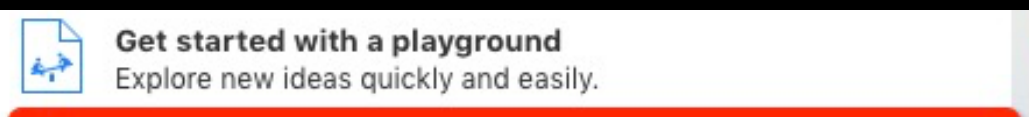
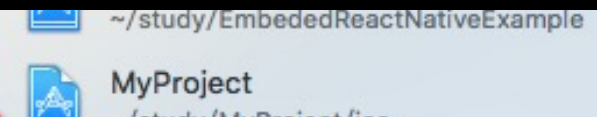




reactnative与现有原生ios项目集成

03 MAY 2016 on reactnative

经过一个月的折腾，我们终于将reactnative项目成功地集成到酷狗8.0的ios端，并完成所有原生app与javascript交互的所有功能，接下来将会继续在独立繁星app中做集成，下面是酷狗live项目回顾页的截图：

大概功能就是：视频部分是原生app，视频以下部分是js，js可以通知视频播放哪个视频，视频可以通知js视频播放，或者播放下一个等等。

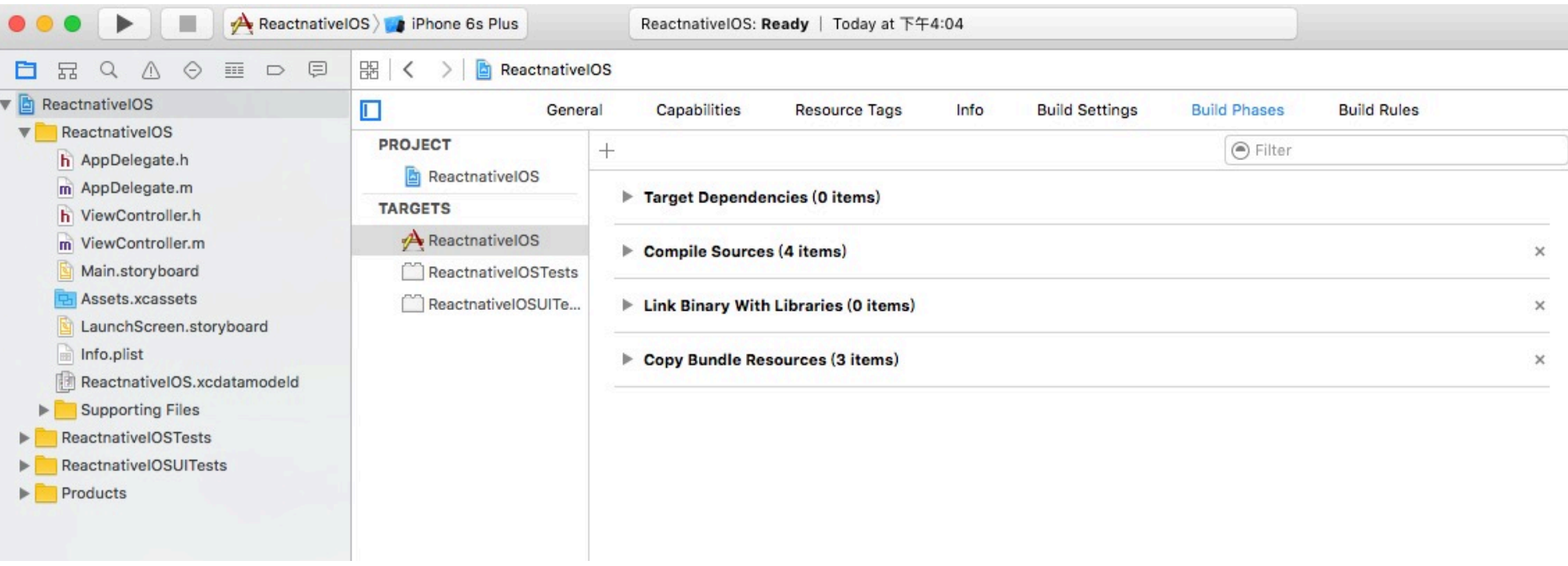
下面我想通过一个简单的ios项目介绍一下原生ios项目是如何与reactnative集成的。 先贴一下官方教程：

Integrating with Existing Apps 植入原生应用

对已有项目，官方推荐用CoCoaPods（一个帮助ios开发人员更方便地管理第三方依赖的工具）进行项目集成，不过悲催的是，当我把我们的酷狗ios项目clone下来才发现，他们并没有使用Cocoapods，而且他们也不推荐使用，因为Cocoapods管理第三方依赖的时候会自动创建一些目录，不便整个团队的管理。好吧，那就只能手动集成咯~

