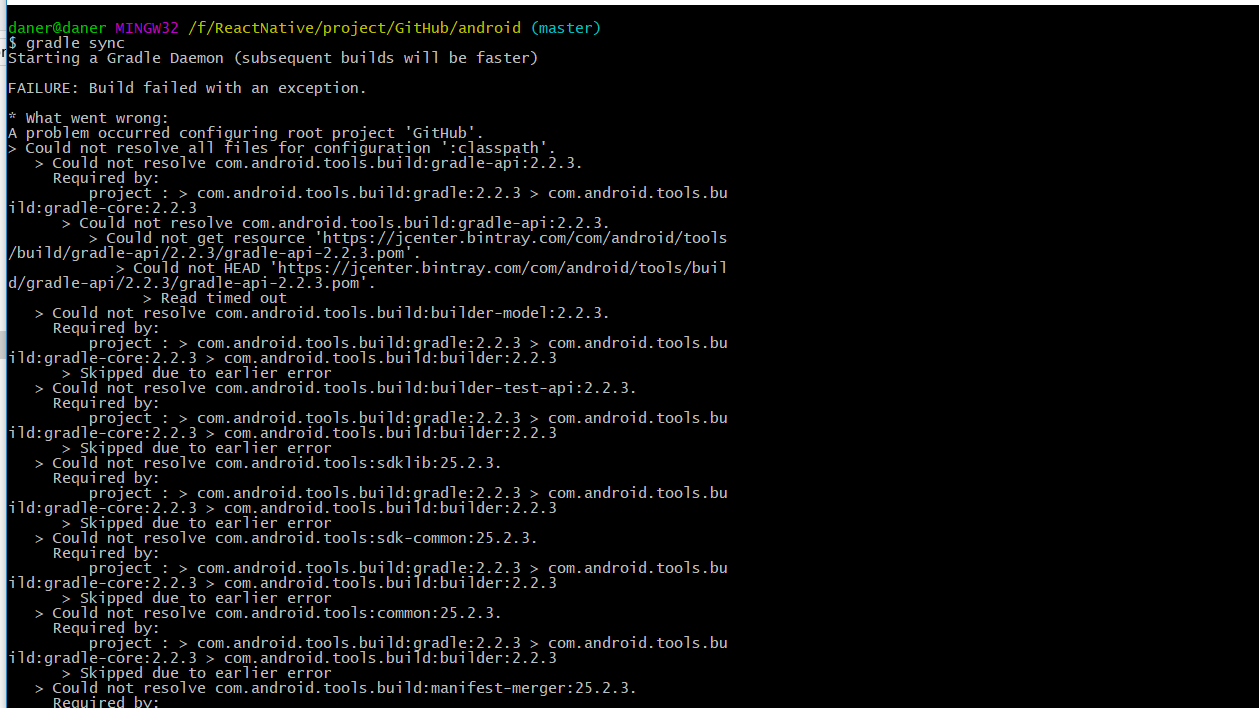
# 第一部分 ReactNative基础篇

## 一 新建ReactNative项目

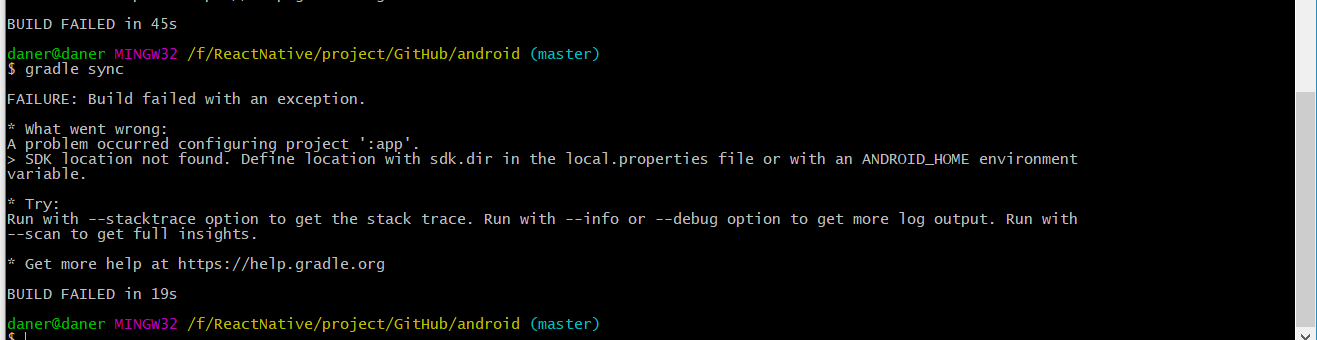
参考博客3 新建RN的工程并创建项目运行

react-native init GitHub

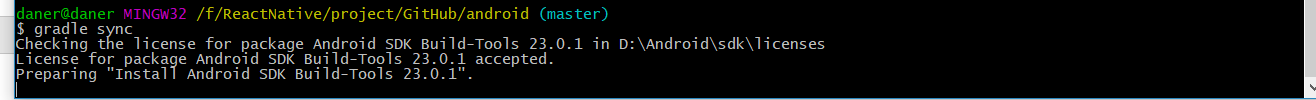


打开梯子

gradle sync

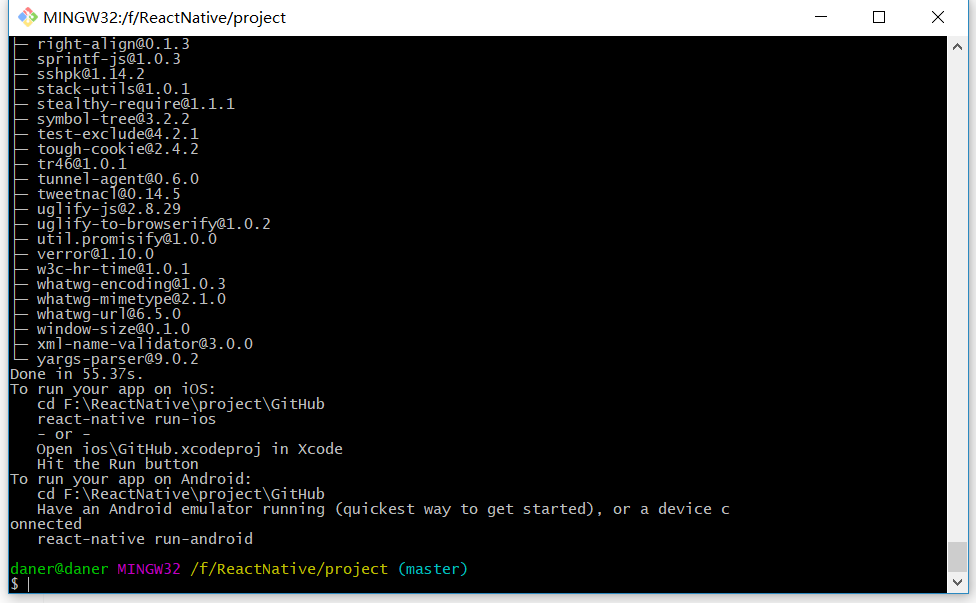


拷贝其他项目的local.properties到项目中





https://stackoverflow.com/questions/47495066/the-compileoptions-bootclasspath-property-has-been-deprecated

