# 介绍

1. [jQuery](https://github.com/jquery/jquery): Dom 工具.
2. [lodash](https://github.com/lodash/lodash): A modern JavaScript utility library delivering modularity, performance, & extras.

# Vue.js

[Vue.js](https://github.com/vuejs/vue) 开发简单直观, 简单实用的东西通常寿命会比较长.

## 周边配套

1. 开发小程序: [Meituan-Dianping/mpvue](https://github.com/Meituan-Dianping/mpvue)
2. 开发原生APP: [weex](https://weex.apache.org/)

## Tips

### ES6

以下几个 ES6 功能应用于 Vue.js 将获得不错的收益[[1]](#footnote-31), 特别是对于无需构建工具的情况.

1. 箭头函数: 让 this 始终指向到 Vue 实例上.
2. 模板字符串: 应用于 Vue 行内模板, 可以方便换行, 无需用加号链接. 也可以应用于变量套入到字符串中.

* Vue.component({  
   template: `<div>  
   <h1></h1>  
   <p></p>  
   </div>`  
   data: {  
   time: `time: ${Date.now()}`  
   }  
  });

1. 模块(Modules): 应用于声明式的组件 Vue.component, 甚至不需要 webpack 的支持.

* import component1 from './component1.js';  
  Vue.component('component1', component1);

1. 解构赋值: 可应用于只获取需要的值, 减少不必要的赋值, 比如只获取 Vuex 中的 commit 而不需要 store.

* actions: {  
   increment ({ commit }) {  
   commit(...);  
   }  
  }

1. 扩展运算符: 数组和对象等批量导出, 而不需要用循环语句. 比如, 将路由根据功能划分为多个文件, 再用扩展展运算符在 index 中合在一起.

### 组件重新渲染

通过设置 v-if 实现, 从 Dom 中剔除再加入.

<demo-component v-if="ifShow"></demo-component>

### 绑定数据后添加属性视图未重新渲染

如果存在异步请求, 在数据上添加属性的情况, 需要先预处理好获取的数据, 然后在将其赋值到 data 中变量. 数据绑定后, 再添加属性, 不会触发界面渲染.

API.getSomething().then(res => {  
 // 1. 先添加属性  
 // handle 表示对数据的处理, 包括对象中属性的添加  
 const handledRes = handle(res);  
 // 2. 然后绑定到 data 中的变量  
 this.varInDate = handledRes;  
});

### 全局引入 SCSS 变量文件[[2]](#footnote-35)

场景: 将常用的变量存储到 vars.scss, 应用变量时需要在每个需要的地方 import.

1. npm install sass-resources-loader --save-dev
2. 更改 build/webpack.base.conf.js, 适用于 vue-cli.

{  
 test: /\.vue$/,  
 loader: 'vue-loader',  
 options: {  
 loaders: {  
 sass: ['vue-style-loader', 'css-loader', {  
 loader: 'sass-loader',  
 options: {  
 indentedSyntax: true  
 }  
 }, {  
 loader: 'sass-resources-loader',  
 options: {  
 resources: path.resolve(\_\_dirname, "./styles/vars.scss")  
 }  
 }],  
 scss: ['vue-style-loader', 'css-loader', 'sass-loader', {  
 loader: 'sass-resources-loader',  
 options: {  
 resources: path.resolve(\_\_dirname, "./styles/vars.scss")  
 }  
 }]  
 }  
 // other vue-loader options go here  
 }  
}

## Compatible

### IE vuex requires a promise polyfill in this browser

npm install --save-dev babel-polyfill

// build/webpack.base.conf.js  
entry: {  
 app: [  
 'babel-polyfill',  
 './src/main.js'  
 ]  
}

[vuex requires a promise polyfill in this browser](https://github.com/vuejs-templates/webpack/issues/474)

# React

npm install -g create-react-app  
create-react-app my-app

[React](https://reactjs.org/) 与 Vue 类似，主要专注于视图层，结合其他的库实现扩展，比如路由、数据状态维护等。

## 常用依赖

1. 路由: [React Router](https://github.com/ReactTraining/react-router)
2. 类型检查: [prop-types](https://github.com/facebook/prop-types)
3. 数据管理: [React Redux](https://github.com/reactjs/react-redux)