下面我使用Xcode创建一个新的项目作为我们原生的ios项目：

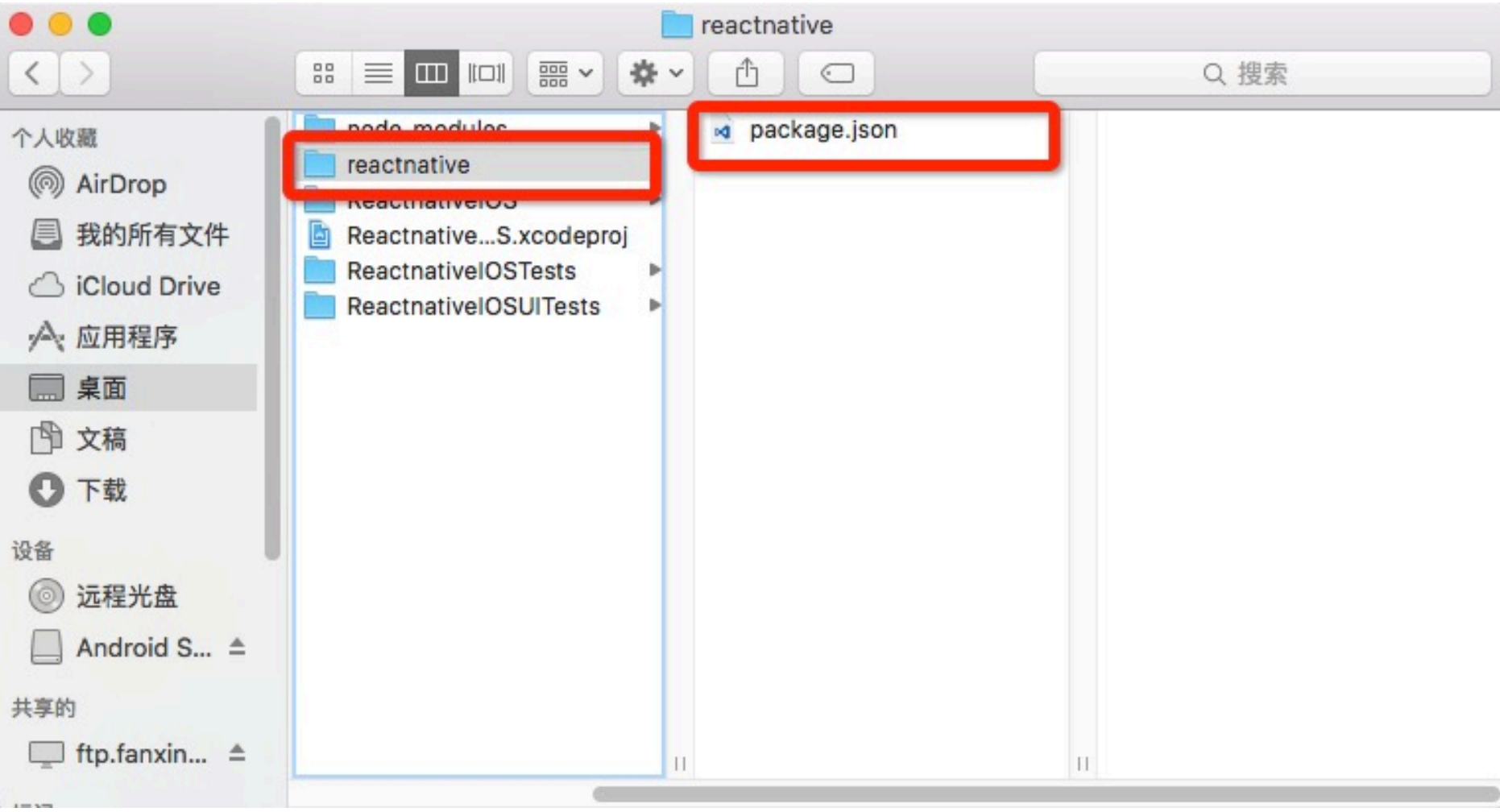
打开Xcode -> Create a new Xcode project -> Single View Application -> ReactnativeIOS



现在，我们已经存在一个名为ReactnativeIOS的ios项目，下面我们看看怎么集成reactnative到该项目中。

1、安装react-native

我们在ReactnativeIOS项目目录建一个reactnative目录，用于存放我们的react-native相关文件，再创建一个package.json文件，用于初始化react-native。




```
//package.json
{
    "name": "ReactnativeIOS",
    "version": "0.0.1",
    "private": true,
    "dependencies": {
        "react": "^0.14.8",
        "react-native": "^0.22.2"
    }
}
```

