Click to prove you're human

bVuevue3+TypeScript0vue3gitvitevitecn.vitejs.dev/guide/pnpmpnpmnpm i -g pnpmpnpm:pnpm create vitenpmnpm create vite@latest:2.2 package.json"scripts": { "dev": "vite --open",},3.1 eslinteslint: eslintpnpm i eslint -D3.1.2 :.eslint.cjsnpx eslint --init3.1.3 vue3pnpm install -D eslint-plugin-import eslint-plugin-vue eslint-plugin-node eslint-plugin-prettier eslint-config-prettier eslint-plugin-node @babel/eslint-parser3.1.4 .eslintignore .eslintignore distnode_modules3.1.5 package.json"scripts": { "lint": "eslint src", "fix": "eslint src --fix",}3.2 prettierprettier3.2.1 pnpm install -D eslint-plugin-prettier prettier eslint-config-prettier3.2.2 .prettierrc.json .prettierrc.json { "singleQuote": true, "semi": false, "bracketSpacing": true, "htmlWhitespaceSensitivity": "ignore", "endOfLine": "auto", "trailingComma": "all", "tabWidth": 2}3.2.3 .prettierignore .prettierignore /dist/*/html/*.local/node_modules/****/*.svg**/*.sh/public/*3.3 stylelintstylelintcsslintcsscsscsssscsspnpm add sass sass-loader stylelint postcss postcss-scss postcss-html stylelint-config-prettier stylelint-config-recess-order stylelint-config-recommended-scss stylelint-config-standard stylelint-config-standard-vue stylelint-scss stylelint-order stylelint-config-standard-scss -D3.3.1 .stylelintrc.cjs .stylelintrc.cjs : @see = { extends: ['stylelint-config-standard', // stylelint 'stylelint-config-html/vue', // vue template 'stylelint-config-standard-scss', // stylelint scss 'stylelint-config-recommended-vue/scss', // vue scss 'stylelint-config-recess-order', // stylelint css, 'stylelint-config-prettier', // stylelintprettier], overrides: [{ files: ['**/*.(scss|css|vue|html)'], customSyntax: 'postcss-scss', }, { files: ['**/*.(html|vue)'], customSyntax: 'postcss-html', },], ignoreFiles: ['**/*.js', '**/*.jsx', '**/*.tsx', '**/*.ts', '**/*.json', '**/*.md', '**/*.yaml',], /** * null => * always => */ rules: { 'value-keyword-case': null, // css v-bind 'no-descending-specificity': null, // 'function-url-quotes': 'always', // URL "always()"|"never()" 'no-empty-source': null, // 'selector-class-pattern': null, // 'property-no-unknown': null, // (true) 'block-opening-brace-space-before': 'always', // 'value-no-vendor-prefix': null, // --webkit-box 'property-no-vendor-prefix': null, // -webkit-mask 'selector-pseudo-class-no-unknown': [// true, { ignorePseudoClasses: ['global', 'v-deep', 'deep'], // element },], },}3.3.2 .stylelintignore .stylelintignore /node_modules/*/dist/*/html/*/public/*3.3.3 "scripts": { "lint:style": "stylelint src/**/*.{css,scss,vue} --cache --fix"}3.4 pnpm,,.bug,scritps/preinstall.jsif (!/pnpm/.test(process.env.npm_execpath || '')) { console.warn(`\u001b[33mThis repository must using pnpm as the package manager ` + ` for scripts to work properly.\u001b[39m`,) process.exit(1)}"scripts": { "preinstall": "node ./scripts/preinstall.js"}3.5 package.json "scripts": { "dev": "vite --open", "build": "vue-tsc && vite build", "preview": "vite preview", "lint": "eslint src", "fix": "eslint src --fix", "format": "prettier --write \"./**/*.{html,vue,ts,js,json,md}\"", "lint:eslint": "eslint src/**/*.{ts,vue} --cache --fix", "lint:style": "stylelint src/**/*.{css,scss,vue} --cache --fix", "preinstall": "node ./scripts/preinstall.js" },4.1 element-plusui element-plus : install element-plus @element-plus/icons-vuemain.tselement-plus,element-plusimport { createApp } from 'vue'import ElementPlus from 'element-plus';import 'element-plus/dist/index.css'//@ts-ignorets()import zhCn from 'element-plus/dist/locale/zh-cn.mjs'const app = createApp(App)app.use(ElementPlus, { locale: zhCn})app.mount('#app')Element Plus// tsconfig.json{ "compilerOptions": { // ... "types": ["element-plus/global"] }}element-plus.4.2 src Node.js npm install @types/node --save-devvite.config.ts // vite.config.tsimport { defineConfig } from 'vite'import vue from '@vitejs/plugin-vue'import path from 'path'// default defineConfig({ plugins: [vue()], resolve: { alias: { '@': path.resolve(__dirname, './src'), // @ src '*': path.resolve(''), }, },})TypeScript // tsconfig.json{ "compilerOptions": { "baseUrl": "./", // "paths": { //baseUrl "@/*": ["src/*"] } }}4.3 import.meta.env .env.development.env.production.env.test.env.test # VITE_ NODE_ENV = 'test'VITE_APP_TITLE = 'ling'VITE_APP_BASE_API = '/test-api'.env.development # VITE_ NODE_ENV = 'development'VITE_APP_TITLE = 'ling'VITE_APP_BASE_API = '/dev-api'.env.production