

环境: node 16.13.1

yarn 1.19.2

vue 3

electron 18.2.4

typescript 4.1.5

地址 <https://github.com/Lateautumn00/zhzy-tveep.git>

搭建vue3

## 一、全局安装vue/cli

```
1 npm install -g @vue/cli
2 // 或
3 yarn global add @vue/cli
```

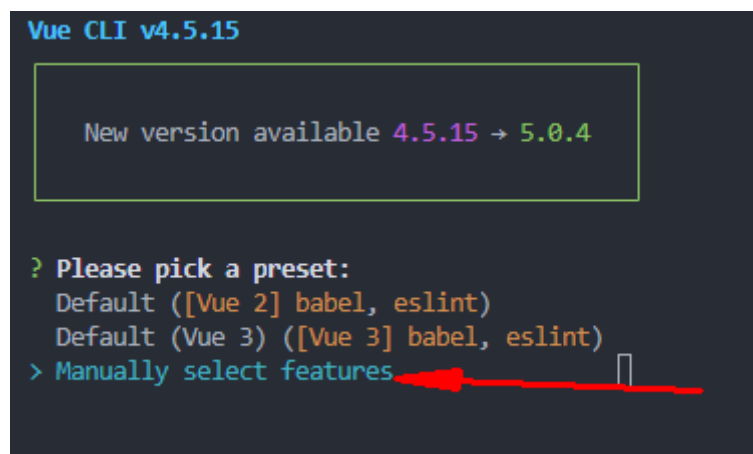
验证全局是否安装成功

```
$ vue -V
@vue/cli 4.5.15
```

## 二、初始化项目

### 1、选择自定义

```
1 vue create zhzy-tveep
```



### 2、使用上下箭头选择，按空格确认选择

```
Vue CLI v4.5.15

New version available 4.5.15 → 5.0.4

? Please pick a preset: Manually select features
? Check the features needed for your project:
  (*) Choose Vue version
  (*) Babel
  (*) TypeScript
  ( ) Progressive Web App (PWA) Support
  (*) Router
  (*) Vuex
  (*) CSS Pre-processors
  (*) Linter / Formatter
> (*) Unit Testing
  ( ) E2E Testing
```

3、这里选择3.x

```
Vue CLI v4.5.15

New version available 4.5.15 → 5.0.4

? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-p
rocessors, Linter, Unit
? Choose a version of Vue.js that you want to start the project with
  2.x
> 3.x
```

4、是否使用class风格

```
Vue CLI v4.5.15

New version available 4.5.15 → 5.0.4

? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-p
rocessors, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? (y/N) y
```

5、是否使用babel做转义

Vue CLI v4.5.15

New version available 4.5.15 → 5.0.4

```
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-processor, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)
)? (Y/n) y
```

## 6、是否使用history模式

Vue CLI v4.5.15

New version available 4.5.15 → 5.0.4

```
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-processor, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)
)? Yes
? Use history mode for router? (Requires proper server setup for index fallback in production) (Y/n)
y
```

## 7、选择css预处理器

Vue CLI v4.5.15

New version available 4.5.15 → 5.0.4

```
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-processor, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)
)? Yes
? Use history mode for router? (Requires proper server setup for index fallback in production) Yes
? Pick a CSS pre-processor (PostCSS, Autoprefixer and CSS Modules are supported by default): (Use arrow keys)
> Sass/SCSS (with dart-sass) 自动编译
  Sass/SCSS (with node-sass) 保存后才生效
  Less
  Stylus
```

## 8、选择验证模式

New version available 4.5.15 → 5.0.4

```
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-processor, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)? Yes
? Use history mode for router? (Requires proper server setup for index fallback in production) Yes
? Pick a CSS pre-processor (PostCSS, Autoprefixer and CSS Modules are supported by default): Sass/SCSS (with dart-sass)
? Pick a linter / formatter config:
  ESLint with error prevention only 只做报错提醒
  ESLint + Airbnb config 非严格模式
  ESLint + Standard config 标准 正常模式
> ESLint + Prettier 严格模式
  TSLint (deprecated) 验证typescript格式
```

