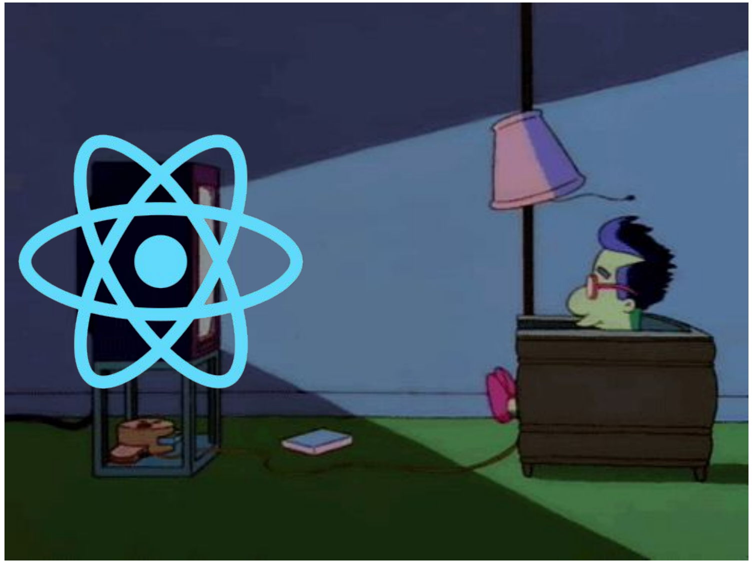
Embracing the "Native" of React Native

Dave Ramirez

FullStack Engineer

Quartet

Enter React Native



Im Not the Only One

DISCORD

UBER EATS

Walmart Save money. Live better.

Native Can Be Hard

"Open up your xcode project and link x and y frameworks under ‘build phases’...”

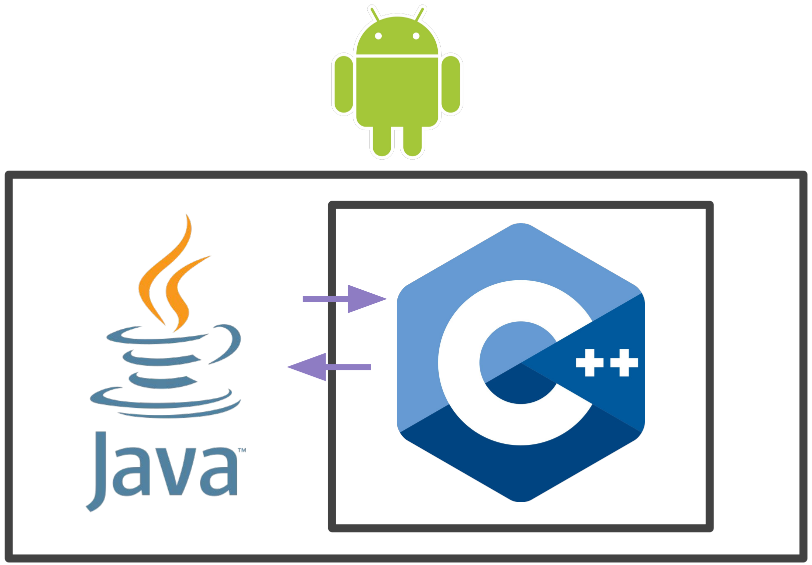
What I'll Cover

- The "Native" of React Native
- How React Native works under the hood
- How to bridge native code
- Performance Considerations

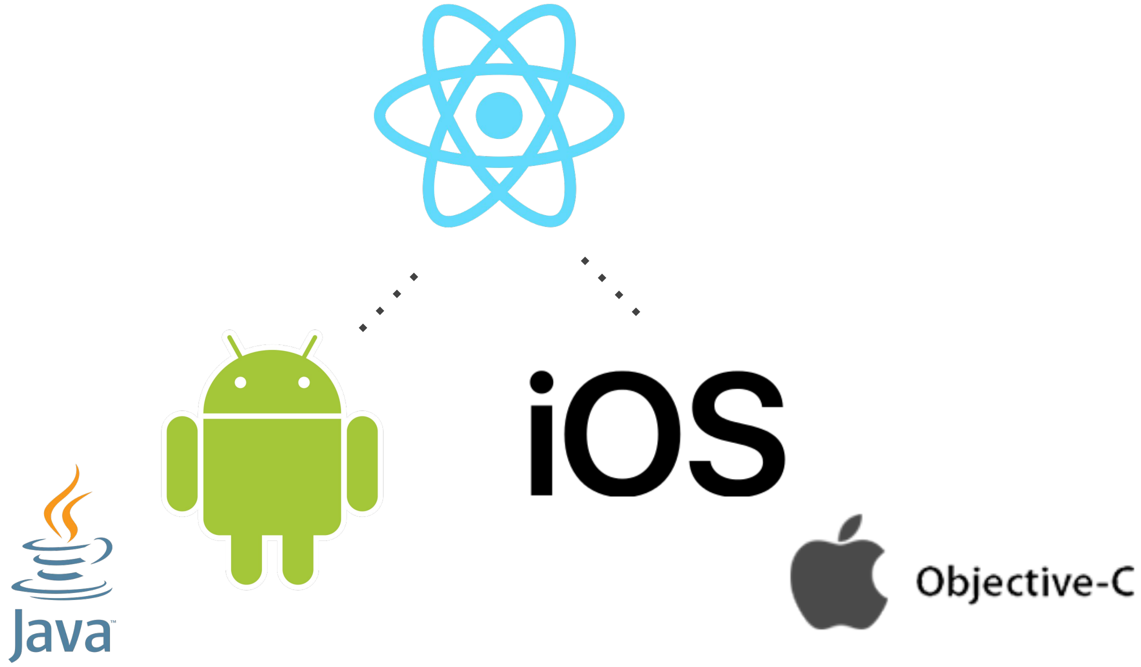
What is Native?

What is Native?

What is Native?



What is Native?



“React Native lets
you build mobile apps
using only JavaScript”
- FaceBook

“React Native lets
you build mobile apps
~~using only JavaScript”~~
- FaceBook

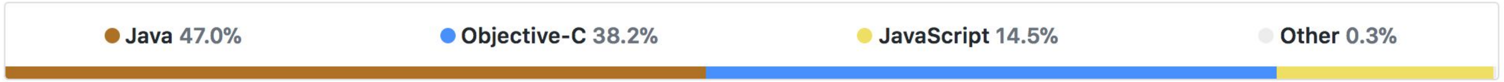
SVG Images:

Navigation:

Camera:

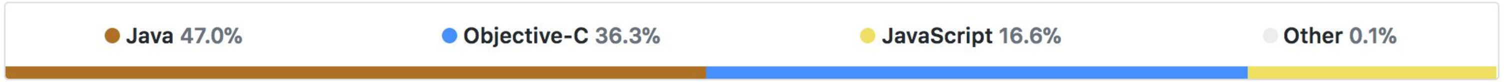
SVG Images:	**react-native-svg**
Navigation:	**react-native-navigation**
Camera:	**react-native-camera**

react-native-svg



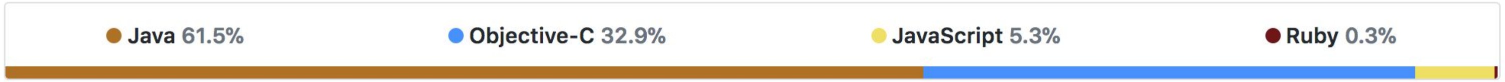
● Java 47.0% ● Objective-C 38.2% ● JavaScript 14.5% ● Other 0.3%

react-native-navigation



● Java 47.0% ● Objective-C 36.3% ● JavaScript 16.6% ● Other 0.1%

react-native-camera

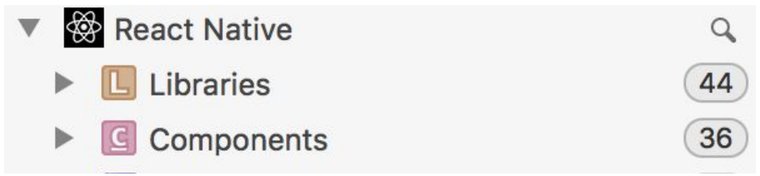


● Java 61.5% ● Objective-C 32.9% ● JavaScript 5.3% ● Ruby 0.3%

“React Native lets you build mobile apps using a lot of JS and also some Objective-C and Java” - Me

Mobile Platforms have huge SDKs

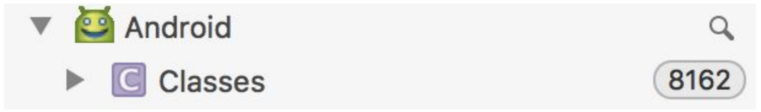
```
80 Documented Libs/Components
```



React Native

Libraries	44
Components	36

```
8162 Classes
```



Android

Classes	8162

Thanks, Dash

We Can Help!

