

AppEmit 说明书

程序名称	AppEmit.exe	
程序版本	v1.3.28	
说明书版本	v1.3.28	
网址	http://www.appemit.com	
Email	appemit@appemit.com	
修改时间	2022 年 1 月 8 日	
修订	v1.3.28	

版权所有 **AppEmit**，保留一切权利。

本文档的任何部分，包括文字、图片、图形等均归属于 **AppEmit**。未经书面许可，任何单位或个人不得以任何方式摘录、复制、翻译、修改本手册的全部或部分。除非另有约定，**AppEmit** 不对本手册提供任何明示或默示的声明或保证。

责任声明

在法律允许的最大范围内，本文档是“按照现状”提供，可能存在瑕疵或错误。**AppEmit** 不对本文档提供任何形式的明示或默示保证，包括但不限于适销性、质量满意度、适合特定目的、不侵犯第三方权利等保证；亦不对使用或是分发本文档导致的任何特殊、附带、偶然或间接的损害进行赔偿，包括但不限于商业利润损失、系统故障、数据或文档丢失产生的损失。

目录

1 概述.....	1
1.1 特点.....	1
1.2 具体功能.....	1
1.3 使用条件.....	2
1.3.1 系统.....	2
1.3.2 用法.....	2
1.4 技术实现.....	3
1.4.1 技术依赖.....	3
1.4.2 步骤.....	3
1.4.3 demo.....	4
1.5 联系.....	4
2 解决方案.....	4
2.1 浏览器.....	4
2.2 Flash.....	4
2.3 多媒体.....	4
2.4 直播.....	4
2.5 Office.....	5
2.6 调用 DLL 或者 ocx.....	5
3 本地实施.....	5
3.1 文件结构及逻辑.....	5
3.2 加载文件.....	5
3.3 主要变量.....	6
3.3.1 常用变量.....	6
3.3.2 Websocket 初始参数 wsInit, 关键是 initSet.....	7
3.3.3 App 对象 AE.AppObj.....	8
3.4 主要方法.....	8
3.4.1 初始化 websocket InitApp(wsUrl,callbackFunc).....	8
3.4.2 打开或者执行命令 OpenApp(Req,interval,sync).....	8
3.4.3 执行代码 runCmd.....	8
3.4.4 函数调用 runCall.....	9
3.4.5 关闭 App CloseApp(AppId,force).....	9
3.4.6 动作处理 AppPosEvent(emit,AppId,pos,t_eventType).....	9
3.4.7 函数回调 callbackFunc.....	9
3.5 通用变量和函数.....	10
3.5.1 JS 中常用函数.....	10
3.5.1.1 获得 AppId 的屏幕坐标 GetAbsoluteLocationEx(AppId).....	10
3.5.1.2 去抖函数 debounce(func,wait,immediate).....	10
3.5.1.3 节流函数 throttle(func, wait, type).....	10
3.5.1.4 其它.....	10
3.5.2 主要操作函数.....	10
3.5.2.1 获取文件 sha1.....	11
3.5.2.2 随机生成 GUID.....	11
3.5.2.3 设置获取 header, whttp_header 独立不共享.....	11
3.5.2.4 设置获取 header http_header.....	11

3.5.2.5 AES 加密并 BASE64 编码.....	11
3.5.2.6 BASE64 解码再 AES 解密.....	12
3.5.2.7 注册控件.....	12
3.5.2.8 判断是否管理员身份运行 appemit.....	13
3.5.3 获得 appemit 参数.....	13
3.5.4 设置 appemit 参数.....	14
3.5.5 AppJs 函数，runCmd 命令执行.....	14
3.6 App 为弹窗（Layui_layer 调用）	14
4 参数示例.....	16
4.1 连接.....	16
4.1.1 ws 协议.....	17
4.1.2 wss 协议.....	17
4.2 初始化数据.....	17
4.2.1 OTP.....	18
4.2.2 在 server 端生成 OTP 规则.....	18
4.2.3 验证 OTP 算法是否一致.....	19
4.2.4 不同语言 AES 算法参考.....	19
4.2.4.1 C#实现的 AES 加密、解密（CBC/PKCS5Padding）	19
4.2.4.2 PHP 实现 AES 加密、解密（CBC/PKCS5Padding）	20
4.2.4.3 PHP7.1 实现 AES 加密、解密（CBC/PKCS5Padding）	21
4.2.4.4 Java 实现的 AES 加密、解密代码（CBC/PKCS5Padding）	21
4.3 通用命令参数.....	22
4.4 硬件.....	24
4.4.1 硬件信息.....	25
4.4.2 硬件温度.....	25
4.4.3 串口 Pcomm.....	25
4.4.3.1 解决办法.....	25
4.4.3.2 串口控件 Pcomm.....	25
4.4.3.3 串口发送接收数据.....	26
4.4.3.4 互动调用串口其它命令.....	27
4.4.4 串口 mscomm32 控件.....	29
4.4.5 USB Hid 通信接口信息.....	30
4.4.6 USB 热拔事件.....	31
4.5 用户间通话.....	32
4.6 打开 APP.....	33
4.6.1 打开文件.....	34
4.6.2 flash.....	35
4.6.2.1 NPAPI 打开 flash 网页 webkit 内核.....	35
4.6.2.2 NPAPI 打开网络 flash 文件.....	37
4.6.2.3 NPAPI 打开 flash rtmp 等媒体文件.....	38
4.6.2.4 ActiveX 打开 flash 文件.....	40
4.6.3 打开网页.....	43
4.6.3.1 提交登陆命令.....	43
4.6.3.2 可以定节点执行任务.....	44
4.6.3.3 在 index.html 调用 embed.html 里的 js 执行命令.....	44

4.6.3.4 被嵌入的网页 embed.html 回调数据到 index.html.....	44
4.6.3.5 支持网页 Object 对象.....	45
4.6.3.6 IE 内核打开网页.....	45
4.6.3.7 IE 内核打开含有 flash_ocx 网页.....	51
4.6.3.8 IE 内核启动前设置 cookie 或者清除 clearCookie.....	51
4.6.3.9 IE 内核启动后设置 cookie 或者清除 clearCookie.....	51
4.6.3.10 webkit 内核打开网页.....	51
4.6.3.11 webkit 内核打开 flash_NP 网页.....	53
4.6.3.12 blink 内核打开网页.....	53
4.6.3.13 blink 内核打开 flash_NP 网页.....	55
4.6.4 打开媒体文件.....	55
4.6.4.1 视频.....	55
4.6.4.2 libvlc 插件播放(支持 RTSP RTMP...).	55
4.6.4.3 Libvlc 播放监控事件.....	55
4.6.5 直播 RTSP、RTMP.....	56
4.6.5.1 RTMP.....	56
4.6.5.2 RTSP.....	56
4.6.6 打开 PDF 文件.....	59
4.6.7 Office.....	60
4.6.7.1 msOffice.....	60
4.6.7.2 Word 其它参数.....	62
4.6.7.3 Excel 其它参数.....	63
4.6.7.4 PowerPoint 其它参数.....	64
4.6.7.5 疑难.....	65
4.6.8 WPS.....	65
4.6.9 Office JavaScript.....	66
4.6.9.1 嵌入网页 JS 控制.....	66
4.6.9.2 浏览器网页执行代码命令控制.....	68
4.6.10 Office DsoFramer.....	68
4.6.10.1 更多示例.....	70
4.6.10.2 更多示例.....	71
4.6.10.3 其它参数.....	71
4.6.10.4 疑难.....	72
4.6.11 网页直接注册并调用 DLL OCX.....	72
4.6.11.1 使用 IE 内核打开 ActiveX.....	73
4.6.11.2 使用 webkit 内核打开 NPAPI.....	73
4.6.11.3 自动注册 ocx、dll 再使用 IE 内核或者 webkit 打开.....	73
4.6.12 公共 DLL\组件调用开发.....	74
4.6.12.1 调用方法.....	74
4.6.12.2 可视化 Flash ocx 调用.....	89
4.6.12.3 报表控件 reportX 调用.....	91
4.6.12.4 第三方 dll 调用.....	93
4.6.12.5 系统 com 组件.....	97
4.6.13 自行 Dll 开发调用.....	98
4.6.13.1 开发.....	98
4.6.13.2 嵌入网页 JS 控制.....	103

4.6.13.3 浏览器网页执行代码命令控制.....	105
4.7 关闭 APP.....	106
4.7.1 关闭 cid 下对应的 APP.....	106
4.7.2 关闭 cid 下所有 Obj 类型 APP.....	106
4.7.3 关闭重新连接前的 runtime 运行服务 APP.....	106
4.8 控制 APP 窗口.....	106
4.8.1 显示.....	106
4.8.2 隐藏.....	107
4.8.3 移动.....	107
4.9 appemit 操作.....	107
4.9.1 文件 config.ini.....	107
4.9.2 错误信息.....	108
4.9.3 重启.....	108
4.9.4 更新.....	108
4.9.5 关于.....	109
4.9.6 版本信息.....	109
5 发布.....	109
5.1 注册设置.....	109
5.2 局域网设置.....	109
5.3 设置账号和授权.....	110
5.4 修改程序和服务名称.....	110
5.5 更新本地数据.....	110
5.6 Js 文件.....	111
5.7 插件部署.....	111
5.8 发布.....	112
5.9 安装.....	113
5.9.1 附加安装.....	113
5.9.1.1 局域网.....	114
5.9.1.2 网页启动程序.....	114
5.9.1.3 多用户.....	114
5.9.1.4 支持 wss.....	114
6 问题.....	114