## 9、选择什么时候进行eslint检查

New version available 4.5.15 → 5.0.4

```
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-processor, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)? Yes
? Use history mode for router? (Requires proper server setup for index fallback in production) Yes
? Pick a CSS pre-processor (PostCSS, Autoprefixer and CSS Modules are supported by default): Sass/SCSS (with dart-sass)
? Pick a linter / formatter config: Prettier
? Pick additional lint features: (Press <space> to select, <a> to toggle all, <i> to invert selection)
>(*) Lint on save 保存时
( ) Lint and fix on commit 提交时
```

## 10、选择断言库

New version available 4.5.15 → 5.0.4

```
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-processor, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)? Yes
? Use history mode for router? (Requires proper server setup for index fallback in production) Yes
? Pick a CSS pre-processor (PostCSS, Autoprefixer and CSS Modules are supported by default): Sass/SCSS (with dart-sass)
? Pick a linter / formatter config: Prettier
? Pick additional lint features: Lint on save
? Pick a unit testing solution: (Use arrow keys)
> Mocha + Chai
  Jest
```

## 11、eslint、babel配置存放位置

New version available 4.5.15 → 5.0.4

```
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-processor, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)? Yes
? Use history mode for router? (Requires proper server setup for index fallback in production) Yes
? Pick a CSS pre-processor (PostCSS, Autoprefixer and CSS Modules are supported by default): Sass/SCSS (with dart-sass)
? Pick a linter / formatter config: Prettier
? Pick additional lint features: Lint on save
? Pick a unit testing solution: Mocha
? Where do you prefer placing config for Babel, ESLint, etc.?
> In dedicated config files 单独存放
  In package.json 存放在package.json中
```

## 12、是否保存这些选项组合

New version available 4.5.15 → 5.0.4

```
? Please pick a preset: Manually select features
? Check the features needed for your project: Choose Vue version, Babel, TS, Router, Vuex, CSS Pre-processor, Linter, Unit
? Choose a version of Vue.js that you want to start the project with 3.x
? Use class-style component syntax? Yes
? Use Babel alongside TypeScript (required for modern mode, auto-detected polyfills, transpiling JSX)? Yes
? Use history mode for router? (Requires proper server setup for index fallback in production) Yes
? Pick a CSS pre-processor (PostCSS, Autoprefixer and CSS Modules are supported by default): Sass/SCSS (with dart-sass)
? Pick a linter / formatter config: Prettier
? Pick additional lint features: Lint on save
? Pick a unit testing solution: Mocha
? Where do you prefer placing config for Babel, ESLint, etc.? In dedicated config files
? Save this as a preset for future projects? (y/N) n
```

### 13、安装完成

```
info "fsevents@2.3.2" is an optional dependency and failed compatibility check. Excluding it from installation.
info fsevents@1.2.13: The platform "win32" is incompatible with this module.
info "fsevents@1.2.13" is an optional dependency and failed compatibility check. Excluding it from installation.
[3/4] Linking dependencies...
[4/4] Building fresh packages...
success Saved lockfile.
Done in 25.60s.
🔗 Running completion hooks...

📄 Generating README.md...

🎉 Successfully created project zhzy-tveep.
👉 Get started with the following commands:

$ cd zhzy-tveep
$ yarn serve
```

### 14、验证是否成功

```
1 cd zhzy-tveep
2 yarn serve
```

```
$ yarn serve
yarn run v1.19.2
$ vue-cli-service serve
INFO Starting development server...
98% after emitting CopyPlugin

DONE Compiled successfully in 16009ms
```

App running at:

- Local: <http://localhost:8080/>
- Network: <http://192.168.1.105:8080/>

Note that the development build is not optimized.  
To create a production build, run `yarn build`.

localhost:8080

[Home](#) | [About](#)



## Welcome to Your Vue.js + TypeScript App

For a guide and recipes on how to configure / customize this project,  
check out the [vue-cli documentation](#).

