React Native入门和基础原理

一、React Native的由来

1. React 是Facebook推出的一套框架。React是一种思想，是Facebook对于Web Components的理解与实现。其中ReactJS是Web端，React Native是iOS端、安卓端。值得注意的是React Native写的应用是Native App，而不是Hybrid App。
2. 也就是说不同于webview形式，用JS、HTML写代码和布局经过React Native 桥接转换成原生ObjectC，用户体验上和原生一样。

二、React Native的优缺点

一、优点：

1. 调试方便：ipa安装好之后，就不需要频繁编译了，只需要reload一下，把js代码从云服务器下载下来就可以呈现改变代码后的效果。而且RN支持hotReload，在调试界面的时候非常方便，修改代码之后保存，界面就自动跟着变化，这一点在调试的时候实在很爽，不过有时候有点慢，需要reload。chrome在线调试也挺不错，可以打断点，看日志。虽然没有xcode或者Android Studio那么浑然一体，但是作为脚本语言的调试工具，也是很厉害了。
2. css-layout布局: 这对于前端程序员来说，降低了不少学习成本，也大大减少了代码量。但是对于iOS或者安卓开发者来说，刚开始接触的时候，得接受一些思想上的转变.
3. 跨平台：大多数代码，只需要写一套，安卓和iOS就都可以运行了，游戏逻辑和数据。界面上一部分有一些平台区分，毕竟是从react包装上来的。跨平台理论上是可以减少开发成本的，减少开发人员数量。
4. 热更新：这可能也是大多数公司选择使用RN的主要原因。频繁的app升级会让用户很烦，而且苹果的审核真是很麻烦。现在很多大型app都使用了RN，毕竟繁多的业务迭代，每次都通过APP审核，也算是噩梦啊。
5. 后台硬：有着Facebook的支撑，相信会发展的很好。

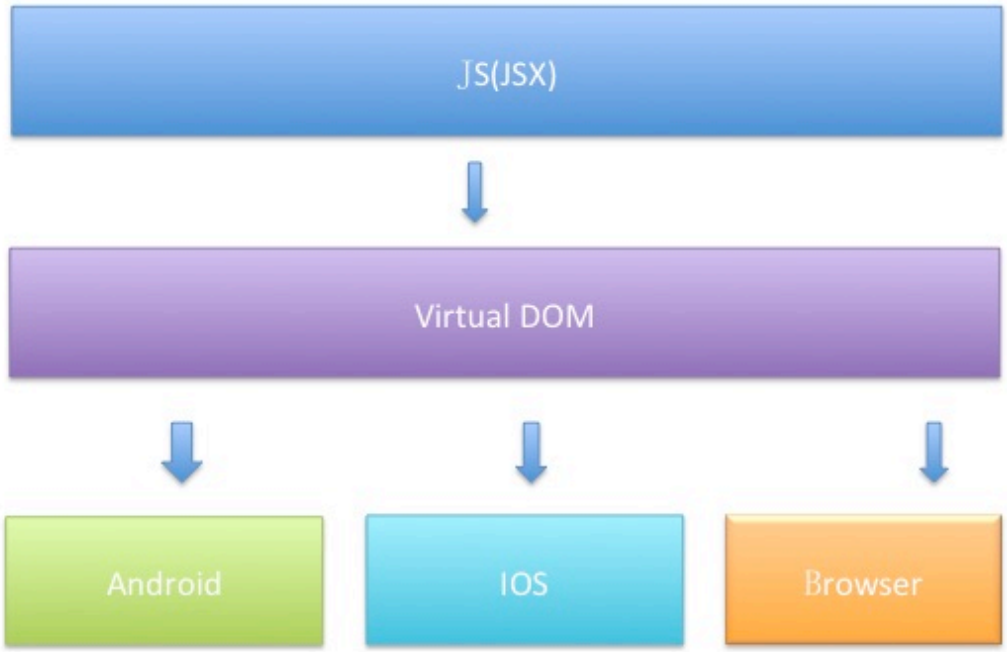
二、缺点：

1. 开发体验一般：虽然js语法很灵活，但毕竟是脚本语言，调试起来还是不方便，不好查错。我们用的表现较好的vscode编辑器，就这都感觉各种跳转很不方便，动不动就得全局搜索，可能是xcode用习惯了吧，脚本语言的编写也会慢慢习惯吧。
2. 文档很粗略：官网上的文档，就只是简单介绍用法和各个控件的属性，对细节的描述很少。当你遇到难解决的问题或者踩到坑了，上面基本找不到答案。（React Native 中文网）
3. 两个平台还没有完全统一：很多控件都是iOS专属，或者安卓专属。还有同一些控件，在不同平台上表现差异很大。
4. 控件不完善：这个其实挺多的，最基本的ListView，功能缺失，坑很多。Text不支持富文本，动画，手势，ScrollView等等等等。

