**RN官网**

RN中文网址<https://reactnative.cn/docs/getting-started.html>

RN英文文档

**Windows => android，环境搭建和运行，**只能运行Android版本APP

1. 必须安装的依赖**Node、Python2、JDK 和 Android Studio**

（1）Node 的版本应大于等于 12

（2）Python 的版本必须为 2.x（不支持 3.x）

（3）JDK 的版本必须是 1.8（目前不支持 1.9 及更高版本）

（4）Android studio SDK 目录下adb工具需要使用23版本及以下，Android studio安装和环境变量配置参考上面网址即可。

2、代码编辑器可以使用**VSCode**

**MAC=> android，MAC系统必须10.15以上**，Mac可以运行安卓和iOS

1. **运行安卓APP必须安装的依赖有：Node、Watchman、JDK 和 Android Studio**

安卓版本需要具有**Android Studio**开发或者使用经验

**（1）nodejs安装，版本是否在 v12 以上**

地址：<https://nodejs.org/en/> 选择稳定版或者最新版

1. **npm install watchman**

npx nrm use taobao # 使用nrm工具切换淘宝源

npx nrm use npm # 如果之后需要切换回官方源可使用

npm install -g yarn

**（3）JDK安装，**Java Development Kit [JDK] 1.8（暂不支持 1.9 及更高版本）

javac -version 查看版本

**（4）Android Studio安装 和 SDK选择** Android 9 (Pie) ，安装后配置路径，**MAc下bash和zsh有差别**

方式1：Bash命令行vi ~/.bash\_profile 文档中添加配置路径

方式2：Zsh命令行.zshrc 文档中添加配置路径

export ANDROID\_HOME=$HOME/Library/Android/sdkexport PATH=$PATH:$ANDROID\_HOME/toolsexport PATH=$PATH:$ANDROID\_HOME/tools/binexport PATH=$PATH:$ANDROID\_HOME/platform-toolsexport PATH=$PATH:$ANDROID\_HOME/emulator

1. **运行IOS的APP必须安装的依赖有Node、Watchman、Xcode 和 CocoaPods，**Ios版本需要具有Xcode开发或者使用经验

（1）**nodejs安装，版本是否在 v12 以上**

地址：<https://nodejs.org/en/> 选择稳定版或者最新版

**（2）npm install watchman**

npx nrm use taobao # 使用nrm工具切换淘宝源

npx nrm use npm # 如果之后需要切换回官方源可使用

npm install -g yarn

1. **[Xcode](https://developer.apple.com/xcode/downloads/)安装，[Xcode](https://developer.apple.com/xcode/downloads/) 10 或更高版本,相应的MAC系统也要进行升级，否则安装不了最新版XCode**。你可以通过 App Store 或是到[Apple 开发者官网](https://developer.apple.com/xcode/downloads/)上下载。这一步骤会同时安装 Xcode IDE、Xcode 的命令行工具和 iOS 模拟器。
2. **CocoaPods安装**，是用 Ruby 编写的包管理器。从 0.60 版本开始 react native 的 iOS 版本需要使用 CocoaPods 来管理依赖。

sudo gem install cocoapods

#或者brew install cocoapods

**3.创建项目**

1.初始化项目**npx react-native init App**

2.打开Android studio进行编译android 目录文件**make Project**，查看build是否成功。

3.终端查看是否有设备 对于 Android 5.0+设备（包括模拟器）来说，将设备通过 USB 连接到电脑上后，可以使用[adb命令行工具](http://developer.android.com/tools/help/adb.html)来设定从设备到电脑的端口转发：**adb devices**