执行安装： `$ cd reactnative` `$ npm install`

安装成功后，reactnative目录会产生一个node_modules，里面就是react-native依赖的所有项目包。

2、创建index.ios.js文件

在reactnative目录下创建`index.ios.js`文件：

```
//index.ios.js

'use strict';

var React = require('react-native');
var {
    Text,
    View
} = React;

var styles = React.StyleSheet.create({
    container: {
        flex: 1,
        backgroundColor: 'red'
    }
});

class SimpleApp extends React.Component {
    render() {
        return (
            <View style={styles.container}>
                <Text>This is a simple application.
</Text>
            </View>
        )
    }
}
```

```
React.AppRegistry.registerComponent('SimpleApp', () =>
SimpleApp);
```

这就是我们js程序的入口文件。ok，以上我们已经完成react-native的准备工作，接下来要开始集成啦~

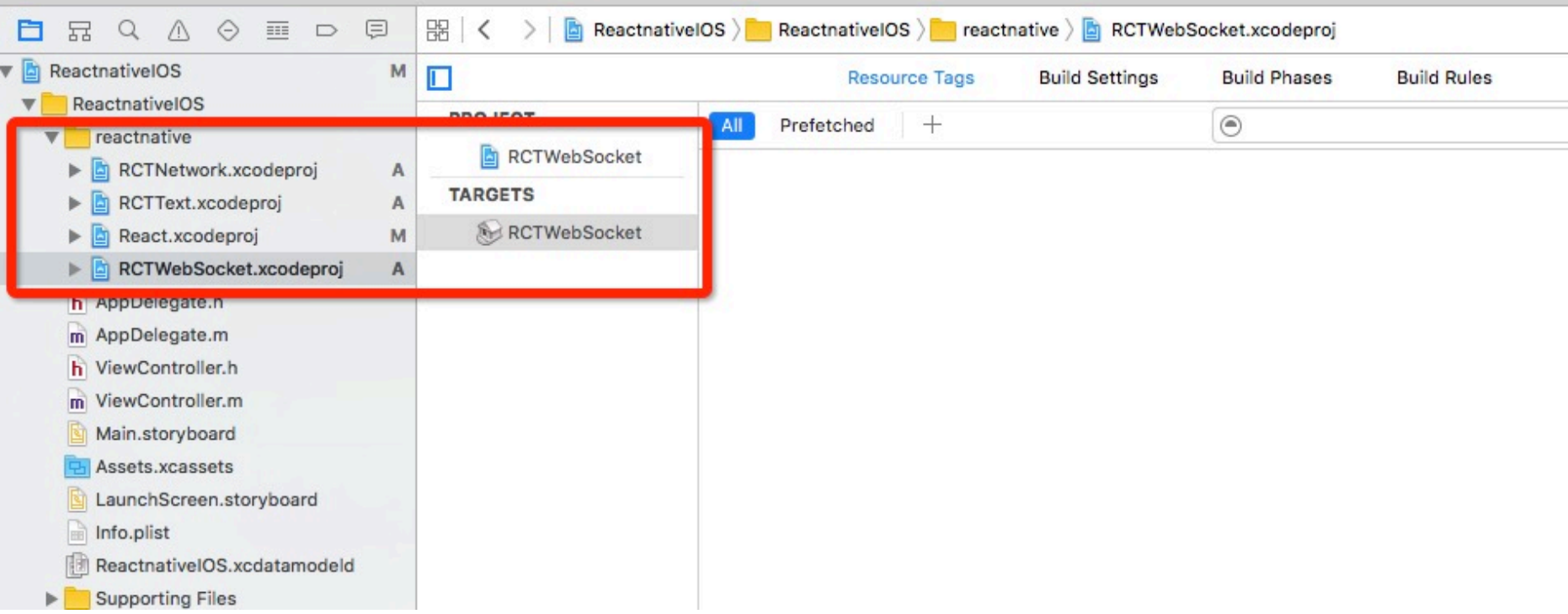
3、手动集成react-native

如果你的项目使用了Cocoapods，可以跳过这一步，直接看第4步，使用Cocoapods集成react - native。

- **添加react-native工程文件** 由于项目没有使用Cocoapods进行第三方依赖包管理，所有我们需要手动将react - native工程包添加到我们的原生ios工程中。打开`reactnative/node_modules/react-native`目录，找到相关的项目包，将React相关的工程包手动添加到项目中：

