무료 전자 책

배우기

react-native

Free unaffiliated eBook created from
Stack Overflow contributors.

#react-native

1:

React Native JavaScript . React UI .

React Native " ", "HTML5 " " " . Objective-C Java . React Native iOS Android UI . JavaScript React .

.

-
-
- GitHub

: React Native

Examples

Mac

brew

.

```
/usr/bin/ruby -e "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Xcode IDE

Mac App Store .

https://developer.apple.com/download/

: **Xcode.app Xcode-beta.app** `xcodebuild` . .

```
sudo xcode-select -switch /Applications/Xcode.app/Contents/Developer/
```

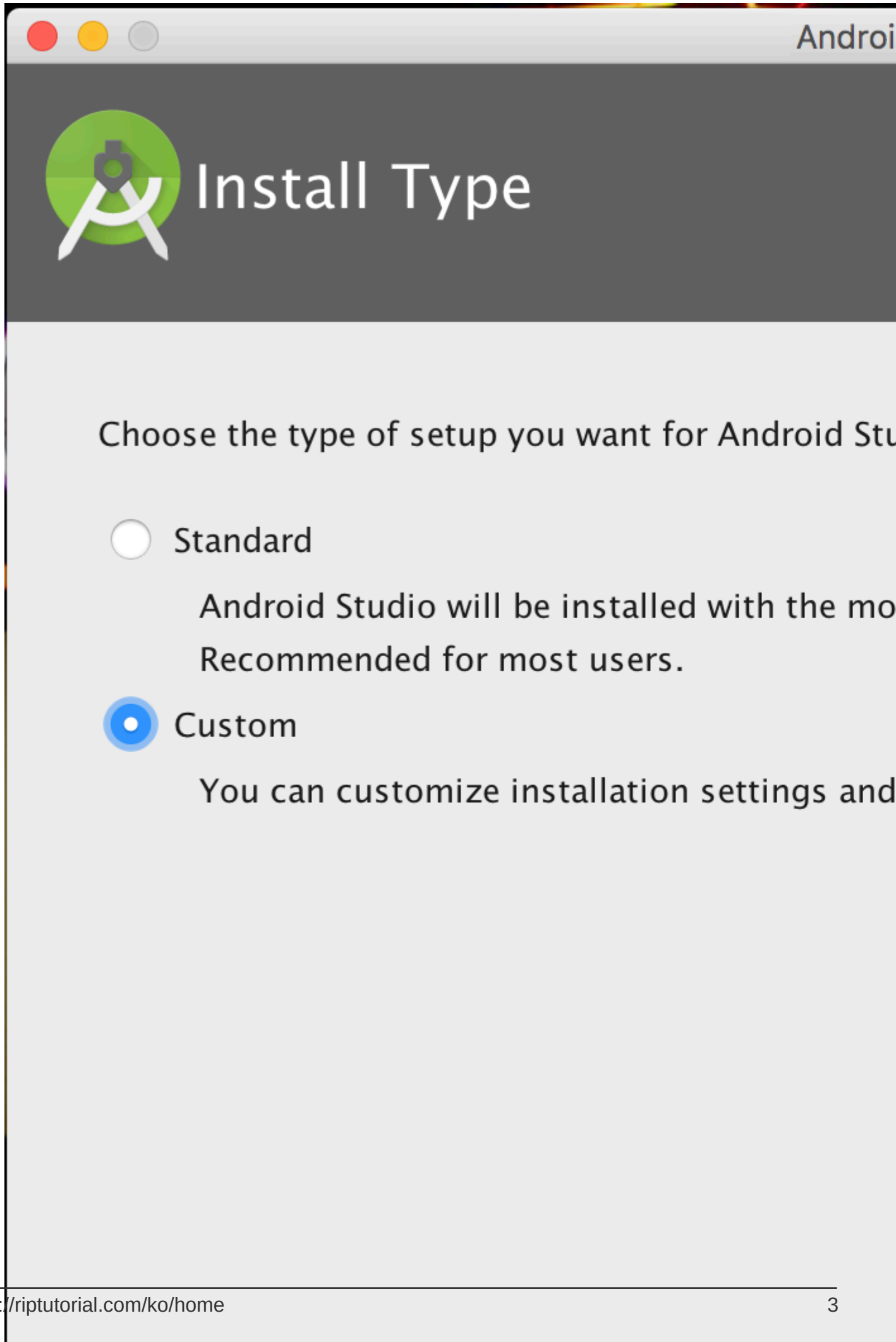
Android

- **git**

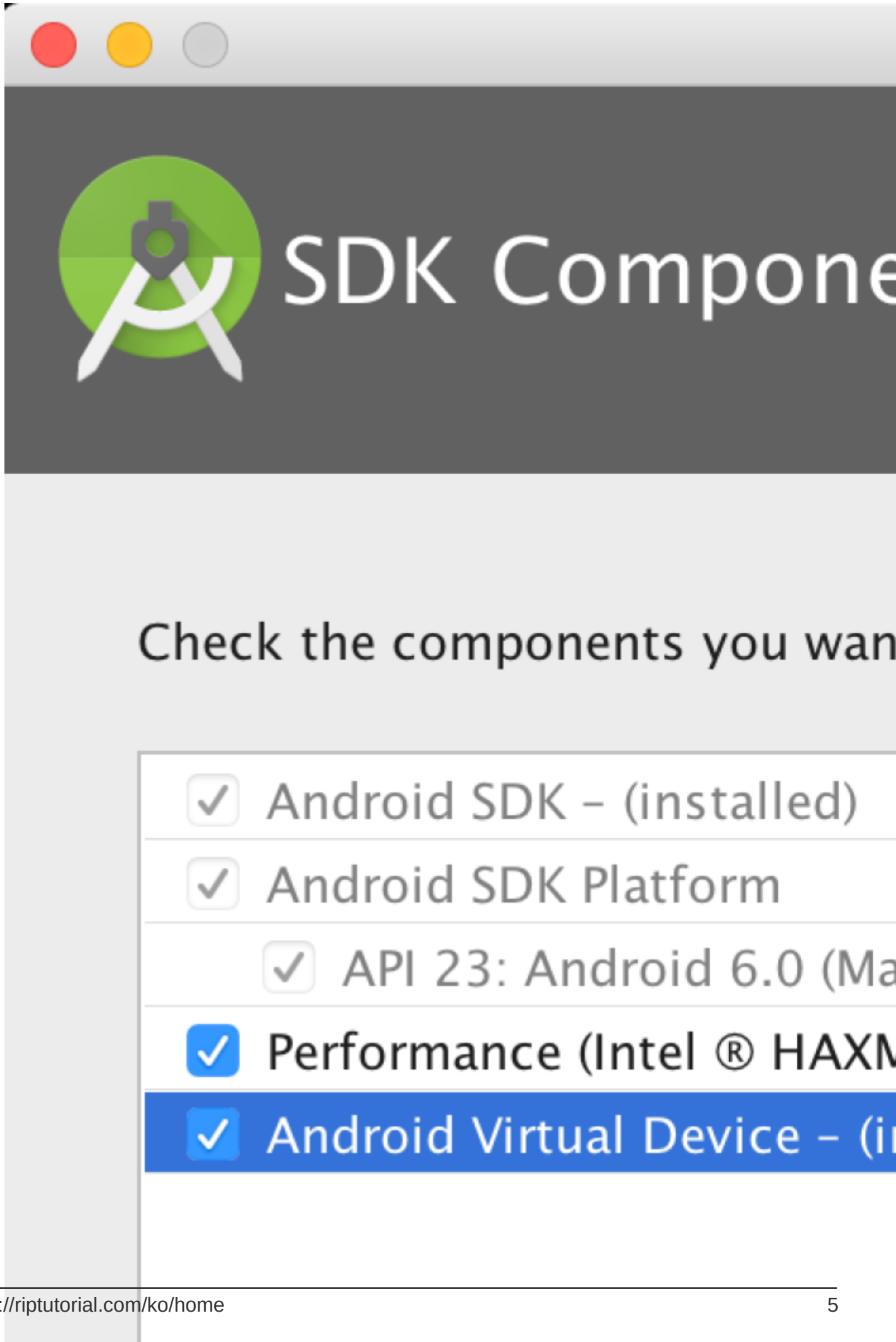
  * Xcode , Git , .