4.打开调试模式 **adb reverse tcp:8081 tcp:8081**

5.终端根目录启动项目 **react-native start**

6.连接手机调试运行安卓版本，注释：第一次会下载很多依赖，所以项目跑起来会时间较长，可能超过半小时。

**npx react-native run-android**

#yarn react-native run-android

7.连接手机调试运行运行iOS版本，注释：第一次会下载很多依赖，所以项目跑起来会时间较长，可能超过半小时。

**cd ios** 进入iOS目录

**pod install** 安装依赖

**cd ..** 返回根目录

**npx react-native run-ios**

**#yarn react-native run-ios**

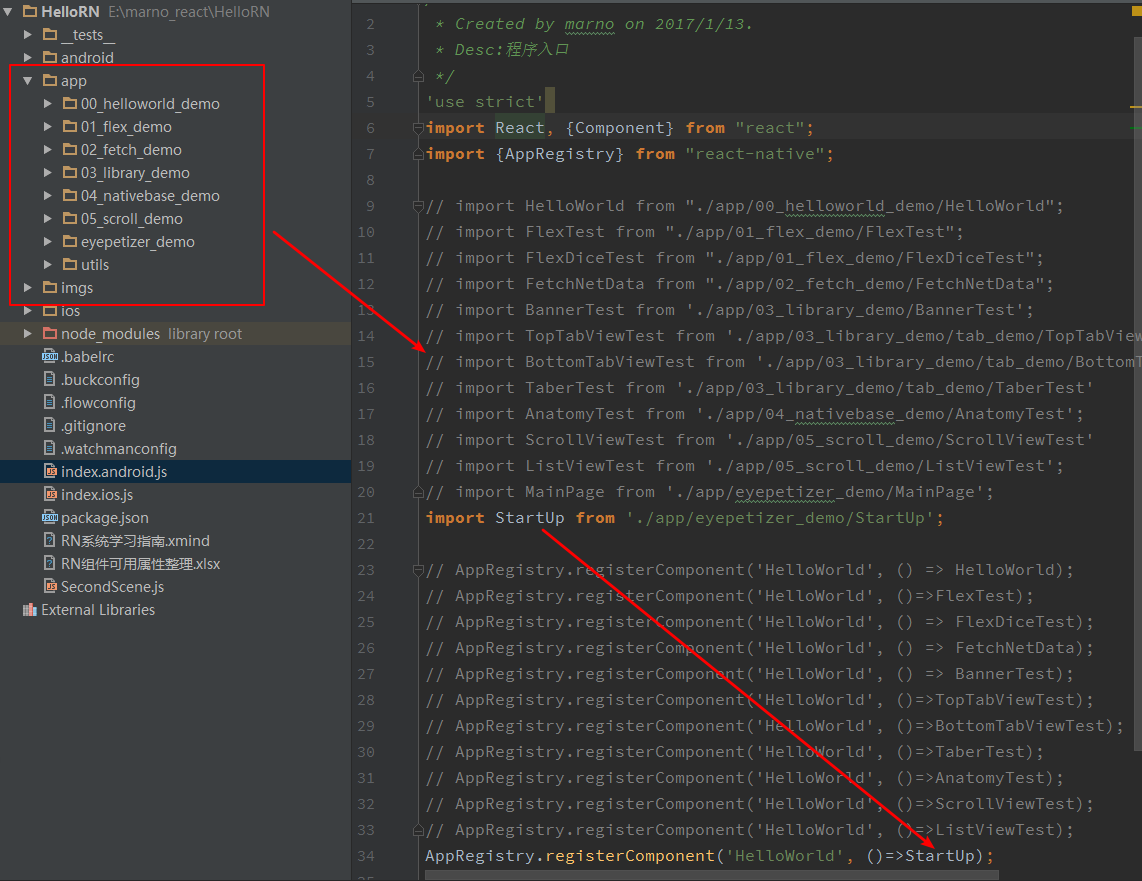
### 修改项目

现在你已经成功运行了项目，我们可以开始尝试动手改一改了：

* 使用你喜欢的文本编辑器打开App.js并随便改上几行
* 按两下 R 键，或是用 Menu 键（通常是 F2，在 Genymotion 模拟器中是⌘+M）打开开发者菜单，然后选择 Reload JS 就可以看到你的最新修改。

**1.如何编写demo**

是不是我们编写每一个知识点的demo都要通过 react-native init XXXX 的方式去新建一个项目呢？当然不是，我们可以把所有代码都写到一个demo里，并通过分包进行管理（app的文件夹是自己创建的，名字自定），这样自己看起来也比较方便。然后每个 demo 都通过 import 的方式导入到index.android.js/ index.ios.js 文件中进行使用（0.49版本之前使用，现在统一入口index.js）。

  
  
第一个项目

import React, { Component } from 'react';

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

class HelloWorldApp extends Component {

  render() {

    return (<Text>Hello World!</Text> );