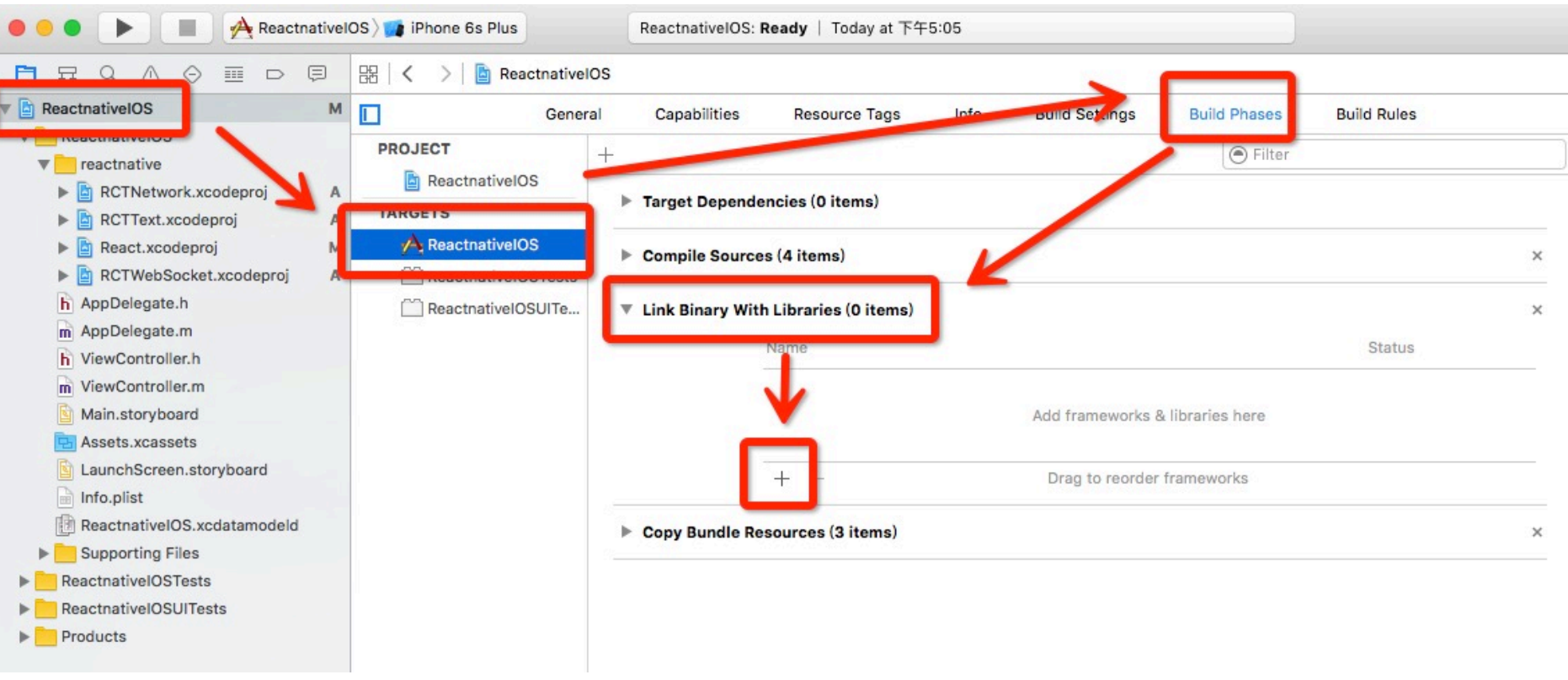
```
添加`react-native/React/React.xcodeproj`到项目中
添加`react-
native/Libraries/Network/RCTNetwork.xcodeproj`到项目中
添加`react-native/Libraries/Text/RCTText.xcodeproj`到项
目中
添加`react-
native/Libraries/WebSocket/RCTWebSocket.xcodeproj`到项目
中
添加`react-
native/Libraries/ActionSheetIOS/RCTActionSheet.xcodepro
j`到项目中
```