### UI

* [element-react](https://eleme.github.io/element-react/)
* [ant-design](https://ant.design/)

### 优化

* [reselector](https://github.com/reactjs/reselect)
* [immutable.js](https://github.com/facebook/immutable-js)
* [seamless-immutable.js](https://github.com/rtfeldman/seamless-immutable)

## 文件组织

## React 与 Redux

reducer 和 action 分文件夹放置，reducer 通过 combineReducers 合并，然后再通过 createStore 创建 store。store 通过 Provider 放置在 App 的顶层元素，并分发给所有子组件。对每个具体的功能，分别按需引入需要的 state （reducer 中定义）和 action，并用 connect 将此两项和视图层（react 实现）连接起来，实现完整的组件。由此，实现了数据、视图、事件之间的分离，通过 connect 实现连接。

## 扩展

1. [GitHub, erikras/ducks-modular-redux](https://github.com/erikras/ducks-modular-redux)

# React Native

1. 主页: <https://facebook.github.io/react-native>
2. GitHub: <https://github.com/facebook/react-native>
3. 示例项目: [amazing-react-projects](https://github.com/jiwonbest/amazing-react-projects)
4. Demo Project: [react-native](https://github.com/HereChen/template/tree/master/react-native)

## 环境配置

### 系统环境

1. 安装 [nodejs](https://nodejs.org).
2. npm install -g react-native-cli.

#### Android

1. JDK (并配置环境变量)
2. 安装 Android Studio <http://www.android-studio.org>
3. 通过 SDK Manager 下载 SDK, 并配置环境变量.

REM set var  
set ANDROID\_HOME=C:\Users\chenl\AppData\Local\Android\Sdk  
  
REM set Android home path  
setx /m ANDROID\_HOME "%ANDROID\_HOME%"  
  
REM set path  
setx /m path "%path%;%ANDROID\_HOME%\tools;%ANDROID\_HOME%\platform-tools;"

#### iOS

1. App Store 安装 XCode.
2. 其他工具安装

* brew install node  
  brew install watchman  
  npm install -g react-native-cli

### 编辑器

1. Visual Studio Code. 安装扩展 React Native Tools 用于调试.
2. Atom. 安装[nuclide](https://atom.io/packages/nuclide).

### 参考

1. <https://facebook.github.io/react-native/docs/getting-started.html>

## 基本命令

1. 新建工程: react-native init demo-project.
2. Android 运行: react-native run-android.
3. iOS 运行: react-native run-ios.

新建工程后首先 npm install 安装依赖. 示例项目 python 和 node-gyp-bin 相关错误可以尝试先执行 yarn add node-sass 或者 npm install -f node-sass (<https://github.com/sass/node-sass/issues/1980>).

## 打包

### Android 打包

#### 生成签名密钥

$ keytool -genkey -v -keystore my-release-key.keystore -alias my-key-alias -keyalg RSA -keysize 2048 -validity 10000  
Enter keystore password:  
Keystore password is too short - must be at least 6 characters  
Enter keystore password: chenlei  
Re-enter new password: chenlei  
What is your first and last name?  
 [Unknown]: HereChen  
What is the name of your organizational unit?  
 [Unknown]: HereChen  
What is the name of your organization?  
 [Unknown]: HereChen  
What is the name of your City or Locality?  
 [Unknown]: Chengdu  
What is the name of your State or Province?  
 [Unknown]: Sichuan  
What is the two-letter country code for this unit?  
 [Unknown]: 51  
Is CN=HereChen, OU=HereChen, O=HereChen, L=Chengdu, ST=Sichuan, C=51 correct?  
 [no]: yes  
  
Generating 2,048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 10,000 days  
 for: CN=HereChen, OU=HereChen, O=HereChen, L=Chengdu, ST=Sichuan, C=51  
Enter key password for <my-key-alias>  
 (RETURN if same as keystore password):  
[Storing my-release-key.keystore]

#### gradle设置

1. my-release-key.keystore 文件放到工程 android/app 文件夹下.
2. 编辑 android/app/gradle.properties, 添加如下信息.