- Write our own Native Modules
- Expose our own Native UI Components

React Native does NOT

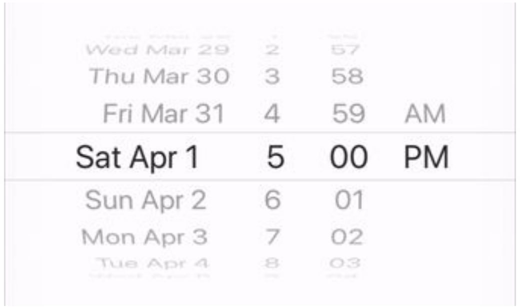
- Render to a DOM
- Compile JS code into Java / Objective-C

React Native DOES

- Render native views via native SDKs

`<DatePickerIOS/>` →

UIDatePicker



```
render() {
  return (
    <p style={{color: ‘blue’}}>
      Hello World</p>
  )
}
```

```
render() {
  return (
    <p style={{color: ‘blue’}}>
      Hello World</p>
  )
}
```

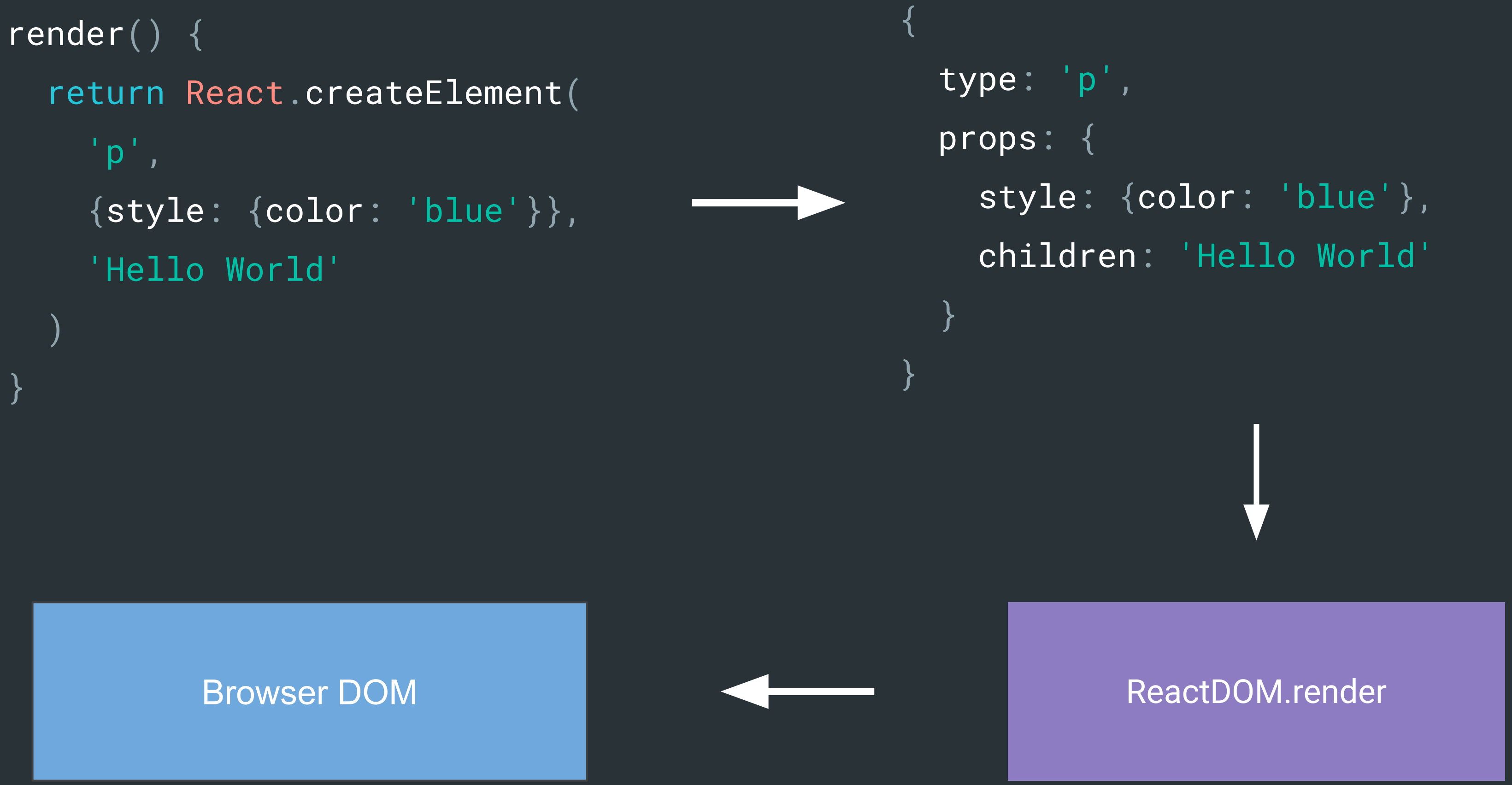
BABEL

```
render() {
  return React.createElement(
    'p',
    {style: {color: 'blue'}},
    'Hello World'
  )
}
```

```
render() {
  return React.createElement(
    'p',
    {style: {color: 'blue'}},
    'Hello World'
  )
}
```

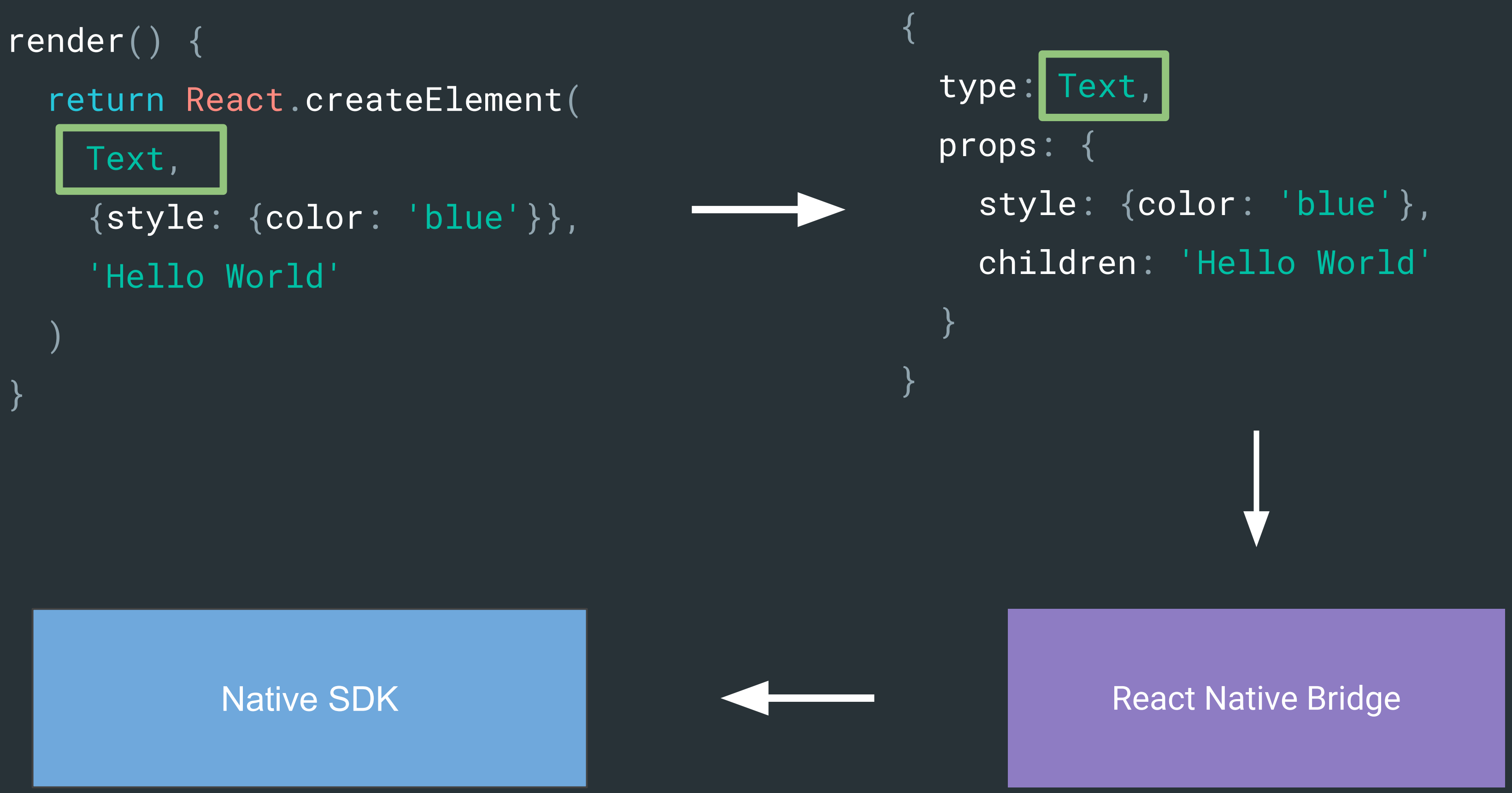
$\longrightarrow$

```
{
  type: ‘p’,
  props: {
    style: {color: 'blue'},
    children: 'Hello World'
  }
}
```