如图所示：

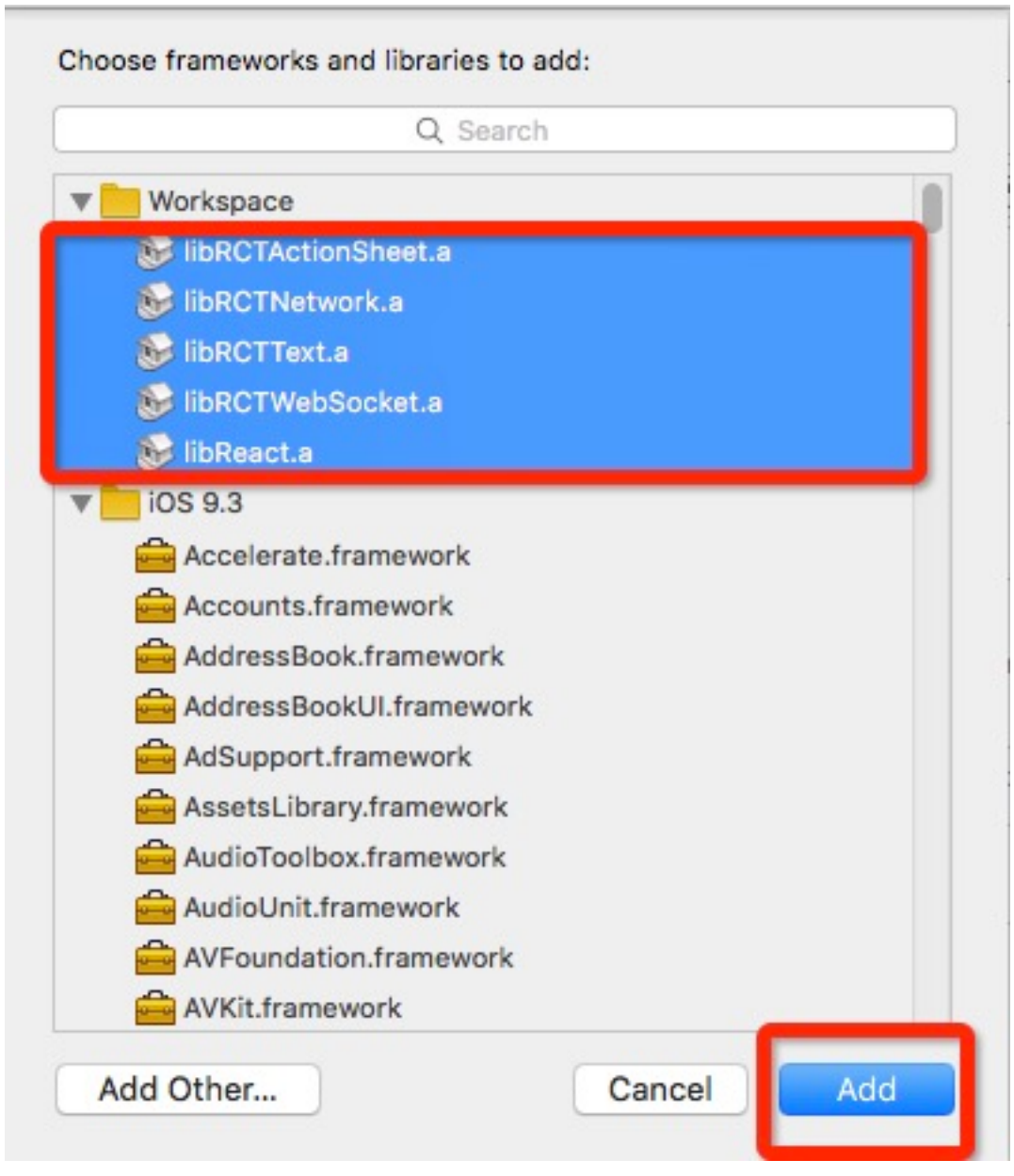


添加的原则是，你需要什么就添加什么工程包。好了，我们已经将React相关工程包手动添加到ReactnativeIOS工程项目的reactnative目录下。

- **添加相关frameworks文件** 接下来要将相关的frameworks文件添加到工程中，`ReactnativeIOS -> TARGETS -> ReactnativeIOS -> Build Phases -> Link Binary Width Libraries`。



点击`+`号，将所有`.a`文件选中并添加。



- **添加搜索头文件的地址** `ReactnativeIOS -> TARGETS -> ReactnativeIOS -> Build Settings -> Header Search Paths`，添加一条`$(SRCROOT)/reactnative/node_modules/react-native/React`，选择`recursive`。

这样我们就react-native集成到现有的ios工程中了。

以上是手动集成react-native工程到现有ios中，如果项目中使用了Cocoapods或者你现在就想用Cocoapods，怎么做呢？

4、Cocoapods集成react-native

如果你已经完成了第三步的手动集成，可以跳过这一步。

- **安装Cocoapods** `$ gem install cocoapods`

- **创建Podfile** Cocoapods通过解析Podfile指定的工程文件进行依赖安装，我们在工程目录下创建一个Podfile文件：

```
# 取决于你的工程如何组织，你的node_modules文件夹可能会在别的地方。
# 请将:path后面的内容修改为正确的路径。
pod 'React', :path =>
'./reactnative/node_modules/react-native', :subspecs =>
[
  'Core',
  'RCTNetwork',
  'RCTText',
  'RCTWebSocket',
  # 添加其他你想在工程中使用的依赖。
]
```