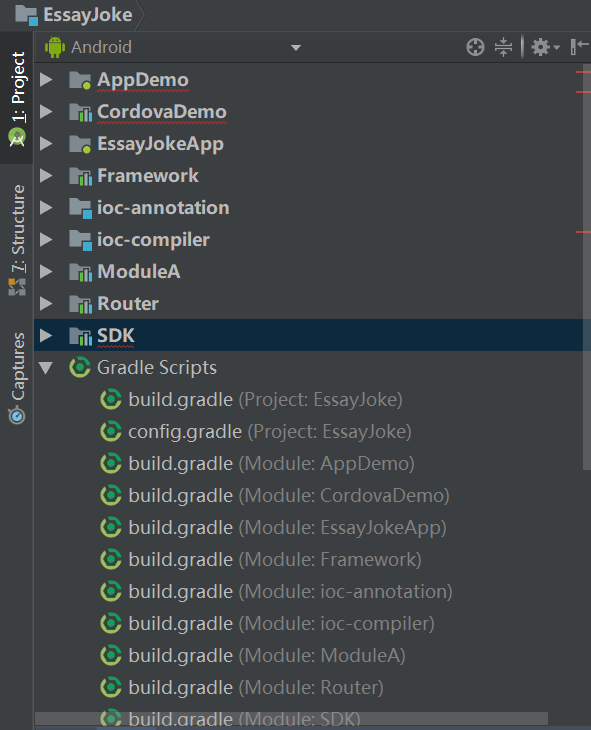


## 二 Android已有项目集成React Native项目

### 1 AS中集成ReactNative步骤

首先搭建好Android开发环境，并有一个新建的或者已经存在的Android项目

在未创建ReactNative项目大概是这个样子

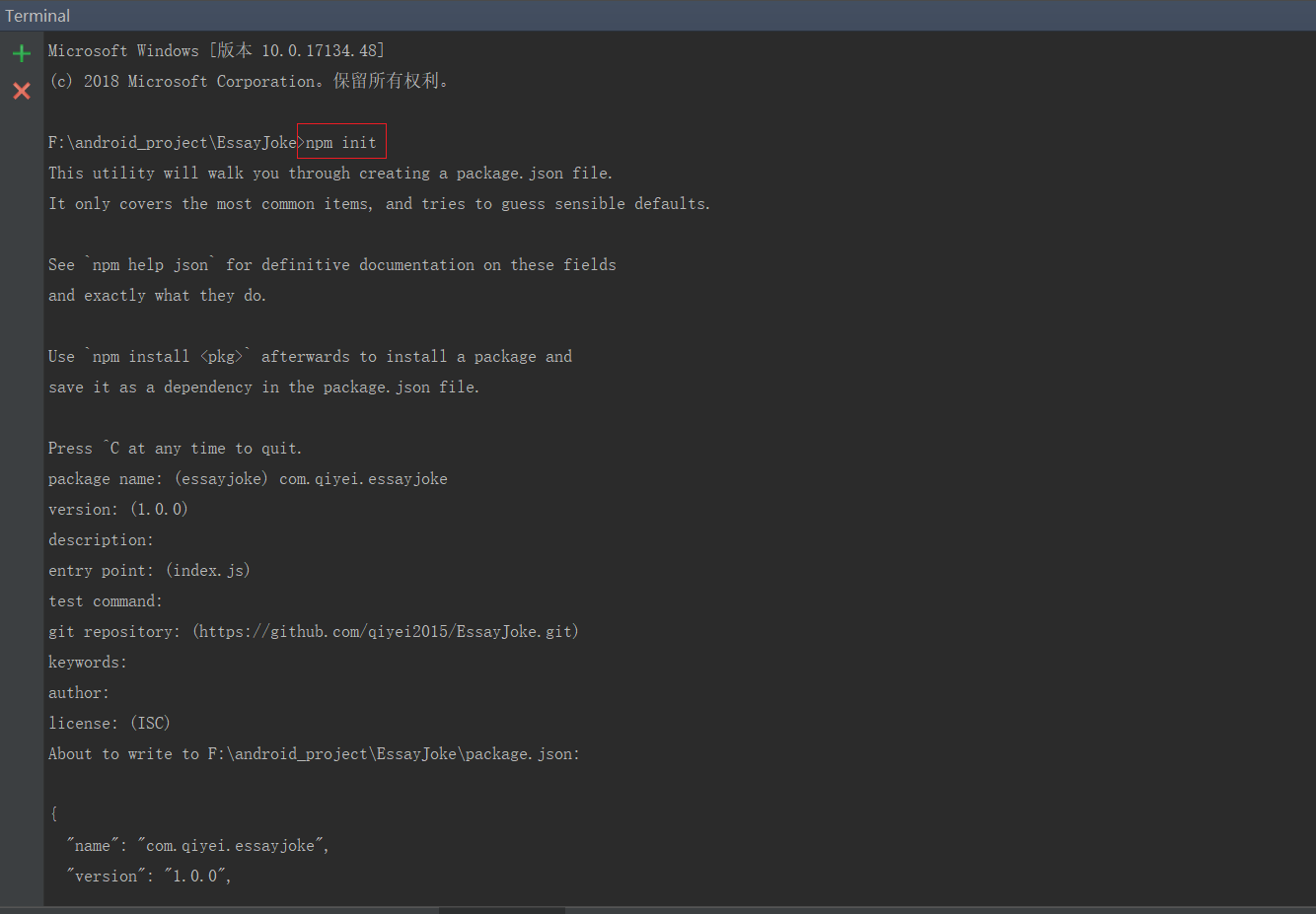


接着创建ReactNative Module用于存放我们RN相关的内容，极大程度的与原生开发解耦

