

React Native 混合开发(iOS篇)



慕课网
已认证帐号

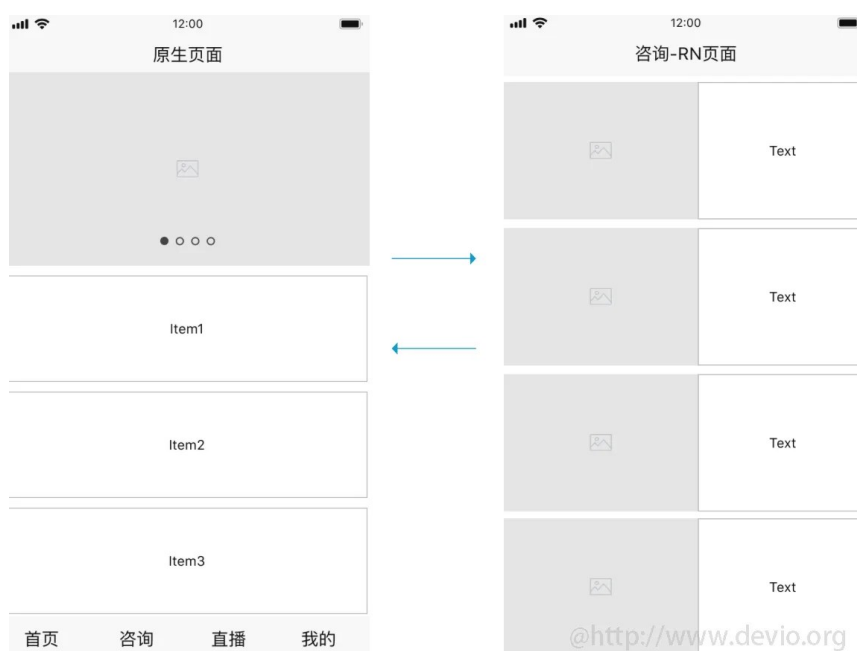
+ 关注

5 人赞同了该文章

在React Native的应用场景中，有时候一个APP只有部分页面是由React Native实现的，比如：我们常用的携程App，它的首页下的很多模块都是由React Native实现的，这种开发模式被称为混合开发。

混合开发的一些其他应用场景：

在原有项目中加入RN页面，在RN项目中加入原生页面

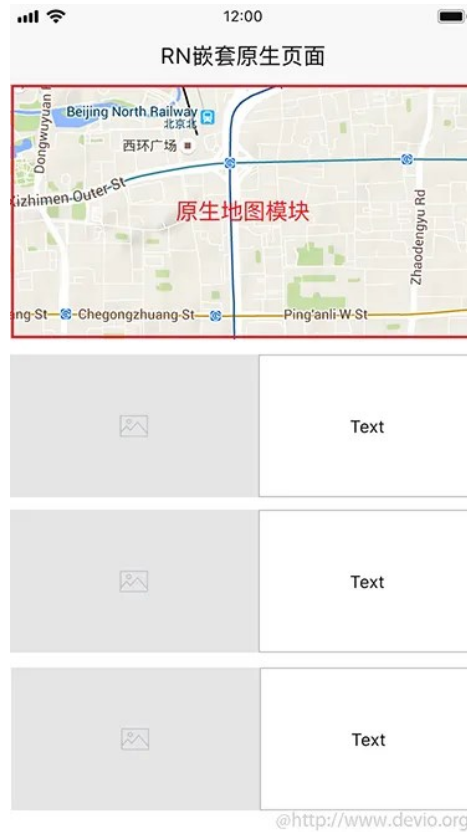


原生页面中嵌入RN模块





RN页面中嵌入原生模块



将React Native集成到现有的iOS应用中需要如下几个主要步骤：

- 首先，你需要有一个React Native项目；
- 为已存在的iOS应用添加React Native所需要的依赖；
- 创建index.js并添加你的React Native代码；
- 创建一个ViewController来承载React Native，在这个ViewController中创建一个RCTRootView来作为React Native服务的容器；
- 启动React Native的Packager服务，运行应用；
- (可选)根据需要添加更多React Native的组件；
- 运行、调试、打包、发布应用；
- 升职加薪、迎娶白富美，走向人生巅峰！；

