

# FRAMEWORKS

陈磊

2018 年 3 月 2 日

## 目录

<b>1</b>	<b>Element</b>	<b>1</b>
1.1	组件使用 . . . . .	1
1.1.1	自定义表单校验 . . . . .	1
1.2	兼容性 . . . . .	2
1.2.1	IE 图标不显示 . . . . .	2
<b>2</b>	<b>React</b>	<b>2</b>
2.1	依赖 . . . . .	2
2.1.1	UI . . . . .	2
2.1.2	优化 . . . . .	2
<b>3</b>	<b>React Native</b>	<b>2</b>
3.1	环境配置 . . . . .	3
3.1.1	系统环境 . . . . .	3
3.1.2	编辑器 . . . . .	3
3.1.3	参考 . . . . .	3
3.2	基本命令 . . . . .	4
3.3	打包 . . . . .	4
3.3.1	Android 打包 . . . . .	4
3.3.2	iOS 打包 . . . . .	6
3.3.3	参考 . . . . .	6
3.4	入口文件更改 . . . . .	6
3.5	工具/依赖 (dependencies) . . . . .	7
3.5.1	导航 . . . . .	7

1	<i>ELEMENT</i>	2
3.5.2	UI	7
3.5.3	HTTP 请求	7
3.6	调试	7
3.6.1	虚拟机	8
3.6.2	调试工具: Chrome	8
3.6.3	调试工具: VSCode	8
3.6.4	HTTP 调试问题备注	8
3.7	工程结构	8
3.7.1	结构	8
3.7.2	参考	9
3.8	Tips	9
3.9	问题及解决	9
3.10	原理	9
4	<b>React Native vs Weex</b>	<b>10</b>
4.1	对比表格	10
4.2	评论摘抄	11
5	<b>Vue.js</b>	<b>11</b>
5.1	Tips	11
5.1.1	ES6	11
5.1.2	组件重新渲染	12
5.1.3	绑定数据后添加属性视图未重新渲染	12
5.1.4	全局引入 SCSS 变量文件	12
5.2	Compatible	13
5.2.1	IE vuex requires a promise polyfill in this browser	13
6	<b>Weex</b>	<b>14</b>
6.1	搭建开发环境	14
6.2	Demo	14
6.3	问题及解决	15

# 1 Element

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

## 1.1 组件使用

### 1.1.1 自定义表单校验

```
1 export default {
2   data: function () {
3     var checkVars = function (rule, value, callback) {
4       if (!value) {
5         callback(new Error('不能为空'));
6       } else {
7         callback();
8       }
9     };
10    return {
11      rules: {
12        vars: [{
13          required: true,
14          trigger: 'change',
15          validator: checkVars
16        }]
17      }
18    }
19  }
20 }
```

## 1.2 兼容性

### 1.2.1 IE 图标不显示

可用文字替代伪元素中的内容.

# 2 React

```
1 npm install -g create-react-app
2 create-react-app my-app
```

## 2.1 依赖

1. react

2. 路由: react-router
3. 类型检查: prop-types
4. 数据管理: react-redux

### 2.1.1 UI

- element-react
- ant.design

### 2.1.2 优化

- immutable.js
- seamless-immutable.js

## 3 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
4. Demo Project: react-native

### 3.1 环境配置

#### 3.1.1 系统环境

1. 安装 nodejs.
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.

```
1 REM set var
2 set ANDROID_HOME=C:\Users\chen1\AppData\Local\Android\Sdk
3
4 REM set Android home path
5 setx /m ANDROID_HOME "%ANDROID_HOME%"
6
```

```
7 REM set path
8 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

```
1 brew install node
2 brew install watchman
3 npm install -g react-native-cli
```

### 3.1.2 编辑器

1. Visual Studio Code. 安装扩展 React Native Tools 用于调试.
2. Atom. 安装nuclide.

### 3.1.3 参考

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

## 3.2 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>).

## 3.3 打包

### 3.3.1 Android 打包

#### 生成签名密钥