### Installed CLI Plugins

[babel](#) [router](#) [vuex](#) [eslint](#) [unit-mocha](#) [typescript](#)

### Essential Links

[Core Docs](#) [Forum](#) [Community Chat](#) [Twitter](#) [News](#)

### Ecosystem

[vue-router](#) [vuex](#) [vue-devtools](#) [vue-loader](#) [awesome-vue](#)

## 搭建electron

### 1、安装（过程有点慢，耐心等待；建议替换yarn源或翻墙）

```
1 安装electron
2 yarn add electron --dev
3 yarn add electron-devtools-installer --dev
4 yarn add vue-cli-plugin-electron-builder
```

### 2、在src目录下新建 background.ts,作为electron入口

```
1 import { app, protocol, BrowserWindow, ipcMain } from 'electron'
2 import { createProtocol } from 'vue-cli-plugin-electron-builder/lib'
3 import path from 'path'
```

```
4 //import installExtension, { VUEJS3_DEVTOOLS } from 'electron-devtools-in
staller'
5 const isDevelopment = process.env.NODE_ENV !== 'production'
6 // Scheme must be registered before the app is ready
7 protocol.registerSchemesAsPrivileged([
8   { scheme: 'app', privileges: { secure: true, standard: true } }
9 ])
10 let win: any = null
11 async function createWindow() {
12   // Create the browser window.
13   win = new BrowserWindow({
14     useContentSize: true,
15     width: 1220,
16     height: 640,
17     minWidth: 1220,
18     minHeight: 640,
19     //transparent: true, //窗口透明 设置后还原窗口win.restore()无效
20     //backgroundColor: '#000', //背景颜色
21     title: '知乎者也', //标题
22     movable: true,
23     webPreferences: {
24       // Required for Spectron testing
25       // enableRemoteModule: !!process.env.IS_TEST,
26       // Use pluginOptions.nodeIntegration, leave this alone
27       // See nklayman.github.io/vue-cli-plugin-electron-builder/guide/se
curity.html#node-integration for more info
28       nodeIntegration: process.env
29         .ELECTRON_NODE_INTEGRATION as unknown as boolean,
30       contextIsolation: !process.env.ELECTRON_NODE_INTEGRATION,
31     }
32   })
33   if (process.env.NODE_ENV === 'production') {
34     //正式
35     createProtocol('app')
36     win.loadURL('app://./index.html') // Load the index.html when not in
development
37     //win.webContents.openDevTools()
38   } else {
39     //开发
40     // Load the url of the dev server if in development mode
41     await win.loadURL(process.env.WEBPACK_DEV_SERVER_URL as string)
```



```
42     if (!process.env.IS_TEST) win.webContents.openDevTools()
43   }
44   // 当应用所有窗口关闭要做的事情
45   win.on('closed', () => {
46     win = null
47   })
48 }
49 // Quit when all windows are closed.
50 app.on('window-all-closed', () => {
51   // On macOS it is common for applications and their menu bar
52   // to stay active until the user quits explicitly with Cmd + Q
53   if (process.platform !== 'darwin') {
54     app.quit()
55   }
56 })
57 app.on('activate', () => {
58   // On macOS it's common to re-create a window in the app when the
59   // dock icon is clicked and there are no other windows open.
60   if (BrowserWindow.getAllWindows().length === 0) createWindow()
61 })
62 // This method will be called when Electron has finished
63 // initialization and is ready to create browser windows.
64 // Some APIs can only be used after this event occurs.
65 app.on('ready', async () => {
66   //外网 加载慢 去掉
67   // if (isDevelopment && !process.env.IS_TEST) {
68   //   // Install Vue Devtools
69   //   try {
70   //     await installExtension(VUEJS3_DEVTTOOLS)
71   //   } catch (e) {
72   //     console.error('Vue Devtools failed to install:', e.toString())
73   //   }
74   // }
75   if (win === null) createWindow()
76 })
77 // Exit cleanly on request from parent process in development mode.
78 if (isDevelopment) {
79   if (process.platform === 'win32') {
80     process.on('message', (data) => {
```

```

81     if (data === 'graceful-exit') {
82         app.quit()
83     }
84 })
85 } else {
86     process.on('SIGTERM', () => {
87         app.quit()
88     })
89 }
90 }
91