NODE_ENV = 'production'VITE_APP_TITLE = 'ling'VITE_APP_BASE_API = '/prod-api'package.json"scripts": { "dev": "vite --open", "build:test": "vue-tsc && vite build --mode test", "build:pro": "vue-tsc && vite build --mode production", "preview": "vite preview"},4.4 sass/lessscss,styleLintsass sass-loader,scsslang="scss"less npm add -D less src/stylesindex.scssindex.scssreset.scss@use './reset.scss';main// import '@/styles/index.scss'src/styles/index.scss..style/variable.scssvariable.scssvite.config.ts:@import "@use "@/styles/variable.scss" as *;";;!export default defineConfig({ plugins: [//...], resolve: {//... }, css: { preprocessorOptions: { scss: { additionalData: `@use "@/styles/variable.scss" as *;`, javascriptEnabled: true, }, }, },})4.5 mock: install -D vite-plugin-mock mockjs vite.config.js import { UserConfigExport, ConfigEnv } from 'vite'import { viteMockServe } from 'vite-plugin-mock'import vue from '@vitejs/plugin-vue'export default ({ command })=> { return { plugins: [vue(), viteMockServe({ localEnabled: command === 'serve', }),], }}mock:mockmockuser.ts//function createUserList() { return [{ userId: 1, avatar: ' , username: 'admin', password: '111111', desc: '', roles: [''], buttons: ['cuser.detail'], routes: ['home'], token: 'Admin Token', }, { userId: 2, avatar: ' , username: 'system', password: '111111', desc: '', roles: [''], buttons: ['cuser.detail', 'cuser.user'], routes: ['home'], token: 'System Token', },]}export default [// { url: '/api/user/login',// method: 'post',// response: ({ body }) => { // const { username, password } = body; //, const checkUser = createUserList().find((item) => item.username === username && item.password === password,) // if (!checkUser) { return { code: 201, data: { message: '' } } } // const { token } = checkUser return { code: 200, data: { token } } }, }, // { url: '/api/user/info', method: 'get', response: (request) => { //token const token = request.headers.token; //token const checkUser = createUserList().find((item) => item.token === token) // if (!checkUser) { return { code: 201, data: { message: '' } } } // return { code: 200, data: {checkUser} } }, },]4.6 axiosaxiospnpm install axios4.6.1 axiosutils/request.ts//axios:import axios from 'axios'import { ElMessage } from 'element-plus'// import useUserStore from '@/store/modules/user'//// import useUserStore from '@/store/modules/user'//// import useUserStore from '@/store/modules/user'//axios.create({ // baseURL: import.meta.env.VITE_APP_BASE_API, ///api timeout: 5000, // withCredentials: true, // cookie // headers: { // // 'Content-Type': 'application/json', // 'token': x-auth-token',//token // 'X-Requested-With': 'XMLHttpRequest', // },})//:requestrequest.interceptors.request.use((config) => { //:token, // let userStore = useUserStore() // if (userStore.token) { // config.headers.token = userStore.token // } //config,headers, // return config})//:request.interceptors.response.use((response) => { // // return response.data }, (error) => { //:http //: let message = '' //http let status = error.response.status switch (status) { case 401: message = 'TOKEN' break case 403: message = '' break case 404: message = '' break case 500: message = '' break default: message = '' break } // ElMessage({ type: 'error', message, }) return Promise.reject(error) },)//export default request// vite.config.tsexport default defineConfig(({ command, mode }) => { // let env = loadEnv(mode, process.cwd()) return { // server: { proxy: { [env.VITE_APP_BASE_API]: { // target: env.VITE_SERVE, // changeOrigin: true, // rewrite: (path) => path.replace(/^\/api/, ''), }, }, }, }}) routerrouterpnpm install vue-router@4 src routes routes.ts // export const constantRoute = [{ path: '/home', component: () => import('@/views/home/index.vue'), name: '', meta: { title: '', hidden: false, icon: 'HomeFilled', }, }, { path: '/404', component: () => import('@/views/404/index.vue'), name: '404', meta: { title: '404', hidden: true, icon: 'DocumentDelete', }, }, { // 404 path: '/:pathMatch(.