1. 创建一个React Native项目

在做混合开发之前我们首先需要创建一个没有Android和iOS模块的React Native项目。我们可以通过两种方式来创建一个这样的React Native项目：

- 通过 `npm` 安装react-native的方式添加一个React Native项目；
- 通过 `react-native init` 来初始化一个React Native项目；

通过 `npm` 安装react-native的方式添加一个React Native项目

第一步：创建一个名为 `RNHybrid` 的目录，然后在该目录下添加一个包含如下信息的 `package.json`：

```
{
```

```

    "name": "RNHybrid",
    "version": "0.0.1",
    "private": true,
    "scripts": {
      "start": "node node_modules/react-native/local-cli/cli.js start"
    }
  }
}

```

第二步：在为package.json添加react-native

在该目录下执行：

```
npm install --save react-native
```

执行完上述命令之后，你会看到如下警告：

```

jphdeMacBook-Pro:RNHybrid jph$ vim package.json
jphdeMacBook-Pro:RNHybrid jph$ npm install --save react-native @http://www.devio.org
> fsevents@1.2.4 install /Users/jph/Downloads/RNHybrid/node_modules/fsevents
> node install
[fsevents] Success: "/Users/jph/Downloads/RNHybrid/node_modules/fsevents/lib/binding/Release/node-v57-darwin-x64/fse.node" already installed
Pass --update-binary to reinstall or --build-from-source to recompile
npm WARN created a lockfile as package-lock.json. You should commit this file.
npm WARN react-native@0.55.4 requires a peer of react@16.3.1 but none is installed. You must install peer dependencies yourself.
npm WARN eslint-plugin-react-native@3.2.1 requires a peer of eslint@^3.17.0 || ^4.0.0 but none is installed. You must install peer dependencies yourself.
+ react-native@0.55.4
added 717 packages in 67.517s

```

其中，有一条警告 `npm WARN react-native@0.55.4 requires a peer of react@16.3.1 but none is installed` 告诉我们需要安装 `react@16.3.1`：

```
npm install --save react@16.3.1
```

至此，一个不含Android和iOS模块的React Native项目便创建好了。

提示：npm 会在你的目录下创建一个 `node_modules`，`node_modules` 体积很大且是动态生成了，建议将其添加到 `.gitignore` 文件中；

通过react-native init来初始化一个React Native项目

除了上述方式之外，我们也可以通过 `react-native init` 命令来初始化一个React Native项目。

react-native init RNHybrid

上述命令会初始化一个完成的名为RNHybridiOS的React Native项目，然后将我们里面的 `android` 和 `ios` 目录删除，替换成已存在Android和iOS项目。

2. 添加React Native所需要的依赖

在上文中我们已经创建了一个React Native项目，接下来我们来看一下如何将这个React Native项目和我们已经存在的Native项目进行融合。

在进行融合之前我们需要将已经存在的Native项目放到我们创建的RNHybrid下，比如：我有一个名为 `RNHybridiOS` 的iOS项目，将其放到RNHybrid目录下：

```

RNHybrid
├── RNHybridiOS
├── package.json
├── node_modules
└── .gitignore

```

第一步：配置CocoaPods依赖

接下来我们需要为已经存在的RNHybridiOS项目添加 React Native依赖，在RNHybridiOS目录下创建一个 `Podfile` 文件(如果已经添加过可跳过)：

```
pod install
```

然后，我们在 `Podfile` 文件中添加如下代码：

```

target 'RNHybridiOS' do
  # Uncomment the next line if you're using Swift or would like to use dynamic framework