* MYAPP\_RELEASE\_STORE\_FILE=my-release-key.keystore  
  MYAPP\_RELEASE\_KEY\_ALIAS=my-key-alias  
  MYAPP\_RELEASE\_STORE\_PASSWORD=chenlei  
  MYAPP\_RELEASE\_KEY\_PASSWORD=chenlei

1. 编辑 android/app/build.gradle, 添加如下信息.

* ...  
  android {  
   ...  
   defaultConfig { ... }  
   signingConfigs {  
   release {  
   storeFile file(MYAPP\_RELEASE\_STORE\_FILE)  
   storePassword MYAPP\_RELEASE\_STORE\_PASSWORD  
   keyAlias MYAPP\_RELEASE\_KEY\_ALIAS  
   keyPassword MYAPP\_RELEASE\_KEY\_PASSWORD  
   }  
   }  
   buildTypes {  
   release {  
   ...  
   signingConfig signingConfigs.release  
   }  
   }  
  }  
  ...

#### 生成 apk

cd android && ./gradlew assembleRelease

打包后在 android/app/build/outputs/apk/app-release.apk.

#### 安装 apk 方式

1. Genymotion 可以拖拽 apk 进行安装.
2. adb install app-release.apk 安装.

如果报签名错误, 可先卸载之前的 debug 版本.

### iOS 打包

iOS 版本编译需要在 Mac 上进行.

#### 签名

没有证书....

#### 生成 ipa

以下流程以 Xcode 9 为例.

1. 打开工程: Xcode 打开 ios 文件夹下 \*.xcodeproj 文件(工程).
2. 选择编译机型: Xcode 虚拟机选择栏中选择 Generic iOS Device.
3. 编译设置: Xcode -> Product -> Scheme -> Edit Scheme -> Run -> Info -> Build Configuration 选择 Rlease
4. JS 改为离线(打包进APP)???

TODO: 命令行打包

### 参考

1. [Generating Signed APK, Facebook Open Source](https://facebook.github.io/react-native/docs/signed-apk-android.html)
2. [打包APK, React Native中文网](https://reactnative.cn/docs/0.51/signed-apk-android.html)
3. [ReactNative之Android打包APK方法（趟坑过程）, ZPengs, 2017.02.09, 简书](https://www.jianshu.com/p/1380d4c8b596)

## 入口文件更改

从0.49开始, 只有一个入口, 不区分 ios 和 android. <https://github.com/facebook/react-native/releases/tag/v0.49.0>

React Native CLI 新建的工程, 默认入口是 index.js. 在 android\app\build.gradle 中更改入口.

project.ext.react = [  
 entryFile: "index.android.js"  
]

对应更改 android\app\src\main\java\com\\*\*\MainApplication.java.

protected String getJSMainModuleName() {  
 return "index.android";  
}

## 工具/依赖(dependencies)

### 导航

<https://facebook.github.io/react-native/docs/navigation.html>

1. [react-navigation](https://github.com/react-navigation/react-navigation) 提供了常用的导航方式(Stack, Tab, Drawer), 推荐.
2. [NavigatorIOS](https://facebook.github.io/react-native/docs/navigatorios.html) 为内建的导航, 仅在 IOS 上可用.

### UI

尚未找到两端(Web, Native)完整好用的 UI, 若后端采用 ant-design 可用 ant-design-mobile.

1. [ant-design-mobile](https://github.com/ant-design/ant-design-mobile) 每个组件是否支持 Native 有说明.
2. [react-native-elements](https://github.com/react-native-training/react-native-elements)
3. [NativeBase](https://github.com/GeekyAnts/NativeBase)

### HTTP 请求

<https://facebook.github.io/react-native/docs/network.html>

1. [fetch](https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API) 为内建接口.
2. [**axios**](https://github.com/axios/axios) 为使用校广泛的第三方请求库, 推荐使用.

## 调试

<https://facebook.github.io/react-native/docs/debugging.html>

根据提示, 可以菜单按钮选择重新加载或热加载. Android 可摇晃手机显示菜单.

### 虚拟机

1. [Genymotion](https://www.genymotion.com/download/), 需要先注册, 然后选择 for personal 使用. 如果系统开启了 Hyper-V, 需要先关闭.
2. Android Studio 内建虚拟机, 同样需要关闭 Hyper-V.
3. [Visual Studio Emulator for Android](https://www.visualstudio.com/vs/msft-android-emulator/) 需要开启 Hyper-V.