```

### 3、修改package.json文件，新增入口属性以及运行命令

```

1  "scripts": {
2      "test:unit": "vue-cli-service test:unit",
3      "lint": "vue-cli-service lint",
4      "web:serve": "vue-cli-service serve --open",
5      "web:build": "vue-cli-service build",
6      "electron:build": "vue-cli-service electron:build",
7      "electron:serve": "vue-cli-service electron:serve",
8      "postinstall": "electron-builder install-app-deps",
9      "postuninstall": "electron-builder install-app-deps"
10 },

```

### 4、根目录下新建vue.config.js 用与保存打包配置

```

1  module.exports = {
2      runtimeCompiler: true,
3      publicPath: './',
4      //electron 13 把"build":{}从package.json移除，在vue.config.js里写
5      pluginOptions: {
6          electronBuilder: {
7              nodeIntegration: true,
8              builderOptions: {
9                  productName: 'zhzy', //打包名称
10                 appId: 'cn.zhzy.app',
11                 copyright: 'Copyright © 2022',
12                 publish: [
13                     {
14                         provider: 'generic',
15                         url: ''
16                     }
17                 ]
18             }
19         }
20     }
21 }

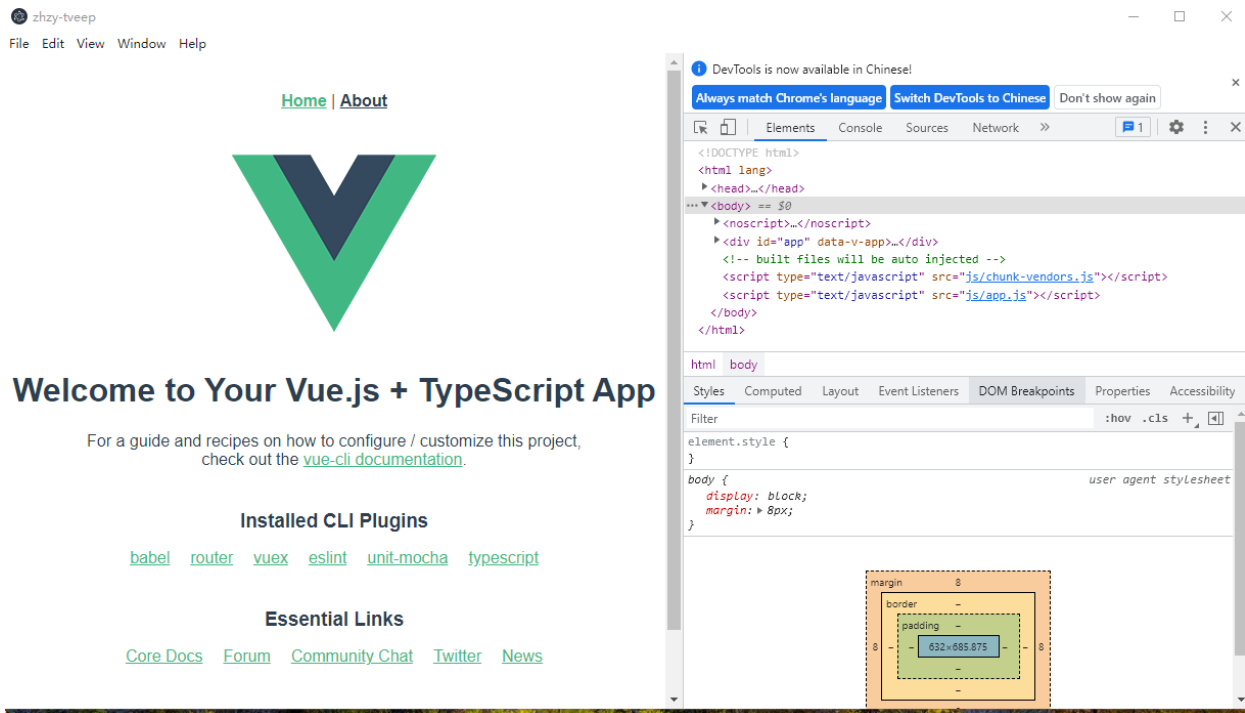
```

```
16         }
17     ],
18     nsis: {
19         allowToChangeInstallationDirectory: true,
20         createDesktopShortcut: true,
21         createStartMenuShortcut: true,
22         shortcutName: 'zhzy',
23         perMachine: true,
24         oneClick: false
25     },
26     dmg: {
27         contents: [
28             {
29                 x: 410,
30                 y: 150,
31                 type: 'link',
32                 path: '/Applications'
33             },
34             {
35                 x: 130,
36                 y: 150,
37                 type: 'file'
38             }
39         ]
40     },
41     //productName 包名称 version 包版本(package.json) ext后缀
42     mac: {
43         icon: 'public/icons/icon.icns',
44         artifactName: '${productName}_setup_${version}.${ext}'
45     },
46     win: {
47         icon: 'public/icons/icon.ico',
48         artifactName: '${productName}_setup_${version}.${ext}'
49     },
50     linux: {
51         icon: 'public/icons',
52         artifactName: '${productName}_setup_${version}.${ext}'
53     }
```

```
54         }  
55     }  
56 }  
57 }  
58
```