  ```
  brew install git
  ```

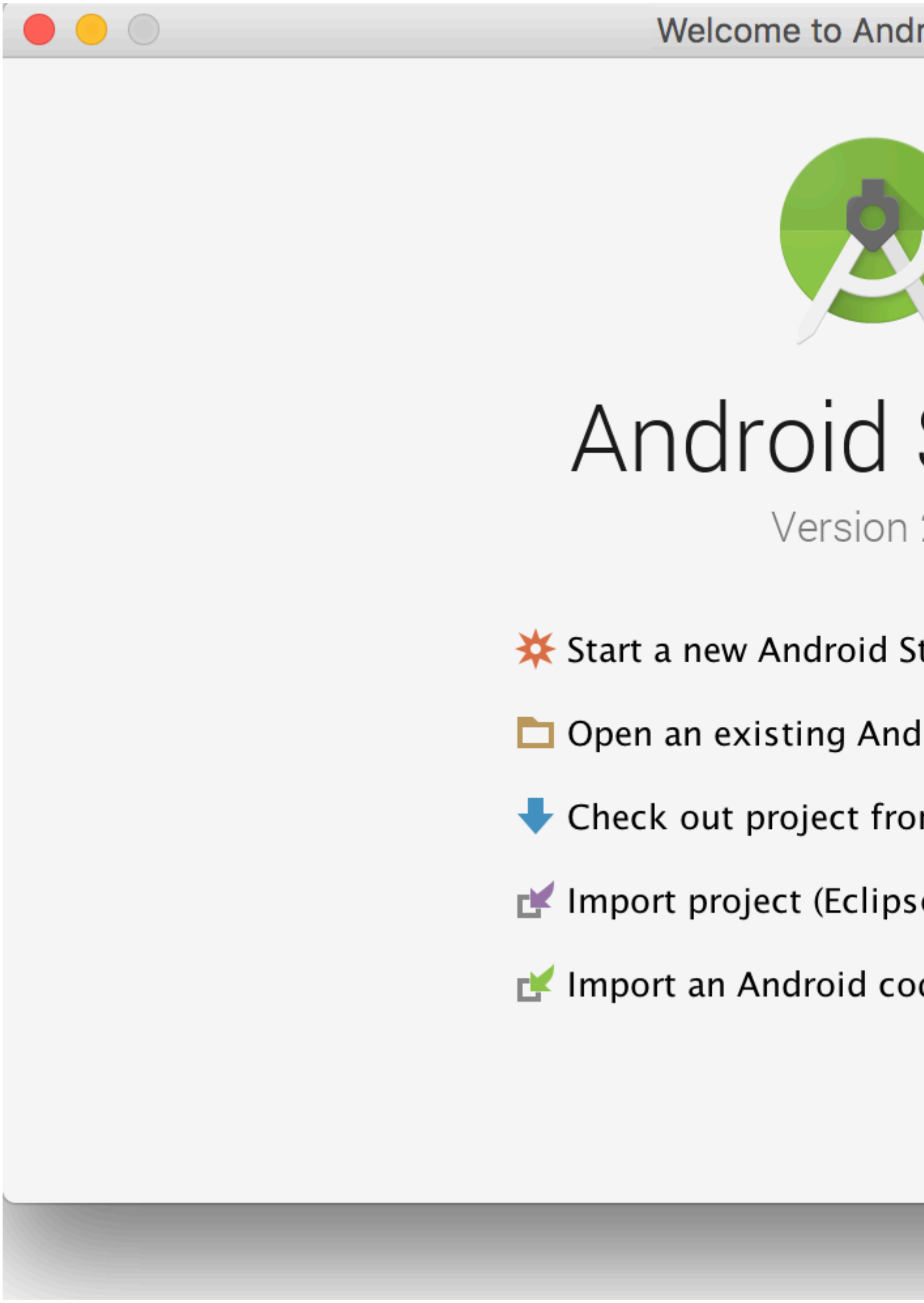
- JDK
- Android Studio


Androi
Install Type
Choose the type of setup you want for Android Stu
Standard
Android Studio will be installed with the mo
Recommended for most users.
Custom
You can customize installation settings and

Android


SDK Compone
Check the components you wan
Android SDK – (installed)
Android SDK Platform
API 23: Android 6.0 (Ma
Performance (Intel ® HAXM
Android Virtual Device – (i

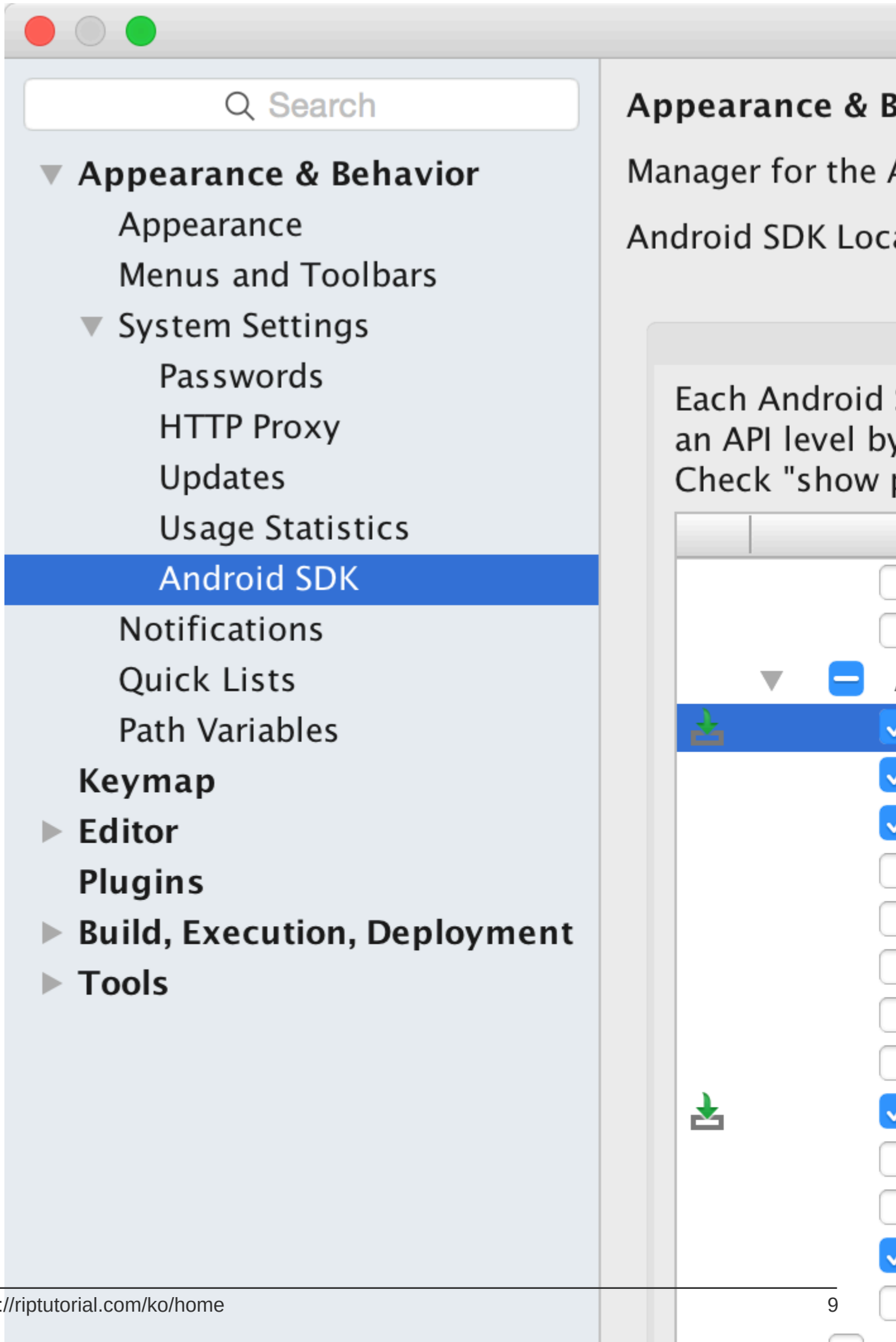
Android Studio -> SDK .


Welcome to Andr
Android S
Version
Start a new Android St
Open an existing And
Check out project fro
Import project (Eclips
Import an Android co

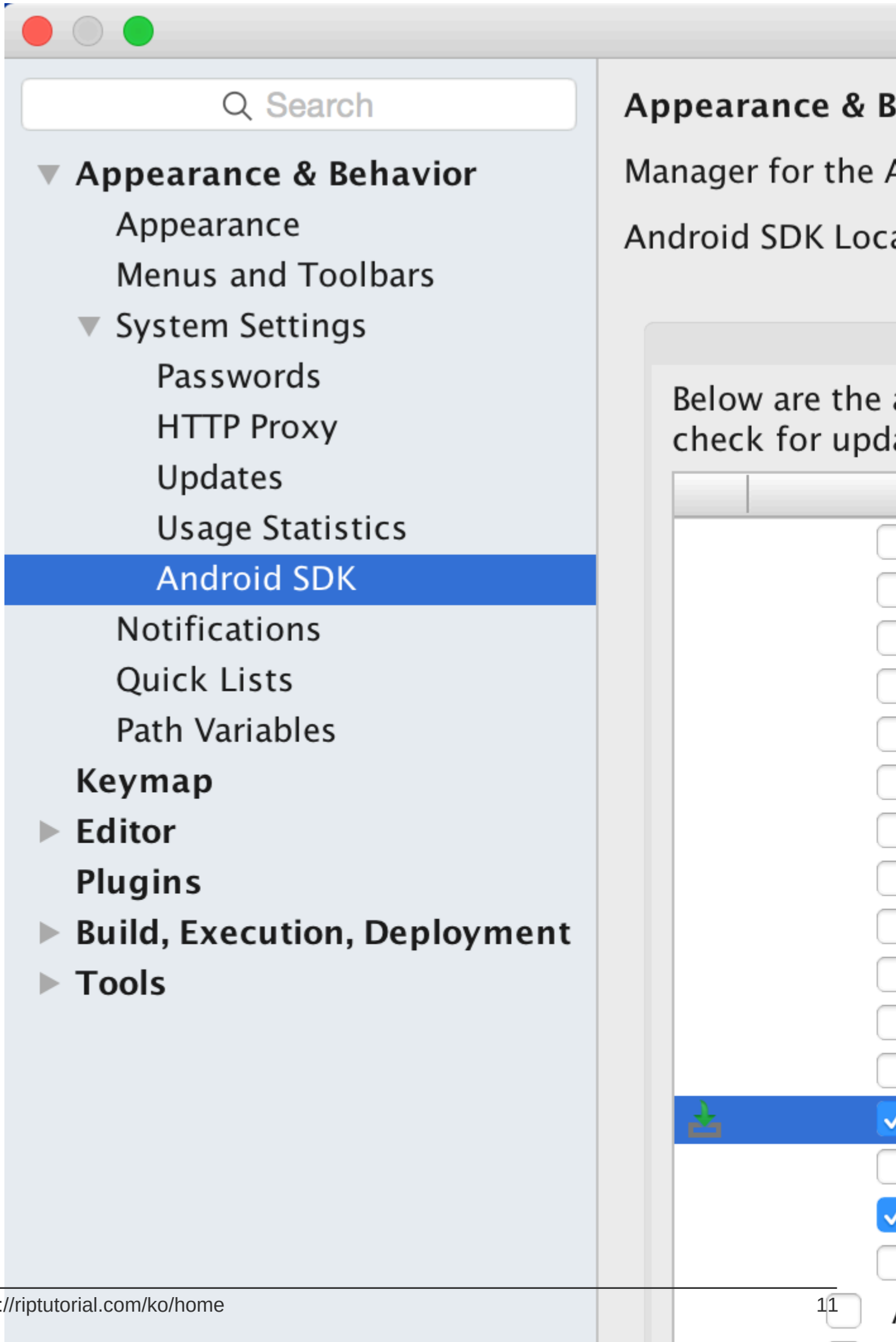
SDK Android 6.0 (Marshmallow) Google API, Intel x86 Atom , Intel x86 Atom_64 Google API Intel x86 Atom_64 .

Google API Intel x86 Atom_64 .

Google API Intel x86 Atom_64 .


Search
Appearance & Behavior
Appearance
Menus and Toolbars
System Settings
Passwords
HTTP Proxy
Updates
Usage Statistics
Android SDK
Notifications
Quick Lists
Path Variables
Keymap
Editor
Plugins
Build, Execution, Deployment
Tools
Appearance & B
Manager for the A
Android SDK Loca
Each Android
an API level by
Check "show

SDK Android SDK Android SDK 23.0.1 .


Search
Appearance & Behavior
Appearance
Menus and Toolbars
System Settings
Passwords
HTTP Proxy
Updates
Usage Statistics
Android SDK
Notifications
Quick Lists
Path Variables
Keymap
Editor
Plugins
Build, Execution, Deployment
Tools
Appearance & B
Manager for the A
Android SDK Loca
Below are the
check for upd

- `ANDROID_HOME`

  ANDROID_HOME Android SDK . ~ / .bashrc, ~ / .bash_profile ( ) :

  Android Studio SDK . / usr / local / opt / android-sdk

```
export ANDROID_HOME=~/Library/Android/sdk
```

Mac

iOS Xcode, android node.js, React Native Watchman Android Studio .

Homebrew node watchman .

```
brew install node
brew install watchman
```

(Watchman) . . .

Node npm React Native .

```
npm install -g react-native-cli
```

sudo .

```
sudo npm install -g react-native-cli.
```

iOS Xcode Mac App Store . Android Studio Android Studio .

Java Gradle Daemon .

React

React Native "AwesomeProject" React Native Reactive-native Run-Ios .

```
react-native init AwesomeProject
cd AwesomeProject
react-native run-ios
```

iOS . react-native run-ios Xcode Nuclide .

.

- index.ios.js index.android.js .
- iOS Command + R ! !

! React Native .

: - React-Native

Windows

: Windows iOS Android .

Windows . .

/

- 10
- (: Powershell Windows)
- Chocolatey (PowerShell)
- JDK (8)
- Android Studio
- HAXM Intel (,)

1)

.

```
choco install nodejs.install
choco install python2
```

npm .

```
npm install -g react-native-cli
```

react-native . 4 . . `C:\Program Files (x86)\Nodist\v-x64\6.2.2` . `C:\Users\admin\AppData\Roaming\npm`

2)

Step by Step .

.

[ ] "" -> -> ->

"Path" 1 react-native .

ANDROID_HOME . " " "ANDROID_HOME" android sdk .

.

3) .

```
react-native init ProjectName
```

4) . android studio . .

```
cd ProjectName
```

```
react-native run-android
```

. . Android Studio SDK .

!

UI r . ctrl + m .

Linux (Ubuntu)

1) Setup Node.JS

nodeJS .

```
curl -sL https://deb.nodesource.com/setup_5.x | sudo -E bash -

sudo apt-get install nodejs
```

```
sudo ln -s /usr/bin/nodejs /usr/bin/node
```

NodeJS :

```
curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -
sudo apt-get install -y nodejs
```

```
curl -sL https://deb.nodesource.com/setup_7.x | sudo -E bash -
sudo apt-get install -y nodejs
```

.

```
node -v
```

npm .

```
sudo npm install -g react-native-cli
```

2)

```
sudo apt-get install lib32stdc++6 lib32z1 openjdk-7-jdk
```

3) Android Studio :

Android SDK Android Studio