5. 升级RN版本需要大动干戈：近期我们做了一次RN版本升级，从原来的0.42升到最近的0.50。真的挺麻烦。新版把PropTyps从React中移了出来，那么之前的引用方式就得变，所有的文件挨个查。之前使用的第三方库，有和PropTyps相关的，都得一一更新。之前很多界面布局的时候，在image上放置了一些其他控件，升级后会报错，然后一一调整。
6. 要做出优质app需要花费大量人力和时间去打磨。

三、React Native的原理

RN这套框架让 JS开发者可以大部分使用JS代码就可以构建一个跨平台APP。原理如图：



1. RN需要一个JS的运行环境， 在IOS上直接使用内置的javascriptcore， 在Android 则使用webkit.org官方开源的jsc.so
2. RN 会把应用的JS代码（包括依赖的framework）编译成一个js文件（一般命名为index.android.bundle), , RN的整体框架目标就是为了解释运行这个js 脚本文件
3. 如果是js 扩展的API， 则直接通过bridge调用native方法
4. 如果是UI界面， 则映射到virtual DOM这个虚拟的JS数据结构中，通过bridge 传递到native ， 然后根据数据属性设置各个对应的真实native的View。

四、安装开发环境

1. Homebrew：Homebrew, Mac系统的包管理器，用于安装NodeJS和一些其他必需的工具软件。终端输入指令：
    - /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
    - 译注：在Max OS X 10.11（El Capitan)版本中，homebrew在安装软件时可能会碰到/usr/local目录不可写的权限问题。可以使用下面的命令修复：
    - sudo chown -R `whoami` /usr/local

2. 使用Homebrew来安装Node.js:React Native目前需要NodeJS 5.0或更高版本。本文发布时Homebrew默认安装的是最新版本，一般都满足要求。终端输入指令: brew install node
3. 安装完node后建议设置npm镜像以加速后面的过程（或使用科学上网工具）。
    - 注意：不要使用cnpm！cnpm安装的模块路径比较奇怪，packager不能正常识别！终端输入指令：
    - npm config set registry https://registry.npm.taobao.org --global
    - npm config set disturl https://npm.taobao.org/dist --global

4. 安装Yarn、React Native的命令行工具（react-native-cli):
    - Yarn是Facebook提供的替代npm的工具，可以加速node模块的下载。
    - React Native的命令行工具用于执行创建、初始化、更新项目、运行打包服务（packager）等任务。终端指令：
    - npm install -g yarn react-native-cli

5. 安装完yarn后同理也要设置镜像源,终端指令：
    - yarn config set registry https://registry.npm.taobao.org --global
    - yarn config set disturl https://npm.taobao.org/dist --global
    - 如果你看到EACCES: permission denied这样的权限报错，那么请参照上文的homebrew译注，修复/usr/local目录的所有权，指令：
    - sudo chown -R `whoami` /usr/local

6. 安装完yarn之后就可以用yarn代替npm了，
    - 例如用yarn代替npm install命令，
    - 用yarn add 某第三方库名代替npm install --save 某第三方库名

7. 安装Watchman：Watchman是由Facebook提供的监视文件系统变更的工具。
    - 安装此工具可以提高开发时的性能（packager可以快速捕捉文件的变化从而实现实时刷新）。
    - 译注：此工具官方虽然是推荐安装，但在实践中，我们认为此工具是必须安装，否则可能无法正常开发。
    - 终端指令：brew install watchman