```

```
# use_frameworks!
```

```
# Your 'node_modules' directory is probably in the root of your project,
# but if not, adjust the `:path` accordingly
pod 'React', :path => '../node_modules/react-native', :specs => [
  'Core',
  'CxxBridge', # Include this for RN >= 0.47
  'DevSupport', # Include this to enable In-App Devmenu if RN >= 0.43
  'RCTText',
  'RCTNetwork',
  'RCTWebSocket', # Needed for debugging
  'RCTAnimation', # Needed for FlatList and animations running on native UI thread
  # Add any other subspecs you want to use in your project
]
# Explicitly include Yoga if you are using RN >= 0.42.0
pod 'yoga', :path => '../node_modules/react-native/ReactCommon/yoga'

# Third party deps podspec link
pod 'DoubleConversion', :podspec => '../node_modules/react-native/third-party-podspecs/DoubleConversion.podspec'
pod 'glog', :podspec => '../node_modules/react-native/third-party-podspecs/glog.podspec'
pod 'Folly', :podspec => '../node_modules/react-native/third-party-podspecs/Folly.podspec'

end
```

接下来在 RNHybridIOS 目录下执行：

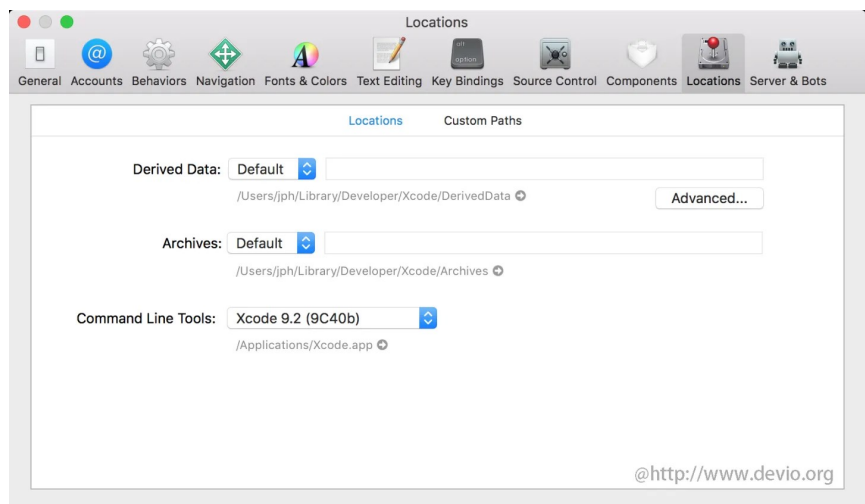
```
pod install
```

执行成功之后，你会看到如下输出：

```
Downloading dependencies
Installing DoubleConversion (1.1.5)
Installing Folly (2016.09.26.00)
Installing React (0.55.4)
Installing boost-for-react-native (1.63.0)
Installing glog (0.3.4)
Installing yoga (0.55.4.React)
Generating Pods project
Integrating client project

[[[] Please close any current Xcode sessions and use 'RNHybridIOS.xcworkspace' for this project from now on.
Sending stats
Pod installation complete! There are 11 dependencies from the Podfile and 6 total pods installed.
```

如果：出现 `xcrun` 的错误，需要安装 Command Line Tools for Xcode，打开XCode -> Preferences -> Locations 选择Command Line Tools：