```
http://developer.android.com/sdk/index.html
```

Android SDK e ENV

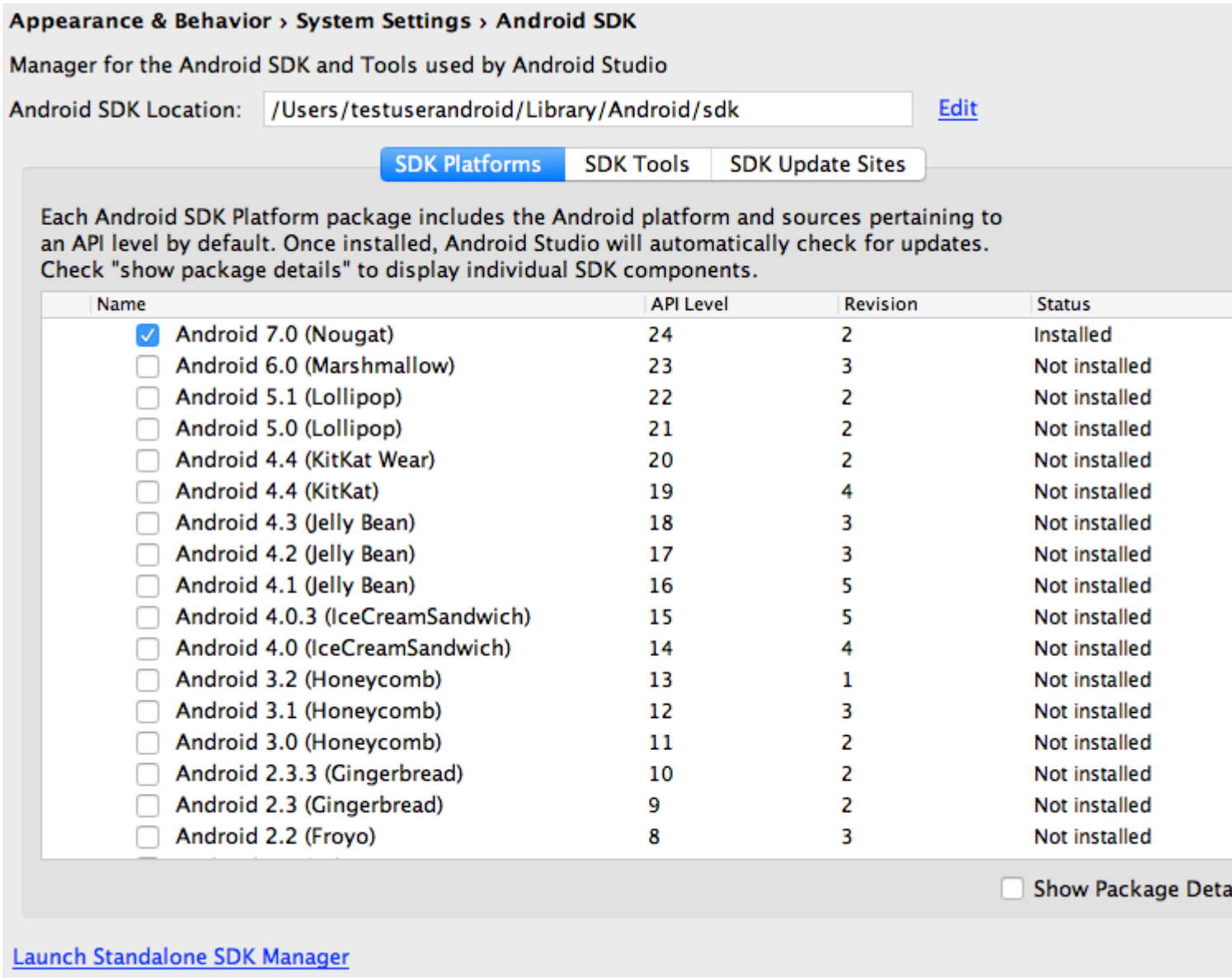
```
export ANDROID_HOME=/YOUR/LOCAL/ANDROID/SDK
export PATH=$PATH:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools
```

4) :

.

```
android
```

SDK Manager "SDK Platforms" "Android 7.0 (Nougat)" . "" .



5)

```
react-native init ReactNativeDemo && cd ReactNativeDemo
```

Obs : `android/app/build.gradle` Android SDK Build Tools .

```
android {
    compileSdkVersion XX
    buildToolsVersion "XX.X.X"
...
```

6)

AVD . .

```
android avd
```

.

.

```
react-native run-android
react-native start
```

: https://riptutorial.com/ko/react-native/topic/857/--

2: Android -

Examples

Android

```
BackAndroid.addEventListener('hardwareBackPress', function() {
    if (!this.onMainScreen()) {
        this.goBack();
        return true;
    }
    return false;
});
```

: `this.onMainScreen()` `this.goBack()` . (https://github.com/immidi/react-native/commit/ed7e0fb31d842c63e8b8dc77ce795fac86e0f712)

BackAndroid

React Native `BackAndroid` `Navigator` .

`componentWillMount` . , . .

`BackAndroid` `Navigator`

```
import React, { Component } from 'react'; // eslint-disable-line no-unused-vars

import {
  BackAndroid,
  Navigator,
} from 'react-native';

import SceneContainer from './Navigation/SceneContainer';
import RouteMapper from './Navigation/RouteMapper';

export default class AppContainer extends Component {

  constructor(props) {
    super(props);

    this.navigator;
  }

  componentWillMount() {
    BackAndroid.addEventListener('hardwareBackPress', () => {
      if (this.navigator && this.navigator.getCurrentRoutes().length > 1) {
        this.navigator.pop();
        return true;
      }
      return false;
    });
  }
```

```
  renderScene(route, navigator) {
    this.navigator = navigator;

    return (
      <SceneContainer
        title={route.title}
        route={route}
        navigator={navigator}
        onBack={() => {
          if (route.index > 0) {
            navigator.pop();
          }
        }}
        {...this.props} />
    );
  }

  render() {
    return (
      <Navigator
        initialRoute={<View />}
        renderScene={this.renderScene.bind(this)}
        navigationBar={
          <Navigator.NavigationBar
            style={{backgroundColor: 'gray'}}
            routeMapper={RouteMapper} />
        } />
    );
  }
};
```

BackHandler

BackAndroid . BackAndroid BackHandler .

```
import { BackHandler } from 'react-native';

{...}
  ComponentWillMount(){
    BackHandler.addEventListener('hardwareBackPress',()=>{
      if (!this.onMainScreen()) {
        this.goBack();
        return true;
      }
      return false;
    });
  }
```

BackHandler (BackAndroid deprecated Navigator)

. . 2 :

1. .
2. .

1 :

```
import { BackHandler } from 'react-native';

constructor(props) {
    super(props)
    this.handleBackButtonClick = this.handleBackButtonClick.bind(this);
}

componentWillMount() {
    BackHandler.addEventListener('hardwareBackPress', this.handleBackButtonClick);
}

componentWillUnmount() {
    BackHandler.removeEventListener('hardwareBackPress', this.handleBackButtonClick);
}

handleBackButtonClick() {
    this.props.navigation.goBack(null);
    return true;
}
```

: componentWillUnmount .

2 :

.

: .

Android - : https://riptutorial.com/ko/react-native/topic/4668/android-----

3: Android APK

CLI APK ( ) .

: ( ). apk .

: https://facebook.github.io/react-native/docs/signed-apk-android.html

Examples

APK

```
keytool -genkey -v -keystore my-app-key.keystore -alias my-app-alias -keyalg RSA -keysize 2048
-validity 10000
```

.

.

```
react-native bundle --platform android --dev false --entry-file index.android.js \
--bundle-output android/app/src/main/assets/index.android.bundle \
--assets-dest android/app/src/main/res/
```

gradle

```
cd android && ./gradlew assembleRelease
```

APK

APK . -r app ( ).

```
adb install -r ./app/build/outputs/apk/app-release-unsigned.apk
```

APK .

```
./app/build/outputs/apk/app-release.apk
```

Android APK : https://riptutorial.com/ko/react-native/topic/8964/android---apk-

4: HTTP

- fetch (url, options) [. then (...) [. catch (...)]]

- Fetch API HTTP API. .
- XMLHttpRequest API HTTP ApiSauce .
- Websocket API "" .

Examples

```
var ws = new WebSocket('ws://host.com/path');

ws.onopen = () => {
  // connection opened

  ws.send('something'); // send a message
};

ws.onmessage = (e) => {
  // a message was received
  console.log(e.data);
};

ws.onerror = (e) => {
  // an error occurred
  console.log(e.message);
};

ws.onclose = (e) => {
  // connection closed
  console.log(e.code, e.reason);
};
```

HTTP API

Fetch . : https://github.com/github/fetch/issues/89 .

XMLHttpRequest https://developer.mozilla.org/en-US/docs/Web/Events/progress .

```
fetch('https://mywebsite.com/mydata.json').then(json => console.log(json));

fetch('/login', {
  method: 'POST',
  body: form,
  mode: 'cors',
  cache: 'default',
}).then(session => onLogin(session), failure => console.error(failure));
```

MDN .

XMLHttpRequest

```
var request = new XMLHttpRequest();
request.onreadystatechange = (e) => {
  if (request.readyState !== 4) {
    return;
  }

  if (request.status === 200) {
    console.log('success', request.responseText);
  } else {
    console.warn('error');
  }
};

request.open('GET', 'https://mywebsite.com/endpoint/');
request.send();
```

fetch API Redux Promise

Redux React-Native . fetch API redux-thunk .

```
export const fetchRecipes = (action) => {
  return (dispatch, getState) => {
    fetch('/recipes', {
        method: 'POST',
        headers: {
          'Accept': 'application/json',
          'Content-Type': 'application/json'
        },
        body: JSON.stringify({
          recipeName,
          instructions,
          ingredients
        })
    })
    .then((res) => {
      // If response was successful parse the json and dispatch an update
      if (res.ok) {
        res.json().then((recipe) => {
          dispatch({
            type: 'UPDATE_RECIPE',
            recipe
          });
        });
      } else {
        // response wasn't successful so dispatch an error
        res.json().then((err) => {
          dispatch({
            type: 'ERROR_RECIPE',
            message: err.reason,
            status: err.status
          });
        });
      }
    })
    .catch((err) => {
      // Runs if there is a general JavaScript error.
      dispatch(error('There was a problem with the request.'));
    });
  };
```

```
};
```

Socket.io

socket.io-client .

```
npm i socket.io-client --save
```

```
import SocketIOClient from 'socket.io-client/dist/socket.io.js'
```

.

```
constructor(props){
    super(props);
    this.socket = SocketIOClient('http://server:3000');
  }
```

. 5 ping (ping) . .

```
_sendPing(){
    //emit a dong message to socket server
    socket.emit('ding');
}

_getReply(data){
    //get reply from socket server, log it to console
    console.log('Reply from server:' + data);
}
```

.

```
constructor(props){
    super(props);
    this.socket = SocketIOClient('http://server:3000');

    //bind the functions
    this._sendPing = this._sendPing.bind(this);
    this._getReply = this._getReply.bind(this);
}
```

_getReply . _getReply socket . .

```
this.socket.on('dong', this._getReply);
```

'dong' .

HTTP

library axios .

. axios.js :

```
import * as axios from 'axios';

var instance = axios.create();
instance.defaults.baseURL = serverURL;
instance.defaults.timeout = 20000;]
//...
//and other options

export { instance as default };
```

.

'' .

```
import axios from '../axios';
import {
    errorHandling
} from '../common';

const UserService = {
        getCallToAction() {
        return axios.get('api/user/dosomething').then(response => response.data)
            .catch(errorHandling);
    },
}
export default UserService;
```

: axios-mock-adapter .

. axois'es . axios.js .

```
import MockAdapter from 'axios-mock-adapter';

var mock = new MockAdapter(instance);
mock.onAny().reply(500);
```

.

Redux

. .

interceptors.js .

```
export function getAuthToken(storeContainer) {
    return config => {
        let store = storeContainer.getState();
        config.headers['Authorization'] = store.user.accessToken;
        return config;
    };
}
```

.

```
axios.interceptors.request.use(getAuthToken(this.state.store));
```

.

Axios .

HTTP : https://riptutorial.com/ko/react-native/topic/2375/http-

5: ListView RefreshControl

:

RefreshControl : https://facebook.github.io/react-native/docs/refreshcontrol.html

ListView : https://facebook.github.io/react-native/docs/listview.html

Examples

```
  _refreshControl(){
    return (
      <RefreshControl
        refreshing={this.state.refreshing}
        onRefresh={()=>this._refreshListView()} />
    )
  }
```

: (true, false).

onRefresh : ListView / ScrollView .

onRefresh

```
  _refreshListView(){
    //Start Rendering Spinner
    this.setState({refreshing:true})
    this.state.cars.push(
      {name:'Fusion',color:'Black'},
      {name:'Yaris',color:'Blue'}
    )
    //Updating the dataSource with new data
    this.setState({ dataSource:
        this.state.dataSource.cloneWithRows(this.state.cars) })
    this.setState({refreshing:false}) //Stop Rendering Spinner
  }
```

dataSource . fetch async / await .

ListView

RefreshControl ScrollView ListView . ListView .

```
'use strict'
import React, { Component } from 'react';
import { StyleSheet, View, ListView, RefreshControl, Text } from 'react-native'


class RefreshControlExample extends Component {
  constructor () {
    super()
```

```
    this.state = {
      refreshing: false,
      dataSource: new ListView.DataSource({
        rowHasChanged: (row1, row2) => row1 !== row2 }),
      cars : [
        {name:'Datsun',color:'White'},
        {name:'Camry',color:'Green'}
      ]
    }
  }

   componentWillMount(){
     this.setState({ dataSource:
       this.state.dataSource.cloneWithRows(this.state.cars) })
   }

  render() {
    return (
      <View style={{flex:1}}>
        <ListView
          refreshControl={this._refreshControl()}
          dataSource={this.state.dataSource}
          renderRow={(car) => this._renderListView(car)}>
        </ListView>
      </View>
    )
  }

  _renderListView(car){
    return(
      <View style={styles.listView}>
        <Text>{car.name}</Text>
        <Text>{car.color}</Text>
      </View>
    )
  }

  _refreshControl(){
    return (
      <RefreshControl
        refreshing={this.state.refreshing}
        onRefresh={()=>this._refreshListView()} />
    )
  }

  _refreshListView(){
    //Start Rendering Spinner
    this.setState({refreshing:true})
    this.state.cars.push(
      {name:'Fusion',color:'Black'},
      {name:'Yaris',color:'Blue'}
    )
    //Updating the dataSource with new data
    this.setState({ dataSource:
        this.state.dataSource.cloneWithRows(this.state.cars) })
    this.setState({refreshing:false}) //Stop Rendering Spinner
  }

}

const styles = StyleSheet.create({
```

```
  listView: {
    flex: 1,
    backgroundColor:'#fff',
    marginTop:10,
    marginRight:10,
    marginLeft:10,
    padding:10,
    borderWidth:.5,
    borderColor:'#dddddd',
    height:70
  }

})

module.exports = RefreshControlExample
```

ListView RefreshControl : https://riptutorial.com/ko/react-native/topic/6672/listview--refreshcontrol

6: WebView

Webview HTML . .

Examples

webview

```
import React, { Component } from 'react';
import { WebView } from 'react-native';

class MyWeb extends Component {
  render() {
    return (
      <WebView
        source={{uri: 'https://github.com/facebook/react-native'}}
        style={{marginTop: 20}}
       />
    );
  }
}
```

WebView : https://riptutorial.com/ko/react-native/topic/8763/webview

7:

Examples

```
import React, { Component } from 'react'
import { View, Text, AppRegistry } from 'react-native'

class Example extends Component {
  render () {
    return (
      <View>
        <Text> I'm a basic Component </Text>
      </View>
    )
  }
}

AppRegistry.registerComponent('Example', () => Example)
```

.