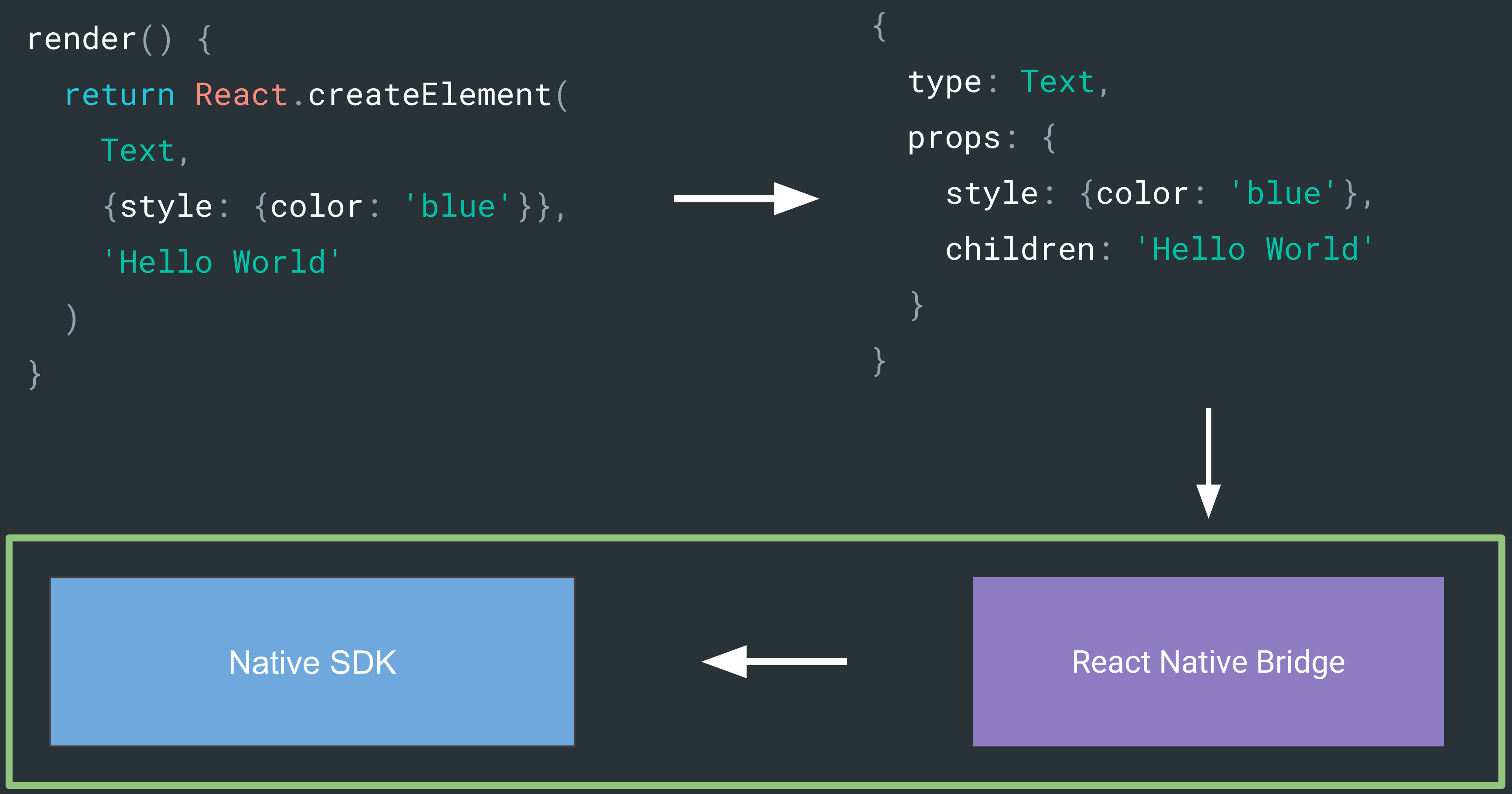
```
render() {
  return React.createElement(
    'p',
    {style: {color: 'blue'}},
    'Hello World'
  )
}
```

```
{
  type: 'p',
  props: {
    style: {color: 'blue'},
    children: 'Hello World'
  }
}
```



```
render() {
  return React.createElement(
    Text,
    {style: {color: 'blue'}},
    'Hello World'
  )
}
```

```
{
  type: Text,
  props: {
    style: {color: 'blue'},
    children: 'Hello World'
  }
}
```



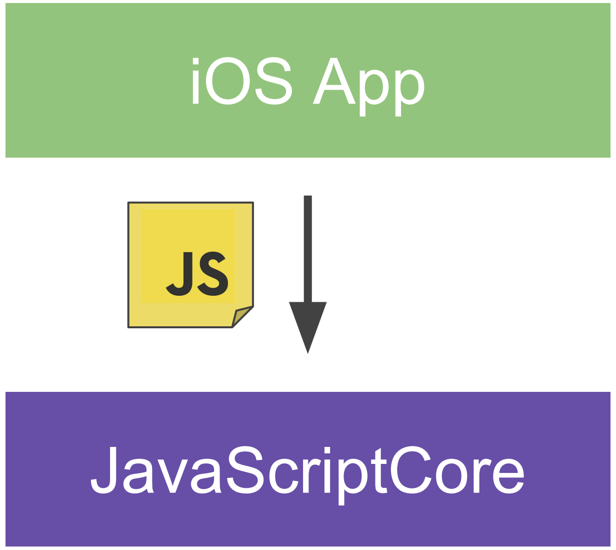
```
render() {
  return React.createElement(
    Text,
    {style: {color: 'blue'}},
    'Hello World'
  )
}
```

```
{
  type: Text,
  props: {
    style: {color: 'blue'},
    children: 'Hello World'
  }
}
```

Disclaimer: I don’t work for Facebook

JavaScriptCore

- JavaScript engine
- Powers Safari
- Also Powers React Native
- C, Objective-C, Swift APIs



Objective-C

```
JSContext *context = [[JSContext alloc] init];

[context evaluateScript:
    @"const sum = (a,b) => { return a + b }"];

JSValue *sum = [context
    evaluateScript:@"sum(4,5)"];
```