记得要添加所有你需要的依赖。举例来说，元素如果不添加RCTText依赖就不能运行。

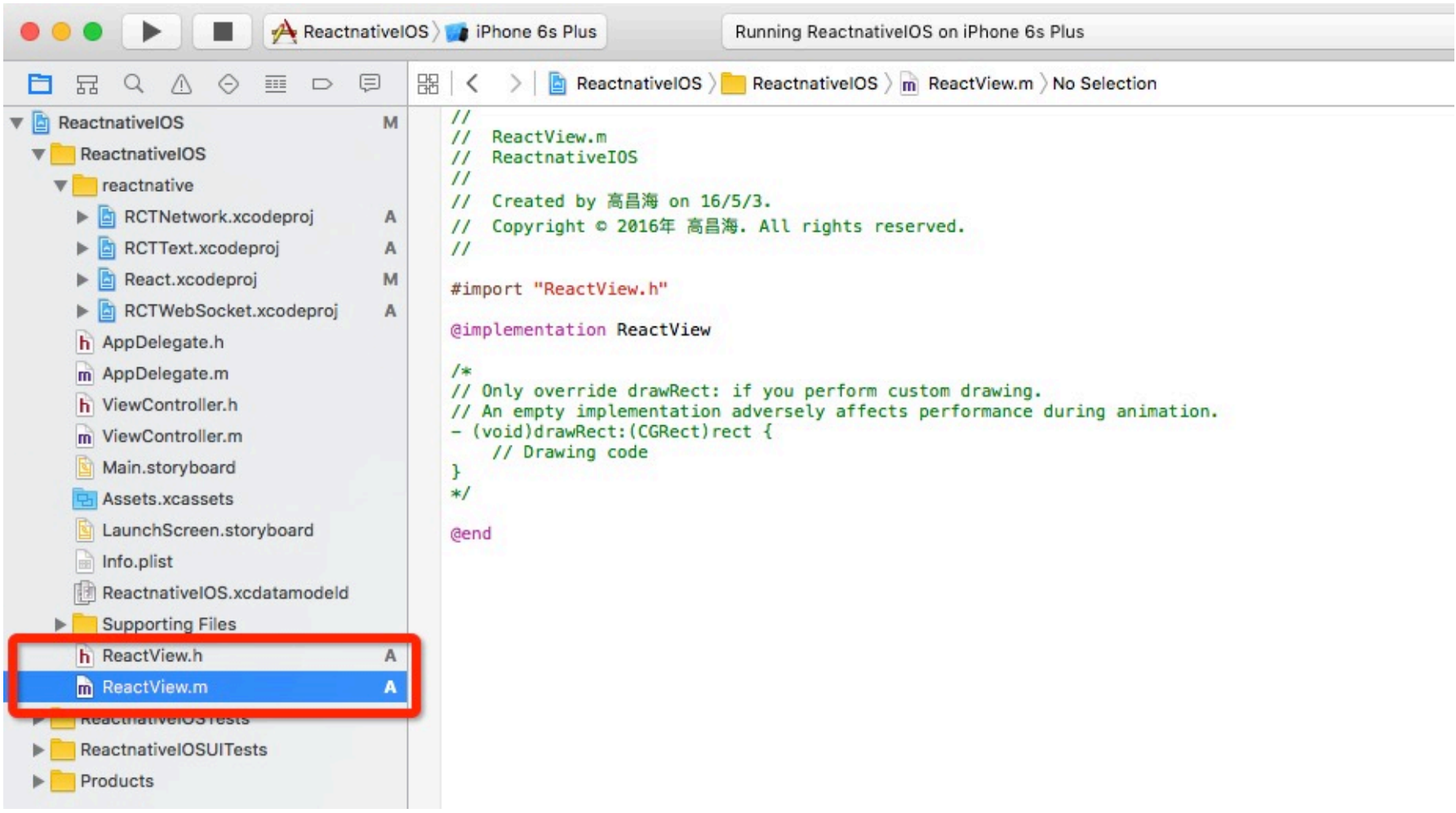
- **依赖文件安装** 进入工程目录，执行`$ pod install`，就完成了react - native工程包的集成。

- **打开ReactnativeIOS.xcworkspace** Cocoapods安装成功之后，工程目录会产生一个`ReactnativeIOS.xcworkspace`的工程文件，我们使用Xcode打开这个工程打开这个工程文件。

5、添加react-native应用

下面我们在原生ios应用中添加一个视图容器，用于展示react-native实现的内容。

- **在原生ios应用添加容器视图** 我们在工程文件下创建一个名为ReactView的UIView文件: `ReactnativeIOS目录 -> 右键 -> New File -> Cocoa Touch Class -> ReactView` ,



