



Speak React Native

React Native course by **U+.**

Overview

- Forms
- Keyboard behavior
- Push notifications
- InApp purchases

Forms

Formik

- A lightweight library for handling **form state**, **validation** and **submission**
- Without unnecessary magic or blackbox functionality

Formik

- Install the library

```
$ yarn add formik
```

Usage

- `<Formik />` component
- Formik gives us an object with form values and validation info on submit
- Rendering form components
 - As children `<Formik>...</Formik>`

Usage

- `<Formik />` component
- Formik gives us an object with form values and validation info on submit
- Rendering form components
 - As children `<Formik>...</Formik>`
 - As render prop `<Formik render={...} />`

Usage

- `<Formik />` component
- Formik gives us an object with form values and validation info on submit
- Rendering form components
 - As children `<Formik>...</Formik>`
 - As render prop `<Formik render={...} />`
 - As component prop `<Formik component={MyForm} />`

Basic example

```
<Formik
```

```
  onSubmit={this.handleSubmitForm} // accepts argument object with values
```

```
>
```

```
{props => (
```

```
  <View>
```

```
    <TextInput
```

```
      onChangeText={props.handleChange("email")}
```

```
      value={props.values.email}
```

```
    />
```

```
    <Button onPress={props.handleSubmit}>Send</Button>
```

```
  </View>
```

```
)}
```

```
</Formik>
```

Basic example

```
<Formik
```

```
  onSubmit={this.handleSubmitForm} // accepts argument object with values
```

```
>
```

```
{props => (
```

```
  <View>
```

```
    <TextInput
```

```
      onChangeText={props.handleChange("email")}
```

```
      value={props.values.email}
```

```
    />
```

```
    <Button onPress={props.handleSubmit}>Send</Button>
```

```
  </View>
```

```
)}
```

```
</Formik>
```

Validations

- 2 ways - `validate` and `validationSchema` props
- Can be both synchronous and asynchronous
- `validate` prop accepts validation function which returns an object containing errors (validation passes if empty object)
- `validationSchema` is for usage with external libs like Yup

Validation example

```
const validate = (values) => {  
  const errors = {}  
  if (!/^[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}$/i.test(values.email)) {  
    errors.email = "Invalid email address"  
  }  
  return errors  
}
```

```
<Formik validate={validate}>...</Formik>
```

Validation errors

- Error messages accessible in `<Formik />` component via `errors` field in props
- The error messages are defined in our validation function
- We can use `<ErrorMessage />` component directly from Formik (contains basic logic for showing/hiding)

Validation errors example

```
<Formik ... >
  {props => (
    <View>
      <TextInput
        onChangeText={props.handleChange("email")}
        value={props.values.email}
      />
      <ErrorMessage name="email" component={Text} />
    </View>
  )}
</Formik>
```

Keyboard behavior

Keyboard issues

- We encounter many forms of incorrect keyboard behavior when using RN inputs
 - Inputs below keyboard
 - Non scrollable content above keyboard
 - Keyboard not hiding on tap outside of it
 - Inputs not visible on focus
 - Different behavior on Android and iOS

KeyboardAwareScrollView

- Solves many of the issues caused by native keyboards
- Install the library

```
$ yarn add react-native-keyboard-aware-scroll-view
```

Usage

```
import { KeyboardAwareScrollView } from  
"react-native-keyboard-aware-scroll-view"
```

```
<KeyboardAwareScrollView  
  keyboardShouldPersistTaps="handled"  
  enableOnAndroid  
>  
  // child components  
</KeyboardAwareScrollView>
```

Android support

- Android support requires a small change in `AndroidManifest.xml`
- Set `windowSoftInputMode` to “adjustPan”