1 概述

AppEmit 是应用程序（尤其是浏览器）与本地程序间互相通信的易扩展的轻量级中间件。主要采用了 HTML5 国际标准的 Web Socket 进行通话，默认为异步，JSON 格式传递参数。

➤ 主要功能：

- 1) 在几乎所有最新版本的浏览器播放含有 flash 的网页或 Flash 文件，包括 swf 交互动画、flv 影视等
- 2) 在浏览器打开、操作本地文件，比如阅读 PDF；创建、阅读、编辑 Office 文件，且支持 JavaScript 代码操作
- 3) 支持各种最新浏览器多种方式打开 RTMP、RTSP 等媒体文件
- 4) 在浏览器中调用第三方 DLL、OCX 等 ActiveX 组件以及系统 winApi 函数，tcc、python、lua 等
- 5) 开发本地硬件 DLL 的插件，实现在网页中操作控制本地的读卡器、打印机、扫描仪、高拍仪、U 盾等各种硬件设备
- 6) 各个应用程序之间通信，比如聊天
- 7) 在 Chrome 里嵌入 IE 内核网页，保护源码，可以不修改原有的 ActiveX 读取 html，同时支持开源内核 webkit 和 blink

➤ 解决问题

- 1) 国际市场份额 68%以上的 chrome 浏览器（数据来源 Netmarketshare；国内 25%以上）在 2020 年 12 月后不再支持 flash，而微软的 edge 也不支持 ActiveX。
- 2) 客户习惯使用浏览器来处理各种业务。
- 3) 游戏商、银行、医院、电力、硬件等企业客户在各种浏览器中开发调用 dll、ocx、ActiveX、com 组件、flash 等文件的场景需要。

➤ 相关链接

程序名称 AppEmit.exe

网址 <http://www.appemit.com>

测试地址 <http://www.appemit.com/demo/index.html>

Github <https://github.com/appemit/appemit>

Email 联系 appemit(at)appemit.com

内容分发下载地址：

<https://appemit.coding.net/api/share/download/a00ac968-de8c-4744-a5c0-b17ea9efca9c>

github 下载地址

<https://cdn.jsdelivr.net/gh/appemit/appemit@master/dist/AppEmit.zip>

1.1 特点

- 精小简洁 不包含 plugin 只有 5M。
- 功能强悍 能满足程序间尤其是浏览器的通信，解决 NPAPI、ActiveX、DLL、RTSP 等问题
- 安全可靠 主要流程加密授权，大量商用不宕机，不记录保存监控任何业务数据和逻辑
- 扩展力强 通过公用或者三方的插件扩展各个商业领域

1.2 具体功能

支持同步、异步（默认）处理消息或者消息组

支持一个页面打开多个 APP

支持本地文件和网络文件访问

支持局域网和万维网的配置

支持获取硬件信息，包括系统、CPU、主板、显卡、内存、硬盘、网络等

支持 USB 监控

支持 pcomm 串口异步多线程调用

支持用户间通信、发送接收消息

支持使用 IE、webkit、blink 内核打开，互动网页

支持打开、互动 flash,包括 ActiveX flash 和 NPflash

支持多媒体播放、转码

支持各种最新浏览器多种方式打开 RTMP、RTSP 等媒体文件

支持打开、编辑、代码互动 microsoft office、金山 office 的 world、excel、ppt，支持本地文档和网络文档处理

支持打开、编辑、代码互动 AutoCad, CATIA

支持打开 3D PDF

支持调用第三方 dll、com 等链接库

支持开发 dll 等

1.3 使用条件

需要公司的所有客户的电脑本地都安装 appemit.exe，是客户端程序。

1.3.1 系统

当前版本只能在 Windows 系统使用，支持 XP 以上，32 位和 64 位都支持。

软件/插件	功能/方式	依赖条件	说明
AppEmit		Windows 系统 XP 以上	32 位, 可以运行在 x64、x86 上
flash	调用本地 flash ActiveX 控件, 打开 swf 文件	本地安装 flash ActiveX 控件	
	打开网络中含有 swf 的网页 打开网络中的文件 swf	自动安装插件	或者下载完全插件版本可局域网使用。后续同。
IE	打开网络中含有 swf 的网页 采用不同内核 IE\webkit 打开网络或者本地网页	-	
PDF	阅读网络或者本地 PDF 文件	-	
Office	打开、编辑 Office 文档	本地安装 Office	
媒体文件	打开网络中的文件 视频格式: mp4、flv、m3u8、rtmp 视频编码: H.264 音频编码: AAC、MP3 音频格式: MP3 等文件	-	

1.3.2 用法

在 windows 系统中，下载免安装程序 AppEmit（不含插件小于 5M），运行 AppEmit.exe 即可。设置了开机自启动，应避免被杀毒软件关闭。

名称	大小
bin	
demo	
plugins	
AppEmit.exe	1,873 KB
config.ini	1 KB
uninstall.exe	574 KB

同时只能开启一个 AppEmit.exe 进程。

然后使用浏览器打开 demo 里面的 html 文件，即可测试。

引入 appemit.min.js，初始数据生成 AE 对象，使用关键步骤 **AE.InitApp()** 和 **AE.OpenApp()** 即可实现基本功能。

1.4 技术实现

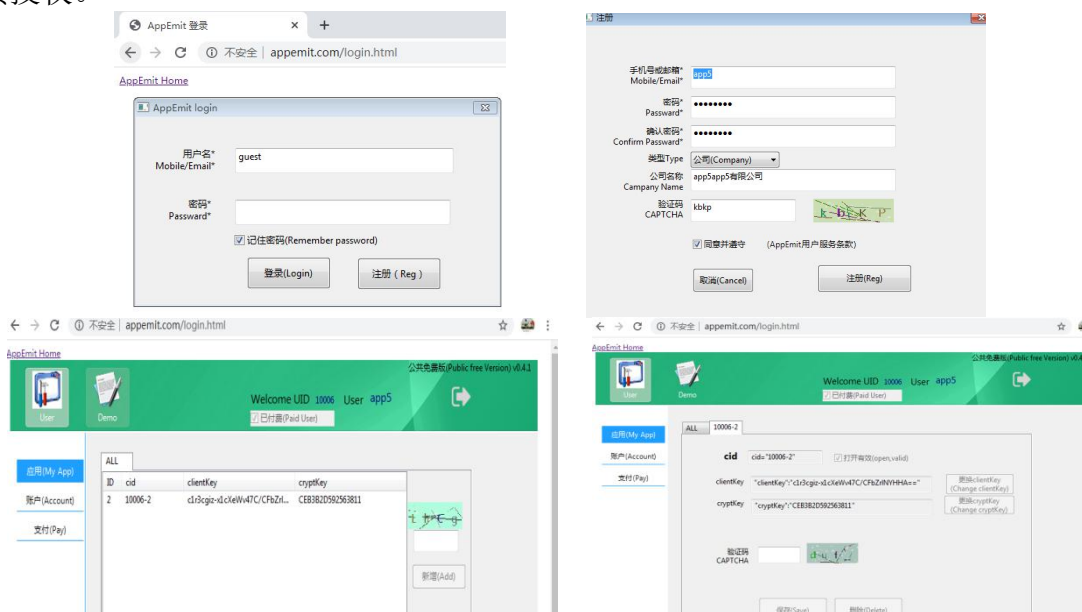
1.4.1 技术依赖

采用开源或公共控件，安全可靠。

- Web Socket 采用开源控件 HP Socket（<https://github.com/ldcsaa/HP-Socket>），支持 ssl。
- Dll 文件开放了 C 接口，可以在此基础二次开发控件,主要调用
 - HPSocket4C_U.dll
 - HPSocket4C-SSL_U.dll
- IE 内核调用 flash Active 控件
- 非 IE 内核调用 flash NPAPI 控件
- webkit 内核采用开源软件 wke 控件
- blink 内核采用开源软件 miniblink 控件，miniblink 使用新版 chromium 内核

1.4.2 步骤

1. 使用临时账户 cid=00000-1 测试(略过此步骤)。网页注册后获得设置 cid, clientKey, 获得连接授权。



一个账户 cid 目前最多 15 个，设置 cid 后不要任意修改，更新数据有 1-2 小时延迟。

2. 设置 clientKey 授权, (clientKey 为私有，发布后需要保密混淆加密 js)初始化数据以及授权等

```

var wsInit = {
...

initSet: {  "emit": "init",
            "clientKey": "temp-0000000000",  //
            "clientInfo": clientInfo,
            "wsUrl": wsUrl,
            // "flag": 0,
            //  "sid": "123456",           // 用户 session 或者用户名 ID，唯一可以准确通话
            "gid": "[1,2]", // 用户群 ID，一个用户可以加入多个群
            // "utf_escape": false,       // 默认 false，反馈的 data 编码转义

            }
};

var AE = new AppEmit(wsInit);