#### 1 在应用中添加js代码

在项目根目录输入以下命令

npm init



在这中间要求输入包名，其他的默认即可

这一步完成之后，在项目的根目录下就会生成package.json这个文件

接下来还需要把启动脚本放进去：

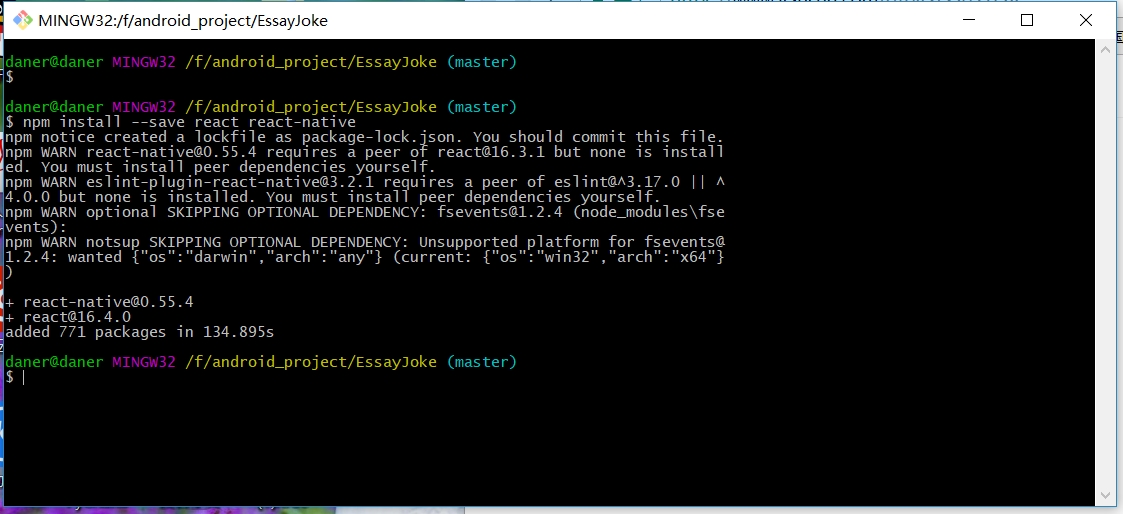
"start": "node node\_modules/react-native/local-cli/cli.js start"