```
1 $ keytool -genkey -v -keystore my-release-key.keystore -alias my-
  ↪ key-alias -keyalg RSA -keysize 2048 -validity 10000
2 Enter keystore password:
```

```

3 Keystore password is too short - must be at least 6 characters
4 Enter keystore password: chenlei
5 Re-enter new password: chenlei
6 What is your first and last name?
7   [Unknown]: HereChen
8 What is the name of your organizational unit?
9   [Unknown]: HereChen
10 What is the name of your organization?
11   [Unknown]: HereChen
12 What is the name of your City or Locality?
13   [Unknown]: Chengdu
14 What is the name of your State or Province?
15   [Unknown]: Sichuan
16 What is the two-letter country code for this unit?
17   [Unknown]: 51
18 Is CN=HereChen, OU=HereChen, O=HereChen, L=Chengdu, ST=Sichuan, C
    ↪ =51 correct?
19   [no]: yes
20
21 Generating 2,048 bit RSA key pair and self-signed certificate (
    ↪ SHA256withRSA) with a validity of 10,000 days
22     for: CN=HereChen, OU=HereChen, O=HereChen, L=Chengdu, ST=
        ↪ Sichuan, C=51
23 Enter key password for <my-key-alias>
24     (RETURN if same as keystore password):
25 [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.

```

1 MYAPP_RELEASE_STORE_FILE=my-release-key.keystore
2 MYAPP_RELEASE_KEY_ALIAS=my-key-alias
3 MYAPP_RELEASE_STORE_PASSWORD=chenlei
4 MYAPP_RELEASE_KEY_PASSWORD=chenlei

```

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

```

1 ...
2 android {
3     ...

```

```
4     defaultConfig { ... }
5     signingConfigs {
6         release {
7             storeFile file(MYAPP_RELEASE_STORE_FILE)
8             storePassword MYAPP_RELEASE_STORE_PASSWORD
9             keyAlias MYAPP_RELEASE_KEY_ALIAS
10            keyPassword MYAPP_RELEASE_KEY_PASSWORD
11        }
12    }
13    buildTypes {
14        release {
15            ...
16            signingConfig signingConfigs.release
17        }
18    }
19 }
20 ...
```

### 生成 apk

```
1 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.

### 3.3.2 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 \*.xcodproj 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 Release
  4. JS 改为离线 (打包进 APP)???
- TODO: 命令行打包

### 3.3.3 参考

1. Generating Signed APK, Facebook Open Source
2. 打包 APK, React Native 中文网
3. ReactNative 之 Android 打包 APK 方法(趟坑过程), ZPengs, 2017.02.09, 简书

## 3.4 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.

```
1 project.ext.react = [  
2   entryFile: "index.android.js"  
3 ]
```

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

```
1 protected String getJSMainModuleName() {  
2   return "index.android";  
3 }
```

## 3.5 工具/依赖 (dependencies)

### 3.5.1 导航

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

1. react-navigation 提供了常用的导航方式 (Stack, Tab, Drawer), 推荐.
2. NavigatorIOS 为内建的导航, 仅在 IOS 上可用.



### 3.5.2 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 每个组件是否支持 Native 有说明.
2. react-native-elements
3. NativeBase

### 3.5.3 HTTP 请求

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

1. fetch 为内建接口.
2. **axios** 为使用较广泛的第三方请求库, 推荐使用.

## 3.6 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.

### 3.6.1 虚拟机

1. Genymotion, 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 需要开启 Hyper-V.

### 3.6.2 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`.

### 3.6.3 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.

### 3.6.4 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.

## 3.7 工程结构

### 3.7.1 结构