```

3. 连接 Appemit 服务

```
AE.InitApp("ws://localhost:80/appemit?cid=00000-1&sid=1&flag=1")
```

4. 发送命令

```
AE.OpenApp({"emit": "hardWare", "Obj": "pc", "par0": {"dev": ["os", "base"]}})
```

1.4.3 demo

在 demo 下主要是 html 的举例，

- 包括获取 pc 信息，实现通话的 index.html
- 以及播放 flash 的 AppEmbed.html
- 播放 RTSP 的 rtsp.html
- 操作 office 的 office.html

1.5 联系

邮件: appemit@appemit.com

2 解决方案

2.1 浏览器

支持 IE webkit blink 内核，可以直接操作 Object 对象。

2.2 Flash

支持 swf flv

- 1 支持网络本地文件读取播放
- 2 支持 activeX 或者 NPAPI 网页 flash

2.3 多媒体

支持 mp4 swf flv rtmp rtsp

2.4 直播

RTSP:

可采用 vlc 的 libvlc 库插件播放
 vlc 的 NPAPIwebkit 网页播放，支持多开
 rtsp 转化为 webRTC 浏览器直接播放
 rtsp 转化为 ogg 浏览器直接播放。

2.5 Office

支持打开 编辑 微软\金山 word excel ppt

2.6 调用 DLL 或者 ocx

- 1) 标准的 OCX、dll:支持标准的 com 组件 ActiceX,ocx 调用，支持 NPAPI 的 dll。对于在 IE 和低版本的 chrome 调用成功的 object 对象可以直接使用 web 嵌入打开,使用 IE 内核打开 activeX 或者 webkit 打开 NPAPI 网页。这种方法最简单，使用已有的 js 文档就好，但是需要注册对应的组件。
- 2) 非标准的 OCX、dll,通过 AppEmit 读取接口，调用 dll，可以在 js 里面直接操作，可以不注册也能调用。

3 本地实施

3.1 文件结构及逻辑

下载解压后，免安装，其文件结构为

```
├bin
├demo
├plugins
├AppEmit.exe
├config.ini
└uninstall.exe
```

可以打开 appemit.com 的主页或者 demo\htmlDemo 里面的 demo 来测试。

appemit.html(或者 index.html...)为主页，调用 appemit.min.js 来与 appemit 实现本地程序或者浏览器通话。

appemit.html 和 appemit.min.js 没有嵌套。

- 1) Html 启动时候执行 AE.InitApp，连接 websocket，此步骤执行一次
- 2) 然后可以反复发送命令 AE.OpenApp(ReqPar)，执行对应的业务
- 3) 在关闭浏览器时，默认执行关闭 websocket，此步骤执行一次。

3.2 加载文件

在开发者的主页，加载 [appemit.min.js](#) 就可以了，执行 [AE.InitApp](#) 一次。

通常情况下,appemit.min.js 和开发者的主页 html 是在一个文件 body 里面，没有嵌套。因为 appemit.min.js 需要自动识别 APP 对应节点 AppEmbed1 的位置。

如果需要嵌套，可以参考 layui_layer.html 和 iframe.html，动态的加载 appemit.html。

如果使用了绝对位置，设置一个 AE.absolutePos。

如果 AppEmbed1 的位置始终固定，AE.fixedPos_NoPageOffset 设置为 true。

调用命令 `AE.OpenApp` 可以多次执行。
 关闭网页的时候，默认执行 `AE.CloseApp`。

使用 `vue` 应该是在模板里面调用 `AE.OpenApp`，`AE.InitApp` 是在主页里面执行一次，这样更加合理。

3.3 主要变量

具体见 `appemit.js`

3.3.1 常用变量

名称	值	可调整	说明
<code>ws_port</code>	<code>[80, 8617, 8618, 8619, 8811];</code>	1	必需， <code>[80, 8617, 8618, 8619, 8811]</code> ；ws 端口。可修改，与 <code>config.ini</code> 文件设置一致，websocket 可用的未注册的端口。依次尝试，若全部失败则在最后一个端口(不小于 2000，并剔除 未指定端口排除表的端口)基础，再尝试 <code>port_try_Maxcnt</code> 次+1 打开
<code>wss_port</code>	<code>[443, 7131, 7132, 7133, 7366];</code>	1	必需，wss 端口。可修改。同上。
<code>port_try_Maxcnt</code>	<code>10;</code>	1	非必需，为指定端口最大尝试次数，可以调整为 0 或者 65536 不限制尝试。
<code>excludePorts</code>	<code>[2049];</code>	1	必需，为指定端口排除表，与 <code>config.ini</code> 文件设置一致，已排除小于 2000 的端口，可追加 chrome 不可用的 websocket 端口。只能追加。https:
<code>port_try_Maxcnt</code>	<code>10;</code>	1	非必需，为指定端口最大尝试次数，可以调整为 0 或者 65536 不限制尝试。
<code>excludePorts</code>	<code>[2049];</code>	1	必需，为指定端口排除表，与 <code>config.ini</code> 文件设置一致，已排除小于 2000 的端口，可追加 chrome 不可用的 websocket 端口。只能追加。https:
<code>initSet</code>	<code>{ "emit": "init", "clientKey": "...", "OTP": "...", "wsUrl": null, "flag": 0, "sid": "1", "gid": "[1, 2]", "utf_escape": false, , }</code>	1	设置 websocket 的初始参数
<code>JsOutData</code>	<code>{};</code>	1	非必需 所有的反馈结果
<code>elementId</code>	<code>AppEmbed;</code>	1	必需。绑定的 div 或者 textarea 等节点名称前缀，可以修改，要求和 html 一致
<code>rid</code>	<code>0;</code> 字符串，数值	1	必需。业务请求命令序列号。 默认从 0 一直自动加 1。本次发送请求的标识 <code>request id</code> ，可以修改。 为了回调做判断可以设置 <code>rid</code> 做标志(后面命令的 <code>rid</code> 延续前面的状态++1)。

			如果不方便打乱 rid, 在 runCmd 命令可以增加一个 AppMark, 唯一标识每一条命令的每一次执行。
noRid	0	1	非业务请求命令序列号, 从 0 递减。
fixedPos_NoPageOffset	false;	1	对应的 APP 节点固定不变, 设置 true 将不计算 PageOffset
absolutePos	{"left":0,"top":0};	1	嵌套 object 或者使用绝对位置#div 来 load 加载 html, 需要实时修改为嵌套的位置来更新 elementId 的位置
hideScale	{"one":0.7,"more":0.3}	1	滚动后需要隐藏 App 时的被卷入 App 长度通用比例阈值。默认单窗口 0.7, 多窗口 0.3。如果需要具体到宽度和高度, 则设置 hideScaleXY, 以 hideScaleXY 优先为准。
ws	null;	0	必需, websocket
SvrOpenMark	false;	0	必需。打开 appemit 服务端口标识
AppObj	{};	0	必需。保存所有 APP 的固定属性, 如果没有必要, 考虑性能只打开一个 APP
SERVICE	appemit;	0	服务标识
EVENTTYPE	{"scroll":1,"move":2,"visible":4,"resize":8,"zoom":16};	0	1 滚动 2 移动 4 隐显 8 大小 16 缩放比例
CurEventType	null;	0	记录动作事件
eventType_Follow	{};	0	所有事件组合, 自动计算出 {CurEventType: AppFollow} { 1: [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31] 2: [2, 3, 6, 7, 10, 11, 14, 15, 18, 19, 22, 23, 26, 27, 30, 31] 4: [4, 5, 6, 7, 12, 13, 14, 15, 20, 21, 22, 23, 28, 29, 30, 31] 8: [8, 9, 10, 11, 12, 13, 14, 15, 24, 25, 26, 27, 28, 29, 30, 31] 16: [16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31] }
clientInfo	null;	0	浏览器信息
curRatio	1;	0	浏览器默认比率

3.3.2 Websocket 初始参数 wsInit, 关键是 initSet

```

initSet={                                     //必需。 websocket 初始化参数
"emit":"init",                               //不可更改。
"clientKey": "temp-0000000000",              //【需要更改】，最好混淆加密本 js 文件(比如
https://obfuscator.io), 不要公开 clientKey 等私密数据
//"OTP":"24DqU57GuWXi_q0lepZ25E60t5Ue_Cewc7P24ica3r0", // 开发者自行获取自己公司
的服务器动态生成 OTP。设置了 OTP, clientKey 可以不
用。生成算法具体见说明书。

//wsUrl:null,
// "flag":0, //调试标识。默认为 0 生产环境 1 调试 2 生产环境压缩 3 调试压缩
// "sid":"1", //必需。用户 session 或者用户名 ID, 用户之间通话则必需唯一, 才能互相
准确通话 //sid gid 发布后统一设置在此。在调试时也可以在
wsUrl 的参数里面设置
"gid": "[1,2]", //非必需。用户群(频道、子公司、组)ID, 一个用户可以加入多个群(频道、子