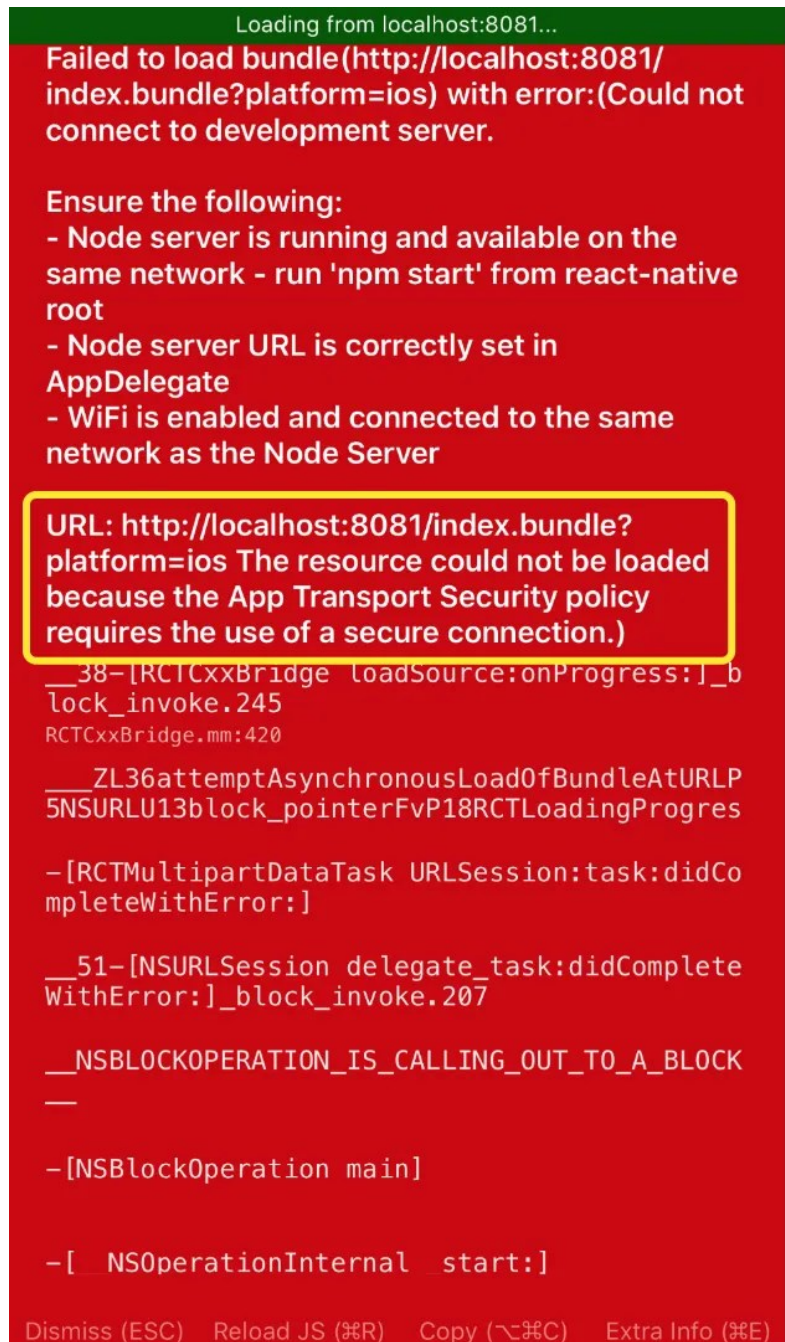
如果：出现 `Unable to find a specification for 'boost-for-react-native' depended upon by Folly` 的错误，则需要在目录下执行 `pod update` 即可。

```
jphdeMacBook-Pro:RNHybridIOS jph$ pod install
Analyzing dependencies
Fetching podspec for 'DoubleConversion' from '../node_modules/react-native/third-party-podspecs/DoubleConversion.podspec'
Fetching podspec for 'Folly' from '../node_modules/react-native/third-party-podspecs/Folly.podspec'
Fetching podspec for 'React' from '../node_modules/react-native'
Fetching podspec for 'glog' from '../node_modules/react-native/third-party-podspecs/glog.podspec'
Fetching podspec for 'yoga' from '../node_modules/react-native/ReactCommon/yoga'
[[[] Unable to find a specification for 'boost-for-react-native' depended upon by 'Folly'

[[[] Automatically assigning platform 'ios' with version '11.2' on target 'RNHybridIOS' because no platform was specified. Please specify a platform for this target in your Podfile. See 'https://guides.cocoapods.org/working-with-xcode/1.3/4-working-with-xcode-11.html' for more information.]]
```

第二步：设置App Transport Security Settings

由于我们的 RNHybridiOS 应用需要加载本地服务器上的JS Bundle，而且是http的协议传输，所以需要设置 App Transport Security Settings，让其支持http传输，否则会出现如下错误：