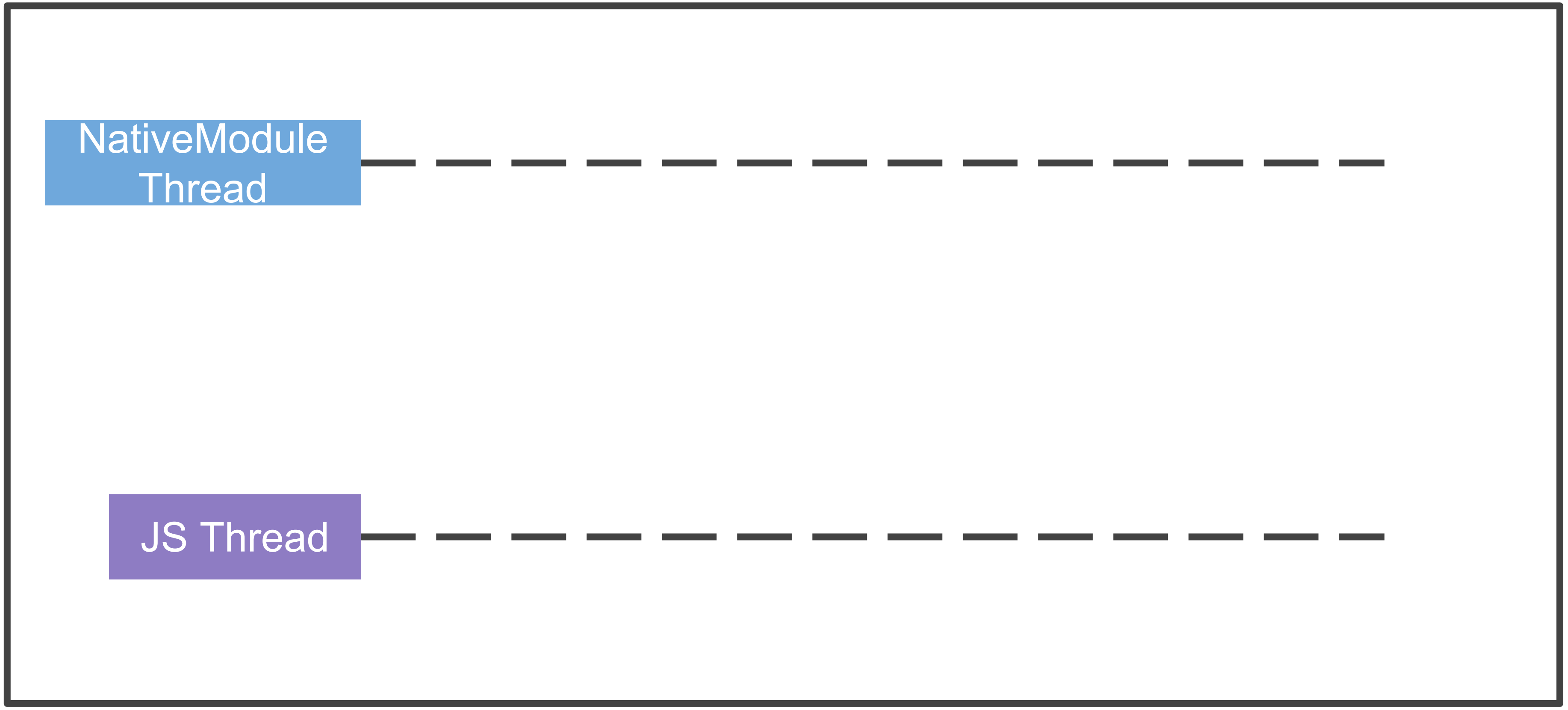
Objective-C

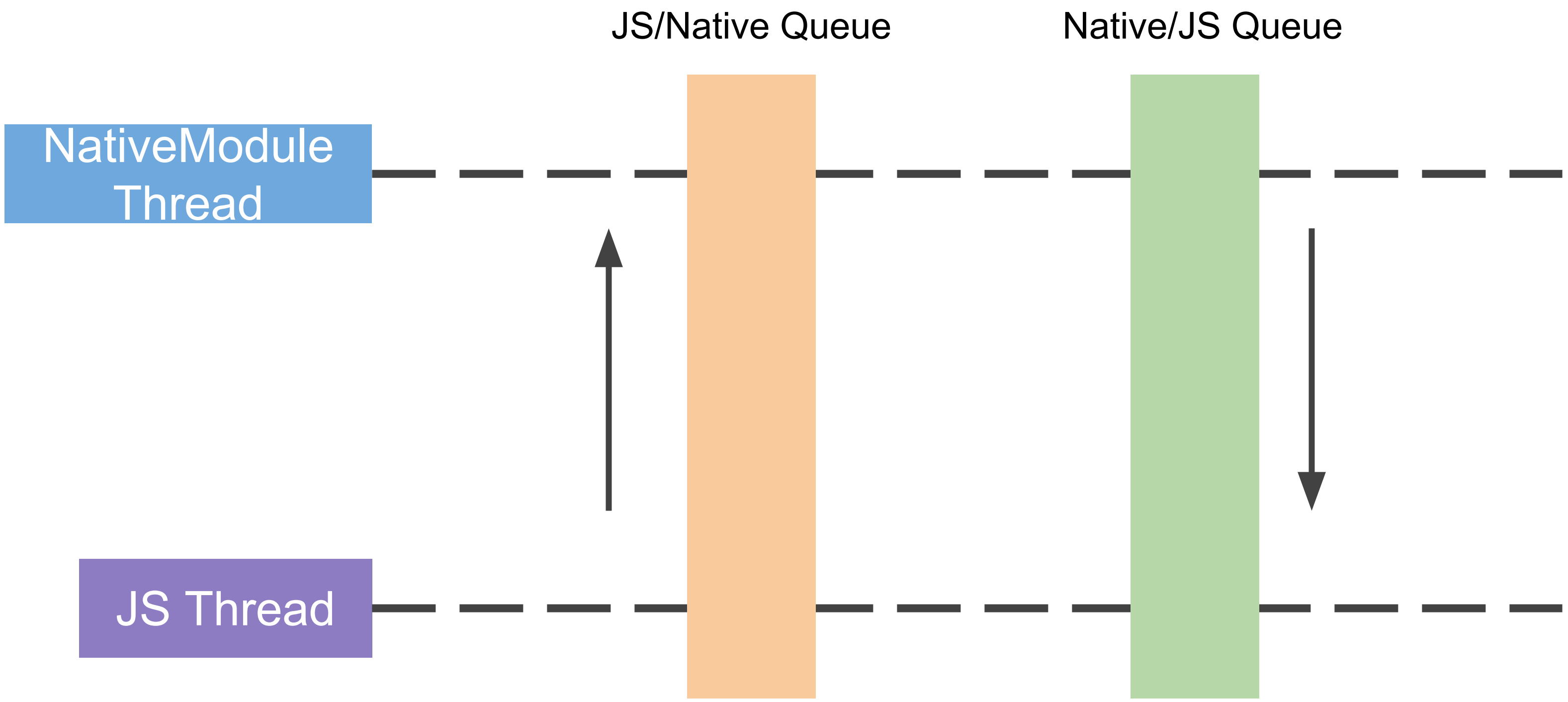
```
JSContext *context = [[JSContext alloc] init];

[context evaluateScript:
    @"const sum = (a,b) => { return a + b }"];

JSValue *tripleNum = [context
    evaluateScript:@"sum(4,5)"];
```

