**electron**

文档地址: https://weishuai.gitbooks.io/electron-/content/api/shell.html

环境搭建 : nodejs中的npm类似net中的nuget

第一步:下载git, 安装的时候选择“系统自带CMD：使用windows系统的命令行工具”，即选择第二个“use git from the windows command prompt”。

第二步:找到git安装路径中bin的位置，如：D:\Program Files\Git\bin

        找到git安装路径中git-core的位置，如：D:\Program Files\Git\libexec\git-core;

        注："D:\Program Files\Git\"是安装路径，可能与你的安装路径不一样，要按照你自己的路径替换"D:\Program Files\Git\",在2.23的最新版本中，第二个路径应当添加为C:\Program Files\Git\mingw64\libexec\git-core

第三步:右键“计算机”->“属性”->“高级系统设置”->“环境变量”->在下方的“系统变量”中找到“path”->选中“path”并选择“编辑”->将3.1种找到的bin和git-core路径复制到其中->保存并退出

        注：“path”中，每个路径之间要以英文输入状态下的分号——“;”作为间隔

第四步:打开notepad 写入:

registry=https://registry.npm.taobao.org

sass\_binary\_site=https://npm.taobao.org/mirrors/node-sass/

phantomjs\_cdnurl=htttp://npm.taobao.org/mirrors/phantomjs

ELECTRON \_MIRROR=http://npm.taobao.org/mirrors/electron/

第五步:

打开cmd,进入准备安装electron的文件夹

1. git clone <https://github.com/szwacz/electron-boilerplate.git>
2. cd electron-boilerplate
3. npm install//非常慢 安装失败时候尝试: **npm install chromedriver --chromedriver\_cdnurl=http://cdn.npm.taobao.org/dist/chromedriver**
4. npm start 启动

文件结构: app 编译后的类似 bin/debug src源码文件 node\_modules类似引用的程序集 tasks编译指令

打包成exe文件: 根目录下执行指令=> npm run release 生成到 dist文件夹下 ,如需打包可使用 InstallShield\InnoSetup工具做安装包 默认打包的是64位程序,如需打包成32位程序需修改 package.json的 “release”:”build”=>“release”:”build –ia32 --win”

自定义绑定 html监听事件,实现修改src源码的html编译后自动copy到app中,否则修改html源码需要修改app中的html代码

**渲染进程和主进程:**html和html中的js代码运行在渲染进程中,background.js运行在主进程中,console.log在主进程打印到控制台,在渲染进程打印到console调试

渲染进程和主进程交互需要使用ipc,

渲染进程调用主进程:

主进程中

import{ipcMain}from ‘electron’

ipcMain.on(‘rendermsg’,(event,arg)=>{console.log(‘rendermsg’+arg);})

渲染进程中

const{ipcRenderer}=require(‘electron’);

btn1.onclick=function(){

ipcRenderer.send{‘rendermsg’,{name:’rupeng’,id:3}}

}

send是异步调用的.如果在渲染进程中使用sendSync进行同步调用,那么通过sendSync进行同步调用的返回值获得在主进程中通过event.returnValue设置的返回值.尽量避免使用sendSync.

主进程调用渲染进程:

首先要拿到要操作的窗口对象,或者BrowserWindow.getAllWindows()获取所有的窗口,在遍历查找要用的窗口.

主进程中

setInterVal(function(){

mainWindwo.send(‘hello’,{name,rupeng});

},3000);

渲染进程中

ipcRenderer.on(‘hello’,function(event,arg){

alert(arg);

});

下面的写法两个进程都可以调用(需要渲染进程启用nodeIntegration,默认是启用的)

1.文件系统:

var fs=require(‘fs’);

fs.readFile(‘d:/1.txt’,’utf8’,function(err,data){

alert(data);

})

2.shell用操作系统默认的编辑器打开文件

const{shell}=require(‘eletron’);

shell.openItem(‘d:/1.txt’);

用默认浏览器打开 shell.openExternal(‘url’)

资源管理器定位文件 shell.showItemInFoder(fullPath)

3.托盘图标

在app文件夹下放一个icon.ico图标,然后在主进程中

constelectron=require(‘electron’);

const Tray=electron.Tray;

var appIcon=null;

app.on(‘read’,function(){

var appIconPath=path.join(dirname,’icon.ico’);

appIcon=new Tray(appIconPath);

var contextMenu=Menu.buildFromTemplate([

{label:’item1’,type:’radio’,click:function(){ console.log(‘Item1 clicked’);}},

{label:’item2’,type:’radio’},

{label:’item3’,type:’radio’,checked:true},

{label:’item4’,type:’radio’}

]);

appIcon.seToolTip(‘This is my application’);

appIcon.setContextMenu(contextMenu);

})

因为electron会把渲染进程的浏览器默认配置nodeIntegration,这样就默认引入了module\require变量,回合jquery\requirejs等冲突,如需解决加入下面的js并放到所有框架之前

if(typeof(‘require’!=’undefined’)){

window.nodeRequire=require;

delete window.require;

delete window.exports;

delete window.module;

}

然后通过 const electron=nodeReqire(‘electron’)来进行require

electron通讯服务器端代码:

1. signalR默认是不处理首字母小写的,因此c#中的大写属性传递到js中很难看.Filters下SignalRContractResolver.cs,然后再startup中配置,参考 <HTTPS://stackoverflow.com/questions/30005575/signalr-use-camel-case>
2. mvc中配置JsonNetActionFilter设置mvc的首字母小写;
3. Filters中SignalRHubPipelineModule记录SignalR未处理异常;
4. Helpers下:UserCenterApi是用来访问WebApi中的UserCenter服务器,读取配置文件,引用了WebApi中开发的UserCenter .Net Sdk 改了一些bug 把UserCenter配置到iis中执行.
5. Helpers下:登录\获取组信息\获取组成员信息通过UserCenter来进行,但是UserCenter只是提供了 接口,所以需要Im服务器提供登录状态的保存.项目中使用jwt来吧登录信息放到cookie中.
6. Helpers下RedisHelper把Redis的创建封装了.
7. HomeController:Login把登录用户信息加密成jwtTOken保存到cookie中;GroupMain是显示班级聊天室主页,当前用户如果不属于这一组则不能打开;LoadGroups是提供软件主页ajax加载用户所属组用的;LoadGroupUsers是用来在班级聊天室主页显示班级成员用的,

vuejs操作DOM对象 需要放到

this.$nextTick(()=>{

var chatbox=document.getElementById(‘chatbox’);

chatbox.scrollTop=chatbox.scrollHeight;

})