注意下启动脚本的路径，如果集成的项目目录结构不一样的话会报找不到启动脚本之类的错误，到时根据具体目录结构修改就好了。

#### 2 安装React 和React Native

根目录输入以下命令

npm install --save react react-native

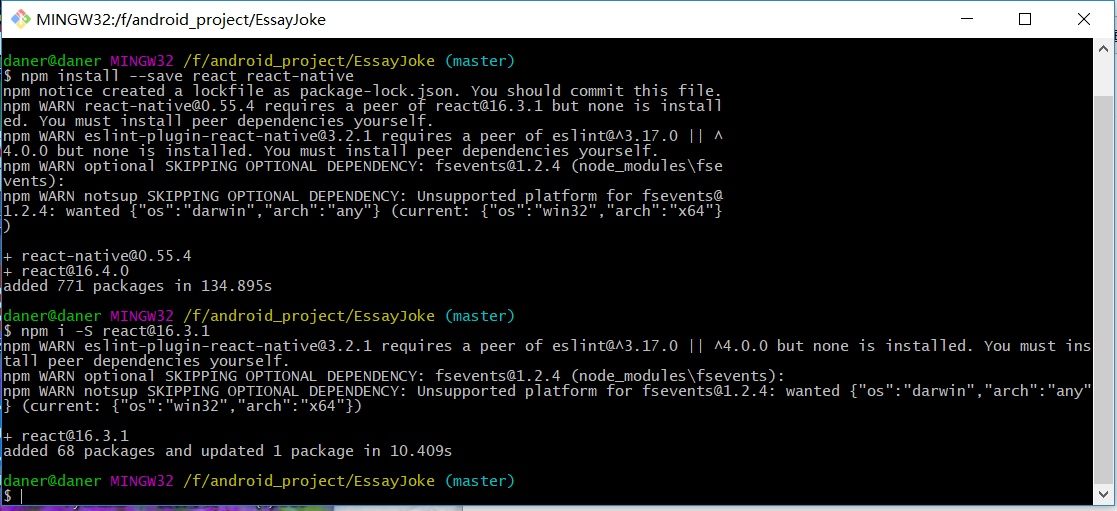


大约一两分钟的样子（如果卡到这里了，看看安装时是不是忘了配置镜像），完成之后你的根目录下会多了一个node\_modules的文件夹，里面存放了下载好的React 和React Native。

要是控制台输出了版本不一致的警告信息，例如：

npm WARN react-native@0.45.1 requires a peer of react@16.3.1 but none was installed.

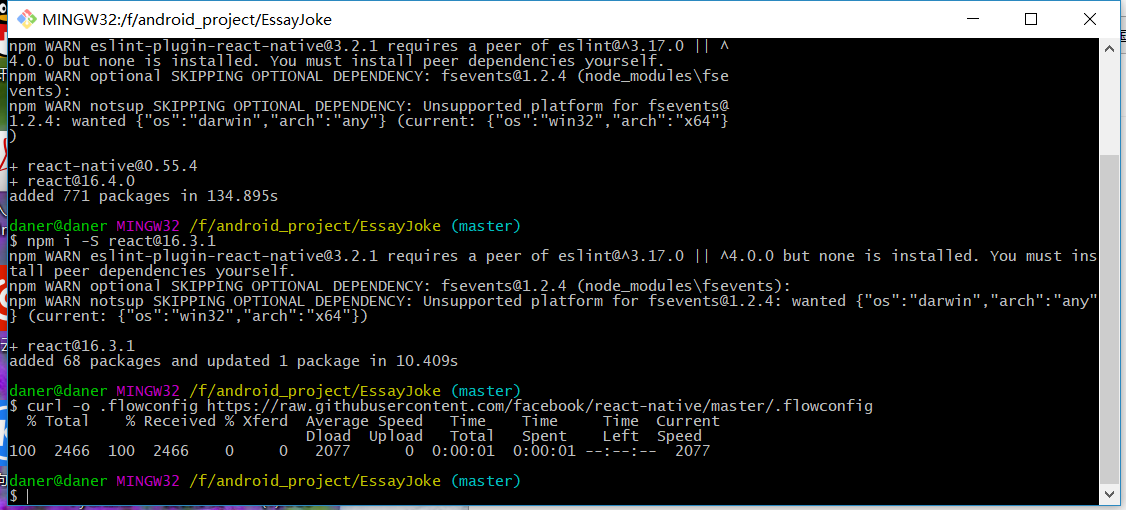
则继续执行 npm i -S react@16.3.1 (这里版本跟警告信息一致)。



#### 3 下载.flowconfig文件(非必须)

curl -o .flowconfig https://raw.githubusercontent.com/facebook/react-native/master/.flowconfig

第三步curl命令，其实质是下载.flowconfig配置文件，这个文件用于约束js代码的写法。非必需，可跳过。



#### 4 新建index.android.js

项目根目录下创建index.android.js文件，目录内容如下：

'use strict';

import React from 'react';

import {

AppRegistry,

StyleSheet,

Text,

View

} from 'react-native';

class EssayJokeReactNativeDemo extends React.Component {

render() {

return (

<View style={styles.container}>

<Text style={styles.hello}>Hello, EssayJoke React-NativeDemo</Text>

</View>

)