Objective-C

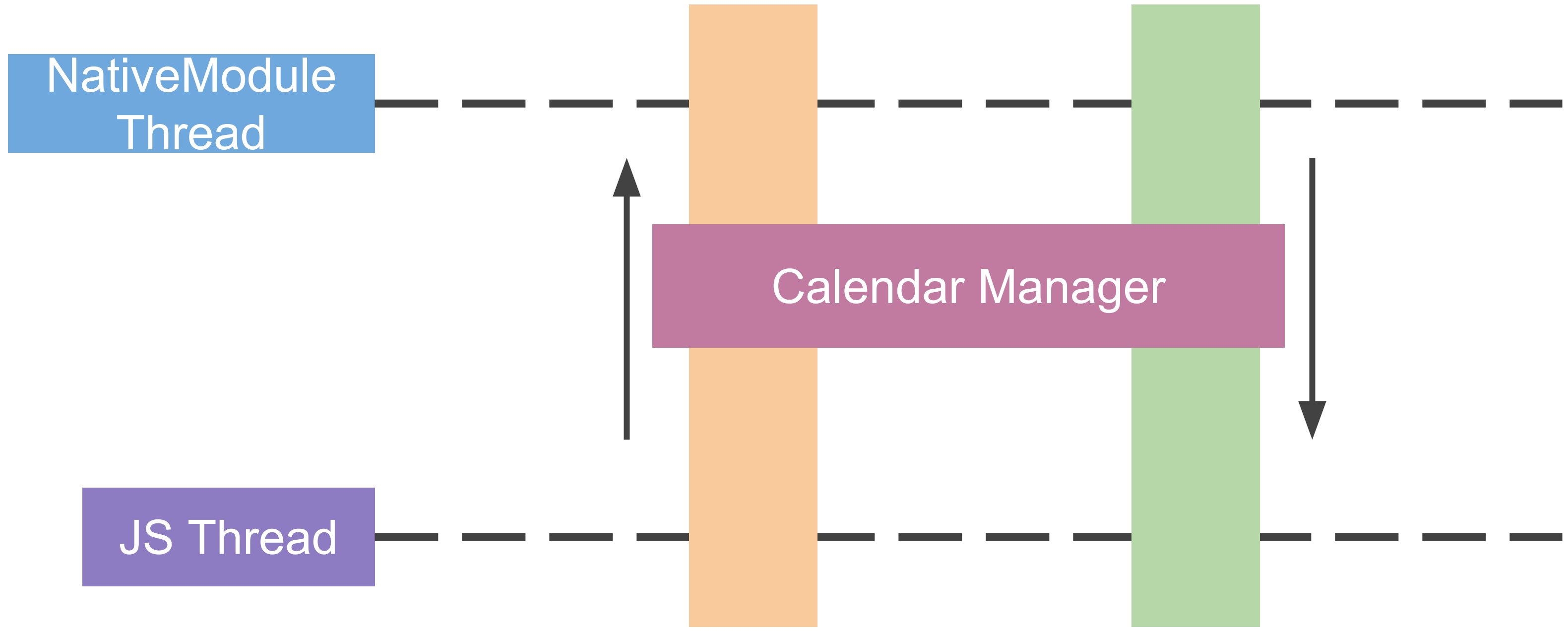
```
JSContext *context = [[JSContext alloc] init];

[context evaluateScript:
    @"const sum = (a,b) => { return a + b }"];

JSValue *sum = [context
    evaluateScript:@"sum(4,5)"];
```

React Native Application




JS/Native Queue
Native/JS Queue
NativeModule Thread
JS Thread

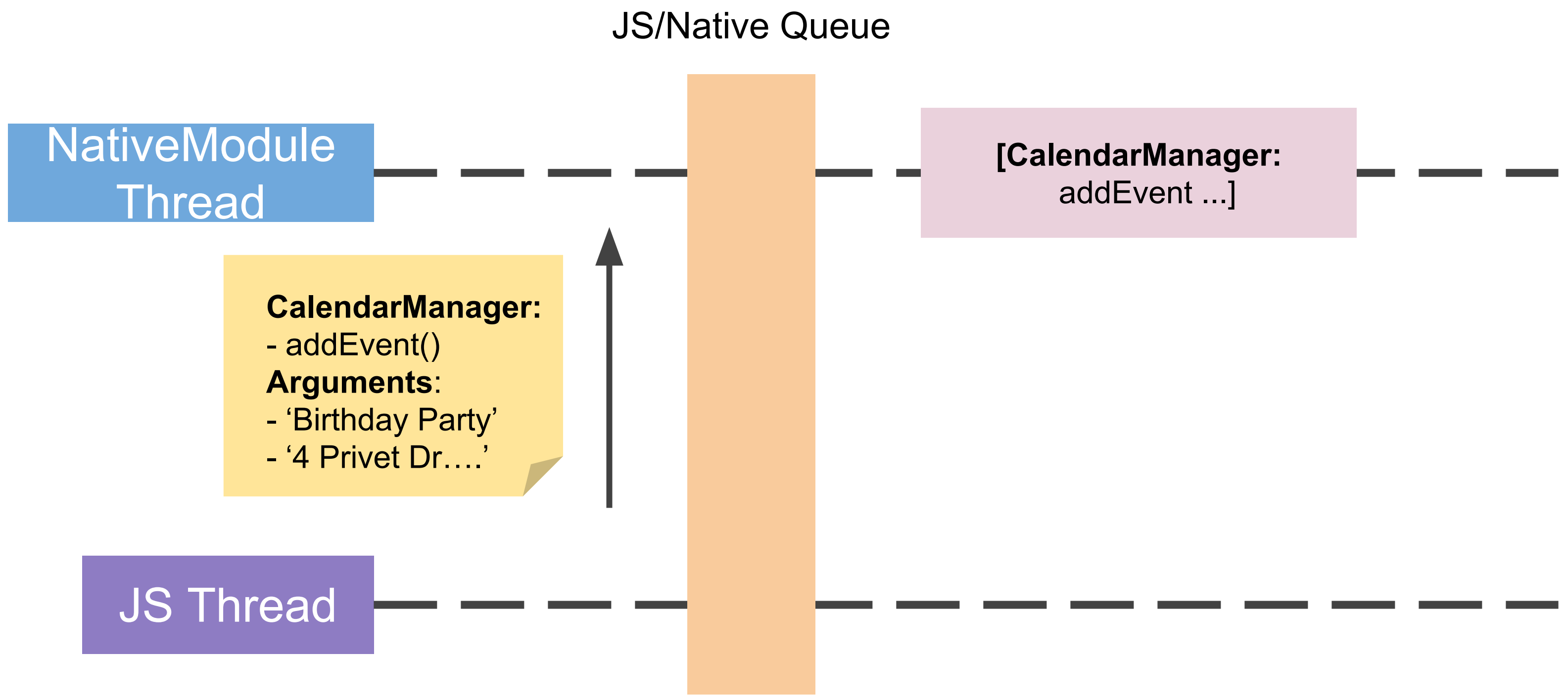
Native Modules:



CalendarManager.addEvent()

```
import {NativeModules} from 'react-native';
const CalendarManager = NativeModules.CalendarManager;
CalendarManager.addEvent(
  'Birthday Party',
  '4 Privet Drive, Surrey'
);
```

