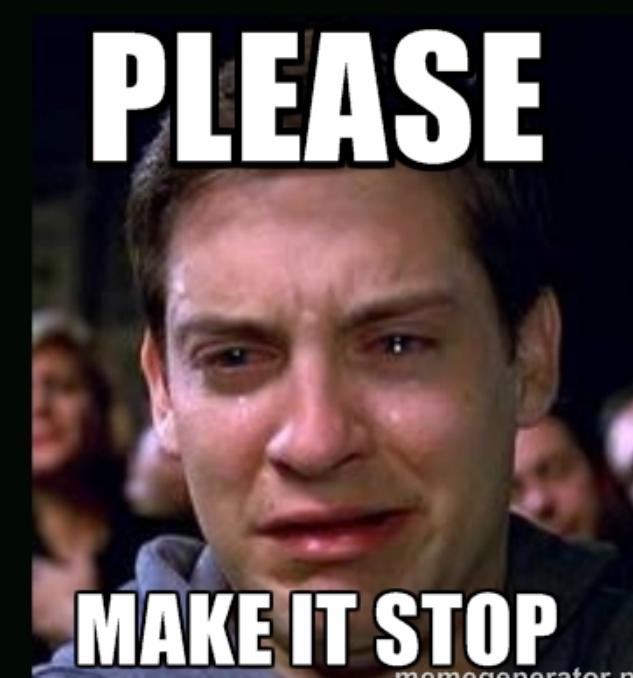
@implementation MyComponentManager
@synthesize bridge = _bridge;

- (void)methodsThatSendsAnEvent
{
    [self.bridge.eventDispatcher sendAppEventWithName:@"CheeseReminder"
                                              body:@{@"content": "I love cheese"}];
}

@end
```

In JS: NativeAppEventEmitter.addListener('CheeseReminder',
(reminder) => console.log(reminder.content));

PLEASE



MAKE IT STOP

memegenerator.net

Creating a custom *UI* component

A UI component is a custom component that *renders* something.

Same as before, but...

Creating a custom *UI component*

- You will create a new class that extends `UIView`
- Your **manager** (kind of view controller, but singleton) has now to be a subclass of `RCTViewManager`.

```
// MyComponentViewManager.h

#import "RCTViewManager.h"

@interface MyComponentViewManager : RCTViewManager
@end
```

Creating a custom *UI component*

- The manager is responsible of the view.

```
// MyComponentViewManager.m

#import "MyComponentViewManager.h"
#import "MyComponentView.h"

@implementation MyComponentViewManager

RCT_EXPORT_MODULE()

- (UIView *)view
{
    return [[MyComponentView alloc] init];
}

...

// Your other RCT_EXPORT_METHOD methods here...

@end
```

Creating a custom *UI* component

- Use RCT_EXPORT_VIEW_PROPERTY() to set things in your view.

```
// MyComponentViewManager.m
RCT_EXPORT_VIEW_PROPERTY(lovingCheese, BOOL)
```



```
// MyComponentView.h
@interface MyComponentView: UIView

@property (nonatomic, assign) bool lovingCheese;

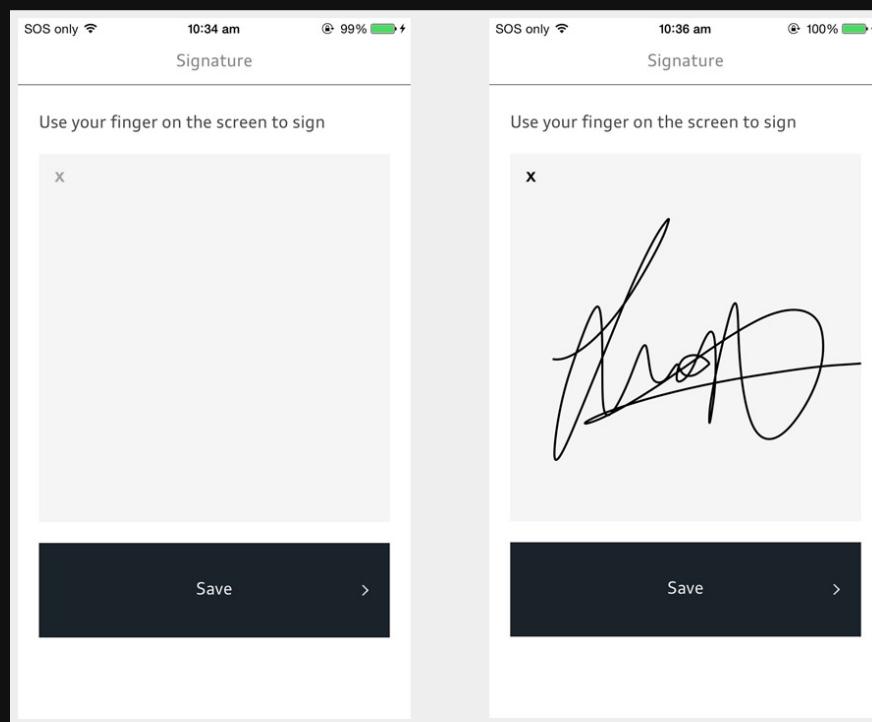
@end
```



```
// MyComponentView.ios.js
MyComponentView.propTypes = {
  lovingCheese: React.PropTypes.bool,
};
```

React-Native-Sketch

A react-native component for touch-based drawing.



React-Native-Sketch

Lessons learned by creating my own custom component:

- It's really cool to publish something to npm
- The frequent changes to the API forces you to be reactive
- It's harder than I thought to get feedbacks & contributions

Conclusion

- Custom components can solve a feature request uncovered. Yet.
- Facebook has created a *great* tool to allow us extend their tool.
- If you *have* to build your own, check the docs regularly.
- Don't be afraid of native code.

Resources

- Native Modules documentation
- Native UI Components documentation
- React-Native View Components (by Brent Vatne)
- Building Custom React Native Components From Scratch (by Jay Garcia)
- How to Bridge an Objective-C View Component (by Jason Brown)
- Awesome React-Native
- JS.Coach

Other components from the community

- react-native-activity-view
- react-native-custom-segmented-control
- react-native-device-info
- react-native-icons
- react-native-overlay
- react-native-search-bar
- react-native-sound

Merci.

Link of this presentation: goo.gl/fpmXem

Questions?