Push notifications

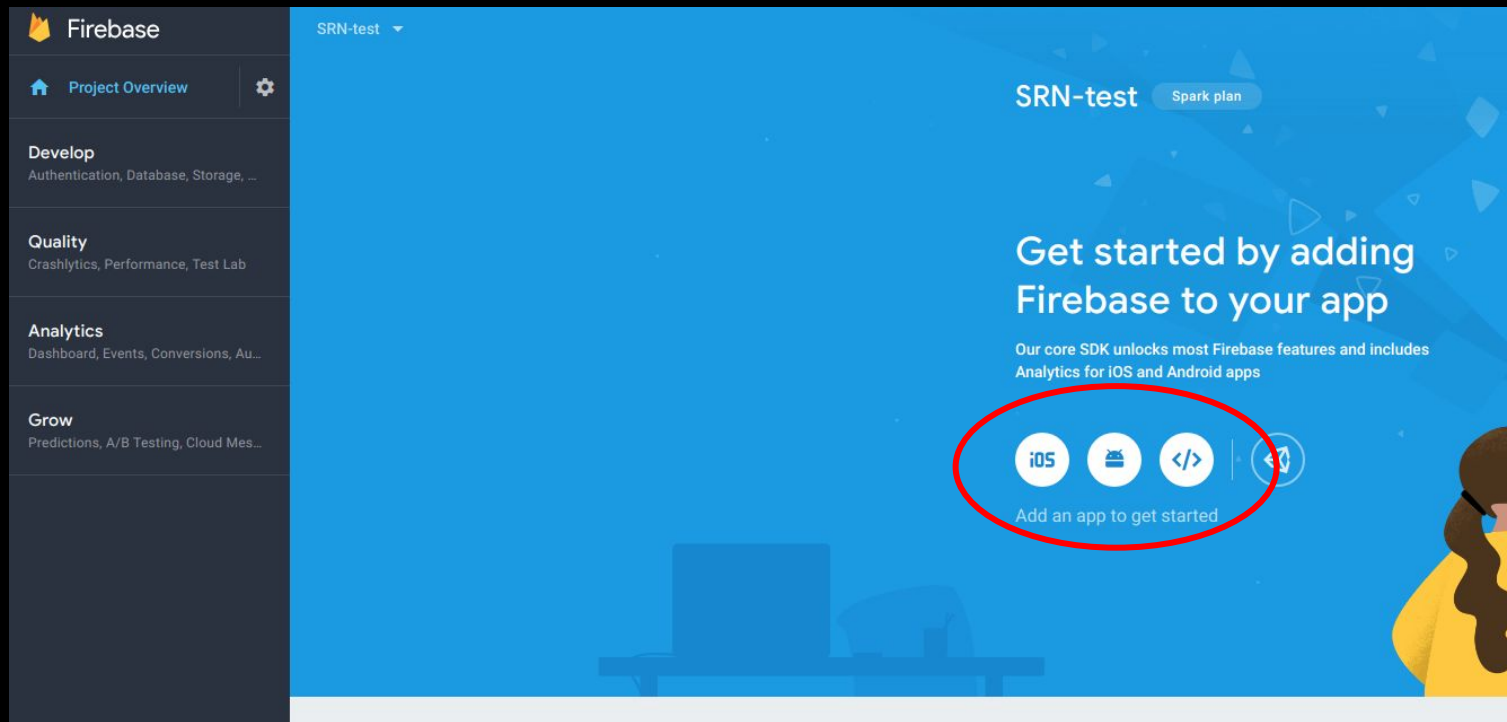
Push notifications

- Background service allowing us to display status bar notifications sent from the server
- Firebase gives us a console for managing and sending notifications on both platforms

Firebase console

- Create new project in [Firebase console](#)
- Choose some name for your project, you can leave the settings to default
- Add a platform to your project (Android/iOS)

Firebase console



Firebase console

1

Register app

iOS bundle ID ⓘ

com.company.appname

App nickname (optional) ⓘ

Freemium iOS App

App Store ID (optional) ⓘ

123456789

Register app

Firebase console

1 Register app

iOS bundle ID ?

App nickname (optional) ?

App Store ID (optional) ?

Running srnwithoutexpo on iPhone 8

srnwithoutexpo > iPhone 8

srnwithoutexpo

General Capabilities Resource Tags Info Build Settings Build Phases

PROJECT

- srnwithoutexpo

TARGETS

- srnwithoutexpo
- srnwithoutexpoTests
- srnwithoutexpo-tvOS
- srnwithoutexpo-tvO...