```

公司、组)

// "utf_escape": false, // 默认 false, 反馈的 data 编码转义

};

3.3.3 App 对象 AE.AppObj

```
{ AppId: //AppId 1 2 3 4 5 {}表示所有
  {Obj: "flash", //flash office comm web file...
    AppShow:true, //显示 APP 载体
    AppFollow:31, // AppShow:true 是才生效
    pos:[l,t,w,h], // 保存最新绝对 pos, 调用参数 1 追加
                    // 最新绝对位置, -1 删除绝对位置,
    elementId:"AppEmbed1", //多个则需要分别设置 AppEmbed2、AppEmbed3、
    AppStatus:0, //1 已打开 0 已关闭
    AppRuntime:0, //1 runtime 服务, F5 刷新不自动关闭
                  APP, 需要手动关闭。默认为 0, F5 刷新, 更新所有的 APP
    hideScaleXY:[0.7,0.7] //滚动后需要隐藏 App 时的被卷入 App 长度具体
                           width,height 比例阈值。hideScaleXY 优先于 AE.hideScale, 对
                           于多开 App 窗口, 可以设置此参数
  },
  AppId2:{...}
}
```

3.4 主要方法

AppEmit 使用流程的主要函数及事件

3.4.1 初始化 websocket InitApp(wsUrl, callbackFunc)

连接浏览器后执行, 连接 websocket, 让 appemit 开始工作。

callbackFunc 可选, 设置相等的条件 equ 满足后, 执行对应的函数

AE.callbackFunc= [{"equ": {"service":"appemit", "rid":

0,"clientAuth":1},"func":function(){AE.OpenApp(Req)}}]

AE.InitApp("ws://localhost:80/appemit?cid=00000-1&sid=1&flag=1",AE.callbackFunc);

3.4.2 打开或者执行命令 OpenApp(Req, interval, sync)

最常用的函数, 执行具体的发送命令。

Req 具体的执行命令参数, 可为数组

Interval 前后命令执行的时间间隔 默认 5 毫秒

Sync 命令组是否同步, 默认 false 异步

下面表示打开 "AppId":1 后完成, 再打开 "AppId":2

```
AE.OpenApp([{"emit":"open","Obj":"web","AppType":1,"AppId":1,"pos":1,"par":{"URL":"http://
www.baidu.com"}},{"emit":"open","Obj":"web","AppType":1,"AppId":2,"pos":1,"par":{"URL":"http://
www.baidu.com"}},null,true)
```

3.4.3 执行代码 runCmd

执行代码可以调用 runCmd 命令, 具体见 AppJs 函数和 dll 章节调用 runCmd 执行代码案例。

```
AE.OpenApp({"emit":"runCmd","Obj":"...", "AppId":1})
```

单独 runCmd 可以在调用的命令中设置 strvar，只是执行本命令 codeStr 的 \${} 替换。
例如替换 \${dir_Cur} 为 appemit.exe 的相对路径。

3.4.4 函数调用 runCall

和 runCmd 执行代码效果一样，执行函数调用可以调用 runCall 命令，反馈为一个变量。
调用通用函数

```
AE.OpenApp({"emit":"runCall","Obj":"app","par":{"objName":"AppJsFun","callFun":{"msgbox":["调用 msgbox","title"]}}})
```

下面的案例是调用 synCard2 的 add2 方法。

```
AE.OpenApp({"emit":"runCall","Obj":"card","AppId":1,"AppShow":false,"src":[],"par":{"objName":"synCard2","plugins": null},"par0":{"callFun":{"add2":3} }})
```

多参数输入，可以使用数组形式。

3.4.5 关闭 App CloseApp(AppId, force)

单独调用关闭

```
AE.OpenApp({"emit":"close","Obj":"media","AppId":1})
```

默认在关闭浏览器执行了关闭所有非 runtime 运行时类型的 App
{} 关闭当前所有非 runtime 运行时 App， 指定 AppId 则关闭对应的。

force 强行关闭任何打开的 App

1) 关闭所有

```
AE.CloseApp({}); 或者 AE.OpenApp({"emit":"closeAll","Obj":"web"})
```

(2) 关闭某一个 APP，使用

```
AE.CloseApp("Obj":"media","AppId":1);
```

或者使用

```
AE.OpenApp({"emit":"close","Obj":"media","AppId":1})
```

3.4.6 动作处理 AppPosEvent(emit, AppId, pos, t_eventType)

如果调用了其它库，在浏览器 resize 之后延迟还改变了 App 的位置大小，则需要使用该函数修正，具体见 demo/htmlDemo/layui_layer.html 中的 MutationObserver。

emit 为 move、resize、hide、show。

AppId 具体的 AppId, {} 表示全集

pos 具体的 ltrwh, 1 表示自动识别

t_eventType emit 动作的 id

```
AE.AppPosEvent("move",{},{},1,AE.EVENTTYPE["move"])
```

3.4.7 函数回调 callbackFunc

在函数开关 callbackFun_cancel 不为 true 的时候执行回调。

可以设置 rid 做标志(后面命令的 rid 延续前面的状态++1)，如果不方便打乱 rid, 在 runCmd 命令可以增加一个 AppMark，唯一标识每一条命令的每一次执行。

还可以通过 "neq" 进一步判断。

如果还不能区分，最好是重写 appemit.min.js 里面的 AE.InitApp 中的 websocket 过程，处理 Jdata


```

AE.callbackFunc= [
{
"equ": { "Obj": "dll", "AppStatus": 1, "rep": 0}
,"neq": { "rid": 1}
,inPar: {..., 可无}
,"func": function(Jdata, inPar){}
}
,{...}
]

```

通过判断 websocket 的 Jdata 判断 equ 相等的条件执行对应的函数。
neq 不相等，非必需。

callbackFun_cancel=true 关闭回调。

3.5 通用变量和函数

3.5.1 JS 中常用函数

3.5.1.1 获得 AppId 的屏幕坐标 GetAbsoluteLocationEx(AppId)

AE.GetAbsoluteLocationEx(AppId)

3.5.1.2 去抖函数 debounce(func, wait, immediate)

function AE.debounce(func, wait, immediate)

//immediate true - 立即执行立即执行版的意思是触发事件后函数会立即执行，然后 n 秒内不触发事件才能继续执行函数的效果

// false - 延迟执行 非立即执行版的意思是触发事件后函数不会立即执行，而是在 n 秒后执行，如果在 n 秒内又触发了事件，则会重新计算函数执行时间

3.5.1.3 节流函数 throttle(func, wait, type)

function AE.throttle(func, wait, type) // type 1 表时间戳版，2 表定时器版 函数触发是在时间段内开始的时候，而定时器版的函数触发是在时间段内结束的时候

3.5.1.4 其它

AE.GetRatio() 获取浏览器比率

AE.isMaxBrowser() 是否最大化

AE.isFullScreen() 是否全屏

3.5.2 主要操作函数

发送命令调用常用函数，获取返回值。

名称	设置	含义	说明
emit	func	必需。获得参数请求。	
Obj	file_sha1	必需。获得参数请求	

	guid en_Aesbase64 de_Aesbase64 regsvr32 postForm whhttp_header http_header		
par			
name	数组		

3.5.2.1 获取文件 sha1

获取文件 sha1，发送命令

```
{"emit":"func","Obj":"file_sha1","par":{"file":"/plugins/common/comm/Pcomm.dll"}}
```

反馈参考如下

```
.   Obj: "file_sha1"
.   data: "38CBACF18C42C624E0C76D2120D7B436E89EF2FC"
.   par: {file: "/plugins/common/comm/Pcomm.dll"}
.   rep: 0
```

3.5.2.2 随机生成 GUID

```
{"emit":"func","Obj":"guid","par":{}}
```

3.5.2.3 设置获取 header，whhttp_header 独立不共享

method null 则默认为 GET

```
{"emit":"func","Obj":"whhttp_header","par":{"url":"http://www.qq.com/"},"par0":{"header":{"a":1},"method":"GET"}}
```

3.5.2.4 设置获取 header http_header

method null 则默认为 GET

```
{"emit":"func","Obj":"http_header","par":{"url":"http://www.qq.com/"},"par0":{"header":{"a":1},"method":"POST"}}
```

3.5.2.5 AES 加密并 BASE64 编码

不同编程语言中 AES 加解密结果要保持一致要注意以下一些要点：

- 1、工作模式 CBC，填充模式 PKCS5，不同语言要保持一致。
- 2、在下面的示例中，加密向量统一设为与密钥相同。
- 3、不同编程语言使用的文本编码要一致，同一个字符串，使用 UTF8 或 GBK 编码在内存中存储的实际数据可能是不一样的。
- 4、如果加密后返回的密文用了 BASE64 编码，那么在解密时同样也先做对应的逆向解码。

```
{"emit":"func","Obj":"en_Aesbase64","par":{"secKey,data,initVector,flags,Mode,Padding}}
```