JS

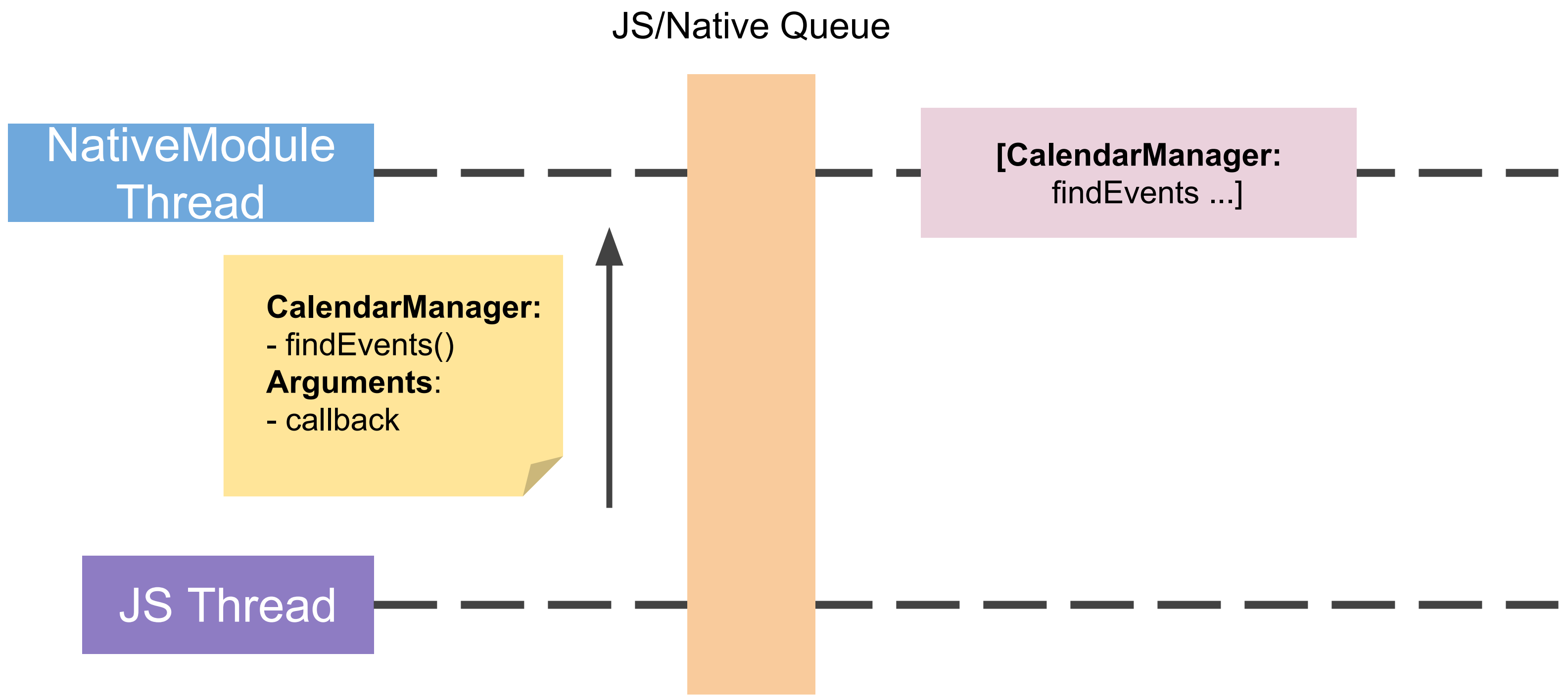

JS/Native Queue
NativeModule Thread
[CalendarManager: addEvent ...]
CalendarManager:
- addEvent()
Arguments:
- ‘Birthday Party’
- ‘4 Privet Dr….’
JS Thread

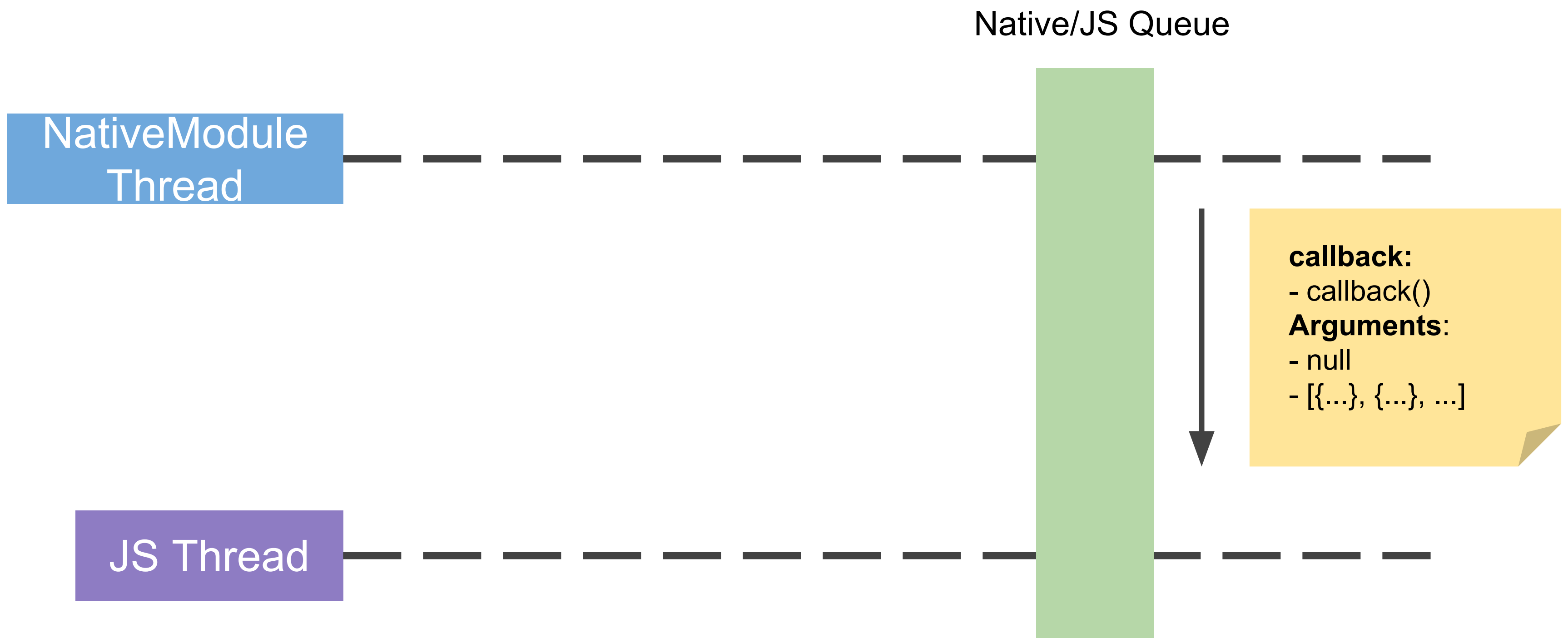
Type Conversions

string	→	NSString
number	→	(NSInteger, float, double, CGFloat, NSNumber)
boolean	→	(BOOL, NSNumber)
array	→	(NSArray)

CalendarManager.findEvents()

```
CalendarManager.findEvents(onEventsFound)
const onEventsFound = (error, events) => {
  if (error) {
    console.error(error)
  } else {
    this.setState({events: events})
  }
}
```


JS/Native Queue
NativeModule Thread
[CalendarManager: findEvents ...]
CalendarManager:
- findEvents()
Arguments:
- callback
JS Thread



callback:
- callback()

Arguments:
- null
- [{...}, {...}, ...]

Rolling Your Own

Why?

- Expose new native functionality
- Fix broken things
- Be Cool

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

iOS

```
public class CalendarManager extends
ReactContextBaseJavaModule {
...
}
```

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

iOS

```
~~public class CalendarManager extends
ReactContextBaseJavaModule {
  ...~~
}
```

```
CalendarManager.addEvent(
  'Birthday Party',
  '4 Privet Drive, Surrey'
);
```

Objective-C

```objc
// CalendarManager.h
#import <React/RCTBridgeModule.h>
@interface CalendarManager : NSObject <RCTBridgeModule>
@end
```

Objective-C

```
// CalendarManager.m
#import "CalendarManager.h"
#import <React/RCTLog.h>
@implementation CalendarManager
RCT_EXPORT_MODULE();
RCT_EXPORT_METHOD(addEvent:(NSString *)name
location:(NSString *)location) {
  RCTLogInfo(@"Pretending to create an event %@ at %@",
name, location);
}
@end
```

Objective-C

```
// CalendarManager.m
#import "CalendarManager.h"
#import <React/RCTLog.h>
@implementation CalendarManager
RCT_EXPORT_MODULE();
RCT_EXPORT_METHOD(addEvent:(NSString *)name
location:(NSString *)location) {
  RCTLogInfo(@"Pretending to create an event %@ at %@",
name, location);
}
@end
```

Objective-C

```
// CalendarManager.m
#import "CalendarManager.h"
#import <React/RCTLog.h>
@implementation CalendarManager
RCT_EXPORT_MODULE();
RCT_EXPORT_METHOD(addEvent:(NSString *)name
location:(NSString *)location) {
  RCTLogInfo(@"Pretending to create an event %@ at %@",
name, location);
}
@end
```

Objective-C

```
// CalendarManager.m
#import "CalendarManager.h"
#import <React/RCTLog.h>
@implementation CalendarManager
RCT_EXPORT_MODULE();
RCT_EXPORT_METHOD(addEvent:(NSString *)name
location:(NSString *)location) {
  RCTLogInfo(@"Pretending to create an event %@ at %@",
name, location);
}
@end
```

```
CalendarManager.findEvents(onEventsFound)
const onEventsFound = (error, events) => {
  if (error) {
    console.error(error)
  } else {
    this.setState({events: events})
  }
}
```

JS

Objective-C

```objc
// CalendarManager.m
...
RCT_EXPORT_METHOD(
 findEvents:(RCTResponseSenderBlock)callback
) {
  NSArray *events = ...

  NSArray *arguments = @[[NSNull null], events];
  callback(arguments);
}
...
```

Objective-C

```
// CalendarManager.m
...
RCT_EXPORT_METHOD(
 findEvents:(RCTResponseSenderBlock)callback
) {
  NSArray *events = ...

  NSArray *arguments = @[[NSNull null], events];
  callback(arguments);
}
...
```