## 5、运行看效果

```
1 yarn run electron:serve
```



## 引入element-plus

### 1、安装

```
1 yarn add element-plus  
2 yarn add @element-plus/icons-vue
```

### 2、新建src/plugins/element.ts文件

```
1 import ElementPlus from 'element-plus'  
2 import 'element-plus/theme-chalk/index.css'  
3 export default (app: any) => {  
4   app.use(ElementPlus)  
5 }
```

### 3、修改main.ts

```
1 import { createApp } from 'vue'  
2 import App from './App.vue'  
3 import router from './router'  
4 import store from './store'
```

```

5 import installElementPlus from './plugins/element'
6 import * as ElementPlusIconsVue from '@element-plus/icons-vue'
7 const app = createApp(App)
8 for (const [key, component] of Object.entries(ElementPlusIconsVue)) {
9   app.component(key, component)
10 }
11 installElementPlus(app)
12 app.use(store).use(router).mount('#app')

```

## 引入axios

### 1、安装

```

1 yarn add axios
2 background.ts webPreferences中增加 webSecurity: false, //解决跨域

```

### 2、新建src/plugins/request.ts

```

1 import axios from 'axios'
2 const service = axios.create({
3   baseURL: '', // 此处写死时无法兼容不同api地址，地址固定一个 这里可写死
4   timeout: 5000
5   // withCredentials: true // send cookies when cross-domain requests
6 })
7 // Request interceptors
8 service.interceptors.request.use(
9   (config) => {
10     // Add X-Access-Token header to every request, you can add other cus
11     // tom headers here
12     // if (UserModule.token) {
13     //   config.headers.Authorization = UserModule.token
14     // }
15     return config
16   },
17   (error) => {
18     Promise.reject(error)
19   }
20 )
21 // Response interceptors
22 service.interceptors.response.use(
23   (response) => {
24     const res = response.data
25     if (res.statusCode !== 1000) {
26       console.error('错误')
27       return Promise.reject(new Error(res.message || 'Error'))
28     }
29     return res
30   },
31   (error) => {
32     Promise.reject(error)
33   }
34 )