默认 initVector=null 即密钥相同； flags=0 ； 工作模式 CBC Mode= 1 ； 填充模式 PKCS5 Padding=1

```
{"emit":"func","Obj":"en_Aesbase64","par":{"secKey":"1234567812345678","data":"Test String"}}
```

反馈

```
.   Obj: "en_Aesbase64"
.   data: "xL1eEwu9WCDRiscUbPPPSA=="
.   par: {data: "Test String", secKey: "1234567812345678"}
```

```
. rep: 0
```

3.5.2.6 BASE64 解码再 AES 解密

```
{"emit":"func","Obj":"de_Aesbase64","par":{"secKey,data,initVector,flags,Mode,Padding}}
```

默认 initVector=null 即密钥相同； flags=0 ； 工作模式 CBC Mode= 1 ； 填充模式 PKCS5
Padding=1

```
{"emit":"func","Obj":"de_Aesbase64","par":{"secKey":"1234567812345678","data":"xL1eEwu9  
WCDRiscUbPPPSA=="}}
```

反馈结果

```
. Obj: "de_Aesbase64"
. data: "Test String"
. par: {data: "xL1eEwu9WCDRiscUbPPPSA==", secKey: "1234567812345678"}
. rep: 0
```

3.5.2.7 注册控件

共有的库可以直接调用。asAdmin=1 win7 会有黑框提示。如果命令方式注册失败，可以增加 bat 参数。或者直接在打开网页参数中设置 embed_object。见 4.6.3.6 bink 内核打开 flash_NP 网页

```
{"emit":"open","Obj":"web","AppType":3,"AppId":1,"pos":1,"par":{"URL":"http://sxiao.4399.com/439  
9swf/upload_swf/ftp14/yzg/20140328/bombit7/zx_game7.htm","plugins":["flash_NP"],"par0":{"righ  
tMenu":-1}}
```

增加"plugins":"flash_NP"，即可引入。

支持网页 Object 对象

```
ReqPar3= {"emit":"func","Obj":"regsvr32","par":regPar};
```

```
var regPar={"pid":"ReportX","reg":true, "asAdmin":0,"dllFile":null  
,"CLASSES_ROOT":"HKEY_CLASSES_ROOT\\ReportProj1.ReportX\\Clsid","clsId":"A5DA6E97  
-1D4C-4708-B705-84A45716B4DD","AuthKey":null};
```

私有的库，需要在 appemit 登陆窗口中我的应用中记录，设置授权和文件校验。

//私有的库 pid 唯一，如果 dll/ocx 文件改变了，需要更新 appemit 登陆窗口中我的应用里面的 sha1 值

```
var regPar={"pid": "292A53CD-DA8F-46A5-808D-B286F2759C37" //pid 名称，如果  
pid 名称是 AE 共有的，则调用共有的插件  
,"reg":true // "reg" null 不执行注册 1 强制操作 true 未注册则注  
册 false 卸载注册（重启 AppEmit 后，就不能加载 dll 了）  
,"asAdmin":0 //"asAdmin" //1 需要管理员权限注册 0 不需要 win7 管  
理员身份会有提示确认窗口  
,"dllFile":"/plugins/private/292A53CD-DA8F-46A5-808D-B286F2759C37  
/ReportX.ocx" //"dllFile" //,"d:/report/ReportX.ocx" //"dllFile  
,"CLASSES_ROOT":"HKEY_CLASSES_ROOT\\ReportProj1.ReportX\\Clsid"  
// "CLASSES_ROOT" 可选，reg= true 检查是否已经注册，一定要是\\分隔  
,"clsId":"A5DA6E97-1D4C-4708-B705-84A45716B4DD"  
// "clsId" 可选，reg= true 检查是否已经注册
```

```
,"AuthKey":"A1-ZneY-2qGoXRfc7h6GZZxBB2gceORBwyhoxsA6GK5agLtIAwLhh6BnK61W8fORVN
```

```
v" // private 目录则需要校验,若网站设置了 sha1 则还会校验 sha1
};
```

```
ReqPar3= {"emit":"func","Obj":"regsvr32","par":regPar};
```

3.5.2.8 判断是否管理员身份运行 appemit

反馈为字符串。

```
{"emit":"func","Obj":"isRunAsAdmin","par":0}
```

```
. Obj: "isRunAsAdmin"
. data: "false"
```

3.5.3 获得 appemit 参数

获得常用参数和状态数据，比如端口是否被占用。

```
{"emit":"getPar","Obj":"app","par":{"name":["clientAuth","utcNum","other..."]}}
```

或者全部 {"emit":"getPar","Obj":"app","par":{"name":["All"]}}

名称	设置	含义	说明
emit	getPar	必需。获得参数请求。	
Obj	app		
par			
name	字符串或者数组	["port", "protocol", "clientAuth", "clientInfo", "sid", "cid", "gid", "rid", "flag", "utf_escape", "utcNum", "intranetIP", "dir_Cur", "dir_AppData_Roaming", "dir_AppData_Local", "dir_Temp", ...]	"All" 反馈所有 "port" "protocol" "clientAuth" "clientInfo" "sid" "cid" "gid" "rid" "flag" "utf_escape" "utcNum" UTC 时间 "intranetIP" 内网 IP "dir_Cur" "dir_AppData_Roaming" "dir_AppData_Local" "dir_Temp"
par0			
portEnable	端口数字或者数组	[123,80,3423,3,"342"]	反馈端口是否没有被占用 portEnable: 3: true 80: false 123: true 342: true 3423: true

3.5.4 设置 appemit 参数

```
{"emit":"setPar","Obj":"app","par":{"sid":2}}
```

名称	设置	含义	说明
emit	getPar	必需。获得参数请求。	
Obj	app		
par	对象	{ "sid":2} "clientInfo","sid",;"gid","flag","utf_escape",	只能再次修改可修改的参数.即初始化的参数,"cid"\ clientAuth 等不能设置修改

3.5.5 AppJs 函数，runCmd 命令执行

定义了一些常用的函数接口，供 js 调用，使用 runCmd 命令执行。

```
{"emit":"runCmd","Obj":"app","codeStr":"AppJs.msgbox(AppJs.utcNum(),'utc');return {data=123}}"
```

"Obj":"app"表示通用。如果是在执行某个 Obj，则设置为对应的 Obj 名称，以及"AppId"即可。

对象 AppJs 的方法

JSONstringify(tab,prettyPrint,unicodeEscaped,objreferences)

JSONparse(...)

JSONtryParse(...)

msgbox(str,title,style,hwndOwner)

dlgOpen(...)

dlgSave(...)

utcNum()

restClient

inet_whohttp

inet_http

同时可以定节点执行命令

```
"AppStep":{"init":"AppJs.msgbox('测试')", "destroy":null, "closed":null,"loaded":"return {111}"}
```

3.6 App 为弹窗 (Layui_layer 调用)

具体见 demo/htmlDemo/layui_layer.html。

使用 layer 最好嵌入类型为 1，这样可以方便调用变量。

```
var layer_fixed=true;    //true = position: absolute;    false= position: relative

layui.use(['layer', 'form'], function() {
    var layer = layui.layer
    ,form = layui.form;

    layer.open({
        type: 1
```

```

, title: '当你选择该窗体时，即会在最顶端'
, offset: 'auto' // ['80px', '400px']
, area: ['800px', '500px']
, shade: 0
, fixed: layer_fixed
//, move: false

//, content: ['iframe_in.html', 'no'] //这里 content 是一个 URL, 如果你不想让 iframe
出现滚动条, 你还可以 content: ['', 'no']
, content: contentStr
, success: function(layero, index){

    if (!layer_fixed) {
        AE.absolutePos=$(layero.selector).offset(); //使用了相对位置但是固定
        不动, 需要修正
    }else{
        AE.fixedPos_NoPageOffset=true; // 固定则 不计算 PageOffset
        AE.AppFollow=30; // 固定则不响应滚动事件, 全部响应默认 31
        // fixed=true offset: 'auto' F12 出现 console 的情况以及 windowresize 更新后
        layer 延迟 auto
        var MutationObserver = window.MutationObserver ||
window.webkitMutationObserver || window.MozMutationObserver;
        observer = new MutationObserver(fixedOffsetAuto_callback);
        targetNode = document.getElementById( (layero.selector).substring(1,
layero.selector.length));
        observer.observe(targetNode, { attributes: true, childList: false, subtree:
false });
    }

    AE.InitApp(".", AE.callbackFunc); //启动 websocket, 成功后 AE.callbackFunc
执行 input_App
}
, cancel: function(index, layero){
    AE.CloseApp({}); //关闭当前所有 App , 指定 AppId 则关闭对应的
    if (observer) observer.disconnect(); //关闭监控
}
, moveEnd: function(layero){

    if (!layer_fixed) AE.absolutePos=$(layero.selector).offset(); //使用了
    相对位置但是固定不动, 需要修正
    AE.AppPosEvent("move", {}, 1, AE.EVENTTYPE["move"]); //更新所有 App 位置, 执
    行 move 事件
}
, resizing: function(layero){
    if (!layer_fixed) AE.absolutePos=$(layero.selector).offset(); //使用了
    相对位置但是固定不动, 需要修正