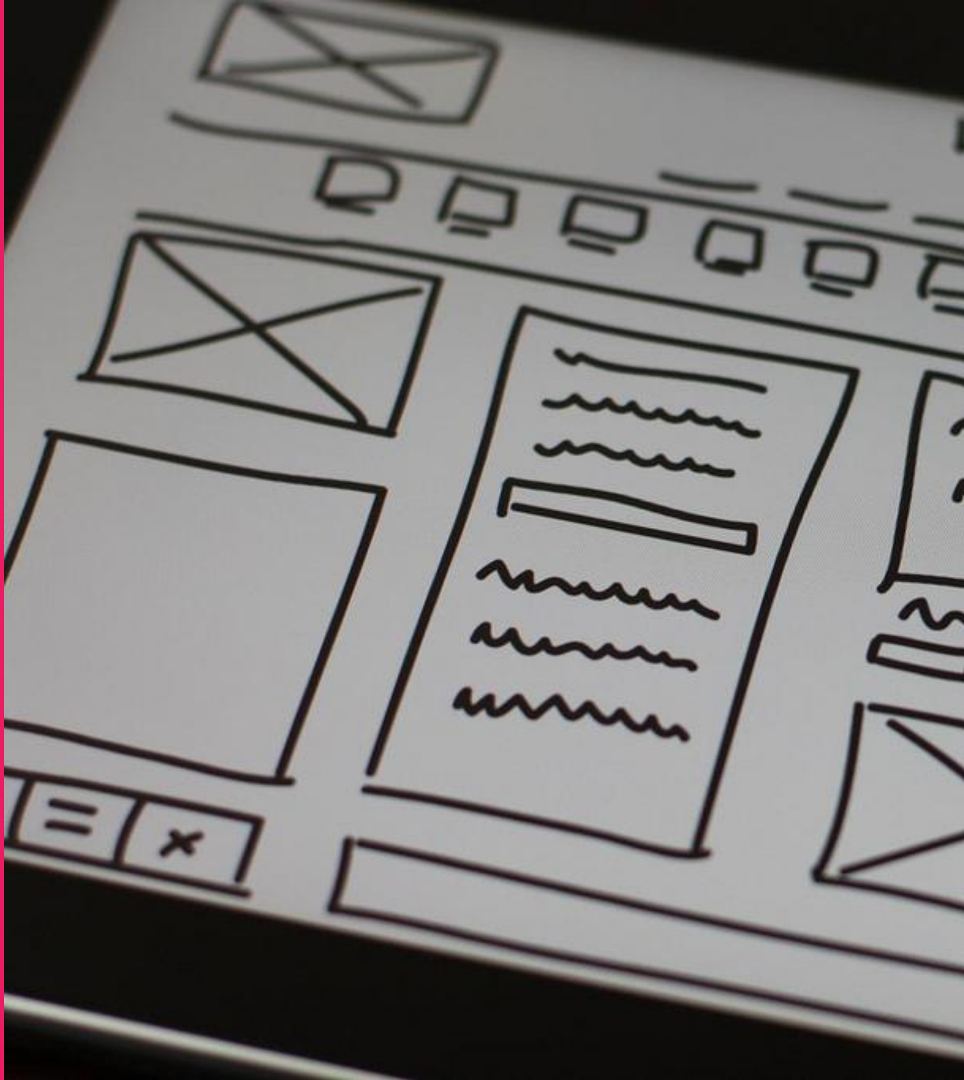
Objective-C

```
// CalendarManager.m
...
RCT_EXPORT_METHOD(
 findEvents:(RCTResponseSenderBlock)callback
) {
  NSArray *events = ...

  NSArray *arguments = @[[NSNull null], events];
  callback(arguments);
}
...
```

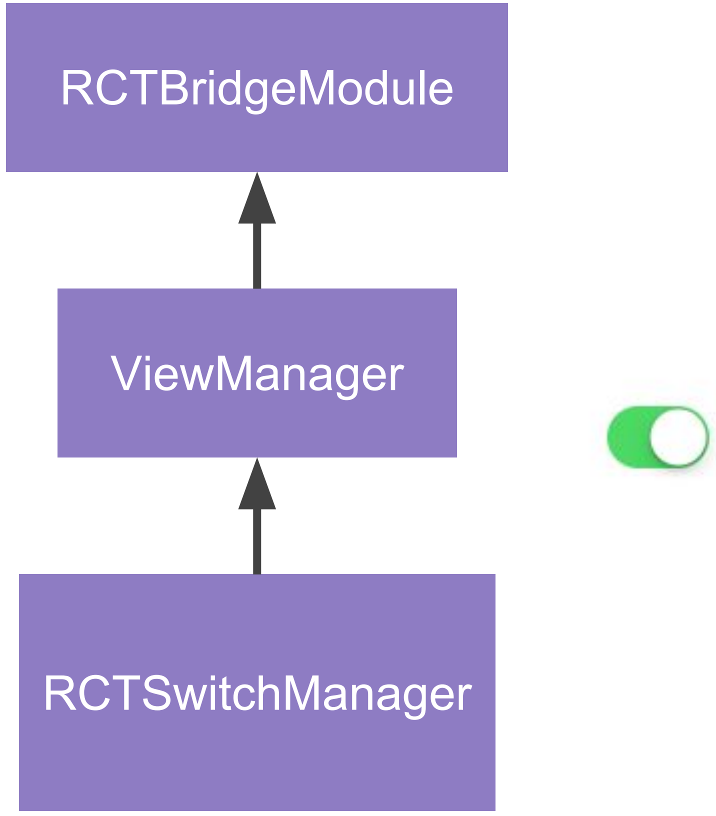
Objective-C

```objc
// CalendarManager.m
...
RCT_EXPORT_METHOD(
 findEvents:(RCTResponseSenderBlock)callback
) {
  NSArray *events = ...

  NSArray *arguments = @[[NSNull null], events];
  callback(arguments);
}
...
```

What about UI Components?



ViewManagers ARE Native Modules



Objective-C

```objc
#import <UIKit/UIKit.h>
#import <React/RCTComponent.h>

@interface RCTSwitch : UISwitch

@property (nonatomic, assign) BOOL wasOn;
@property (nonatomic, copy) RCTBubblingEventBlock
onChange;

@end
```

Objective-C

```objc
#import "RCTSwitch.h"

#import "RCTEventDispatcher.h"
#import "UIView+React.h"

@implementation RCTSwitch

- (void)setOn:(BOOL)on animated:(BOOL)animated {
 _wasOn = on;
 [super setOn:on animated:animated];
}
```

Objective-C

```
#import <React/RCTViewManager.h>

@interface RCTSwitchManager : RCTViewManager

@end
```

Objective-C

```
@implementation RCTSwitchManager

RCT_EXPORT_MODULE()

- (UIView *)view
{
  RCTSwitch *switcher = [RCTSwitch new];
  [switcher addTarget:self
  action:@selector(onChange:)
  forControlEvents:UIControlEventValueChanged];
  return switcher;
}
- (void)onChange:(RCTSwitch *)sender {...}
...
```

Objective-C

```
@implementation RCTSwitchManager

RCT_EXPORT_MODULE()

- (UIView *)view
{
  RCTSwitch *switcher = [RCTSwitch new];
  [switcher addTarget:self
  action:@selector(onChange:)
  forControlEvents:UIControlEventValueChanged];
  return switcher;
}
- (void)onChange:(RCTSwitch *)sender {...}
...
```

Objective-C

```
@implementation RCTSwitchManager

RCT_EXPORT_MODULE()

- (UIView *)view
{
  RCTSwitch *switcher = [RCTSwitch new];
  [switcher addTarget:self
  action:@selector(onChange:)
  forControlEvents:UIControlEventValueChanged];
  return switcher;
}
- (void)onChange:(RCTSwitch *)sender {...}
...
```

Objective-C

```
@implementation RCTSwitchManager

RCT_EXPORT_MODULE()

- (UIView *)view
{
  RCTSwitch *switcher = [RCTSwitch new];
  [switcher addTarget:self
  action:@selector(onChange:)
  forControlEvents:UIControlEventValueChanged];
  return switcher;
}
- (void)onChange:(RCTSwitch *)sender {...}
...
```