修改ReactView.m

```
#import "ReactView.h"
#import <RCTRootView.h>

@implementation ReactView

/*
// Only override drawRect: if you perform custom
drawing.
// An empty implementation adversely affects
performance during animation.
- (void)drawRect:(CGRect)rect {
    // Drawing code
```

```
}
*/

- (instancetype)initWithFrame:(CGRect)frame
{
    if (self = [super initWithFrame:frame]) {
        NSString * strUrl =
@"http://localhost:8081/index.ios.bundle?
platform=ios&dev=true";
        NSURL * jsCodeLocation = [NSURL
URLWithString:strUrl];

        RCTRootView * rootView = [[RCTRootView alloc]
initWithBundleURL:jsCodeLocation

moduleName:@"SimpleApp"

initialProperties:nil

launchOptions:nil];

        [self addSubview:rootView];

        rootView.frame = self.bounds;
    }
    return self;
}
@end
```

`ReactView.m`中通过`http://localhost:8081/index.ios.bundle?platform=ios&dev=true`加载bundle文件，由RCTRootView解析转化

原生的UIView，然后通过initWithFrame将frame暴露出去。

- **在原生ios应用中引用RCTRootView** 上面我们创建了一个RCTRootView，怎么在原生应用中引入该View呢？ 打开`ViewController.m`，在`viewDidLoad`方法中应用我们的ReactView。

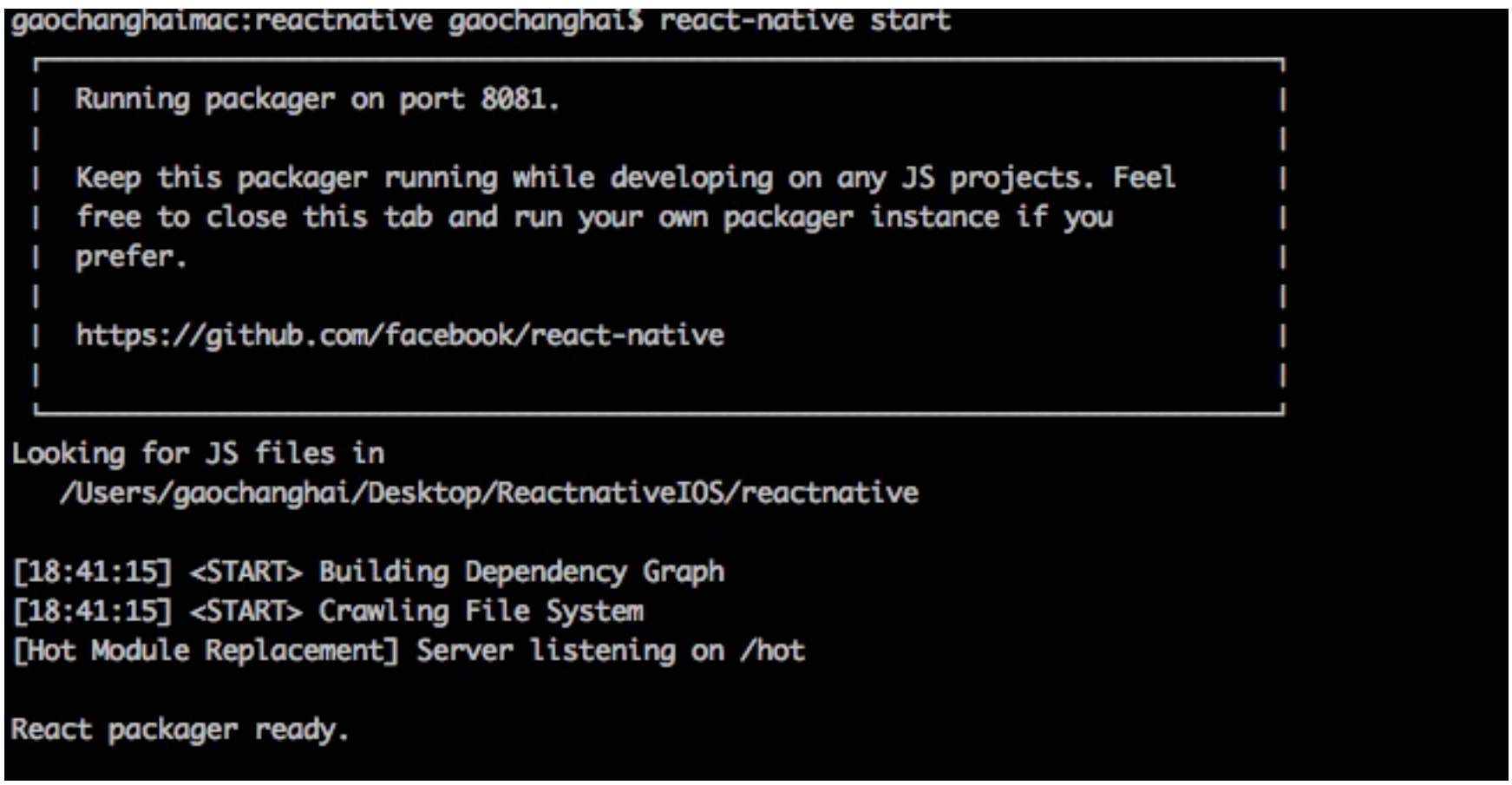
```
#import "ViewController.h"
#import "ReactView.h"

@interface ViewController ()

@property (weak, nonatomic) IBOutlet ReactView
*reactView;

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view,
typically from a nib.
    ReactView * reactView = [[ReactView alloc]
initWithFrame:CGRectMake(0, 40,
CGRectGetWidth(self.view.bounds), 100)];

    [self.view addSubview:reactView];
}

- (void)didReceiveMemoryWarning {
```

```
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

@end
```