```

```

        AE.AppPosEvent("resize", {}, 1, AE.EVENTTYPE["resize"]);    //更新所有 App 位置, 执行 resize 事件
    }

});
});

```

4 参数示例

4.1 连接

ws://localhost:80/appemit?cid=00000-1&sid=1&flag=1

名称	设置	含义	说明
协议	ws	ws SSL 为 wss	wss://local.appemit.com:443/appemit?cid=00000-1&sid=1&flag=1
网址	Localhost 127.0.0.1		
port	[80,8617,8618,8619,9780]	数值或者数组，默认 80。	依次尝试打开端口。若所有端口被占用 尝试关闭最后一个 port 的进程，打开最后一个端口 都可以在 config.in 修改
	[443,5124,5125,5126,43100]	ssl 默认 443。	同上。
path	appemit	必需	
para	cid	必需。00000-1 为免费账号。	
	sid	字符串，通常情况可选。唯一 session 或者用户名 ID	测试后最好在 js 中实现隐藏。如果需要调用私有 APP，则必须有，否则无法互相通话。
	flag	可选。默认 0，非调试。 1 调试	
	strvar	默认为 0	参数字符串含有变量标识。推荐默认 0，1 则会所有命令执行替换 \${} \${} 的固定参数。 单独 runCmd 可以在调用的命令中设置 strvar，只是执行本命令 codeStr 的 \${} \${} 替换。

4.1.1 ws 协议

4.1.2 wss 协议

需要申请 ssl 认证，为指定域名提供证书，指向 127.0.0.1，并设置对应 config.ini 文件。
具体步骤：

- 1、DNS 配置 local.name.com A 方式指向 127.0.0.1
- 2、下载 local.name.com 的 SSL 的证书，拷贝 Apache 下的 crt 、key 文件到 bin/cert 中
- 3、设置 config.ini ，[userSet]字段下配置 certFile，keyFile，要求名称一致。
certFile=/bin/cert/local_AE_2021.crt
keyFile=/bin/cert/local_AE_2021.key
(还可以 局域网配置文件 aaset.stlan 设置 SSL 的证书,但是是以 config.ini 为最终修改状态。)
- 4、重启程序。

注意证书的时间年限应该很长，要不后续更新证书文件麻烦。

后续使用 wss://local.name.com:443/appemit?cid=00000-1&sid=1&flag=1 来访问。

测试 wss://local.appemit.com:443/appemit?cid=00000-1&sid=1&flag=1

生产环境不要使用 wss://local.appemit.com，该 SSL 是免费的 TrustAsia TLS RSA CA SSL 认证，一年到期，要断几天，不推荐使用。

如果断网使用 wss 需要设置 config.ini

ETC_HOSTS={"local.name.com":"127.0.0.1"}

如果程序没有管理员权限配置失败，手动需要修改 hosts 文件,127.0.0.1 local.name.com 。

4.2 初始化数据

设置授权信息等

{"emit":"init",...}

```
initSet = {
  "emit":"init",
  "clientKey": "temp-00000000000", //
  "OTP":"服务器动态生成",
  "clientInfo":clientInfo,
  "wsUrl": wsUrl,
  // "flag":0,
  // "sid":"123456", // 用户 session 或者用户名 ID,唯一可以准确通
  "gid": "[1,2]", //用户群 ID, 一个用户可以加入多个群
  // "utf_escape":false, //默认 false, 反馈的 data 编码转义
```

话

};

名称	设置	含义	说明
emit	init	必需。初始化请求。	
clientKey	temp-00000 00000	必需，客户端，与 cid 对应。	保密，js 应该混淆加密。 授权成功反馈 1
OTP		One-time password	使用服务器 des 算法，对 clientKey 的安全保护 授权成功反馈 2
clientInfo	对象	必需。使用浏览器。 默认	
flag	标识	默认为整体情况 0 生产环境 1 调试 2 压缩生产环境 3 调试压缩	同时可以在具体的一条 命令中增加 flag 标识，只 处理本消息情况。比如在 整体 flag=0 时,发送带有 flag=1 的命令则此条命令 的提示信息也会反馈，但 是压缩解压只能整体设 置。
wsUrl	wsUrl	必需。默认	可以在 config.in 修改
sid	字母数字下 划线短线	必需。用户或者 session，唯一才 可以正常通话。	生产环境，同一设置于 此。
gid	数组	非必需。 群	一个 sid 可有不同 gid
utf_escape	false	默认 false	反馈的 data 编码转义

4.2.1 OTP

企业用户以上可以使用更安全的加密方法 OTP(One-time password)。OTP 是 AppEmit 对 clientKey 的安全保护措施，安全可靠，简单易用。将会有效杜绝他人通过在页面上获取 clientKey 的方式，进行非法操作。

当在代码中 AppEmit 初始化时候，传入一个新参数 OTP，以 OTP 为验证，clientKey 此时可以不传递。

OTP 是一个在服务器端按 OTP 的算法规则来生成的加密字符串，AppEmit 将会对 OTP 进行合法性验证，因为每个 OTP 只可以使用一次，即使他人从页面代码中获得了 clientKey，也无法推送或接收消息。

登录主页注册，可以获得 clientKey 和 cryptKey。在服务器端 cryptKey 用于在生成 OTP 时，作为加密的密钥。

4.2.2 在 server 端生成 OTP 规则

1. 声明一个字符串，内容为"000"+网络 UTC 时间（不是本地时间）毫秒数。
也可以使用下面命令
{"emit": "getPar", "Obj": "app", "par": "utcNum"}
反馈 data: {utcNum: 1599706089}来确保一致。
注意：appemit 的使用 ntp 获得网络时间，若果局域网或者网络不通，则使用客户端时间转化为 UTC 时间。
2. 将 cryptKey 作为密钥，用 AES(CBC)算法对字符串进行加密，工作模式 CBC，填充模式 PKCS5，加密向量统一设为与密钥相同，编码为 UTF-8
3. 使用 Base64 对加密结果进行编码，考虑 URL 传递，在此基础上做下一步变换
+ => -
/ => _
去除尾部的 _ =

其结果就是 OTP。

4.2.3 验证 OTP 算法是否一致

cryptKey: 83AA400AF464C76D

UTC 毫秒数: 0001599193983184

OTP: cRwkend-1a00fBhEkOq2oDxADnN89uwqswaFR-Z6fHw

4.2.4 不同语言 AES 算法参考

不同编程语言中 AES 加解密结果要保持一致要注意以下一些要点：

- 1、工作模式 CBC，填充模式 PKCS5，不同语言要保持一致。
- 2、在下面的示例中，加密向量统一设为与密钥相同。
- 3、不同编程语言使用的文本编码要一致，同一个字符串，使用 UTF8 或 GBK 编码在内存中存储的实际数据可能是不一样的。默认编码为 UTF-8。
- 4、如果加密后返回的密文用了 BASE64 或 16 进制编码，那么在解密时同样也先做对应的逆向解码。

4.2.4.1 C#实现的 AES 加密、解密（CBC/PKCS5Padding）

```
using System;
using System.Text;
using System.Security.Cryptography;

namespace TestApp
{
    class Aes
    {
        static void Main(string[] args)
        {
            String str = "Test String";
            String encryptData = Aes.Encrypt(str, "83AA400AF464C76D",
"83AA400AF464C76D");
            Console.WriteLine(encryptData);

            String dstr = Aes.Decrypt("xL1eEwu9WCDRiscUbPPPSA==",
"83AA400AF464C76D", "83AA400AF464C76D");
            Console.WriteLine(dstr);
            Console.ReadKey();
        }

        public static string Encrypt(string toEncrypt, string key, string iv)
        {
            byte[] keyArray = UTF8Encoding.UTF8.GetBytes(key);
            byte[] ivArray = UTF8Encoding.UTF8.GetBytes(iv);
            byte[] toEncryptArray = UTF8Encoding.UTF8.GetBytes(toEncrypt);

            RijndaelManaged rm = new RijndaelManaged();
            rm.Key = keyArray;
            rm.IV = ivArray;
            rm.Mode = CipherMode.CBC;
            rm.Padding = PaddingMode.PKCS7;
```

```

        ICryptoTransform cTransform = rm.CreateEncryptor();
        byte[] resultArray = cTransform.TransformFinalBlock(toEncryptArray, 0,
toEncryptArray.Length);
        return Convert.ToBase64String(resultArray, 0, resultArray.Length);
    }

    public static string Decrypt(string toDecrypt, string key, string iv)
    {
        byte[] keyArray = UTF8Encoding.UTF8.GetBytes(key);
        byte[] ivArray = UTF8Encoding.UTF8.GetBytes(iv);
        byte[] toEncryptArray = Convert.FromBase64String(toDecrypt);

        RijndaelManaged rm = new RijndaelManaged();
        rm.Key = keyArray;
        rm.IV = ivArray;
        rm.Mode = CipherMode.CBC;
        rm.Padding = PaddingMode.PKCS7;

        ICryptoTransform cTransform = rm.CreateDecryptor();
        byte[] resultArray = cTransform.TransformFinalBlock(toEncryptArray, 0,
toEncryptArray.Length);
        return UTF8Encoding.UTF8.GetString(resultArray);
    }
}