```

```

27     } else {
28         return res
29     }
30 },
31 (error) => {
32     console.error('错误')
33     return Promise.reject(error)
34 }
35 )
36
37 export default service

```

### 3、新建配置文件app/config/index.ts

```

1 interface Config {
2     url: string
3 }
4 const config: Config = {
5     url: ''
6 }
7 if (process.env.NODE_ENV === 'production') {
8     config.url = 'http://127.0.0.1'
9 } else {
10     config.url = 'http://127.0.0.1'
11 }
12 export default config

```

### 4、新建app/api/demo.ts

```

1 import request from '@plugins/request'
2 import config from '@config/index'
3 export const getList = (params: any) =>
4     request({
5         url: `${config.url}/sys/getTime`,
6         method: 'get',
7         params
8     })
9 export const create = (data: any) =>
10     request({
11         url: `${config.url}/sys/getTime`,
12         method: 'post',
13         data
14     })
15

```

## 5、使用 见 FrameHome.vue

## 6、新建app/types文件夹，将shims-vue.d.ts移入并新建axios.d.ts

### 隐藏菜单栏

### 在background.ts增加

```
1 autoHideMenuBar: true, //是否隐藏菜单栏
```

### 无边框模式

#### 1、在background.ts增加

```
1 frame: false, //无边框 不使用无边框模式 需要屏蔽掉 并将router/index.ts 路由中的替换 原始边框路由
```

#### 2、background.ts增加

```
1 //登录窗口最小化
2 ipcMain.on('app-min', function () {
3   win.minimize()
4 })
5 //登录窗口最大化、还原窗口
6 ipcMain.on('app-max', function () {
7   if (win.isMaximized()) {
8     win.restore()
9   } else {
10    win.maximize()
11  }
12 })
13 ipcMain.on('app-close', function () {
14   //win.close() //在设置无边框后失效
15   win = null //主窗口设置为null 防止内存溢出
16   app.exit() //直接退出应用程序
17 })
18
```

#### 3、在src目录下新建preload.ts

```
1 // import { contextBridge, ipcRenderer } from 'electron'
2 // contextBridge.exposeInMainWorld('electron', { ipcRenderer }) //使用这种会报错，因为contextIsolation: false是非安全的，设置为true 此文件就获取不到
3 import { ipcRenderer } from 'electron'
4 ;(window as any).ipcRenderer = ipcRenderer
```