Identity

Display Name: srnwithoutexpo

Bundle Identifier: **org.reactjs.native.example.srnwithoutexpo**

Version: 1.0

Build: 2

Firebase console

- iOS - download `GoogleService-Info.plist` and add it to your project (File > Add Files in XCode)
- Android - download `google-services.json` and add it to your project (android/app/google-services.json)

Local setup

- Install React Native Firebase

```
$ yarn add react-native-firebase
```

- Link native files

```
$ react-native link react-native-firebase
```

Local setup

- For additional steps of react-native-firebase setup, follow [Android guide](#) or [iOS guide](#) from docs
- You will also need to setup native packages and permissions for notifications, both on [Android](#) and [iOS](#)

Initializing in app

- First, we check if we have notification permissions

```
firebase
    .messaging()
    .hasPermission()
    .then(enabled => {
        if (enabled) {
            // we can listen to messages
        } else {
            // no permissions
        }
    })
})
```

Requesting permissions

- If we don't have permissions yet, we can request them

```
firebase
  .messaging()
  .requestPermission()
  .then(() => {
    // we are good to go
  }).catch(error => console.log('User
rejected'))
```

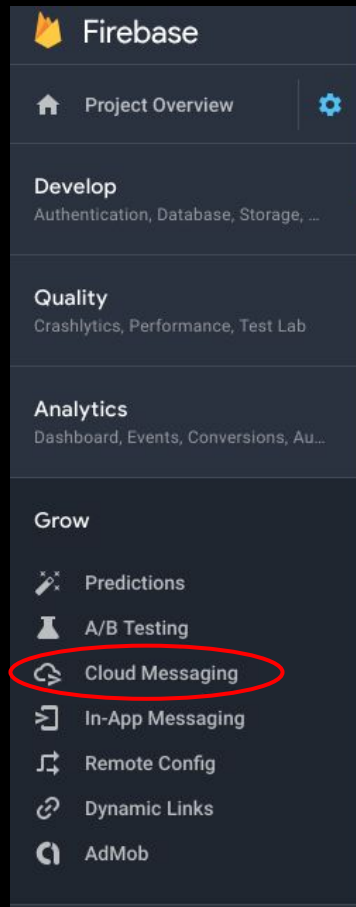
Listening to notifications

- Once we have permissions, we can listen to incoming notifications

```
firebase
  .notifications()
  .onNotification((notification:
Notification) => {
    // custom logic on notification
  })
```

Test the config

- Firebase console
lets us test our
configuration with
debug
notifications



Test the config

Cloud Messaging

Send targeted notifications to drive user engagement

Send your first message

1

Notification

Notification title (optional) ⓘ

Notification text


Notification label (optional) ⓘ

[Test on device](#)

Test the config

- You will need to add [FCM tokens](#) generated by react-native-firebase here in order to test notifications

Test on device

You can test this campaign by entering or selecting the [FCM registration tokens](#)  of your development device below.

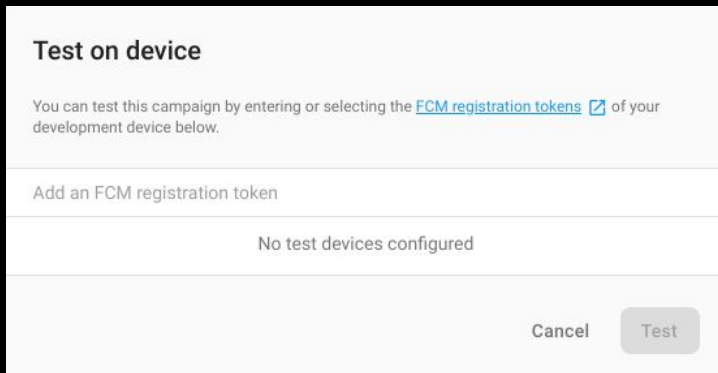
Add an FCM registration token

No test devices configured

Cancel Test

Test the config

- You will need to add [FCM tokens](#) generated by react-native-firebase here in order to test notifications



The screenshot shows a 'Test on device' dialog box. At the top, it says 'Test on device'. Below that, a message states: 'You can test this campaign by entering or selecting the [FCM registration tokens](#) of your development device below.' There is a text input field with the placeholder 'Add an FCM registration token'. Below the input field, it says 'No test devices configured'. At the bottom right, there are two buttons: 'Cancel' and 'Test'.