```

4. 2. 4. 2PHP 实现 AES 加密、解密（CBC/PKCS5Padding）

//AES 加密

```

function aes_encrypt($encryptKey,$encryptStr) {
    $localIV = $encryptKey;
    $encryptKey = $encryptKey;

    $module = mcrypt_module_open(MCRYPT_RIJNDAEL_128, "", MCRYPT_MODE_CBC,
$localIV);
    mcrypt_generic_init($module, $encryptKey, $localIV);

    $block = mcrypt_get_block_size(MCRYPT_RIJNDAEL_128, MCRYPT_MODE_CBC);
    $pad = $block - (strlen($encryptStr) % $block);
    $encryptStr .= str_repeat(chr($pad), $pad);

    $encrypted = mcrypt_generic($module, $encryptStr);
    mcrypt_generic_deinit($module);
    mcrypt_module_close($module);

    return base64_encode($encrypted);
}

```

//AES 解密

```

function aes_decrypt($encryptKey,$encryptStr) {
    $localIV = $encryptKey;

```

```

$encryptKey = $encryptKey;

$module = mcrypt_module_open(MCRYPT_RIJNDAEL_128, "", MCRYPT_MODE_CBC,
$localIV);
mcrypt_generic_init($module, $encryptKey, $localIV);

$encryptedData = base64_decode($encryptStr);
$encryptedData = mdecrypt_generic($module, $encryptedData);

$e = ord($encryptedData[strlen($encryptedData)-1]);
if($e<=16)$encryptedData=substr($encryptedData, 0,strlen($encryptedData)-$e);
return $encryptedData;
}

$result = aes_encrypt("83AA400AF464C76D",'0001599193983184');
$decryptString = aes_decrypt("83AA400AF464C76D",$result);
echo $result;
echo $decryptString;

```

4. 2. 4. 3PHP7. 1 实现 AES 加密、解密（CBC/PKCS5Padding）

```

<?php

//AES 加密
function aes_encrypt($key,$str)
{
    return base64_encode( openssl_encrypt($str,
'AES-128-CBC',$key,OPENSSL_RAW_DATA,$key) );;
}

//AES 解密
function aes_decrypt($key,$str)
{
    return openssl_decrypt(base64_decode($str), 'AES-128-CBC', $key, OPENSSL_RAW_DATA,
$key);
}

$result = aes_encrypt("83AA400AF464C76D","0001599193983184");
$str = aes_decrypt("83AA400AF464C76D",$result);

echo $result."<br>";
echo $str;
?>

```

4. 2. 4. 4Java 实现的 AES 加密、解密代码（CBC/PKCS5Padding）

```

import javax.crypto.spec.IvParameterSpec;
import javax.crypto.Cipher;

```

```

import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;

public class AESCrypt {

    public static String encrypt(String source, String key) throws Exception {

        byte[] sourceBytes = source.getBytes("UTF-8");
        byte[] keyBytes = key.getBytes("UTF-8");

        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        IvParameterSpec ivParameterSpec = new IvParameterSpec(keyBytes);
        cipher.init(Cipher.ENCRYPT_MODE, new SecretKeySpec(keyBytes,
"AES"),ivParameterSpec);

        byte[] decrypted = cipher.doFinal(sourceBytes);
        return new sun.misc.BASE64Encoder().encode(decrypted);
    }

    public static String decrypt(String encryptStr, String key) throws Exception {
        byte[] sourceBytes = new sun.misc.BASE64Decoder().decodeBuffer(encryptStr);
        byte[] keyBytes = key.getBytes("UTF-8");

        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        IvParameterSpec ivParameterSpec = new IvParameterSpec(keyBytes);
        cipher.init(Cipher.DECRYPT_MODE, new SecretKeySpec(keyBytes,
"AES"),ivParameterSpec);

        byte[] decoded = cipher.doFinal(sourceBytes);
        return new String(decoded, "UTF-8");
    }
}

```

4.3 通用命令参数

说明，参数分组为{a,par:{b},par0:{c}}
abc 里面的字段名称都要不同。

参数形式如下

名称	设置	含义	说明
emit		init 初始化 open 打开 App runCmd 互动调用 App 运行命令 close closeAll 关闭 App msg 发送消息	open 反馈 AppStatus 接收的数据 1/0 系统自动反馈 1 打开 0 关闭

		hardware 获取硬件信息 lasterr	
Obj		pc app appemit media vlc flash office hardWare ...	
AppId	1	1,2,3 autoAdd	一个页面一个 APP，默认为 1。 open 方法自动增加 1，再次 open 默认关闭以前的 AppId=1 的窗口，重新设置为 1。 如果不要关闭则设置为 AppId="autoAdd"，累计新开。 最好自行设定。
AppType	1, -1	1,2,3,4... -1,-2,-3,-4	正数嵌入 负数浮动
windowPars	[12,28,20,28,"0xFFF5F5F5"]	默认值 [12,28,20,28,"0xFFF5F5F5"]	AppType -1 浮动时设置窗口的参数，[字体大小,按钮宽度,按钮高度,标题栏高度,背景色,前景色]
pos	1, -1 [left,top,width,height]	位置 1 更新位置 -1 删除位置数据 [left,top,width,height] 保留	
data			
src			
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 0 不变化 31 响应所有事件 1+2+4+8+16 30 不响应滚动事件 2+4+8+16 23 不响应浏览器窗口 resize 事件 3 滚动 1+移动事件 2 ...	AppShow 为 true 时，APP 和浏览器的响应其变化方式为 按照 AE.CurEventType 1 滚动 2 移动 4 隐显 8 大小 16 缩放比例 求和,对应的值求和即得到 AE.eventType_Follow
AppRuntime		非必需。 0 默认，普通程序，退出浏览器关	

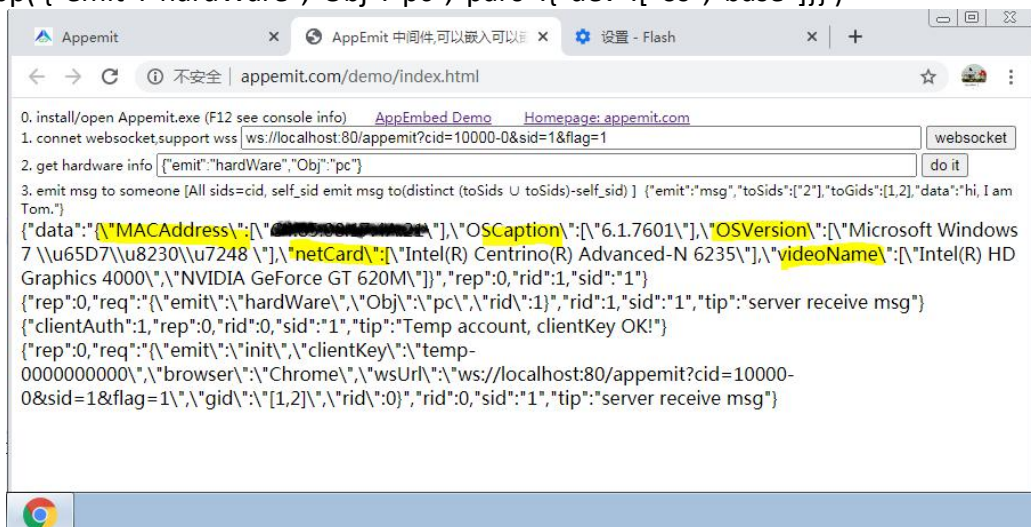
		闭程序 1 服务程序，退出浏览器默认不关闭服务	
AppMark		字符串或者数值	唯一标识每一条命令的每一次执行，用于区分回调函数 在 runCmd 请求命令是可以使用。
par		必填参数	
plugins		字符串或者数组。	引入调用的插件，如果没有则会自动下载已经上传插件库的插件 常用的插件已经默认加入。私有或者第三方的插件需要引入。
objName			
...			
par0		可填参数	
force			对于 runtime 运行服务类型，1 表示强制关闭以前的，重新打开
header			
userAgent			
crossDomain			
...			