#### 4、在background.ts webPreferences新增

```
1 preload: path.join(__dirname, 'preload.js'), //预加载
```

无边框模式下自定义关闭、最大化、还原、最小化 完整代码 请拉取代码查看components  
Header.vue

版本升级

## 1、安装

```
1 yarn add electron-updater
```

## 2、src目录下新建upgrade.ts

```
1 import { ipcMain } from 'electron'
2 import { autoUpdater } from 'electron-updater'
3 let mainWindow: any = null
4 export function upgradeHandle(window: any, feedUrl: any) {
5   const msg = {
6     error: '检查更新出错 ...',
7     checking: '正在检查更 ...',
8     updateAva: '检测到新版本 ...',
9     updateNotAva: '已经是最新版本 ...',
10    downloadProgress: '正在下载新版本 ...',
11    downloaded: '下载完成, 开始更新 ...'
12  }
13  mainWindow = window
14  autoUpdater.autoDownload =
15    process.env.VUE_APP_UPGRADE === 'automatic' ? true : false //true 自
    动升级 false 手动升级
16  //设置更新包的地址
17  autoUpdater.setFeedURL(feedUrl)
18  //监听升级失败事件
19  autoUpdater.on('error', function (message: any) {
20    sendUpdateMessage({
21      cmd: 'error',
22      title: msg.error,
23      message: message
24    })
25  })
26  //监听开始检测更新事件
27  autoUpdater.on('checking-for-update', function (message: any) {
28    sendUpdateMessage({
29      cmd: 'checking-for-update',
30      title: msg.checking,
31      message: message
32    })
```



```
33  })
34  //监听发现可用更新事件
35  autoUpdater.on('update-available', function (message: any) {
36      sendUpdateMessage({
37          cmd: 'update-available',
38          title: msg.updateAva,
39          message: message
40      })
41  })
42  //监听没有可用更新事件
43  autoUpdater.on('update-not-available', function (message: any) {
44      sendUpdateMessage({
45          cmd: 'update-not-available',
46          title: msg.updateNotAva,
47          message: message
48      })
49  })
50  // 更新下载进度事件
51  autoUpdater.on('download-progress', function (message: any) {
52      sendUpdateMessage({
53          cmd: 'download-progress',
54          title: msg.downloadProgress,
55          message: message
56      })
57  })
58  //监听下载完成事件
59  autoUpdater.on(
60      'update-downloaded',
61      function (
62          event: any,
63          releaseNotes: any,
64          releaseName: any,
65          releaseDate: any,
66          updateUrl: any
67      ) {
68          sendUpdateMessage({
69              cmd: 'update-downloaded',
70              title: msg.downloaded,
71              message: {
72                  releaseNotes,
```

```

73         releaseName,
74         releaseDate,
75         updateUrl
76     }
77 })
78 //退出并安装更新包
79 autoUpdater.quitAndInstall()
80 }
81 )
82 //接收渲染进程消息，开始检查更新
83 ipcMain.on('checkForUpdate', () => {
84     //执行自动更新检查
85     autoUpdater.checkForUpdates()
86 })
87 ipcMain.on('downloadUpdate', () => {
88     // 下载
89     autoUpdater.downloadUpdate()
90 })
91 }
92 //给渲染进程发送消息
93 function sendUpdateMessage(text: any) {
94     mainWindow.webContents.send('message', text)
95 }
96

```

### 3、background.ts增加

```

1 import { upgradeHandle } from '@/upgrade'
2 if (process.env.NODE_ENV === 'production') {
3     //正式
4     createProtocol('app')
5     win.loadURL('app://./index.html') // Load the index.html when not in
    development
6     if (process.env.VUE_APP_UPLOAD !== '')
7         upgradeHandle(win, process.env.VUE_APP_UPLOAD) //检测版本更新
8     //win.webContents.openDevTools()
9 }

```

### 4、通过.env.development/.env.production设置 开发与正式环境

```

1 # 配置生产环境 系统变量 建议配置一些 常用 或者 数据较单一的配置
2 NODE_ENV=production
3 # 版本号 应与package.json version 保持统一
4 VUE_APP_VERSION=0.1.0

```

```
5 # 升级地址
6 VUE_APP_UPLOAD= http://192.168.1.51:8011/upload/
7 # 升级方式 automatic 自动升级 manual 手动升级
8 VUE_APP_UPGRADE>manual
9
```

5、将打包后的dist\_electron 目录下的 exe文件 和 latest.yml上传到 服务器

VUE\_APP\_UPLOAD

解决打包后白屏

```
1 const router = createRouter({
2   // history: createWebHistory(process.env.BASE_URL),
3   history: process.env.IS_ELECTRON
4     ? createWebHashHistory(process.env.BASE_URL)
5     : createWebHistory(process.env.BASE_URL), //解决打包后白屏
6   routes
7 })
```

更换图标

window

如下方式可解决打包后窗口左上角图标不显示问题

1、将生成的icon粘贴到当前目录

2、修改vue.config.js

```
"win": {
  "icon": "build/icons/icon.ico" (为256*256图片)
}
```

3、运行yarn run electron:serve or yarn run electron:build

mac

图标格式icns

推荐在线转换格式 <https://cloudconvert.com/png-to-icns>