- Notifications must be tested on real device

Test the config

- Use Postman to send push notifications

FCM Message POST Examples (0) ▾

POST ▾ https://fcm.googleapis.com/fcm/send Params Send ▾ Save ▾

Authorization Headers (2) Body ● Pre-request Script Tests Code

	Key	Value	Description	...	Bulk Edit	Presets ▾
<input checked="" type="checkbox"/>	Authorization	key=AAAAqfBdztY:APA91bHOyacTO4hnCBJuniZWlxIKW...				
<input checked="" type="checkbox"/>	Content-Type	application/json				
	New key	Value	Description			

Body Cookies Headers (14) Test Results Status: 200 OK Time: 861 ms

Pretty Raw Preview Save Response

```
{"multicast_id":4940192930731003050,"success":1,"failure":0,"canonical_ids":0,"results":[{"message_id":"0:1511509224900645%fc67cdfacccfb49c"}]}
```

InApp purchases

How does it work

- Inapp payments have different types (subscriptions/products) and pricings (several tiers)
- We create these products in Apple / Play store admin
- Both Apple and Google charge a percentage of the payment (i.e. we don't get the whole amount)

Create subscription in Google Play

1

In-app products

Development tools

Release management

Store presence

Store listing

Store listing experiments

Pricing & distribution

Content rating

MANAGED PRODUCTS ?
0 active, 0 inactive

SUBSCRIPTIONS ? 2

Subscription settings

Search by product name or ID

3

CREATE SUBSCRIPTION

Name & ID	Price	Billing period	Active ?	Last updated	Status
Premium předplatné (com.usertechnologies.selected.android.pr	CZK 20.00	3 months		Jun 27, 2018	Active
Premium předplatné (com.usertechnologies.selected.android.pr	CZK 247.00	Monthly		Oct 17, 2018	Active

Create subscription in AppStore

App Store Connect

My Apps

#Selected

Martin Doubek
User Technologies s.r.o.

App Store

1 Features

TestFlight

Activity

App Analytics

Sales and Trends

IN-APP PURCHASES

2 In-App Purchases

App Store Promotions

SUBSCRIPTION GROUPS

Subscriptions

VIEW ALL GROUPS

In-App Purchases

In-App Purchase (1) + 3

Search

App-Specific Shared Secret

Reference Name ^	Type	Product ID	Status
Premium subscription	Auto-Renewable Subscription	com.usertechologies.selected.ios.premium	Approved

Create subscription in AppStore

- Clear information about subscription must be provided
 - In the AppStore app's description
 - In the application UI

Information you must provide

- Title of publication or service
- Length of subscription (time period and content or services provided during each subscription period)
- Payment will be charged to iTunes Account at confirmation of purchase
- Subscription automatically renews unless auto-renew is turned off at least 24-hours before the end of the current period
- Account will be charged for renewal within 24-hours prior to the end of the current period, and identify the cost of the renewal
- Subscriptions may be managed by the user and auto-renewal may be turned off by going to the user's Account Settings after purchase
- Any unused portion of a free trial period, if offered, will be forfeited when the user purchases a subscription to that publication, where applicable
- A link to the terms of use

React Native IAP

- <https://github.com/dooboolab/react-native-iap>
- Define products

```
const premiumSubscriptionId = Platform.select({  
  ios: 'com.usertechologies.selected.ios.premium',  
  android: 'com.usertechologies.selected.android.premium',  
})
```

React Native IAP

```
const buyPremiumSubscriptionEpic = (action$: any) =>
  action$.ofType('BUY_PREMIUM_SUBSCRIPTION').mergeMap(action => {
    const observable = Observable.create(observer => {
      RNiap.buySubscription(appConfig.premiumSubscriptionId)
        .then(transaction => {
          Api.subscription({purchaseToken: transaction.transactionReceipt})
        })
      ...
    })
  })
```

React Native IAP

```
const buyPremiumSubscriptionEpic = (action$: any) =>
  action$.ofType('BUY_PREMIUM_SUBSCRIPTION').mergeMap(action => {
    const observable = Observable.create(observer => {
      RNiap.buySubscription(appConfig.premiumSubscriptionId)
        .then(transaction => {
          Api.subscription({purchaseToken: transaction.transactionReceipt})
        })
      ...
    })
  })
```

React Native IAP

- You must test the in-app purchases on a real device.
Purchases will always fail on iOS simulator!
- Not supported by Expo

Questions?

Projects



Sources

- <https://medium.com/ios-development-tips-and-tricks/working-with-ios-in-app-purchases-e4b55491479b>
- <https://brightinventions.pl/blog/dont-let-your-ios-app-be-rejected/>