}

}

var styles = StyleSheet.create({

container: {

flex: 1,

justifyContent: 'center',

},

hello: {

fontSize: 20,

textAlign: 'center',

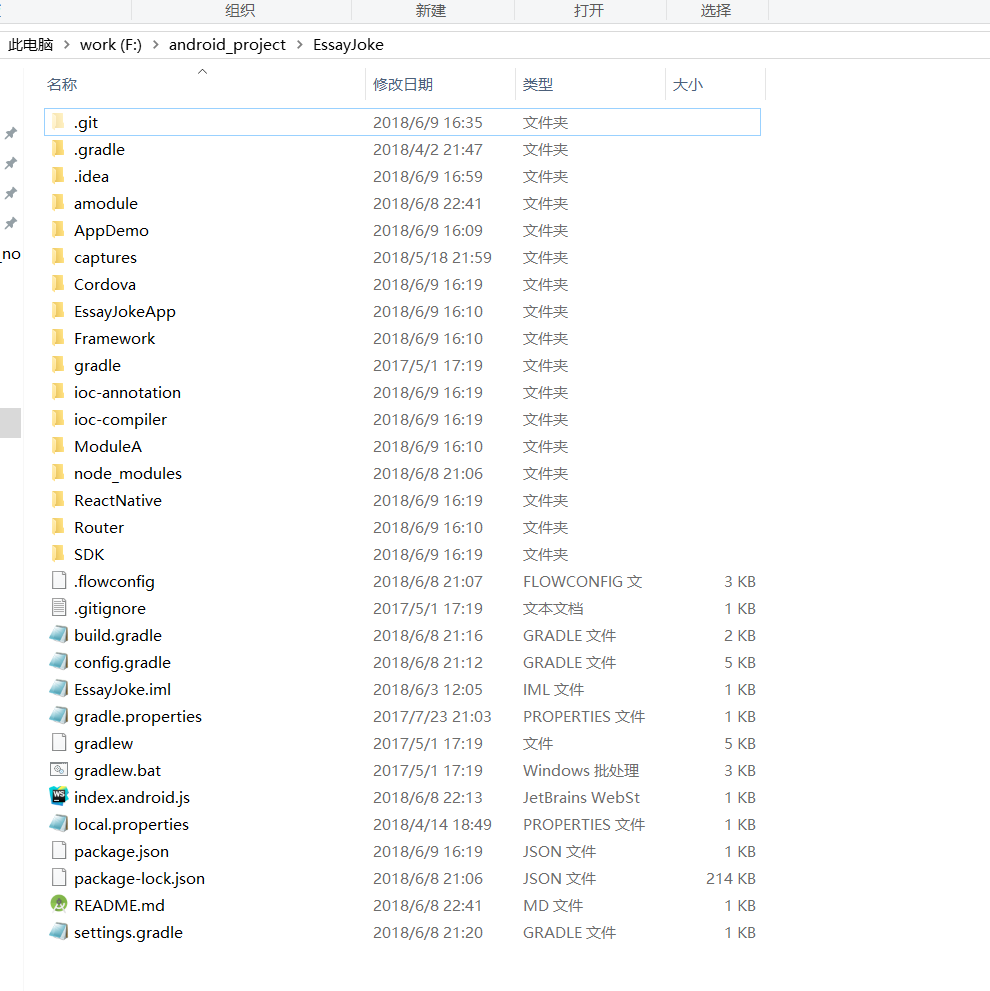
margin: 10,

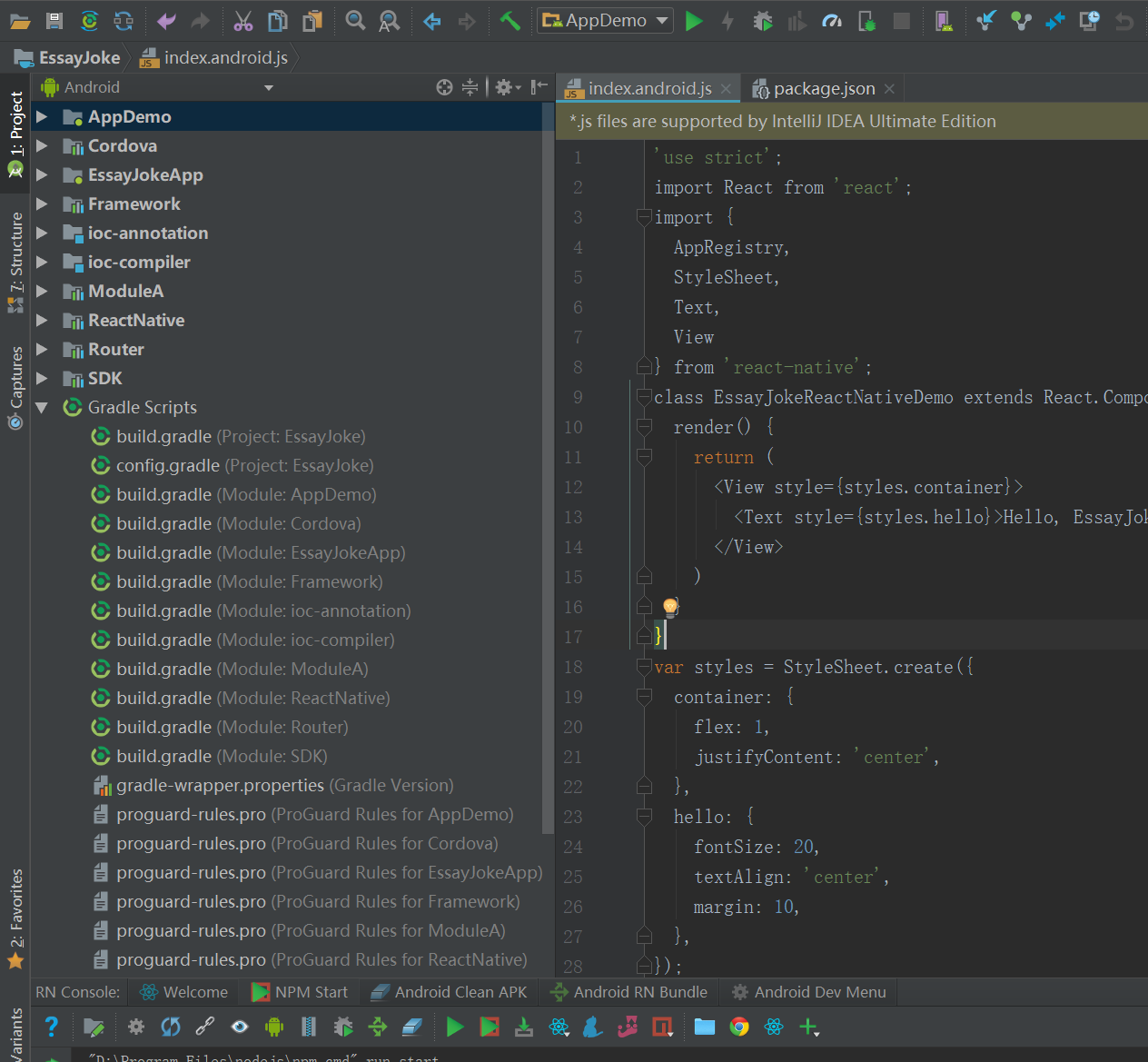
},

});