```
import React, { Component } from 'react'
import { View, Text, AppRegistry } from 'react-native'

class Example extends Component {
  constructor (props) {
    super(props)
    this.state = {
      name: "Sriraman"
    }
  }
  render () {
    return (
      <View>
        <Text> Hi, {this.state.name}</Text>
      </View>
    )
  }
}

AppRegistry.registerComponent('Example', () => Example)
```

Stateless Components . **Dumb Components** . stateful .

`const name = ({props}) => ( ... ) .` .

`App Title` .

```
import React from 'react'
import { View, Text, AppRegistry } from 'react-native'

const Title = ({Message}) => (
  <Text>{Message}</Text>
)
```

```
const App = () => (
  <View>
    <Title title='Example Stateless Component' />
  </View>
)

AppRegistry.registerComponent('App', () => App)
```

. .

: https://riptutorial.com/ko/react-native/topic/5532/-

8: (Android)

:

`Could not connect to development server` => . `adb reverse tcp:8081 tcp:8081` , (adb). `react-native start`

Examples

Android .

1. `adb devices`
   - ? USB .
2. `adb reverse tcp:8081 tcp:8081` :
   - React-Native . (: **`Android Version 5.`**)
3. `react-native run-android` :
   - .
4. `react-native start` :
   - (). React-native .

(Android) : https://riptutorial.com/ko/react-native/topic/5135/----android--

9:

Examples

(IOS)

http://facebook.github.io/react-native/docs/native-modules-ios.html

API React Native . JavaScript Objective-C, Swift C ++ , , .

`RCTBridgeModule` Objective-C .

Xcode **Cocoa Touch Class** (*: NativeModule*) `NSObject` `NSObject` `Objective-C` .

`NativeModuleEx.h` `NativeModuleEx.m` .

`RCTBridgeModule.h` `NativeModuleEx.h` .

```
#import <Foundation/Foundation.h>
#import "RCTBridgeModule.h"

@interface NativeModuleEx : NSObject <RCTBridgeModule>

@end
```

`NativeModuleEx.m` .

```
#import "NativeModuleEx.h"

@implementation NativeModuleEx

RCT_EXPORT_MODULE();

RCT_EXPORT_METHOD(testModule:(NSString *)string )
{
  NSLog(@"The string '%@' comes from JavaScript! ", string);
}

@end
```

`RCT_EXPORT_MODULE()` , . Objective-C .

`RCT_EXPORT_METHOD()` JavaScript `RCT_EXPORT_METHOD()` JavaScript .

JavaScript .

```
import { NativeModules } from 'react-native';
```

```
var NativeModuleEx = NativeModules.NativeModuleEx;

NativeModuleEx.testModule('Some String !');
```

: https://riptutorial.com/ko/react-native/topic/6155/-

10:

Examples

Navigator React Native . Navigator , .

```
<Navigator
  ref={(navigator) => { this.navigator = navigator }}
  initialRoute={{ id: 'route1', title: 'Route 1' }}
  renderScene={this.renderScene.bind(this)}
  configureScene={(route) => Navigator.SceneConfigs.FloatFromRight}
  style={{ flex: 1 }}
  navigationBar={
    // see "Managing the Navigation Bar" below
    <Navigator.NavigationBar routeMapper={this.routeMapper} />
  }
/>
```

, initialRoute . . .

Navigator renderScene .

```
renderScene(route, navigator) {
  if (route.id === 'route1') {
    return <ExampleScene navigator={navigator} title={route.title} />; // see below
  } else if (route.id === 'route2') {
    return <ExampleScene navigator={navigator} title={route.title} />; // see below
  }
}
```

ExampleScene .

```
function ExampleScene(props) {

  function forward() {
    // this route object will passed along to our `renderScene` function we defined above.
    props.navigator.push({ id: 'route2', title: 'Route 2' });
  }

  function back() {
    // `pop` simply pops one route object off the `Navigator`'s stack
    props.navigator.pop();
  }

  return (
    <View>
      <Text>{props.title}</Text>
      <TouchableOpacity onPress={forward}>
        <Text>Go forward!</Text>
      </TouchableOpacity>
      <TouchableOpacity onPress={back}>
        <Text>Go Back!</Text>
      </TouchableOpacity>
    </View>
```

```
  );
}
```

`configureScene` prop `Navigator` . `route` . .

- Navigator.SceneConfigs.PushFromRight ()
- Navigator.SceneConfigs.FloatFromRight
- Navigator.SceneConfigs.FloatFromLeft
- Navigator.SceneConfigs.FloatFromBottom
- Navigator.SceneConfigs.FloatFromBottomAndroid
- Navigator.SceneConfigs.FadeAndroid
- Navigator.SceneConfigs.HorizontalSwipeJump
- Navigator.SceneConfigs.HorizontalSwipeJumpFromRight
- Navigator.SceneConfigs.VerticalUpSwipeJump
- Navigator.SceneConfigs.VerticalDownSwipeJump

. , iOS `UINavigationController interactivePopGestureRecognizer` .

```
configureScene={(route) => {
  return {
    ...Navigator.SceneConfigs.FloatFromRight,
    gestures: {
      pop: {
        ...Navigator.SceneConfigs.FloatFromRight.gestures.pop,
        edgeHitWidth: Dimensions.get('window').width / 2,
      },
    },
  };
}}
```

NavigationBar

`Navigator` `navigationBar` React . `Navigator.NavigationBar` . `routeMapper` .

`routeMapper` `Title` , `RightButton` `LeftButton` . :

```
const routeMapper = {

  LeftButton(route, navigator, index, navState) {
    if (index === 0) {
      return null;
    }

    return (
      <TouchableOpacity
        onPress={() => navigator.pop()}
        style={styles.navBarLeftButton}
      >
        <Text>Back</Text>
      </TouchableOpacity>
    );
  },

  RightButton(route, navigator, index, navState) {
```

```
    return (
      <TouchableOpacity
        onPress={route.handleRightButtonClick}
        style={styles.navBarRightButton}
      >
        <Text>Next</Text>
      </TouchableOpacity>
   );
  },

  Title(route, navigator, index, navState) {
    return (
      <Text>
        {route.title}
      </Text>
    );
  },
};
```

Navigator , .

.

```
npm install --save react-navigation
```

:

```
import { Button, View, Text, AppRegistry } from 'react-native';
import { StackNavigator } from 'react-navigation';

const App = StackNavigator({
  FirstPage: {screen: FirstPage},
  SecondPage: {screen: SecondPage},
});

class FirstPage extends React.Component {
  static navigationOptions = {
    title: 'Welcome',
  };
  render() {
    const { navigate } = this.props.navigation;

    return (
      <Button
        title='Go to Second Page'
        onPress={() =>
          navigate('SecondPage', { name: 'Awesomepankaj' })
        }
      />
    );
  }
}

class SecondPage extends React.Component {
  static navigationOptions = ({navigation}) => ({
    title: navigation.state.params.name,
  });
```

```
  render() {
    const { goBack } = this.props.navigation;
    return (
      <View>
        <Text>Welcome to Second Page</Text>
        <Button
          title="Go back to First Page"
          onPress={() => goBack()}
        />
      </View>
    );
  }
}
```

- -

npm install --save react-native-router-flux npm install --save react-native-router-flux

<Scene>

      <Scene key="home" component={LogIn} title="Home" initial />

key   .

component   ,

title NavBar 'Home' .

initial ?

     :

```
import React from 'react';
import { Scene, Router } from 'react-native-router-flux';
import LogIn from './components/LogIn';
import SecondPage from './components/SecondPage';

const RouterComponent = () => {
  return (
    <Router>
      <Scene key="login" component={LogIn} title="Login Form" initial />
      <Scene key="secondPage" component={SecondPage} title="Home" />
    </Router>
  );
};

export default RouterComponent;
```

App.js ( ) . .

: https://riptutorial.com/ko/react-native/topic/2559/--

11: API

API . Google . .

`Linking` `react-native` .

```
import {Linking} from 'react-native'
```

Examples

openURL .

```
Linking.openURL(url)
.catch(err => console.error('An error occurred ', err))
```

URL .

```
Linking.canOpenURL(url)
.then(supported => {
  if (!supported) {
    console.log('Unsupported URL: ' + url)
  } else {
    return Linking.openURL(url)
  }
}).catch(err => console.error('An error occurred ', err))
```

URI

	`https://stackoverflow.com`	
	`tel:1-408-555-5555`	
	`mailto:email@example.com`	
SMS	`sms:1-408-555-1212`	
Apple	`http://maps.apple.com/?ll=37.484847,-122.148386`	
	`geo:37.7749,-122.4194`	Google
iTunes	iTunes Link Maker	
	`fb://profile`	
YouTube	`http://www.youtube.com/v/oHg5SJYRHA0`	
	`facetime://user@example.com`	
iOS	`calshow:514300000` [1]	iPhoneDevWiki

[1] 2001 1 1 (UTC?) . API Apple .

URL .

```
componentDidMount() {
  const url = Linking.getInitialURL()
  .then((url) => {
    if (url) {
      console.log('Initial url is: ' + url)
    }
  }).catch(err => console.error('An error occurred ', err))
}
```

iOS Link `RCTLinking` .

Android .

API : https://riptutorial.com/ko/react-native/topic/9687/-api-

12:

Examples

`` .

```
  render() {
    let firstName = 'test';
    let lastName = 'name';
    return (
      <View style={styles.container}>
        <Text>{`${firstName} ${lastName}` } </Text>
      </View>
    );
  }
```

:

: https://riptutorial.com/ko/react-native/topic/10781/--

13:

.

Examples

Jest .

.

```
import 'react-native';
import React from 'react';
import Index from '../index.android.js';

import renderer from 'react-test-renderer';

it('renders correctly', () => {
  const tree = renderer.create(
    <Index />
  );
});
```

.

```
import React, { Component } from 'react';
import {
  AppRegistry,
  StyleSheet,
  Text,
  View
} from 'react-native';

export default class gol extends Component {
  render() {
    return (
      <View>
        <Text>
          Welcome to React Native!
        </Text>
        <Text>
          To get started, edit index.android.js
        </Text>
        <Text>
          Double tap R on your keyboard to reload,{'\n'}
          Shake or press menu button for dev menu
        </Text>
      </View>
    );
  }
}

AppRegistry.registerComponent('gol', () => gol);
```

Jest

- 0.38, - init Jest . package.json .

```
    "scripts": {
    "start": "node node_modules/react-native/local-cli/cli.js start",
    "test": "jest"
    },
    "jest": {
     "preset": "react-native"
    }
```

run npm test or jest run npm test or jest run npm test or jest . :

: https://riptutorial.com/ko/react-native/topic/8281/-

14:

- ;

Examples

Android JS

. Google . js .

console.log ()

`console.log()` . Android .

```
react-native log-android
```

iOS .

```
react-native log-ios
```

.

: https://riptutorial.com/ko/react-native/topic/5105/

15:

. .

Examples

IOS .

```
import React, { Component } from 'react';
import { Text, Navigator, TouchableHighlight } from 'react-native';

export default class NavAllDay extends Component {
  render() {
    return (
      <Navigator
        initialRoute={{ title: 'Awesome Scene', index: 0 }}
        renderScene={(route, navigator) =>
          <Text>Hello {route.title}!</Text>
        }
        style={{padding: 100}}
      />
    );
  }
}
```

Navigator . renderScene . initialRoute .

: https://riptutorial.com/ko/react-native/topic/8279/

16:

Component.render .

Examples

JSX

JSX () .

https://github.com/yannickcr/eslint-plugin-react/blob/master/docs/rules/jsx-no-bind.md :

JSX . . .

jsx :

```
<TextInput
  onChangeValue={  value => this.handleValueChanging(value) }
/>
```

```
<button onClick={ this.handleClick.bind(this) }></button>
```

:

```
<TextInput
  onChangeValue={  this.handleValueChanging }
/>
```

```
<button onClick={ this.handleClick }></button>
```

handleValueChanging .