至此，我们所有的集成工作已经完成，接下来就是运行项目啦。

6、启动开发服务器

在运行我们的项目之前，我们需要先启动我们的开发服务器。进入`reactnative目录`,然后启动。`$ cd reactnative` `$ react-native start`



看到上面的界面就标示我们的服务已经启动啦！

7、更新App Transport Security

直接运行项目会报`Could not connect to development server`错

误，官网中有这段话：

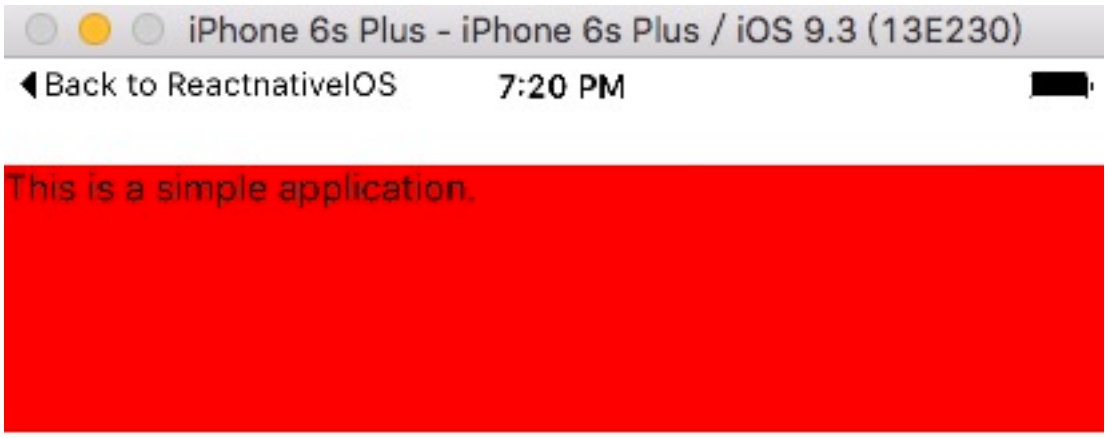
> *在iOS 9以上的系统中，除非明确指明，否则应用无法通过http协议连接到localhost主机。 我们建议你在Info.plist文件中将localhost列为App Transport Security的例外。 如果不这样做，在尝试通过http连接到服务器时，会遭遇这个错误 - Could not connect to development server.*

打开工程中的`Info.list`文件，添加下面配置即可：

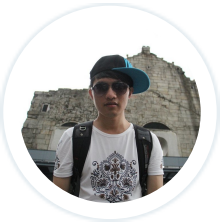
```
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSExceptionDomains</key>
    <dict>
        <key>localhost</key>
        <dict>

<key>NSTemporaryExceptionAllowsInsecureHTTPLoads</key>
            <true/>
        </dict>
    </dict>
</dict>
```

8、运行ios项目

通过Xcode点击项目或者`command + R`运行项目，如果顺利的话，就会看到成功运行的界面：



大功告成！掌握了原生ios项目集成raectnative的方法之后，我们就可以尽情地在原生项目中撸js代码啦～



贱客

Read more posts by this author.

GuangZhou *https://github.com/garygchai*

Share this post

READ THIS NEXT

YOU MIGHT ENJOY

前端编程模式

可能在大部分人来讲，前端就是可见的页面数据呈现正确就行。然而这样是不正确的，页面呈现是一部分，更多的是整体的可维护性。本篇讲述的主题就是应用后端开发思想进行前端开发。 后端开发，使用最广的就是java语言，而java给人的第一印象就是面向对象。面向对象的特性就是：封装，继承，多态。在实现面向对象过程中，很自然会产生MVC模型，以及分层结构：UI(视图接口), BLL(...

Webpack 扩展浅谈

Webpack 是什么 一句话介绍，Webpack 是一个当下最热门且生态完善的 Web 前端构建工具。为什么这么说呢？从开源项目使用情况来看，Grunt, Gulp, FIS, Browserify, Webpack, Rollup 等等，最近讨论及大家推荐最多的当属 Webpack 了，...

繁星**UED** © 2016

Proudly published with **Ghost**