由于 App Transport Security Settings 网上设置的教程有很多，在这里就不重复了，需要的同学可以Google一下xcode http。

3. 创建index.js并添加你的React Native代码

通过上述两步，我们已经为RNHybridiOS项目添加了React Native依赖，接下来我们来开发一些JS代码。

在RNHybrid目录下创建一个 index.js 文件并添加如下代码：

```
import { AppRegistry } from 'react-native';
import App from './App';

AppRegistry.registerComponent('App1', () => App);
```

上述代码，AppRegistry.registerComponent('App1', () => App); 目的是向React Native注册一个名为 App1 的组件，然后我会在第四步给大家介绍如何在iOS中加载并显示出这个组件。

另外，在上述代码中我们引用了一个 App.js 文件：

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

```

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

type Props = {};
export default class App extends Component<Props> {
  render() {
    return (
      <View style={styles.container}>
        <Text style={styles.welcome}>
          this is App
        </Text>
      </View>
    );
  }
}

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

```

这个 App.js 文件代表了我们React Native的一个页面，在这个页面中显示了 `this is App` 的文本内容。

4. 为React Native创建一个ViewController和RCTRootView来作为容器

经过上述3、4步，我们已经为RNHybridiOS项目添加了React Native依赖，并且创建一些React Native代码和注册了一个名为 App1 的组件，接下来我们来学习下如何在RNHybridiOS项目中使用这个 App1 组件。

创建RNPageController

首先我们需要创建一个ViewController和RCTRootView来作为React Native的容器。

```

#import "RNPageController.h"

#import <React/RCTRootView.h>

#import <React/RCTBundleURLProvider.h>

#import <React/RCTEventEmitter.h>

@interface RNPageController ()

@end

@implementation RNPageController

- (void)viewDidLoad {
    [super viewDidLoad];
    [self initRCTRootView];
}

- (void)initRCTRootView{
    NSURL *jsCodeLocation;

    // jsCodeLocation = [NSURL URLWithString:@"http://localhost:8081/index.bundle?platform=ios"]

    jsCodeLocation = [[RCTBundleURLProvider sharedSettings] jsBundleURLForBundleRoot:@"index"
                                                                    options:nil];

    //这个"App1"名字一定要和我们在index.js中注册的名字保持一致

    RCTRootView *rootView = [[RCTRootView alloc] initWithBundleURL:jsCodeLocation
                                                                moduleName:@"App1"
                                                                launchOptions: nil];

    self.view=rootView;
}

@end

```


参数说明

- `initWithBundleURL`：用于设置 `jsCodeLocation`，有上述三种设置方式，在开发阶段推荐使用 `RCTBundleURLProvider` 的形式生成 `jsCodeLocation`，`release` 只会使用静态 `js bundle`；
- `moduleName`：用于指定 RN 要加载的 JS 模块名，也就是上文中所讲的在 `index.js` 中注册的模块名；
- `launchOptions`：主要在 `AppDelegate` 加载 JS Bundle 时使用，这里传 `nil` 就行；
- `initialProperties`：接受一个 `NSDictionary` 类型的参数来作为 RN 初始化时传递给 JS 的初始化数据

5. 运行 React Native

经过上述的步骤，我们已经完成了对一个现有 iOS 项目 `RNHybridiOS` 添加了 RN，并且创建了一个 `RNPageController` 来加载我们在 JS 中注册的名为 `App1` 的 RN 组件。

接下来我们来启动 RN 服务器，运行 `RNHybridiOS` 项目打开 `RNPageController` 来查看效果：

```
npm start
```