### 调试工具: Chrome

1. Remote JS Debugging 开启JS调试.
2. 浏览器端进去 http://localhost:8081/debugger-ui/, 并开启开发工具.
3. 可在 Sources 中设置断点或者代码中写入 debugger.

### 调试工具: VSCode

1. 安装扩展: React Native Tools.
2. F5 生成 launch.json 文件.
3. 进入调试菜单(Ctrl + Shift + D), 选择 Debug Android.
4. 设置断点或者写入 debugger 开始调试, 在 output 栏输出.

### HTTP 调试问题备注

应用 Fiddler 调试 HTTP, 模拟器设置了代理后, APP 无法热加载 JS bundle. 目前只有用 Chrome 或者断点的方式来调试.

## 工程结构

### 结构

android/ # Android 工程  
ios/ # IOS 工程  
src/ # 开发前端资源  
 -- assets/ # 静态资源  
 -- components/ # 组件  
 -- api/ # 接口  
 -- route/ # 导航(路由)  
 -- config/ # 常量配置  
 -- pages/ # 页面/功能  
 -- utils/ # 常用工具  
 -- reducers 相关  
 -- index.js # APP 入口  
index.js # 入口文件

### 参考

1. [Organizing a React Native Project](https://medium.com/the-react-native-log/organizing-a-react-native-project-9514dfadaa0)
2. [React native project setup — a better folder structure](https://hackernoon.com/manage-react-native-project-folder-structure-and-simplify-the-code-c98da77ef792)

## Tips

1. Android 查看当前的 Android 设备 adb devices.
2. Android 虚拟机: Ctrl + M 打开菜单 (Android Studio自带虚拟机没有菜单和摇晃手机, 可以这种方式打开菜单).
3. iPhone 虚拟机啊重新加载资源: command + R.

## 问题及解决

1. VSCode Debug 无法加载的情况, 首先重启 VSCode 再启动项目.
2. 添加antd-mobile后报错, 无法解析 react-dom, 依赖中加入react-dom并安装即可.
3. 集成react-native-navigation需要注意Android SDK版本, 版本过低可能出现编译错误(Error:Error retrieving parent for item: No resource found).

## 原理

1. React Native将代码由JSX转化为JS组件，启动过程中利用instantiateReactComponent将ReactElement转化为复合组件ReactCompositeComponent与元组件ReactNativeBaseComponent，利用 ReactReconciler对他们进行渲染[[3]](#footnote-119)。
2. UIManager.js利用C++层的Instance.cpp将UI信息传递给UIManagerModule.java，并利用UIManagerModule.java构建UI[[4]](#footnote-121)。
3. UIManagerModule.java接收到UI信息后，将UI的操作封装成对应的Action，放在队列中等待执行。各种UI的操作，例如创建、销毁、更新等便在队列里完成，UI最终 得以渲染在屏幕上[[5]](#footnote-122)。

# Weex

1. 主页: <http://weex.apache.org>
2. GitHub: <https://github.com/apache/incubator-weex/>

问题: 入口在哪儿?

**案例**

1. [网易严选](https://github.com/zwwill/yanxuan-weex-demo)
2. [点我达骑手Weex最佳实践](https://mp.weixin.qq.com/s/dowOE_QpZrtV5GH9EAgyHg)
3. [weexteam/weex-hackernews](https://github.com/weexteam/weex-hackernews)

## 搭建开发环境

npm install -g weex-toolkit

## Demo

**web**

weex create weex  
cd weex  
npm install  
npm run dev & npm run serve

**命令**

https://github.com/weexteam/weex-pack

# debug  
weex debug  
  
# add platform  
weex platform add android  
weex platform add ios  
  
# run  
weex run web  
weex run android  
weex run ios  
  
# build  
weex build web

## 问题及解决

1. https://maven.google.com/ 链接不上, 更改\platforms\android\build.gradle文件, 换成 https://dl.google.com/dl/android/maven2/。
2. adb: failed to stat app/build/outputs/apk/playground.apk: No such file or directory, 替换 platforms/android/app/build.gradle 文件中的 weex-app.apk 为 playground.apk.
3. weex debug 报错可先安装 npm install -g weex-devtool.