*)*', redirect: '/404', name: 'any', meta: { title: '404', hidden: true, icon: 'DocumentDelete', }, },]src routes index.ts import { createRouter, createWebHistory } from 'vue-router'import { constantRoute } from './routers'let router = createRouter({ // History history: createWebHistory(), routes: constantRoute, // scrollBehavior() { return { left: 0, top: 0, } },})export default routerpiniastorepiniapnpm i pinia@2.0.347.1 src\store\index.ts//import { createPinia } from 'pinia'//const pinia = createPinia()//export default pinia7.2 src\store\modules\user.ts//import { defineStore } from 'pinia'//const useUserStore = defineStore('User', { // state: () => { return { } }, //| actions: {}, getters: {},})//export default useUserStore7.3 main.ts// storeimport pinia from './store'// app.use(pinia)Echartspnpm install echarts --saveEchartsimport { createApp } from 'vue'import App from './App.vue'//echartsimport * as echarts from 'echarts';const app= createApp(App)//app.config.globalProperties.$echarts = echartsapp.mount('#app') Electron Electron Electron Electron Node.js fs app.getPath fs app.getPath fs app.getPath data app.getPath(userData) windows C:\Users\\AppData\Roaming\Mac /Users//Library/Application Support/ linux /home//.config/ node.js path path.join(app.getPath('userData'), 'myFile.txt') fs const { app} = require('electron');const fs = require('fs');const path = require('path');const filePath = path.join(app.getPath('userData'),'/myfile.txt'); // Example file pathtry { fs.writeFileSync(filePath, 'hello world', 'utf-8');} catch (e) { console.error('Failed to save the file!');} Dialog API fs Electron Dialog API const { dialog} = require('electron').remote;const fs = require('fs');const options = { title: 'Save file', defaultPath: 'my_filename', buttonLabel: 'Save', filters: [{ name: 'txt', extensions: ['txt'] }, { name: 'All Files', extensions: ['*'] },],};dialog.showSaveDialog(null, options).then(({ filePath}) => { if (filePath) { fs.writeFileSync(filePath, 'hello world', 'utf-8'); }}); fs Dialog Dialog Electron ipcmain // -> main.js// .... ohter code// ... SaveFile const { app, BrowserWindow, ipcMain, dialog} = require("electron");const path = require("path");const fs = require('fs');/** * @param options: { title: String, defaultPath: String, buttonLabel: String, filters: area} * @param content: String * @returns Promise */ipcMain.handle('saveFile', async (event, content, options) => { let path; try { const { filePath } = await dialog.showSaveDialog(null, options); path = filePath; } catch(e) { return Promise.reject({error: e, success: false}); } if(path) { try { fs.writeFileSync(path, content, 'utf-8'); return Promise.resolve({error: null, success: true}); } catch(e) { return Promise.reject({error: e, success: false}); } } else { return Promise.reject({error: e, success: false, canceled: true}); }});// ....// Vue // Vue ipc import {reactive, ref, onMounted} from "vue"; const textContent = ref('hello world'); const { ipcRenderer } = require('electron'); const exportImg = async () => { try { // -> // await ipcRenderer.invoke('saveFile', 'hello', { title: '', buttonLabel: '', defaultPath: 'text.txt', filters: [{ name: 'All Files', extensions: ['*'] }] }) } catch(e) { console.error('', e) } } Electron node.js fs path dialog data ipc KBjs Flashclipboard.js npm npm install clipboard --saveZIPscript CDNDOMHTMLHTMLnew ClipboardJS('.btn'); //.btn?HTML5 data-clipboard-target data-clipboard-action copy/cut / Mussum ipsum cacilds... Cut to clipboard cut data-clipboard-text Copy to clipboarddata-clipboard-text data-clipboard-target data-clipboard-text success / errorsuccessevent.action copy/cut event.text copy/cut event.triger DOMerrorevent.action copy/cut event.triger DOMvar clipboard = new ClipboardJS('.btn');clipboard.on('success', function(e) { console.info('Action:', e.action); console.info('Text:', e.text); console.info('Trigger:', e.trigger); e.clearSelection();});clipboard.on('error', function(e) { console.error('Action:', e.action); console.error('Trigger:', e.trigger);});clipboard.jsCSSGitHubPrimerHTMLAPI target, new ClipboardJS('.btn', { target: function(trigger) { return trigger.nextElementSibling || document.getElementById('name'); }});textStringnew ClipboardJS('.btn', { text: function(trigger) { return trigger.getAttribute('aria-label') || 'default text '; }});Bootstrap Modals container new ClipboardJS('.btn', { container: document.getElementById('modal')});DOMvar clipboard = new ClipboardJS('.btn');clipboard.destroy();SelectionexecCommand apiChrome 42 +Edge12 +Firefox 41 +IE 9 +Opera 29 +Safari 10 +clipboard.js!Ctrl+CClipboardJS.isSupported()clipboard.jsUI/Chrome Firefox GitHub, MDN, Gist, StackOverflow, StackExchange, npm, and even Mediu "copy to clipboard"

How to save high resolution from canva. Canva high resolution images. How to save high quality photos from canva. How to save high quality pictures from canva.