在 `RNHybrid` 的根目录运行上述命令，来启动一个 RN 本地服务：

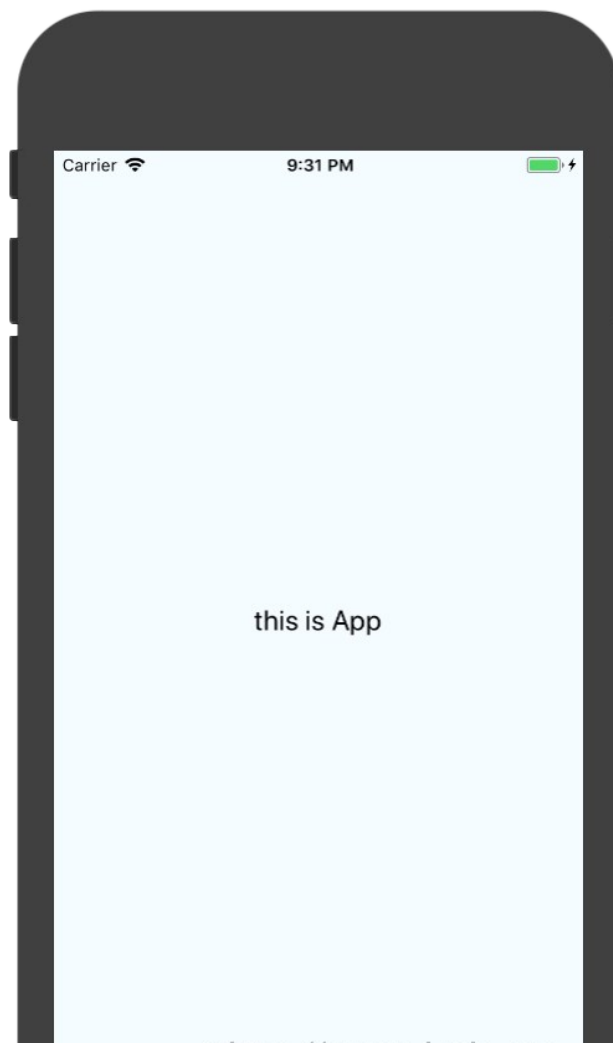
```
ohdeMacBook-Pro:RNHybrid jph$ npm start
RNHybrid@0.0.1 start /Users/jph/Documents/GitRepo/RNHybrid
node node_modules/react-native/local-cli/cli.js start
canning folders for symlinks in /Users/jph/Documents/GitRepo/RNHybrid/node_modules (9n

Running Metro Bundler on port 8081.

Keep Metro running while developing on any JS projects. Feel free to
close this tab and run your own Metro instance if you prefer.

https://github.com/facebook/react-native
```

然后我们打开 Xcode，点击运行按钮或者通过快捷键 `Command+R` 来将 `RNHybridiOS` 安装到模拟器上：



6. 添加更多React Native的组件

我们可以根据需要添加更多的React Native的组件:

```
import { AppRegistry } from 'react-native';
import App from './App';
import App2 from './App2';

AppRegistry.registerComponent('App1', () => App);
AppRegistry.registerComponent('App2', () => App);
```

然后, 在Native中根据需要加载指定名字的RN组件即可。

7. 调试、打包、发布应用

调试

调试这种混合的RN应用和调试一个纯RN应用时一样的, 都是 `Command + D` 打开 RN 开发者菜单, `Command + R` 进行reload JS, 另外大家也可以通过学习课程来掌握更多RN调试的技巧。

打包

虽让, 通过上述步骤, 我们将RN和我们的RNHybridiOS项目做了融合, 但打包RNHybridiOS你会发现里面并不包含JS部分的代码, 如果要将JS代码打包进iOS ipa包中, 可以通过如下命令:

```
react-native bundle --entry-file index.js --platform ios --dev false --bundle-output
```