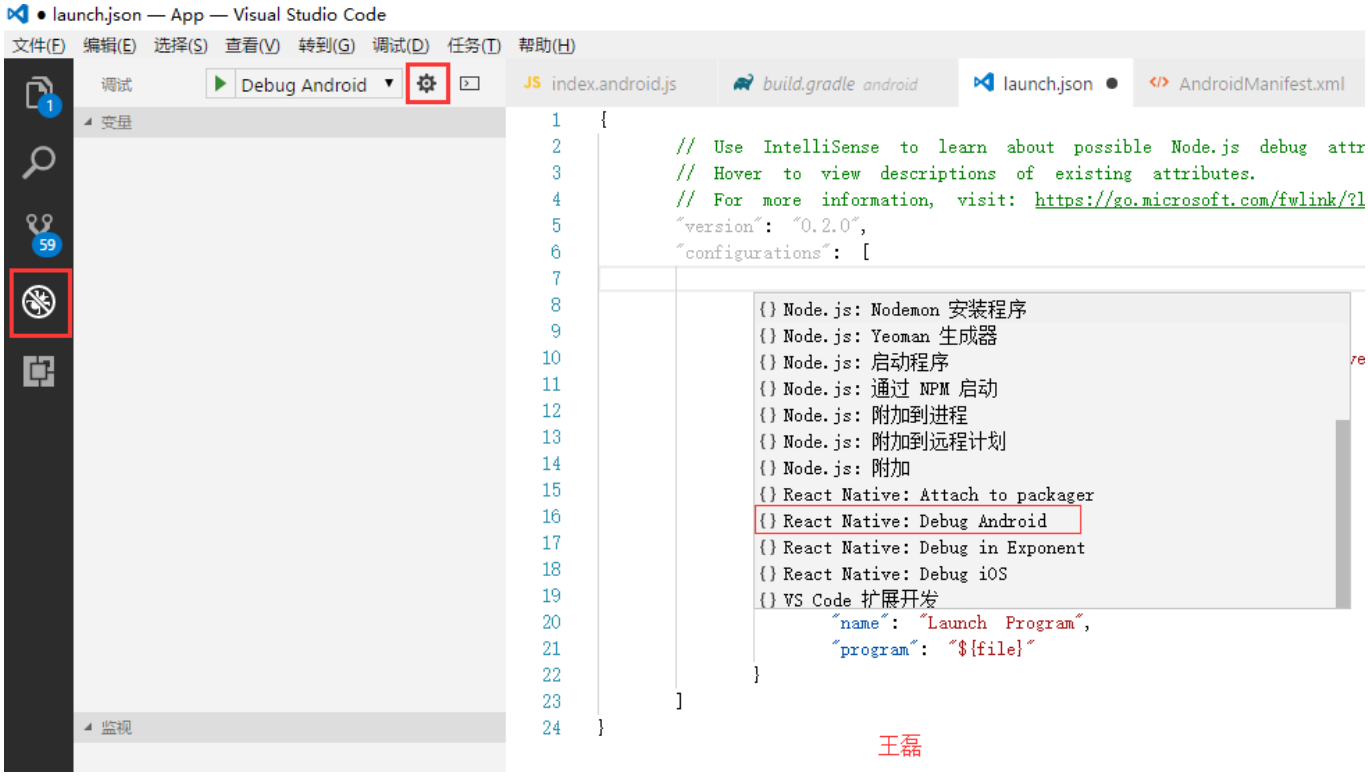
四、开发工具介绍

1. Nuclide + atom ： 由Facebook提供的基于atom的集成开发环境，可用于编写、运行和 调试React Native应用（不推荐）
2. WebStorm：收费的，体量较大，功能较多，基本无需配置
3. Sublime Text:相对轻量，但或多或少需要下载插件和配置

4. Visual Studio Code:相对轻量，但或多或少需要下载插件和配置
5. 总结：相对开发RN,推荐Visual Studio Code使用效果更好。

五、Visual Studio Code（VSCode)下载和使用

1. 官方vscode for MAC下载地址: VSCode
2. 添加RN开发插件
   - React Native Tools：微软官方出的ReactNative插件,非常好用
   - Reactjs code snippets：react的代码提示，如componentWillMount方法可以通过cwm直接获得
   - Auto Close Tag：自动闭合标签
   - Auto Rename Tag：自动重命名标签，配合上面的插件使用，基本上能赶上IntelliJ IDEA系的功能了
   - Path Intellisense：文件路径提示补全
3. 配置VSCode调试ReactNative，摆脱手动输入命令运行RN项目的苦恼



五.开始创建一个RN项目

- 开始创建项目，终端指令：
  - react-native init AwesomeProject
  - cd AwesomeProject
  - react-native run-ios/run-android