

前端技术分享交流会（2020-12-09）

一、electron介绍



什么是electron

Electron 是 GitHub 发布的跨平台桌面应用开发工具，支持 Web 技术开发桌面应用，其本身是基于 C++ 开发的，GUI 核心来自于 Chrome，而 JavaScript 引擎使用 v8。

`Electron = Chromium + Node.js + Native API`

- Chromium：为Electron提供了强大的UI能力，可以不考虑兼容性的情况下，利用强大的Web生态来开发界面。
- Node.js：让Electron有了底层的操作能力，比如文件的读写，甚至是集成C++等等操作，并可以使用大量开源的 `npm` 包来完成开发需求。
- Native API：Native API让Electron有了跨平台和桌面端的原生能力，比如说它有统一的原生界面，窗口、托盘这些

什么时候使用

- 如果没有专门的桌面应用开发者，而需要前端兼顾来进行开发时，用Electron就是一个不错的选择。
- 一个应用同时开发Web端和桌面端的时候，那使用Electron来进行开发就对了。
- 开发一些效率工具，比如说我们的VSCode，比如说一些API类的工具，用Electron都是不错的选择。

那些应用是Electron开发的

- VSCode：程序员最常用的开发者工具。
- Atom：是Github开发的文本编辑器，我想大部分的前端程序员都应该使用过。

依赖环境

- node
- npm

- electron



二、electron目录介绍

<https://github.com/PanJiaChen/electron-vue-admin>

```
1  |— .electron-vue    // electron-vue配置文件
2  |— build/icons      // 打包配置文件（菜单等）
3  |— config           // 环境配置
4  |— dist             // 打包文件
5  |— src              // 资源
6  |   |— main         // electron配置文件
7  |   |— renderer     // vue代码
8  |   |— index.ejs    // 入口文件
9  |— package.json
```

2、electron简单版结构目录

<https://github.com/mischieff/electron-html>

```
1  |— build            // 打包后exe安装包
2  |— main.js          // electron配置文件
3  |   |— mainMenu.js  // 菜单文件
4  |— views            // html 文件 (css、js、images等)
5  |— favicon.ico      // 应用图标
6  |— mian.js          // 入口文件
7  |— package.json
```

三、electron中package.js详解

scripts执行命令

- `dev` : 开发环境
- `package` : window系统打包
- `build:mac` : ios系统打包
- `build:win` : windows系统打包

```

1  "scripts": {
2      "dev": "electron . ",
3      "package": "electron-packager ./ myapp --out ./OutApp --platform=win32 --
      overwrite --icon=./favicon.ico --overwrite",
4      "build:mac": "electron-builder --platform=mac --arch=x64",
5      "build:win": "electron-builder"
6  }

```

build打包

```

1  "build": { // electron-builder配置
2      "productName": "智能约谈", //项目名 这也是生成的exe文件的前缀名
3      "appId": "xxxxx", //包名
4      "copyright": "xxxx", //版权 信息
5      "compression": "store", // "store" | "normal" | "maximum" 打包压缩情况(store 相对
      较快), store 39749kb, maximum 39186kb
6      "directories": {
7          "output": "build" // 输出文件夹
8      },
9      "asar": false, // asar打包
10     "extraResources": { // 拷贝dll等静态文件到指定位置
11         "from": "./app-update.yml",
12         "to": "./b.txt"
13     },
14     "win": {
15         "icon": "xxx/icon.ico" //图标路径,
16         "extraResources": { // 拷贝dll等静态文件到指定位置 (用于某个系统配置)
17             "from": "./app-update.yml",
18             "to": "./b.txt"
19         }
20     },
21     "nsis": {
22         "oneClick": false, // 一键安装
23         "guid": "xxxx", //注册表名字, 不推荐修改
24         "perMachine": true, // 是否开启安装时权限限制 (此电脑或当前用户)
25         "allowElevation": true, // 允许请求提升。 如果为false, 则用户必须使用提升的权限重
      新启动安装程序。
26         "allowToChangeInstallationDirectory": true, // 允许修改安装目录
27         "installerIcon": "./build/icons/aaa.ico", // 安装图标
28         "uninstallerIcon": "./build/icons/bbb.ico", // 卸载图标
29         "installerHeaderIcon": "./build/icons/aaa.ico", // 安装时头部图标
30         "createDesktopShortcut": true, // 创建桌面图标
31         "createStartMenuShortcut": true, // 创建开始菜单图标
32         "shortcutName": "xxxx" // 图标名称
33     }
34 }

```

四、electron中main.js入口文件详解

配置应用生命周期

```

1  const { app, BrowserWindow } = require('electron'); // 引入控制应用生命周期的模块 app
2  const path = require('path');

```

```

3
4   let mianWin = null;
5
6   //初始化
7   app.on('ready', () => {
8     //创建主程序窗口
9     mianWin = new BrowserWindow({
10      webPreferences: {
11        nodeIntegration: false,    //内置浏览器里面不会有module和require全局变量，不能使用
nodejs
12        webSecurity: false,    // 禁用浏览器的跨域安全特性（同源策略）
13      },
14      width: 1400,    // 应用宽度
15      height: 900,    // 应该高度
16      backgroundColor: '#010f25', // 背景色
17      fullscreen: true, // 是否全屏
18      frame: false, // 是否有边框
19      movable: true, // 是否可移动
20      resizable: true, //是否可拉伸
21    });
22
23    // 打开开发工具
24    mianWin.webContents.toggleDevTools();
25
26    // 加载页面文件
27    mianWin.loadURL(path.join('file://', __dirname, './views/index.html'));
28
29    // 关闭窗口
30    mianWin.on('close', () => {
31      mianWin = null;
32    });
33
34    // 加载按钮
35    require('./main/mainMenu.js');
36  });

```

五、electron应用菜单

设置应用菜单



首页

文件 编辑

```

1   const { Menu } = require('electron')
2
3   let mianMenuTemplate = [
4     {
5       label: "文件",    // 菜单名称
6       submenu: [        // 二级菜单
7         {
8           label: "新建文件",    // 二级菜单名称
9           type: "checkbox",    // 二级菜单类型

```

```
10         checked: true,      // 二级菜单是否选中
11         accelerator: (() => { // 快捷键 (win mac)
12             return shortcutkey('ctrl+n', 'command+n')
13         })(),
14         click: () => {        // 点击事件
15             console.log(666)
16         }
17     },
18 ]
19 },
20 {
21     label: "编辑"           // 菜单名称
22 }
23 ]
24
25 // 实例化菜单
26 let mainMenu = Menu.buildFromTemplate(mianMenuTemplate)
27
28 // 挂载菜单
29 Menu.setApplicationMenu(mainMenu)
30
31 //快捷键函数
32 function shortcutkey(winkey, mackey) {
33     //mac系统和 win系统判断
34     if (process.platform == 'darwin') {
35         return mackey
36     } else {
37         return winkey
38     }
39 }
```