```
constructor(){
  this.handleValueChanging = this.handleValueChanging.bind(this)
}
```

: https://github.com/andreypopp/autobind-decorator @autobind :

```
@autobind
handleValueChanging(newValue)
{
    //processing event
}
```

: https://riptutorial.com/ko/react-native/topic/10649/--

17:

Examples

```
$ react-native -v
```

```
react-native-cli: 0.2.0
react-native: n/a - not inside a React Native project directory //Output from  different
folder
react-native: react-native: 0.30.0 // Output from the react native project directory
```

RN

app `package.json` .

```
"react-native": "0.32.0"
```

:

```
$ npm install
```

```
$ react-native upgrade
```

```
$ react-native log-android
```

iOS

```
$ react-native log-ios
```

React Native

```
react-native init MyAwesomeProject
```

React Native

```
react-native init --version="0.36.0" MyAwesomeProject
```

Android

```
cd MyAwesomeProject
react-native run-android
```

iOS

```
cd MyAwesomeProject
react-native run-ios
```

React Native Packager

```
$ react-native start
```

React Native . .

8081 . .

```
$ react-native start --port PORTNUMBER
```

Android

Android Android .

```
$ react-native android
```

`android index.android.js .`

: https://riptutorial.com/ko/react-native/topic/2117/--

18:

.

animationType	(' ', ' ', ' ') .
	visiblity bool.
onShow	.
	.
onRequestClose (**android**)	.
onOrientationChange (**IOS**)	.
supportedOrientations (**IOS**)	enum ('portrait', 'portrait-downside', 'landscape', 'landscape-left', 'landscape-right')

Examples

```
import React, { Component } from 'react';
import {
  Modal,
  Text,
  View,
  Button,
  StyleSheet,
} from 'react-native';

const styles = StyleSheet.create({
  mainContainer: {
    marginTop: 22,
  },
  modalContainer: {
    marginTop: 22,
  },
});

class Example extends Component {
  constructor() {
    super();
    this.state = {
      visibility: false,
    };
  }


  setModalVisibility(visible) {
```

```
    this.setState({
      visibility: visible,
    });
  }

  render() {
    return (
      <View style={styles.mainContainer}>
        <Modal
          animationType={'slide'}
          transparent={false}
          visible={this.state.visibility}
        >
          <View style={styles.modalContainer}>
            <View>
              <Text>I'm a simple Modal</Text>
              <Button
                color="#000"
                onPress={() => this.setModalVisibility(!this.state.visibility)}
                title="Hide Modal"
              />
            </View>
          </View>
        </Modal>

        <Button
          color="#000"
          onPress={() => this.setModalVisibility(true)}
          title="Show Modal"
        />
      </View>
    );
  }
}

export default Example;
```

.

```
import React, { Component } from 'react';
import { Text, View, StyleSheet, Button, Modal } from 'react-native';
import { Constants } from 'expo';

export default class App extends Component {
  state = {
    modalVisible: false,
  };

  _handleButtonPress = () => {
    this.setModalVisible(true);
  };

  setModalVisible = (visible) => {
    this.setState({modalVisible: visible});
  }

  render() {
    var modalBackgroundStyle = {
      backgroundColor: 'rgba(0, 0, 0, 0.5)'
    };
```

```
    var innerContainerTransparentStyle = {backgroundColor: '#fff', padding: 20};
    return (
      <View style={styles.container}>
      <Modal
          animationType='fade'
          transparent={true}
          visible={this.state.modalVisible}
          onRequestClose={() => this.setModalVisible(false)}
          >
          <View style={[styles.container, modalBackgroundStyle]}>
            <View style={innerContainerTransparentStyle}>
              <Text>This is a modal</Text>
              <Button title='close'
                onPress={this.setModalVisible.bind(this, false)}/>
            </View>
          </View>
        </Modal>
        <Button
          title="Press me"
          onPress={this._handleButtonPress}
        />

      </View>
    );
  }
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    alignItems: 'center',
    justifyContent: 'center',
    paddingTop: Constants.statusBarHeight,
    backgroundColor: '#ecf0f1',
  }
});
```

: https://riptutorial.com/ko/react-native/topic/8253/

19:

Examples

ListView - . API ListView.DataSource ListView BLOB renderable renderRow .

:

```
getInitialState: function() {
  var ds = new ListView.DataSource({rowHasChanged: (r1, r2) => r1 !== r2});
  return {
    dataSource: ds.cloneWithRows(['row 1', 'row 2']),
  };
},

render: function() {
  return (
    <ListView
      dataSource={this.state.dataSource}
      renderRow={(rowData) => <Text>{rowData}</Text>}
    />
  );
},
```

ListView , , (onEndReached) (onChangeVisibleRows) . .

( ) ListView .

- . rowHasChanged ListView . ListViewDataSource .
- - (pageSize). .

: https://riptutorial.com/ko/react-native/topic/3112/

20: ESLint

ESLint .

Examples

ESLint . ESLint .

javascript, .

ESLint https://github.com/eslint/eslint/tree/master/docs/rules . .eslintr.json " 'eslint : recommended' ESLint . ("extends": ["eslint : recommended"]) ESLint http://eslint.org/docs/developer-guide/development-environment . .

, ES Lint https://github.com/yannickcr/eslint-plugin-react/tree/master/docs/rules . : . : react / display-name react / no-unknown-property . react / jsx-no-bind react / jsx-key .

.

, . https://github.com/intellicode/eslint-plugin-react-native : react-native / no-inline- .

config 'env' . "env": { "browser": true, "es6": true, "amd": true} ,

ESLint .

ESLint : https://riptutorial.com/ko/react-native/topic/10650/--eslint

21:

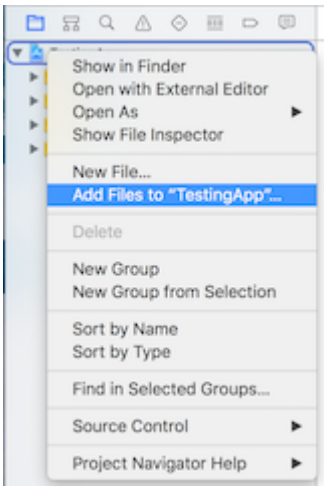
Examples

React Native (Android)

1. `android/app/src/main/assets/fonts/font_name.ttf` .
2. `react-native run-android` Android .
3. `fontFamily: 'font_name'` `fontFamily: 'font_name'`

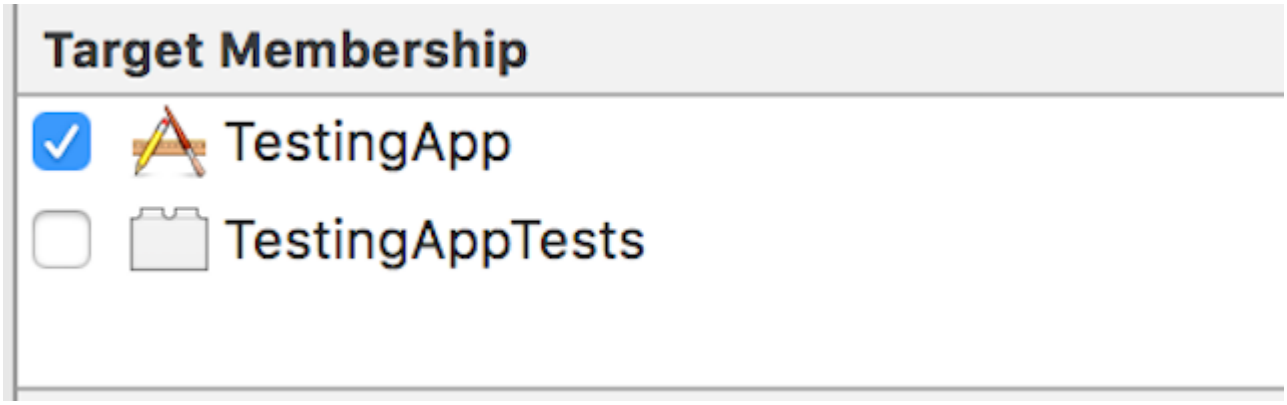
React Native (iOS)

1. Xcode .



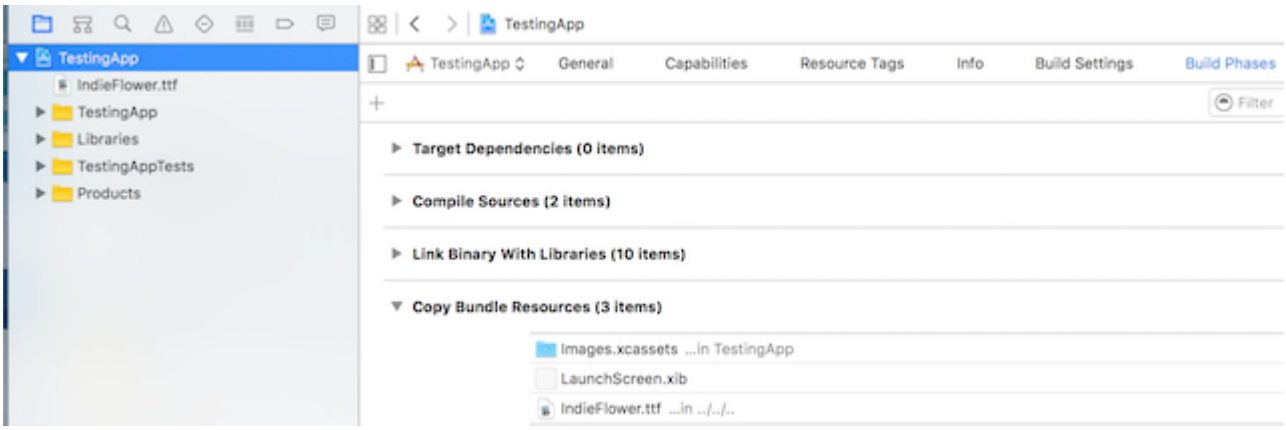
2.

.



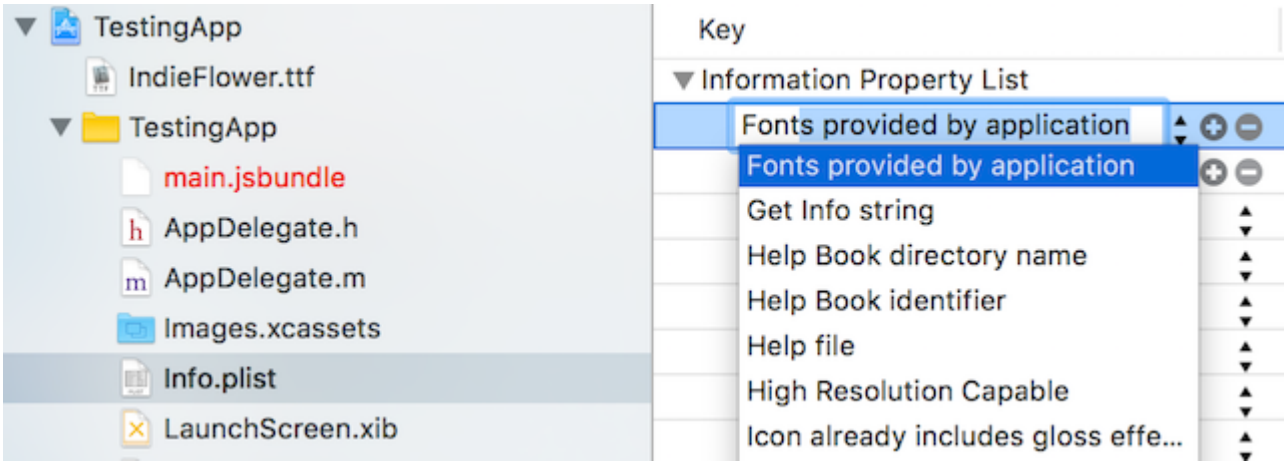
3. Resource

Xcode "Build Phases" "Copy Bundle Resources" . .



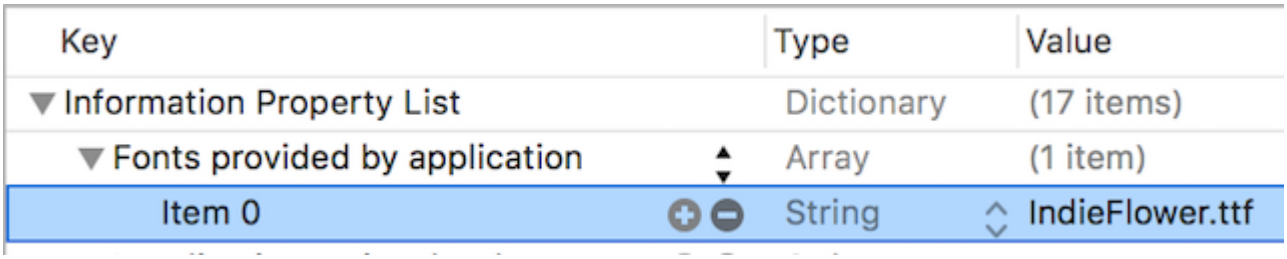
4. Application Plist (Info.plist)

Info.plist " " (+) . " " .



5.

.

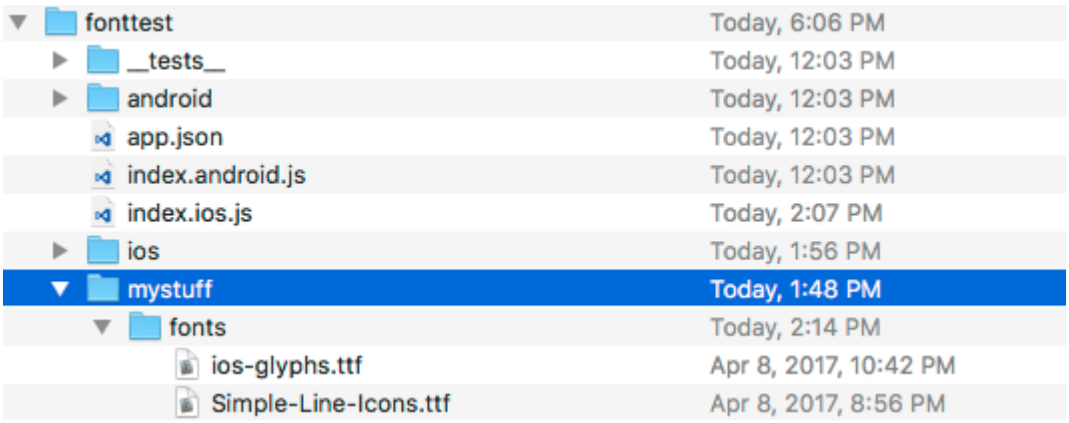


6. .