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

到这里js部分的都弄的差不多了，现在来看整个项目的目录结构如下：





ReactNative Module就是我们ReactNative相关开发的内容都放在该module下

#### 5 ReactNative Module引入依赖RN依赖

在build.gradle引入以下依赖

com\_facebook\_react\_react\_native = "com.facebook.react:react-native:+"

api com\_facebook\_react\_react\_native // From node\_modules.

在项目根目录build.gradle添加依赖仓库地址

allprojects {

repositories {

…

maven {

// All of React Native (JS, Android binaries) is installed from npm

url "$rootDir/node\_modules/react-native/android"

}

}

}

这里maven路径官网上写的是$rootDir/../node\_modules/react-native/android，这个跟目录结构有关，跟我一样在根目录上操作的话是不需要$rootDir后面的..的，它指的路径就是项目根目录下的node\_modules下的路径

#### 6 新建Activity作为原生与RN的跳转

在较老的RN版本中，我们新建一个Activity实现DefaultHardwareBackBtnHandler接口，大体类似下面

public class MyReactActivity extends AppCompatActivity implements DefaultHardwareBackBtnHandler {

private ReactRootView mReactRootView;

private ReactInstanceManager mReactInstanceManager;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

mReactRootView = new ReactRootView(this);

mReactInstanceManager = ReactInstanceManager.builder()

.setApplication(getApplication())

.setBundleAssetName("index.android.bundle")

.setJSMainModuleName("index.android")

.addPackage(new MainReactPackage())

.setUseDeveloperSupport(BuildConfig.DEBUG)

.setInitialLifecycleState(LifecycleState.RESUMED)

.build();

// 注意这里的HelloWorld必须对应“index.android.js”中的

// “AppRegistry.registerComponent()”的第一个参数

mReactRootView.startReactApplication(mReactInstanceManager, "test", null);

setContentView(mReactRootView);

}

@Override

public void invokeDefaultOnBackPressed() {

super.onBackPressed();

}

@Override

protected void onPause() {

super.onPause();

if (mReactInstanceManager != null) {

mReactInstanceManager.onHostPause(this);

}

}

@Override

protected void onResume() {

super.onResume();

if (mReactInstanceManager != null) {

mReactInstanceManager.onHostResume(this, this);

}

}

@Override

protected void onDestroy() {

super.onDestroy();

if (mReactInstanceManager != null) {

mReactInstanceManager.onHostDestroy();

}

}

@Override

public void onBackPressed() {

if (mReactInstanceManager != null) {

mReactInstanceManager.onBackPressed();

} else {

super.onBackPressed();

}

}

@Override

public boolean onKeyUp(int keyCode, KeyEvent event) {

if (keyCode == KeyEvent.KEYCODE\_MENU && mReactInstanceManager != null) {

mReactInstanceManager.showDevOptionsDialog();

return true;

}

return super.onKeyUp(keyCode, event);

}

}

较新的RN版本，例如RN0.55 我们直接继承ReactActivity即可

/\*\*

\* @author Created by qiyei2015 on 2018/6/8.

\* @version: 1.0

\* @email: 1273482124@qq.com

\* @description:

\*/

public class ReactNativeDemoActivity extends ReactActivity{

private static final String MAIN\_COMPONENT = "EssayJokeReactNativeDemo";

/\*\*

\* 返回在index.android.js 中注册的组件名

\* @return

\*/

@Nullable

@Override

protected String getMainComponentName() {

return MAIN\_COMPONENT;

}

}

新建一个Activity，让其继承ReactActivity，并重写getMainComponentName(),返回我们在index.android.js

中注册的HelloWorld这个组件。

在Application中初始化一下

/\*\*

\* Email: 1273482124@qq.com

\* Created by qiyei2015 on 2017/5/8.

\* Version: 1.0

\* Description:

\*/

public class BaseApplication extends Application implements ReactApplication{

private static final String TAG = " Sophix";

/\*\*

\* ReactNative的host

\*/

private final ReactNativeHost mReactNativeHost = new ReactNativeHost(this) {

@Override

public boolean getUseDeveloperSupport() {

return BuildConfig.DEBUG;

}

@Override

protected List<ReactPackage> getPackages() {

return Arrays.<ReactPackage>asList(

new MainReactPackage()

);

}

};

@Override

public void onCreate() {

super.onCreate();

....

}

/\*\*

\* ReactNative的host

\* @return

\*/

@Override

public ReactNativeHost getReactNativeHost() {

return mReactNativeHost;

}

....