```
1 android/      # Android 工程
2 ios/          # IOS 工程
3 src/          # 开发前端资源
4 -- assets/    # 静态资源
5 -- components/ # 组件
6 -- api/       # 接口
7 -- route/     # 导航(路由)
8 -- config/    # 常量配置
9 -- pages/     # 页面/功能
10 -- utils/    # 常用工具
11 -- reducers  相关
12 -- index.js  # APP 入口
13 index.js     # 入口文件
```

### 3.7.2 参考

1. Organizing a React Native Project
2. React native project setup—a better folder structure

## 3.8 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`.

### 3.9 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`).

### 3.10 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<sup>1</sup>.
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<sup>2</sup>.
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<sup>3</sup>.

## 4 React Native vs Weex

### 4.1 对比表格

属性	React Native	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

---

<sup>1</sup>ReactNative 源码篇: 渲染原理

<sup>2</sup>ReactNative 源码篇: 渲染原理

<sup>3</sup>ReactNative 源码篇: 渲染原理

属性	React Native	Weex
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
2. Weex 简介
3. Weex & React Native

## 4.2 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?

## 5 Vue.js

### 5.1 Tips

#### 5.1.1 ES6

以下几个 ES6 功能应用于 Vue.js 将获得不错的收益<sup>4</sup>, 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.

```
1 Vue.component({
2   template: `<div>
3     <h1></h1>
4     <p></p>
5   </div>`
6   data: {
7     time: `time: ${Date.now()}`
8   }
9 });
```

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

```
1 import component1 from './component1.js';
2 Vue.component('component1', component1);
```

4. 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`.

```
1 actions: {
2   increment ({ commit }) {
3     commit(...);
4   }
5 }
```

5. 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, 再用扩展运算符在 `index` 中合在一起.

---

<sup>4</sup>ANTHONY GORE, 4 Essential ES2015 Features For Vue.js Development, 2018-01-22

### 5.1.2 组件重新渲染

通过设置 `v-if` 实现, 从 Dom 中剔除再加入.

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

### 5.1.3 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.

```
1 API.getSomething().then(res => {  
2   // 1. 先添加属性  
3   // handle 表示对数据的处理, 包括对象中属性的添加  
4   const handledRes = handle(res);  
5   // 2. 然后绑定到 data 中的变量  
6   this.varInDate = handledRes;  
7 });
```

### 5.1.4 全局引入 SCSS 变量文件<sup>5</sup>

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`.

```
1 {  
2   test: /\.vue$/,  
3   loader: 'vue-loader',  
4   options: {  
5     loaders: {  
6       sass: ['vue-style-loader', 'css-loader', {  
7         loader: 'sass-loader',  
8         options: {  
9           indentedSyntax: true  
10        }  
11      }, {  
12        loader: 'sass-resources-loader',
```

<sup>5</sup>[https://www.reddit.com/r/vuejs/comments/7o663j/sassscss\\_in\\_vue\\_where\\_to\\_store\\_variables/?st=JC9T45PB&sh=4f87ec9d](https://www.reddit.com/r/vuejs/comments/7o663j/sassscss_in_vue_where_to_store_variables/?st=JC9T45PB&sh=4f87ec9d)

```

13         options: {
14             resources: path.resolve(__dirname, "./styles/vars.
                ↪ scss")
15         }
16     }],
17     scss: ['vue-style-loader', 'css-loader', 'sass-loader', {
18         loader: 'sass-resources-loader',
19         options: {
20             resources: path.resolve(__dirname, "./styles/vars.
                ↪ scss")
21         }
22     }]
23 }
24 // other vue-loader options go here
25 }
26 }

```

## 5.2 Compatible

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

```
1 npm install --save-dev babel-polyfill
```

```

1 // build/webpack.base.conf.js
2 entry: {
3   app: [
4     'babel-polyfill',
5     './src/main.js'
6   ]
7 }

```

vuex requires a promise polyfill in this browser

## 6 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
2. 点我达骑手 Weex 最佳实践
3. weexteam/weex-hackernews

## 6.1 搭建开发环境

```
1 npm install -g weex-toolkit
```

## 6.2 Demo

web

```
1 weex create weex
2 cd weex
3 npm install
4 npm run dev & npm run serve
```

命令

<https://github.com/weex-team/weex-pack>

```
1 # debug
2 weex debug
3
4 # add platform
5 weex platform add android
6 weex platform add ios
7
8 # run
9 weex run web
10 weex run android
11 weex run ios
12
13 # build
14 weex build web
```

## 6.3 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`.