Objective-C

```
@implementation RCTSwitchManager

RCT_EXPORT_MODULE()

- (UIView *)view
{
  RCTSwitch *switcher = [RCTSwitch new];
  [switcher addTarget:self
  action:@selector(onChange:)
  forControlEvents:UIControlEventValueChanged];
  return switcher;
}
- (void)onChange:(RCTSwitch *)sender {...}
...
```

Objective-C

```
...
- (void)onChange:(RCTSwitch *)sender
{
  if (sender.wasOn != sender.on) {
  if (sender.onChange) {
  sender.onChange(@{ @"value": @(sender.on) });
 }
  sender.wasOn = sender.on;
 }
 ...
```

Objective-C

```
...
- (void)onChange:(RCTSwitch *)sender
{
  if (sender.wasOn != sender.on) {
  if (sender.onChange) {
  sender.onChange(@{ @"value": @(sender.on) });
 }
  sender.wasOn = sender.on;
 }
 ...
```

Objective-C

```
...
RCT_EXPORT_VIEW_PROPERTY(onTintColor, UIColor);
RCT_EXPORT_VIEW_PROPERTY(tintColor, UIColor);
RCT_EXPORT_VIEW_PROPERTY(thumbTintColor, UIColor);
RCT_REMAP_VIEW_PROPERTY(value, on, BOOL);
RCT_EXPORT_VIEW_PROPERTY(onChange,
RCTBubblingEventBlock);
...
```

```
<Switch value={...} onChange={...}/ tintColor={}/>
```

Objective-C

```
...
RCT_EXPORT_VIEW_PROPERTY(onTintColor, UIColor);
RCT_EXPORT_VIEW_PROPERTY(tintColor, UIColor);
RCT_EXPORT_VIEW_PROPERTY(thumbTintColor, UIColor);
RCT_REMAP_VIEW_PROPERTY(value, on, BOOL);
RCT_EXPORT_VIEW_PROPERTY(onChange,
RCTBubblingEventBlock);
...
```

```
<Switch value={...} onChange={...}
 thumbTintColor={}/>
```

JS

Objective-C

```
...
RCT_EXPORT_VIEW_PROPERTY(onTintColor, UIColor);
RCT_EXPORT_VIEW_PROPERTY(tintColor, UIColor);
RCT_EXPORT_VIEW_PROPERTY(thumbTintColor, UIColor);
RCT_REMAP_VIEW_PROPERTY(value, on, BOOL);
RCT_EXPORT_VIEW_PROPERTY(onChange,
RCTBubblingEventBlock);
...
```

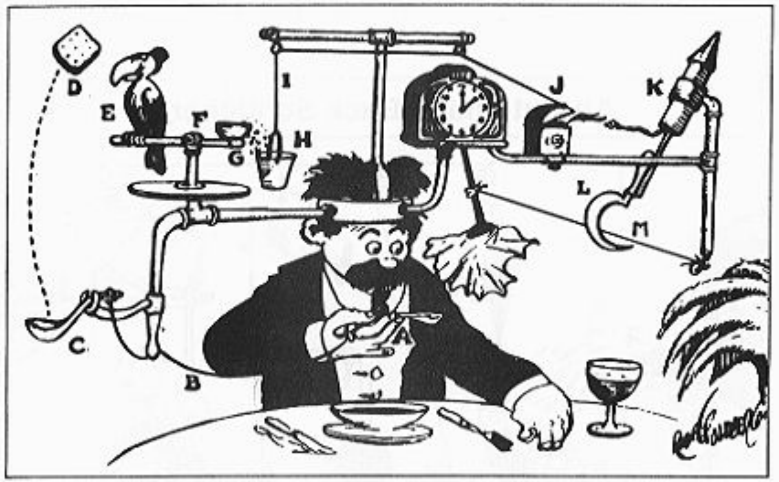
```
<Switch value={...} onChange={...}
 thumbTintColor={}/>
```

JS

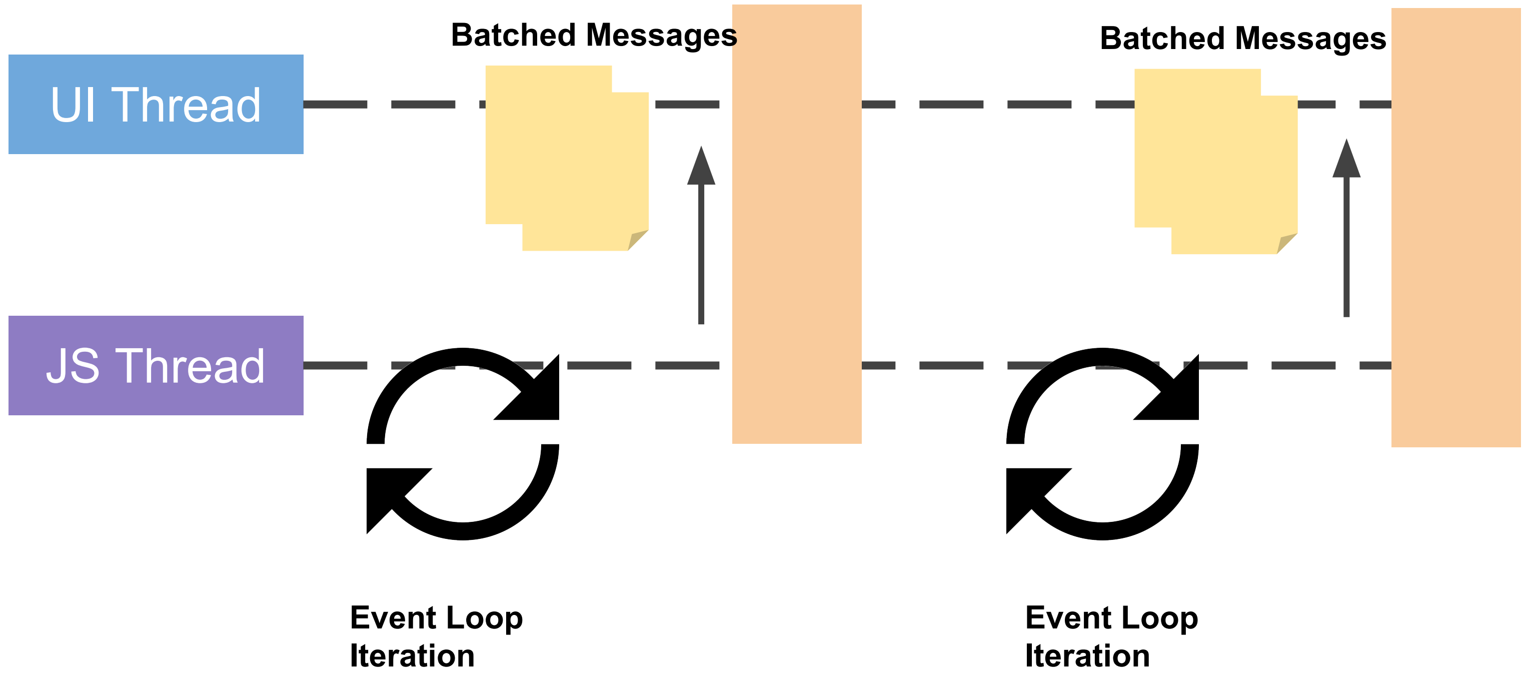
We did it!

This Is Complex

Self-Operating Napkin



Animation!


UI Thread
JS Thread
Batched Messages
Batched Messages
Event Loop Iteration
Event Loop Iteration

Especially Navigation

- We often want to animated navigation
- We often re-render a lot of components

Offload to Native

(When Possible)

- React Native: LayoutAnimation
- React Native: Animated (useNativeDriver: true)
- NavigatorIOS
- 3rd Party nav libs (react-navigation)

App Start Time

- Need to wait for React Native to initialize
- Need to load, run the JS Bundle

RAM bundle + inline requires

```
BloatedComponent =
require('./BloatedCoponent').default;
```

Additional Resources

- https://facebook.github.io/react-native/docs/performance
- http://www.awesome-react-native.com/
- https://www.reactiflux.com/

Thanks

GitHub

/ramirezd42

@daveramirez