记得在运行上述命令之前先创建一个 `release_ios` 目录。

参数说明

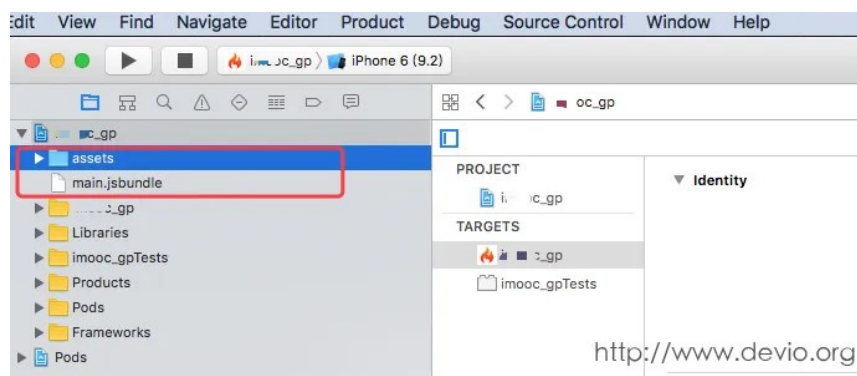
- `--platform ios`: 代表打包导出的平台为iOS;
- `--dev false`: 代表关闭JS的开发者模式;
- `--entry-file index.js`: 代表js的入口文件为 `index.js`;
- `--bundle-output`: 后面跟的是打包后将JS bundle包导出到的位置;
- `--assets-dest`: 后面跟的是打包后的一些资源文件导出到的位置;

上述命令执行完成之后, 会在 `release_ios` 目录下生

成 `main.jsbundle`, `main.jsbundle.meta`, 以及 `assets` 目录(如果RN中用到了一些图片资源的话)。

将js bundle包和图片资源导入到iOS项目中

这一步我们需要用到XCode, 选择assets文件夹与main.jsbundle文件将其拖拽到XCode的项目导航面板中即可。



然后, 修改 `jsCodeLocation`, 添加如下代码:


```
...
    NSURL *jsCodeLocation;http://coding.imooc.com/class/304.html
    //jsCodeLocation = [[RCTBundleURLProvider sharedSettings] jsBundleURLForBundleRoot:@
    +jsCodeLocation = [[NSBundle mainBundle] URLForResource:@"main" withExtension:@"jsbu
```

1 上述代码的作用是让React Native去使用我们刚才导入的jsbundle，这样以来我们就摆脱了对本地nodejs服务器的依赖。

提示：如果在项目中使用了CodePush热更新，那么我们需要就可以直接通过CodePush来读取本地的jsbundle，方法如下：

```
...
    NSURL *jsCodeLocation;
#ifdef DEBUG
        jsCodeLocation = [[RCTBundleURLProvider sharedSettings] jsBundleURLForBundleRoot:
    #else
        jsCodeLocation = [CodePush bundleURL];
    #endif
    ...
```

到目前为止呢，我们已经将js bundle包和图片资源导入到iOS项目中，接下来我们就可以发布我们的iOS应用了。

发布iOS应用

发布iOS应用我们需要有一个99美元的账号用于将App上传到AppStore，或者是299美元的企业级账号用于将App发布到自己公司的服务器或第三方公司的服务器。

接下来我们就需要进行申请APPID 在Tunes Connect创建应用 打包程序 将应用提交到app store 等几大步骤。

因为官方文档中有详细的说明，在这我就不再重复了。

作者：CrazyCodeBoy

链接：imooc.com/article/25316...

来源：慕课网

本文原创发布于慕课网，转载请注明出处，谢谢合作

发布于 2020-11-23 15:38

原生应用

React Native

慕课网



写评论 | 你和作者最近都关注了 iOS 开发 话题



还没有评论，发表第一个评论吧

推荐阅读



说说 react native 的 require 过程

从优先级说起在 package.json 里，除了 main 字段之外，我们一般还可以定义 browser 字段。这样



React Native 搭建开发环境
- 运行 React Native

胡恒铭 发表于不定期更新...

React Native布局详细指南

慕课网

可以指导 webpack 在node环境使用main，在浏览器环境使用browser。在rn当中，我们还可...

alsotang

为什么我不是个 React Native 开发者

Javen... 发表于极光日报

▲ 赞同 5

▼

● 添加评论

🔗 分享

♥ 喜欢

★ 收藏

📄 申请转载

⋮