}

在manifest.xml配置如下：

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.qiyei.reactnative">

<application>

<activity android:name="com.facebook.react.devsupport.DevSettingsActivity" />

<activity android:name=".ReactNativeDemoActivity" />

</application>

</manifest>

由于需要访问 DevSettingsActivity 界面，也需要在 AndroidManifest.xml 中声明

#### 7 同步编译

该步骤可能会遇到以下问题

Warning:Conflict with dependency 'com.google.code.findbugs:jsr305'. Resolved versions for app (3.0.0) and test app (2.0.1) differ. See http://g.co/androidstudio/app-test-app-conflict for details.

在build.bundle里面加入：

allprojects {

...

configurations.all {

resolutionStrategy.force 'com.google.code.findbugs:jsr305:3.0.0'

}

}

#### 8 在ReactNative Module新建assets目录

### 2 在AS中运行RN项目

有了以上准备，我们就直接来运行项目吧

#### 1 启动npm server

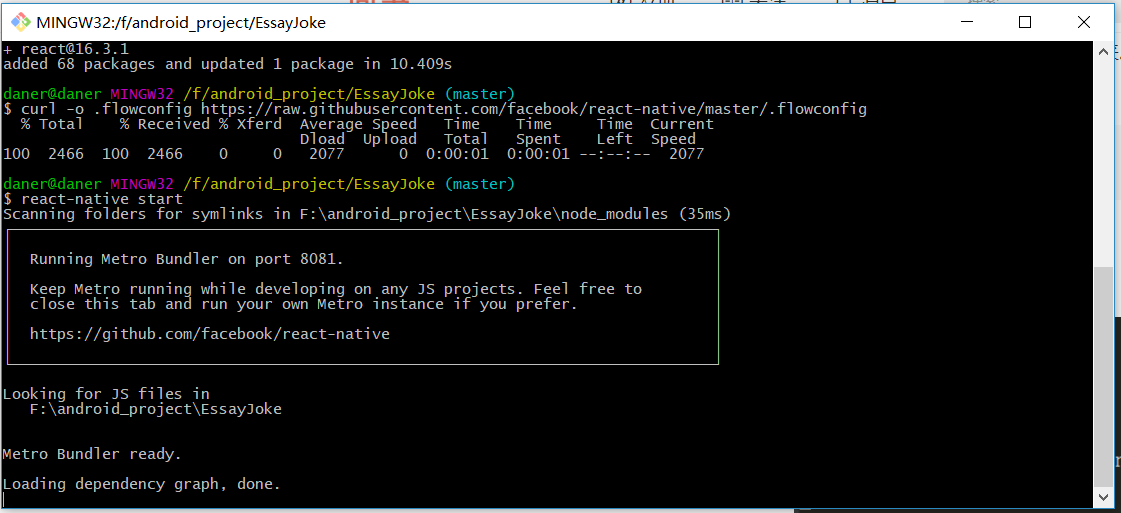
在根目录下输入以下命令

npm start

或者

react-native start

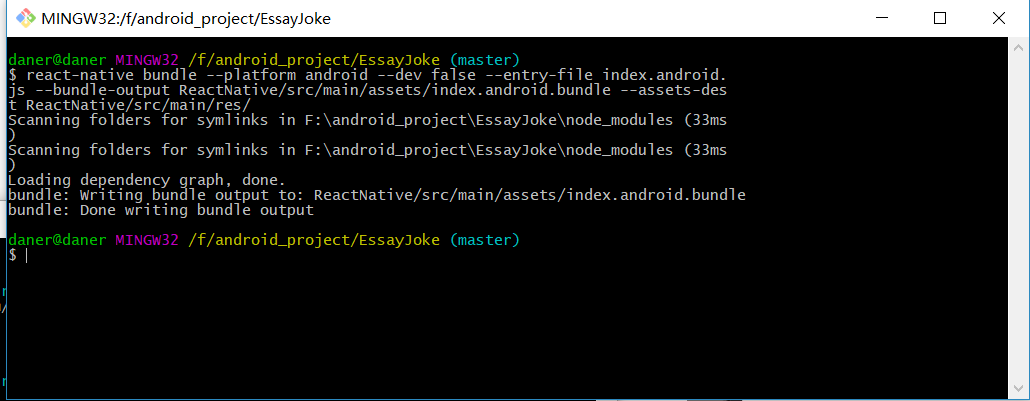
都可以启动后台服务



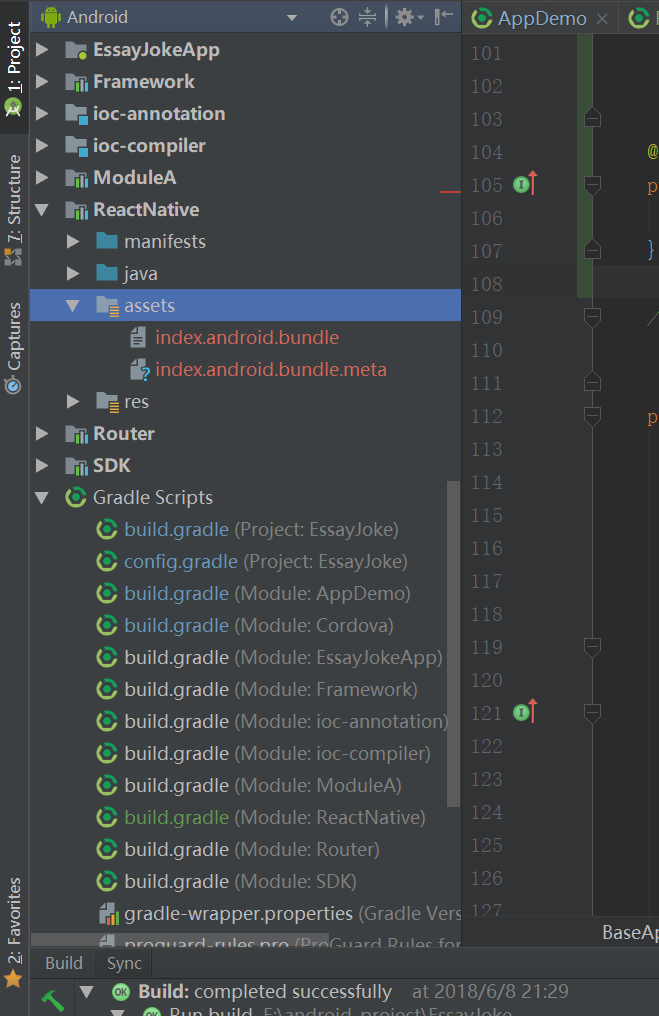
如果卡在了以上界面，直接在根目录下执行以下命令

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

注意，需要将index.android.bundle生成的目录指定对



完成之后assets目录下会生成以下两个文件



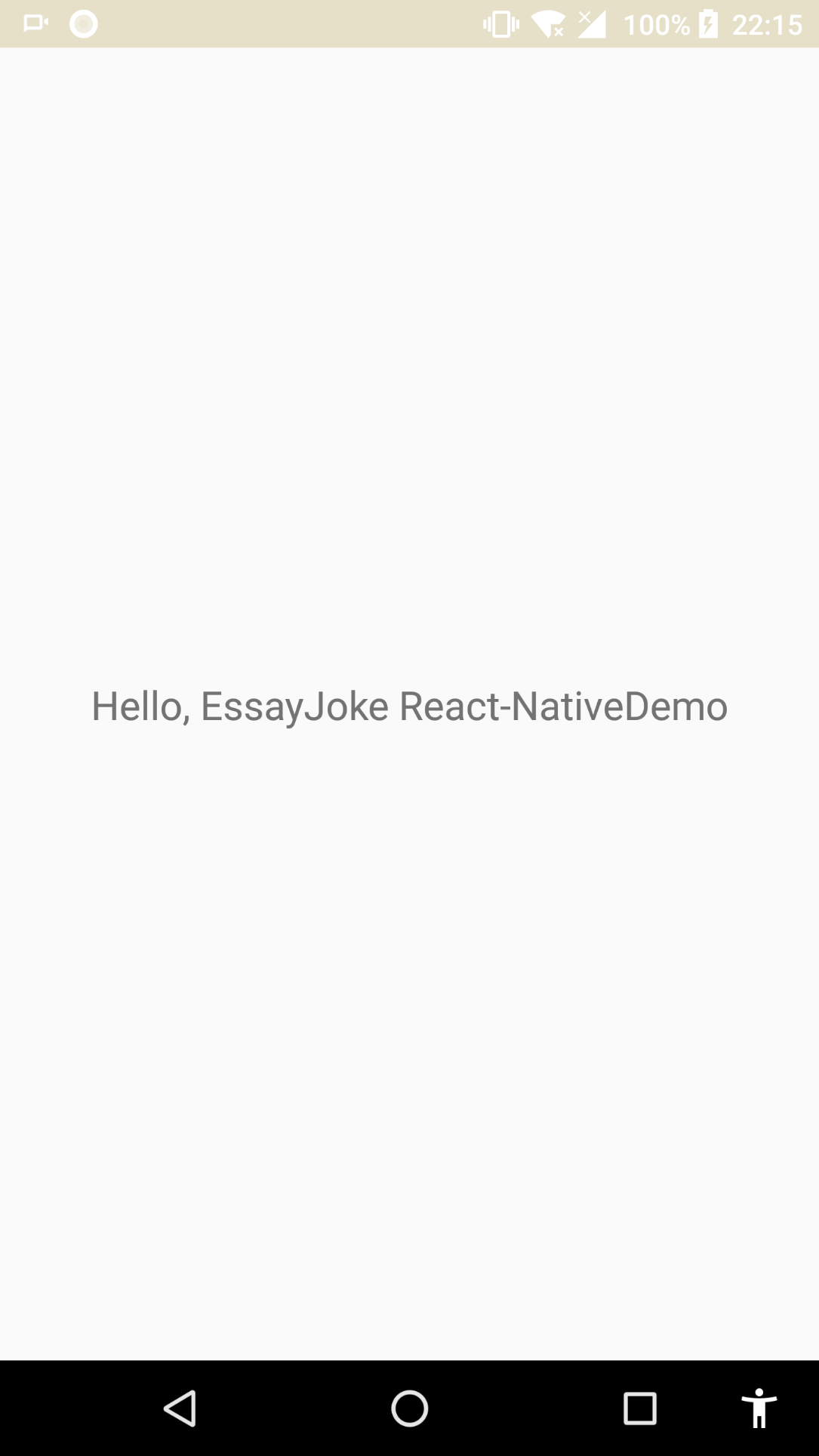
#### 2 监测npm server是否可以访问

在电脑的浏览器中访问以下地址，如果可以访问，一般来说server就启动成功了

http://localhost:8081/index.android.bundle?platform=android

#### 3 运行APP

点击跳转到RN界面，运行效果如下：



注意，运行过程中可能会有一些不顺利的地方，可以搜索网络上的答案解决

# 第二部分 ReactNative

http://idea.codebeta.cn

## 1 RN基础

Props（属性）