```
<Text style={{fontFamily:'IndieFlower'}}>
  Welcome to React Native!
</Text>
```

Android IOS

- . :
  - : "mystuff", "font" .



- `package.json` .

```
{
    ...

    "rnpm": {
        "assets": [
          "path/to/fontfolder"
        ]
    },

    ...
}
```

  - package.json "mystuff / fonts".

```
"rnpm": {
  "assets": [
    "mystuff/fonts"
  ]
}
```

- `react-native link` .

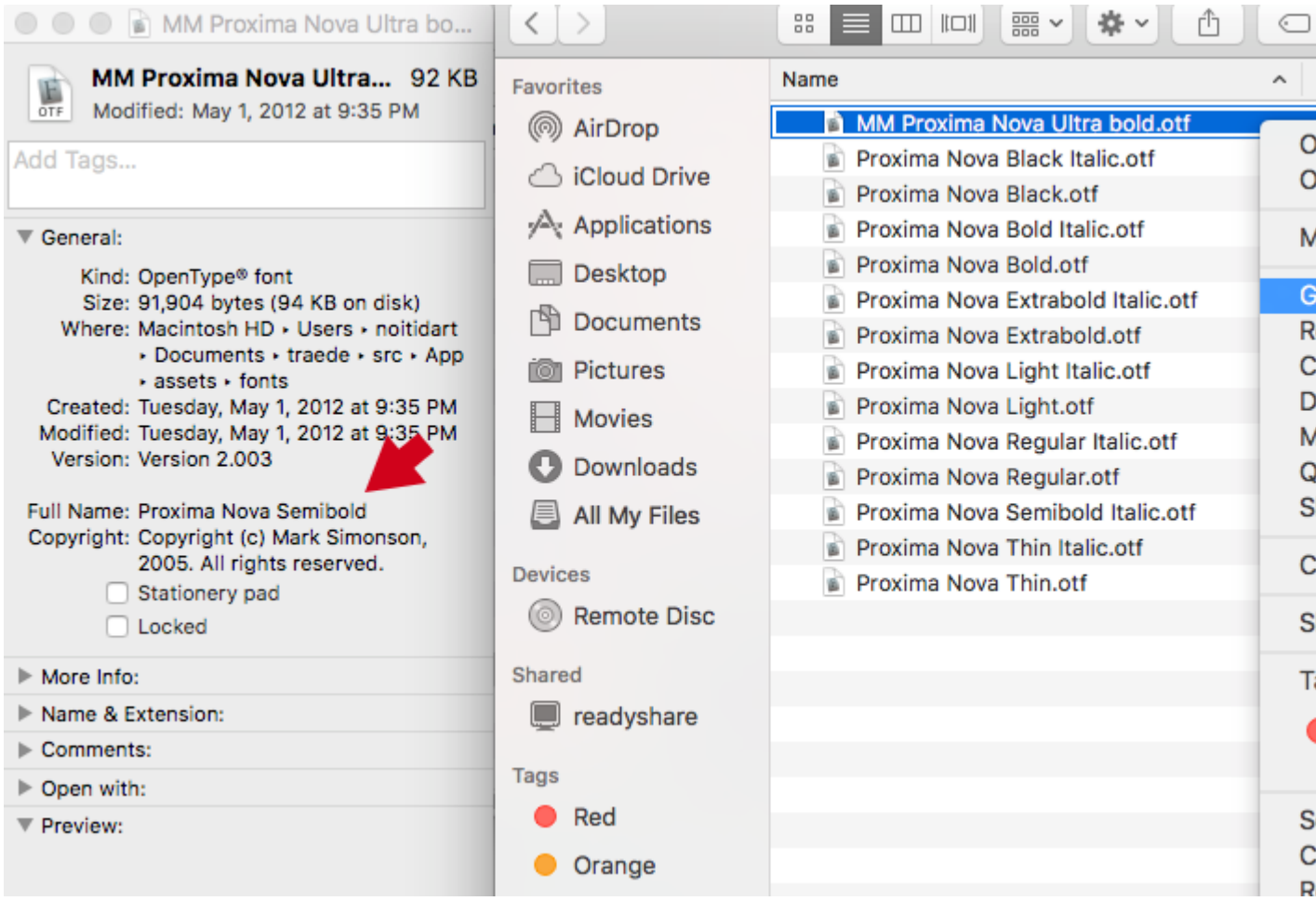
- ```
  <Text style={{ fontFamily: 'FONT-NAME' }}>
      My Text
  </Text>
  ```

  `FONT-NAME` .

  FONT-NAME . : `Roboto-Regular.ttf` `fontFamily: Roboto-Regular` .

  ### iOS

  " " " ". (: https://stackoverflow.com/a/16788493/2529614) `MM Proxima Nova Ultra bold.otf` "Full Name" "Proxima Nova `MM Proxima Nova Ultra bold.otf` " `fontFamily: Proxima Nova Semibold` . -
  ```



- `react-native run-ios` `react-native run-android` `react-native run-android` ( )

: https://riptutorial.com/ko/react-native/topic/4341/--

22:

- setState (| nextState, [ ])

Examples

setState

`setState` . . setState .

`setState` - - .

-

```
this.setState({myKey: 'myValue'});
```

.

```
this.setState((previousState, currentProps) => {
    return {
        myInteger: previousState.myInteger+1
    }
})
```

`setState` .

```
this.setState({myKey: 'myValue'}, () => {
    // Component has re-rendered... do something amazing!
));
```

```
import React, { Component } from 'react';
import { AppRegistry, StyleSheet, Text, View, TouchableOpacity } from 'react-native';

export default class MyParentComponent extends Component {
  constructor(props) {
    super(props);

    this.state = {
      myInteger: 0
    }

  }
  getRandomInteger() {
    const randomInt = Math.floor(Math.random()*100);

    this.setState({
      myInteger: randomInt
    });

  }
```

```
  incrementInteger() {

    this.setState((previousState, currentProps) => {
      return {
        myInteger: previousState.myInteger+1
      }
    });

  }
  render() {

    return <View style={styles.container}>

      <Text>Parent Component Integer: {this.state.myInteger}</Text>

      <MyChildComponent myInteger={this.state.myInteger} />

      <Button label="Get Random Integer" onPress={this.getRandomInteger.bind(this)} />
      <Button label="Increment Integer" onPress={this.incrementInteger.bind(this)} />

    </View>

  }
}

export default class MyChildComponent extends Component {
  constructor(props) {
    super(props);
  }
  render() {

    // this will get updated when "MyParentComponent" state changes
    return <View>
      <Text>Child Component Integer: {this.props.myInteger}</Text>
    </View>

  }
}

export default class Button extends Component {
  constructor(props) {
    super(props);
  }
  render() {

    return <TouchableOpacity onPress={this.props.onPress}>
        <View style={styles.button}>
          <Text style={styles.buttonText}>{this.props.label}</Text>
        </View>
      </TouchableOpacity>

  }
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#F5FCFF',
  },
```

```
  button: {
    backgroundColor: '#444',
    padding: 10,
    marginTop: 10
  },
  buttonText: {
    color: '#fff'
  }
});

AppRegistry.registerComponent('MyApp', () => MyParentComponent);
```

.

```
export default class MyComponent extends Component {
  constructor(props) {
    super(props);

    this.state = {
      myInteger: 0
    }
  }
  render() {
    return  (
      <View>
        <Text>Integer: {this.state.myInteger}</Text>
      </View>
    )
  }
}
```

setState .

: https://riptutorial.com/ko/react-native/topic/3596/

23:

React . React UI . ( ) .

Examples

?

. . **this.props.keyName** . .

. index.android.js index.ios.js .

```
import React, { Component } from 'react';
import { AppRegistry, Text, View } from 'react-native';

class Greeting extends Component {
  render() {
    return (
      <Text>Hello {this.props.name}!</Text>
    );
  }
}

class LotsOfGreetings extends Component {
  render() {
    return (
      <View style={{alignItems: 'center'}}>
        <Greeting name='Rexxar' />
        <Greeting name='Jaina' />
        <Greeting name='Valeera' />
      </View>
    );
  }
}

AppRegistry.registerComponent('LotsOfGreetings', () => LotsOfGreetings);
```

. Button . .

: -

PropType

prop-types . , name isYummy . . propTypes .

```
import React, { Component } from 'react';
import PropTypes from 'prop-types';
import { AppRegistry, Text, View } from 'react-native';

import styles from './styles.js';

class Recipe extends Component {
  static propTypes = {
```

```
    name: PropTypes.string.isRequired,
    isYummy: PropTypes.bool.isRequired
  }
  render() {
    return (
      <View style={styles.container}>
        <Text>{this.props.name}</Text>
        {this.props.isYummy ? <Text>THIS RECIPE IS YUMMY</Text> : null}
      </View>
    )
  }
}

AppRegistry.registerComponent('Recipe', () => Recipe);


// Using the component
<Recipe name="Pancakes" isYummy={true} />
```

PropTypes

propTypes . , . .

```
static propTypes = {
  name: PropTypes.oneOfType([
      PropTypes.string,
      PropTypes.object
  ])
}
```

children ,

```
<Recipe children={something}/>
```

.

```
<Recipe>
  <Text>Hello React Native</Text>
</Recipe>
```

:

```
return (
  <View style={styles.container}>
    {this.props.children}
    {this.props.isYummy ? <Text>THIS RECIPE IS YUMMY</Text> : null}
  </View>
)
```

Recipe <Text> . Hello React Native .

propype

```
children: PropTypes.node
```

defaultProps . props John .

```
class Example extends Component {
  render() {
    return (
      <View>
        <Text>{this.props.name}</Text>
      </View>
    )
  }
}


Example.defaultProps = {
  name: 'John'
}
```

: https://riptutorial.com/ko/react-native/topic/1271/

24:

CSS JSON . `StyleSheet.create(StyleObject)` CSS.

- `<Component style={styleFromStyleSheet} />`
- `<Component style={styleObject} />`
- `<Component style={[style1,style2]} />`

Most React CSS . `text-decoration` `textDecoration` .

CSS . `.`, `View` .
`Text` . `Text` .

Examples

React Native `style` . CSS JavaScript .

```
<Text style={{color:'red'}}>Red text</Text>
```

. .

```
import React, { Component } from 'react';
import { View, Text, StyleSheet } from 'react-native';

const styles = StyleSheet.create({
    red: {
        color: 'red'
    },
    big: {
        fontSize: 30
    }
});

class Example extends Component {
    render() {
        return (
            <View>
                <Text style={styles.red}>Red</Text>
                <Text style={styles.big}>Big</Text>
            </View>
        );
    }
}
```

`StyleSheet.create()` . React Native ID .