  }

}

// 注意，这里用引号括起来的 'HelloWorldApp' 必须和你 init 时创建的项目名一致

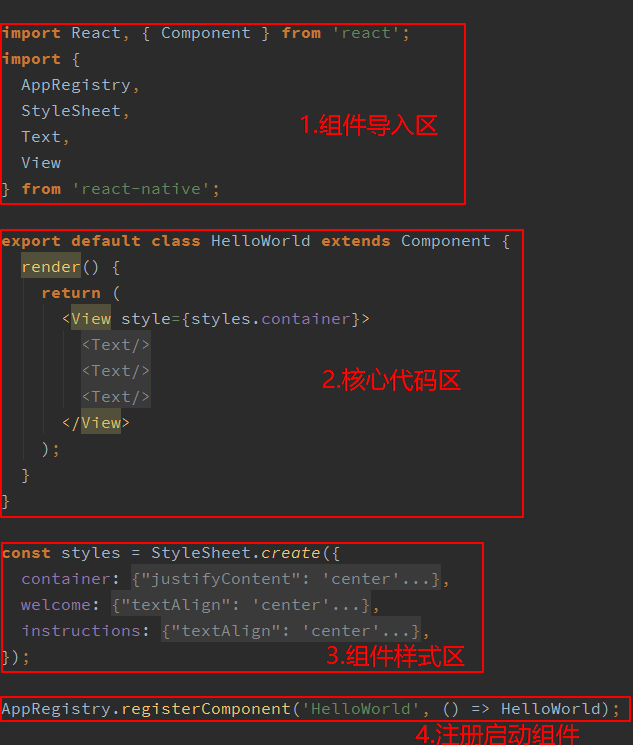
AppRegistry.registerComponent('HelloWorldApp', () => HelloWorldApp);

1. **了解 index.android/ios.js**

**注释：0.49后似乎已经统一入口了！就是index.js，index.android.js和index.ios.js那都是老版本了，而且多看文档！不要老看旧的教程！**

index.android.js 和 index.ios.js 分别作为 Android 手机和 iOS 手机上程序开始的入口，所以我们先来了解一下这个文件的结构。

1. ****组件导入区：**** 所有用到的组件都需要事先进行导入，包括样式也需要进行导入  
   ****2）核心代码区：****所有逻辑代码编写的地方  
   ****3）组件样式区：****render() 方法中用到的组件的样式，可以集中在这里编写  
   ****4）注册启动组件：****组件只有注册后才能运行。这里用到的 AppRegistry 也需要在组件导入区进行导入



1. **Flex布局**
2. **网络请求（fetch）**

这里要说明的是，如果你是 Android 或 iOS 工程师，那你需要习惯一下 RN 处理 Json 数据的方式（或者说 Web 处理 Json 的方式），我们通过网络请求回来的 Json Object 数据就可以直接进行操作，而不像 Native 开发，还需要通过什么额外的工具去进行 Json 的转换。这也是 Web 开发比较方便的地方。无论 Json Object 转 Json 字符串，还是 Json 字符串转 Json Object，都是非常方便的。如果非要把网络数据进行本地存储，那也很方便，通过解构赋值，直接就可以赋值给你创建的 Model 了。

1. **滚动视图**

现在的 App 基本都离不开列表数据展示，所以接下来的内容就让我们一起学习一下滚动视图的使用。在 RN 中滚动视图包含两部分的内容【ScrollView】和【ListView】。

8.1 **ScrollView**

其实和 Android 中的 ScrollView 一样，如果包含在 ScrollView 的组件超出屏幕范围后，就可以对内容进行滚动。而且 ScrollView 可以设置竖向或者横向的滚动 <https://www.jianshu.com/p/58301af1dbf2>

8.2 **ListView**

在 RN 中的 ListView 其实是继承了 ScrollView 的，这导致目前RN有一个比较头疼的问题，就是列表的性能问题

**8.3 RefreshControl（下拉刷新）**

原生下拉刷新 RefreshControl 的使用，该组件支持 Android 和 iOS，但是在两个平台上的呈现方式不太一样。支持 ScrollView 和 ListView 组件的刷新，使用也很简单，直接看代码。

**8.4 上拉加载**

官方并没有提供这样的组件，只能通过自己去实现，照着网上的教程我尝试了一些方式，但是感觉都不是特别满意，效果比较生硬。所以这部分内容暂时空缺，后面在完善。不过上面在下拉刷新那一节中提到的 react-native-gifted-listview，可以支持上拉加载更多。如果有需要，可以集成该框架进行使用。