# React Native vs Weex

## 对比表格

|  |  |  |
| --- | --- | --- |
| 属性 | [React Native](https://github.com/facebook/react-native) | [Weex](https://github.com/apache/incubator-weex/) |
| 开源时间 | 2015/03 | 2016/06 |
| 开源企业 | Facebook | Alibaba |
| 协议 | BSD 3-clause | Apache License 2.0 |
| 主页标语 | Build native mobile apps using JavaScript and React | A framework for building Mobile cross-paltform UIs |
| 核心理念 | Learn Once, Write Anywhere | Write Once, Run Everywhere |
| 前端框架 | React | Vue.js |
| JS Engine | JavaScriptCore(iOS/Android) | JavaScriptCore(iOS) /v8(Android) |
| 三端开发 | 部分组件需要区分平台开发 | 强调三端统一 |
| 代码写法 | JSX(JavaScript + XML) | Web 写法 |
| 调试 | 虚拟机 | 可用 Chrome 查看效果 |
| 社区支持 | 社区活跃, 有多个流行产品的实践 | 目前, 开发者主要在国内, 没有太多的实践案例 |
| 优势 | 生态好, 第三方依赖多, 有可借鉴的经验 | 基于 Vue.js, 上手快, 能更好的保证三端一致 |

以下参考都是 2016 年文章.

1. [compare weex to react native](https://www.gitbook.com/book/xiaomaer/compare-weex-to-react-native/details)
2. [Weex 简介](http://slides.com/ciyinhuang/weex#/)
3. [Weex & React Native](http://zfx5130.me/blog/2016/09/15/Weex-&-React-Native/)

## 评论摘抄

After a few days of experimentation, I realized Weex and its documentation were not yet developed enough to for us to use to deliver top-quality apps. This was my experience with Weex. [Sam Landfried, 2017.10.20, Is VueJS' Weex a Suitable Alternative to React Native?](https://www.bignerdranch.com/blog/is-vuejs-weex-a-suitable-alternative-to-react-native/)

# Element

<https://github.com/ElemeFE/element>

## 组件使用

### 自定义表单校验

export default {  
 data: function () {  
 var checkVars = function (rule, value, callback) {  
 if (!value) {  
 callback(new Error('不能为空'));  
 } else {  
 callback();  
 }  
 };  
 return {  
 rules: {  
 vars: [{  
 required: true,  
 trigger: 'change',  
 validator: checkVars  
 }]  
 }  
 }  
 }  
}

## 兼容性

### IE 图标不显示

可用文字替代伪元素中的内容.

1. [ANTHONY GORE, 4 Essential ES2015 Features For Vue.js Development, 2018-01-22](https://vuejsdevelopers.com/2018/01/22/vue-js-javascript-es6/) [↑](#footnote-ref-31)
2. <https://www.reddit.com/r/vuejs/comments/7o663j/sassscss_in_vue_where_to_store_variables/?st=JC9T45PB&sh=4f87ec9d> [↑](#footnote-ref-35)
3. [ReactNative源码篇：渲染原理](https://github.com/guoxiaoxing/react-native/blob/master/doc/ReactNative%E6%BA%90%E7%A0%81%E7%AF%87/4ReactNative%E6%BA%90%E7%A0%81%E7%AF%87%EF%BC%9A%E6%B8%B2%E6%9F%93%E5%8E%9F%E7%90%86.md) [↑](#footnote-ref-119)
4. [ReactNative源码篇：渲染原理](https://github.com/guoxiaoxing/react-native/blob/master/doc/ReactNative%E6%BA%90%E7%A0%81%E7%AF%87/4ReactNative%E6%BA%90%E7%A0%81%E7%AF%87%EF%BC%9A%E6%B8%B2%E6%9F%93%E5%8E%9F%E7%90%86.md) [↑](#footnote-ref-121)
5. [ReactNative源码篇：渲染原理](https://github.com/guoxiaoxing/react-native/blob/master/doc/ReactNative%E6%BA%90%E7%A0%81%E7%AF%87/4ReactNative%E6%BA%90%E7%A0%81%E7%AF%87%EF%BC%9A%E6%B8%B2%E6%9F%93%E5%8E%9F%E7%90%86.md) [↑](#footnote-ref-122)