`style` . .

```
import React, { Component } from 'react';
import { View, Text, StyleSheet } from 'react-native';
```

```
const styles = StyleSheet.create({
    red: {
        color: 'red'
    },
    greenUnderline: {
        color: 'green',
        textDecoration: 'underline'
    },
    big: {
        fontSize: 30
    }
});

class Example extends Component {
    render() {
        return (
            <View>
                <Text style={[styles.red, styles.big]}>Big red</Text>
                <Text style={[styles.red, styles.greenUnderline]}>Green underline</Text>
                <Text style={[styles.greenUnderline, styles.red]}>Red underline</Text>
                <Text style={[styles.greenUnderline, styles.red, styles.big]}>Big red
underline</Text>
                <Text style={[styles.big, {color:'yellow'}]}>Big yellow</Text>
            </View>
        );
    }
}
```

```
<View style={[(this.props.isTrue) ? styles.bgcolorBlack : styles.bgColorWhite]}>
```

isTrue true , .

: https://riptutorial.com/ko/react-native/topic/7757/

25:

Examples

index.ios.js index.android.js

index.ios.js index.android.js <View> </View> index.android.js . , <Text> Hello World! </Text> .

Hello World! Hello World! !

! Hello World .

!

```
import React, { Component } from 'react';
import { AppRegistry, Text } from 'react-native';

class HelloWorldApp extends Component {
  render() {
    return (
      <Text>Hello world!</Text>
    );
  }
}

AppRegistry.registerComponent('HelloWorldApp', () => HelloWorldApp);
```

: https://riptutorial.com/ko/react-native/topic/3779/-

26: API

Examples

```
class AnimatedImage extends Component {
    constructor(props){
        super(props)
        this.state = {
            logoMarginTop: new Animated.Value(200)
        }
    }
    componentDidMount(){
        Animated.timing(
            this.state.logoMarginTop,
            { toValue: 100 }
        ).start()
    }
    render () {
      return (
        <View>
           <Animated.Image source={require('../images/Logo.png')} style={[baseStyles.logo, {
              marginTop: this.state.logoMarginTop
           }]} />
        </View>
      )
    }
}
```

.

API : https://riptutorial.com/ko/react-native/topic/4415/-api

27:

Examples

react-native Image . :

```
import { Image } from 'react';

<Image source={{uri: 'https://image-souce.com/awesomeImage'}} />
```

:

```
import { Image } from 'react';

<Image source={require('./img/myCoolImage.png')} />
```

: -, . .

```
class ImageExample extends Component {
  render() {
    return (
      <View>
        <Image style={{width: 30, height: 30}}
          source={{uri: 'http://facebook.github.io/react/img/logo_og.png'}}
        />
      </View>
    );
  }
}
```

```
<Image style={[this.props.imageStyle]}
        source={this.props.imagePath
        ? this.props.imagePath
        : require('../theme/images/resource.png')}
    />
```

imagePath .

```
let imagePath = require("../../assets/list.png");

<Image style={{height: 50, width: 50}} source={imagePath} />
```

:

```
<Image style={{height: 50, width: 50}} source={{uri: userData.image}} />
```

```
<Image
    resizeMode="contain"
    style={{height: 100, width: 100}}
```

```
    source={require('../assets/image.png')} />
```

, , .

: https://riptutorial.com/ko/react-native/topic/3956/

28: Firebase

// firebase app api . firebase 'firebase' ;

APIID : "yourAPIKey", authDomain : "authDomainNAme", databaseURL : "yourDomainBaseURL", projectId : "yourProjectID", storageBucket : "storageBUcketValue", messagingSenderId : "senderIdValue"}); firebase.auth (). signInWithEmailAndPassword (,) .then (this.onLoginSuccess)}}}

Examples

React - Firebase ListView

Firebase ListView .

Firebase (Posts.js) .

Posts.js

```
import PostsList from './PostsList';

class Posts extends Component{
    constructor(props) {
        super(props);
        this.state = {
            posts: []
        }
    }

    componentWillMount() {
        firebase.database().ref('Posts/').on('value', function(data) {
            this.setState({ posts: data.val() });
        });
    }

    render() {
        return <PostsList posts={this.state.posts}/>
    }
}
```

PostsList.js

```
class PostsList extends Component {
    constructor(props) {
        super(props);
        this.state = {
            dataSource: new ListView.DataSource({
                rowHasChanged: (row1, row2) => row1 !== row2
            }),
        }
    }
```

```
    getDataSource(posts: Array<any>): ListView.DataSource {
        if(!posts) return;
        return this.state.dataSource.cloneWithRows(posts);
    }

    componentDidMount() {
        this.setState({dataSource: this.getDataSource(this.props.posts)});
    }

    componentWillReceiveProps(props) {
        this.setState({dataSource: this.getDataSource(props.posts)});
    }

    renderRow = (post) => {
        return (
            <View>
                <Text>{post.title}</Text>
                <Text>{post.content}</Text>
            </View>
        );
    }

    render() {
        return(
            <ListView
                dataSource={this.state.dataSource}
                renderRow={this.renderRow}
                enableEmptySections={true}
            />
        );
    }
}
```

`Posts.js` `firebase` `Posts.js` . ( ) .

ListView . .

: [http://stackoverflow.com/questions/38414289/react-native-listview-not-rendering-data-from-firebase][1]

Firebase

firebase app api :

```
import firebase from 'firebase';
componentWillMount() {
firebase.initializeApp({
  apiKey: "yourAPIKey",
  authDomain: "authDomainNAme",
  databaseURL: "yourDomainBaseURL",
  projectId: "yourProjectID",
  storageBucket: "storageBUcketValue",
  messagingSenderId: "senderIdValue"
});
    firebase.auth().signInWithEmailAndPassword(email, password)
  .then(this.onLoginSuccess)
  .catch(() => {
    firebase.auth().createUserWithEmailAndPassword(email, password)
```

```
      .then(this.onLoginSuccess)
      .catch(this.onLoginFail)
  })
}
```

Firebase : https://riptutorial.com/ko/react-native/topic/6391/-firebase-

29:

Examples

js . .

```
//In the page "Home", I want to have the right nav button to show
//a settings modal that resides in "Home" component.

componentWillMount() {
  this.props.route.navbarTitle = "Home";

  this.props.route.rightNavButton = {
    text: "Settings",
    onPress: this._ShowSettingsModal.bind(this)
  };
}
```

```
'use strict';

import React, {Component} from 'react';
import ReactNative from 'react-native';

const {
  AppRegistry,
  StyleSheet,
  Text,
  View,
  Navigator,
  Alert,
  TouchableHighlight
} = ReactNative;


//This is the app container that contains the navigator stuff
class AppContainer extends Component {

    renderScene(route, navigator) {
        switch(route.name) {
            case "Home":
      //You must pass route as a prop for this trick to work properly
            return <Home route={route} navigator={navigator} {...route.passProps}  />
            default:
            return (
        <Text route={route}
        style={styles.container}>
            Your route name is probably incorrect {JSON.stringify(route)}
            </Text>
      );
        }
    }

  render() {
    return (
      <Navigator
        navigationBar={
```

```
          <Navigator.NavigationBar
            style={ styles.navbar }
            routeMapper={ NavigationBarRouteMapper } />
        }

        initialRoute={{ name: 'Home' }}
        renderScene={ this.renderScene }

      />
    );
  }
}


//Nothing fancy here, except for checking for injected buttons.
//Notice how we are checking if there are injected buttons inside the route object.
//Also, we are showing a "Back" button when the page is not at index-0 (e.g. not home)
var NavigationBarRouteMapper = {
  LeftButton(route, navigator, index, navState) {
    if(route.leftNavButton) {
      return (
        <TouchableHighlight
        style={styles.leftNavButton}
        underlayColor="transparent"
        onPress={route.leftNavButton.onPress}>
          <Text style={styles.navbarButtonText}>{route.leftNavButton.text}</Text>
        </TouchableHighlight>
      );
    }
    else if(route.enableBackButton) {
      return (
        <TouchableHighlight
        style={styles.leftNavButton}
        underlayColor="transparent"
        onPress={() => navigator.pop() }>
          <Text style={styles.navbarButtonText}>Back</Text>
        </TouchableHighlight>
      );
    }
  },
  RightButton(route, navigator, index, navState) {
    if(route.rightNavButton) {
      return (
        <TouchableHighlight
        style={styles.rightNavButton}
        underlayColor="transparent"
        onPress={route.rightNavButton.onPress}>
          <Text style={styles.navbarButtonText}>{route.rightNavButton.text}</Text>
        </TouchableHighlight>
      );
    }
  },
  Title(route, navigator, index, navState) {
    //You can inject the title aswell.  If you don't we'll use the route name.
    return (<Text style={styles.navbarTitle}>{route.navbarTitle || route.name}</Text>);
  }
};

//This is considered a sub-page that navigator is showing
class Home extends Component {
```

```
  //This trick depends on that componentWillMount fires before the navbar is created
  componentWillMount() {
        this.props.route.navbarTitle = "Home";

        this.props.route.rightNavButton = {
            text: "Button",
            onPress: this._doSomething.bind(this)
        };
    }

  //This method will be invoked by pressing the injected button.
  _doSomething() {
      Alert.alert(
      'Awesome, eh?',
      null,
      [
        {text: 'Indeed'},
      ]
    )
  }

  render() {
    return (
      <View style={styles.container}>
            <Text>You are home</Text>
      </View>
    );
  }
}

var styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#F5FCFF',
    marginTop: 66
  },
  navbar: {
    backgroundColor: '#ffffff',
  },
  navbarTitle: {
    marginVertical: 10,
    fontSize: 17
  },
  leftNavButton: {
    marginVertical: 10,
    paddingLeft: 8,
 },
  rightNavButton: {
    marginVertical: 10,
    paddingRight: 8,
  },
  navbarButtonText: {
    fontSize: 17,
    color: "#007AFF"
  }
});

AppRegistry.registerComponent('AppContainer', () => AppContainer);
```

: https://riptutorial.com/ko/react-native/topic/6416/----

30:

zo0r npm **react-native-push-notification** . .

npm install --save react-native-push-notification

GitHub Repo .

Examples

PushNotification

```
react-native init PushNotification
```

index.android.js .

```
import React, { Component } from 'react';

import {
  AppRegistry,
  StyleSheet,
  Text,
  View,
  Button
} from 'react-native';

import PushNotification from 'react-native-push-notification';

export default class App extends Component {

    constructor(props){
        super(props);

        this.NewNotification = this.NewNotification.bind(this);
      }

    componentDidMount(){

        PushNotification.configure({

            // (required) Called when a remote or local notification is opened or received
            onNotification: function(notification) {
                console.log( 'NOTIFICATION:', notification );
            },

            // Should the initial notification be popped automatically
            // default: true
            popInitialNotification: true,

            /**
              * (optional) default: true
              * - Specified if permissions (ios) and token (android and ios) will requested or
not,
              * - if not, you must call PushNotificationsHandler.requestPermissions() later
```

```
              */
            requestPermissions: true,
        });

    }

      NewNotification(){

          let date = new Date(Date.now() + (this.state.seconds * 1000));

          //Fix for IOS
        if(Platform.OS == "ios"){
            date = date.toISOString();
        }

        PushNotification.localNotificationSchedule({
            message: "My Notification Message", // (required)
            date: date,// (optional) for setting delay
            largeIcon:""// set this blank for removing large icon
            //smallIcon: "ic_notification", // (optional) default: "ic_notification" with
fallback for "ic_launcher"
        });
    }

      render() {

        return (
            <View style={styles.container}>
                <Text style={styles.welcome}>
                  Push Notification
                </Text>
                <View style={styles.Button} >
                <Button
                  onPress={()=>{this.NewNotification()}}
                  title="Show Notification"
                  style={styles.Button}
                  color="#841584"
                  accessibilityLabel="Show Notification"
                />
                </View>
            </View>
        );
      }
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#F5FCFF',
  },
  welcome: {
    fontSize: 20,
    textAlign: 'center',
    margin: 10,
  },
  Button:{
    margin: 10,
  }
});
```

```
AppRegistry.registerComponent('PushNotification', () => App);
```

/ . .

```
'use strict';

import React, { Component } from 'react';
import {
    StyleSheet,
    Text,
    View,
    Navigator,
    TouchableOpacity,
    AsyncStorage,
    BackAndroid,
    Platform,
} from 'react-native';
import PushNotification from 'react-native-push-notification';

let initialRoute = { id: 'loginview' }

export default class MainClass extends Component
{
    constructor(props)
    {
        super(props);

        this.handleNotification = this.handleNotification.bind(this);
    }

    handleNotification(notification)
    {
        console.log('handleNotification');
        var notificationId = ''
        //your logic to get relevant information from the notification

    //here you navigate to a scene in your app based on the notification info
        this.navigator.push({ id: Constants.ITEM_VIEW_ID, item: item });
    }

    componentDidMount()
    {
        var that = this;

        PushNotification.configure({

            // (optional) Called when Token is generated (iOS and Android)
            onRegister: function(token) {
                console.log( 'TOKEN:', token );
            },

            // (required) Called when a remote or local notification is opened or received
            onNotification(notification) {
                console.log('onNotification')
                console.log( notification );

                that.handleNotification(notification);
            },
```

```
            // ANDROID ONLY: (optional) GCM Sender ID.
            senderID: "Vizido",

            // IOS ONLY (optional): default: all - Permissions to register.
            permissions: {
                alert: true,
                badge: true,
                sound: true
            },

            // Should the initial notification be popped automatically
            // default: true
            popInitialNotification: true,

            /**
              * (optional) default: true
              * - Specified if permissions (ios) and token (android and ios) will requested or
not,
              * - if not, you must call PushNotificationsHandler.requestPermissions() later
              */
            requestPermissions: true,
        });
    }

    render()
    {

        return (
            <Navigator
                ref={(nav) => this.navigator = nav }
                initialRoute={initialRoute}
                renderScene={this.renderScene.bind(this)}
                configureScene={(route) =>
                    {
                        if (route.sceneConfig)
                        {
                            return route.sceneConfig;
                        }
                        return Navigator.SceneConfigs.FadeAndroid;
                    }
                }
            />
        );
    }

    renderScene(route, navigator)
    {

        switch (route.id)
        {
            // do your routing here
            case 'mainview':
                return ( <MainView navigator={navigator} /> );

            default:
                return ( <MainView navigator={navigator} /> );
        }
    }
}
```

: https://riptutorial.com/ko/react-native/topic/9674/-

31:

Examples

OS /

' - ' Platform .

```
import { Platform } from 'react-native'
```

`Platform.OS` OS .

```
const styles = StyleSheet.create({
  height: (Platform.OS === 'ios') ? 200 : 100,
})
```

Android `Platform.Version` .

```
if (Platform.Version === 21) {
  console.log('Running on Lollipop!');
}
```

iOS Platform.Version .

```
if (parseInt(Platform.Version, 10) >= 9) {
    console.log('Running version higher than 8');
}
```

. :

- `MyTask.android.js`
- `MyTask.ios.js`

```
const MyTask = require('./MyTask')
```

: https://riptutorial.com/ko/react-native/topic/3593/-

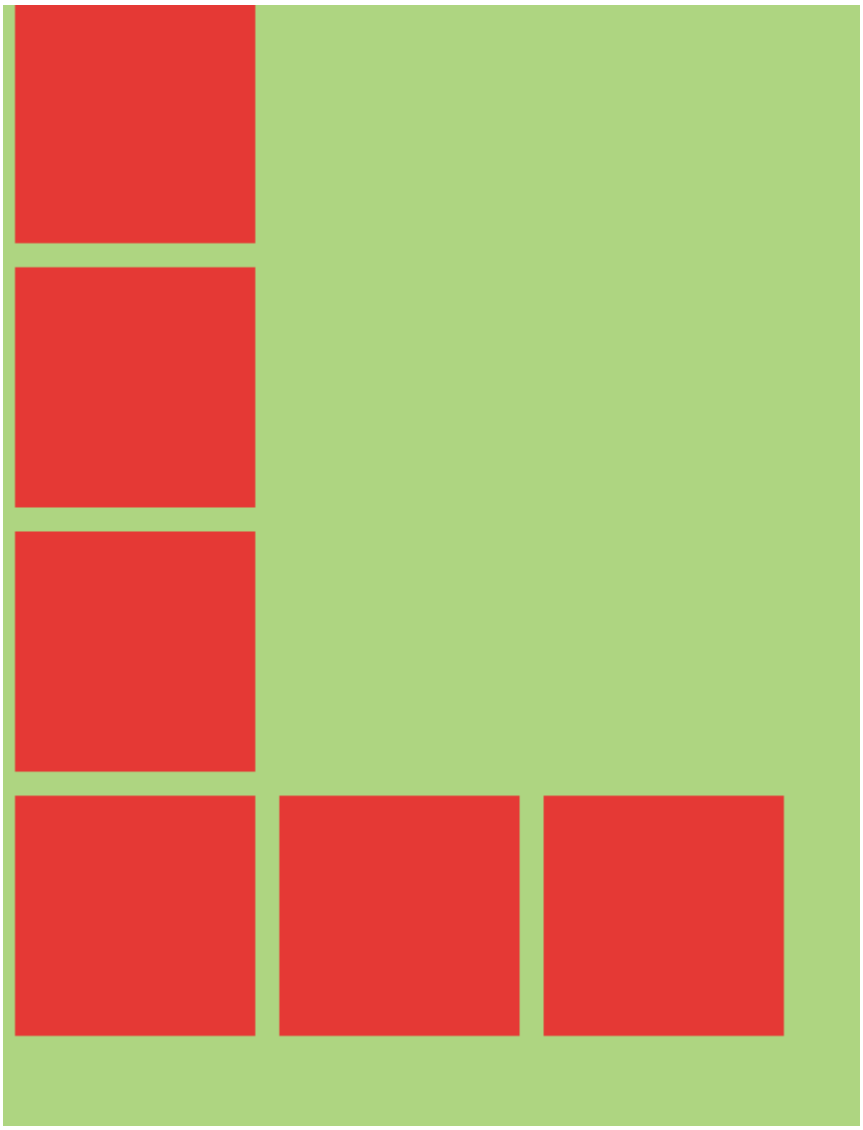
32:

Examples

Flexbox . flexbox . `flexDirection: 'row'` `flexDirection: 'row'` .

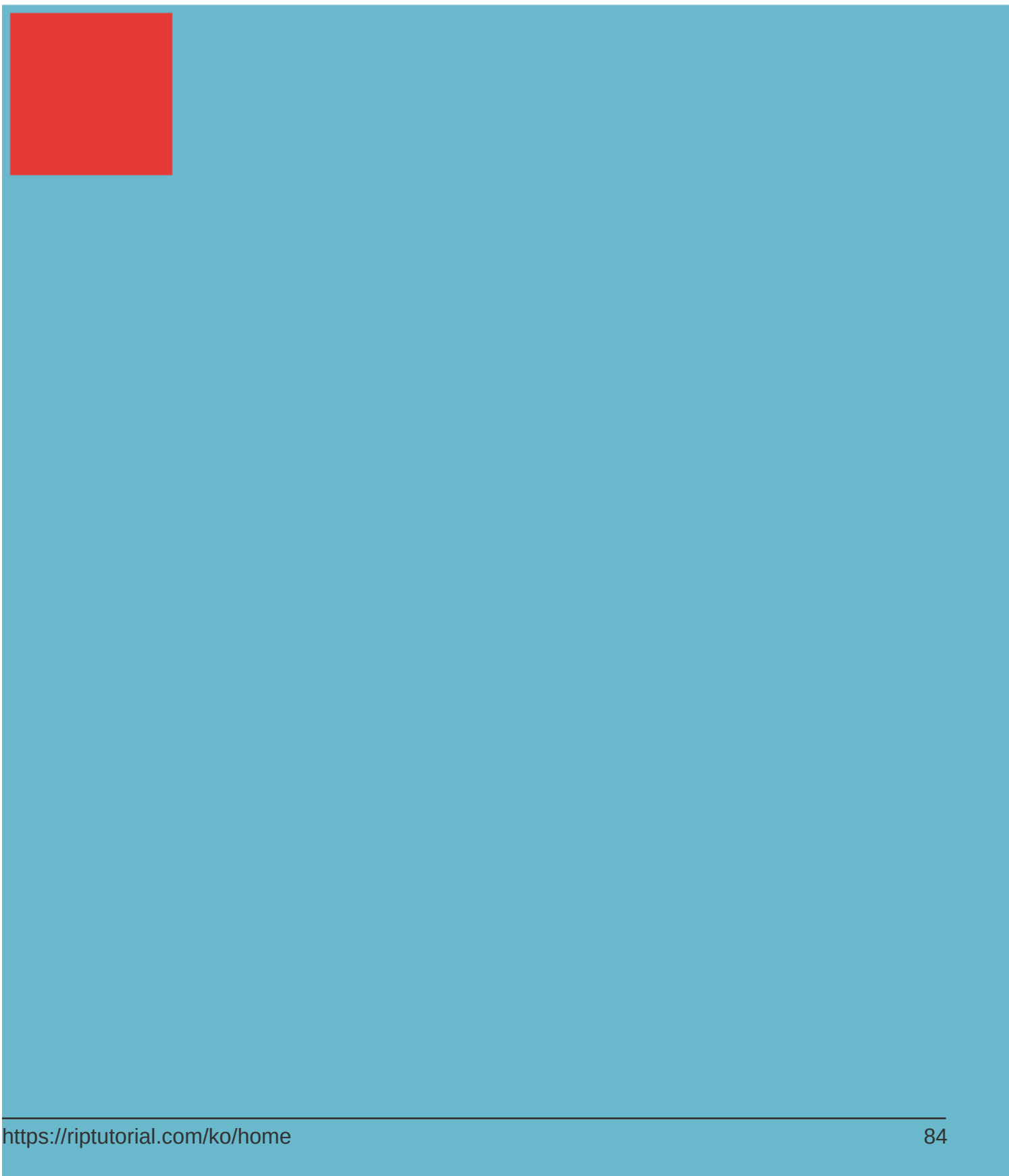
```
const Direction = (props)=>{
  return (
    <View style={styles.container}>
      <Box/>
      <Box/>
      <Box/>
      <View style={{flexDirection:'row'}}>
        <Box/>
        <Box/>
        <Box/>
      </View>
    </View>
  )
}

const styles = StyleSheet.create({
  container: {
    flex:1,
    backgroundColor: '#AED581',
  }
});
```



```
const AlignmentAxis = (props)=>{
  return (
    <View style={styles.container}>
      <Box />
      <View style={{flex:1, alignItems:'flex-end', justifyContent:'flex-end'}}>
        <Box />
        <Box />
      </View>
      <Box />
    </View>
  )
}

const styles = StyleSheet.create({
  container: {
    flex:1,
    backgroundColor: `#69B8CC`,
  },
  text:{
    color: 'white',
    textAlign:'center'
  }
});
```


Carrier
8:17 PM
AlignmentAx

S. No		Contributors
1		Adam, Community, Damien Varron, Dmitry Petukhov, Dr. Nitpick, Idan, Kaleb Portillo, Lucas Oliveira, manosim, Scimonster, Sivart, Tushar Khatiwada, xhg, Yevhen Dubinin
2	Android -	Cássio Santos, manosim, Michael S, Pascal Le Merrer, Sriraman, Virat18
3	Android APK	Aditya Singh
4	HTTP	Alex Belets, Alireza Valizade, AntonB, Chris Pena, Daniel Schmidt, Dmitry Petukhov, Everettss, Jagadish Upadhyay, manosim, MauroPorrasP, respectTheCode, shaN, Tejashwi Kalp Taru, Tobias Lins
5	ListView RefreshControl	Abdulaziz Alkharashi
6	WebView	sudo bangbang
7		Michael Hancock, Sriraman, Tobias Lins
8	(Android)	Jagadish Upadhyay, Lwin Kyaw Myat, Mayeul
9		Andres C. Viesca
10		Ankit Sinha, Michael Helvey, Pankaj Thakur
11	API	Viktor Seč
12		Jigar Shah
13		Ankit Sinha, sudo bangbang
14		Jagadish Upadhyay, mostafiz rahman
15		sudo bangbang
16		Alex Belets
17		Dmitry Petukhov, epsilondelta, Idan, Jagadish Upadhyay, manosim, Mozak, Sriraman, Tim Rijavec
18		Ahmed Ali, Liron Yahdav, Tobias Lins
19		Kaleb Portillo

20	ESLint	Alex Belets
21		Abdulaziz Alkharashi, Lwin Kyaw Myat, Noitidart, Olivia, Sriraman
22		AndyI, David, Jagadish Upadhyay, Tim Rijavec, Tobias Lins
23		CallMeNorm, Chris Pena, corasan, fson, Gianfranco P., henkimon, Hugo Dozois, Idan, Jagadish Upadhyay, Tobias Lins, Yevhen Dubinin, zhenjie ruan
24		Jigar Shah, Martin Cup, Scimonster
25		stereodenis, Zakaria Ridouh
26	API	Shashank Udupa, Sriraman, Tom Walters
27		Jagadish Upadhyay, Jigar Shah, Serdar Değirmenci, Zakaria Ridouh
28	Firebase	Ankit Sinha, corasan
29		Ahmed Al Haddad
30		shaN, Tejashwi Kalp Taru
31		Florian Hämmerle, Gabriel Diez, Jagadish Upadhyay, Zakaria Ridouh
32		Alex Belets, gwint, Jagadish Upadhyay, Scimonster, sudo bangbang