4.4 硬件

使用浏览器打开 demo 下的 index.html。授权连接后，发送获取 PC 信息命令。

```
AE.InitApp("ws://localhost:80/appemit?cid=00000-1&sid=1&flag=1")
```

```
AE.OpenApp({'emit':"hardWare","Obj":"pc","par0":{"dev":["os","base"]}})
```



4.4.1 硬件信息

```
{"emit":"hardWare","Obj":"pc","par0":{"dev":["device","os","display","cpu","net","memory","storage","base"]}}
```

名称	设置	含义	说明
emit	hardWare	必需。通信请求。	
Obj	pc	必需。目标对象。	
par0			
dev	字符串 "base" 或者数组 ["device"]	非必需 ["device","os","display","cpu","net","memory","storage","base"],取某个值都即可。 为空时默认设置为"base"	"base"可获取多个设备,但是对于 windows 精简版本可能无法获取信息。其它字段获取为第一个设备。 免费版本仅支持"base"
rid	整数	必需。表示本次具体请求标号。	如果没有设置,则每次发送命令自动加 1。 后续省略描述。

4.4.2 硬件温度

需要管理员身份运行

```
{"emit":"temperature","Obj":"pc"}
```

4.4.3 串口 Pcomm

4.4.3.1 解决办法

- 1) 标准的串口 ocx 或者 dll 使用 IE 控件或者 webkit 内核打开网页最简单,直接操作 object。
- 2) 调用接口直接访问串口。下面为方法 2 的示例。

4.4.3.2 串口控件 Pcomm

在网页中调用 Pcomm.dll 的串口控件,支持多窗口同时异步跨线程调用串口。

```
{"emit":"open","Obj":"comm","AppType":1,"AppShow":true,"pos":1,"par":{"objName":"Pcomm1","Port":"com1","BaudRate":"9600","ByteSize":"8","Parity":"none","StopBits":"1"},"par0":{"Send_AsciiHex":"Ascii","Recv_AsciiHex":"Ascii","BaudRate_list":["50","75","110","134","150","300","600","1200","1800","2400","4800","7200","9600","19200","38400","57600","115200","230400","460800","921600"],"ByteSize_list":["5","6","7","8"],"Parity_list":["none","even","odd","spc","mrk"],"StopBits_list":["1","1.5","2"]}}
```

名称	设置	含义	说明
emit	open	必需。打开串口事件请求。	
Obj	comm	必需。	
AppId	键值	必需。未定义时默认为 1	一个页面打开多个应用,应用的 AppId 必须不同。
AppType	1	必需。 1 Pcomm 2 comm	有默认的右键菜单。 若为负数-1,则是浮动窗口

pos	1	0 无界面不需要 1 有界面，位置自动识别	
data		非必需。	启动后发送给串口的数据
AppShow	false	必需。空时默认窗口不可见。 有界面必需设置 true。	
AppStatus	1/0	系统自动反馈 1 打开 0 关闭	接收的数据
par			
objName	Pcomm1 字符串	字符串变量，字母开头、数字、 下划线、	用来在 js 调用变量
Port	com1/1		
par0			
Send_AsciiHex	Ascii/Hex	Ascii Hex	全局写入方式
Recv_AsciiHex	Ascii/Hex	Ascii Hex	全局接收格式
toFormat	Ascii/Hex	Ascii Hex	对本次发送的 data 转换格式。
BaudRate	9600		
ByteSize	8		
Parity	none		
StopBits	1		

4.4.3. 3 串口发送接收数据

自动开启 COM1 隐藏界面接收数据

```
{"emit":"open","Obj":"comm","AppType":1,"AppShow":0,"pos":1,"par":{"objName":"Pcomm1","Port":"COM1","BaudRate":"9600","ByteSize":"8","Parity":"none","StopBits":"1"},"par0":{"AppMethod":"POST","Send_AsciiHex":"Ascii","Recv_AsciiHex":"Ascii"}}
```

对 COM1 端口发送数据

```
{"emit":"send","Obj":"comm","AppType":1,"AppId":1,"data":"可以,第 1, from com1","par0":{"Send_AsciiHex":"Hex","toFormat":"Hex"}}
```

名称	设置	含义	说明
emit	send	必需。串口发送事件请求。	
Obj	comm	必需。	
AppId	键值	必需。未定义时默认为 1	一个页面打开多个应用，应用的 AppId 必须不同。
AppType	1	必需。 1 Pcomm 2 comm	有默认的右键菜单。 若为负数-1,则是浮动窗口
pos	1	0 无界面不需要 1 有界面，位置自动识别	
data		非必需。	启动后发送给串口的数据

AppShow	false	必需。空时默认窗口不可见。 有界面必需设置 true。	
data		必需。发送数据内容	
par			
objName	字符串	字符串变量，字母开头、数字、 下划线、	用来在 js 调用变量
par0			
Send_AsciiHex	Ascii/Hex	Ascii Hex	本次写入方式。此次 send 命令 为空则默认为 open 事件中的 Send_AsciiHex
toFormat	Ascii/Hex	Ascii Hex	对发送的 data 转换格式。此次 send 命令为空则默认为 open 事件中的 toFormat

4.4.3.4 互动调用串口其它命令

调用 runCmd 命令来操作 App 对象。

```
{ "emit": "runCmd", "Obj": "comm", "AppType": 1, "AppId": 1, "codeStr": "AppJsObject.Pcomm1.write('动态写入 ascii 数据 1')"
```

下面是调用的接口函数，和原含有略有调整。

open(__) = 修改端口号并打开端口，成功返回 true，\n 失败返回 null, 错误信息, 错误代码
close() = 关闭串口\n 这个函数不会被析构函数自动调用,\n 应在确认不再使用时调用此函数关闭串口
getch() = 读取一个字节码\n 失败返回 null, 错误信息
putch() = 发送一个字节码\n 成功返回长度, 失败返回 null, 错误信息
read() = 读取数据, 可选指定读取缓冲区长度\n 成功返回数据, 失败返回 null, 错误信息\n 如果没有指定长度且没有接收到数据返回 null
readBuffer(.(缓冲区, 读取长度) = 读取数据到 buffer 缓冲区, 读取长度可省略\n 成功返回读取长度, 失败返回 null, 错误信息
readHex() = 读取数据并以十六进制编码显示\n 成功返回数据, 失败返回 null, 错误信息
write(.(数据, 长度) = 写入数据\n 不指定长度时自动获取数据长度\n 成功返回写入长度, 失败返回 null, 错误信息
writeHex(.(数据) = 写入十六进制编码数据
getBaudRate() = 返回串口波特率\n 失败返回 null, 错误信息
setBaudRate(.(波特率) = 设置串口波特率\n 失败返回 null, 错误信息
getMode() = 返回串口的工作模式\n 返回值为三个，分别为: 数据位, 停止位, 校验位\n 失败返回 null, 错误信息
ioctl(.(波特率, 数据位, 停止位, 校验位) = 设置串口的工作模式\n 波特率为数值, 省略时使用默认值 9600\n 数据位为数值, 可选值为 5, 6, 7, 8, 省略时默认值为 8\n 停止位可选值为 1, 2, 不指定时默认为 1\n 校验位使用字符串值指定, 可选值为 "even", "odd", "spc", "mrk", 可省略以设置空校验位
flush(.(接收缓冲区, 发送缓冲区) = 清除接收、发送缓冲区\n 对应参数为 true 清除该缓冲区\n 无参数时清除接收缓冲区
lctrl(.(dtr, rts) = 设置串口 RTS/DTS, 参数使用布尔值
iqueue() = 返回输入缓冲区字符长度\n 失败返回 null, 错误信息

oqueue() = 返回发送缓冲区中剩余的数据长度
 getLineStatus() = 获取串口的 CTS, DST, DCD, RI 线的状态
 setFlow(. (CTS 硬流控, RTS 硬流控, TX 软流控, RX 软流控) = 串口流控
 getFlow() = 返回 4 个值, 分别表示 CTS 硬流控, RTS 硬流控, TX 软流控, RX 软流控是否开启
 dataStatus() = 检查接收数据时是否遇到错误\n0 表示无错误, 小于 0 表示函数执行错误, 大于 0 时各二进制位作用:\nbit 0 on - parity error\nbit 1 on - framing error\nbit 2 on - overrun error\nbit 3 on - overflow error
 abortRead() = 强制中止 read, getch 等读数据函数
 getReadTimeouts() = 返回读数据总超时, 间隔超时两个值, 单位毫秒
 setReadTimeouts(. (总超时, 间隔超时) = 设置读数据超时, 单位毫秒
 abortWrite() = 强制中止 Write, putch 等写数据函数
 getWriteTimeouts() = 返回写数据总超时, 间隔超时两个值, 单位毫秒
 setWriteTimeouts(. (总超时, 间隔超时) = 设置写数据超时, 单位毫秒
 termIrqThread(指定字节码, 回调函数, owner 对象) = @.termIrqThread(__/*线程接收到指定字节码时响应事件*/, function(port){\n var sport = sio.port(port);\n})
 termCntIrqThread(指定长度, 回调函数, owner 对象) = @.termCntIrqThread(__/*接收到指定个字节时响应事件*/, function(port){\n var sport = sio.port(port);\n})
 modemIrqThread(回调函数, owner 对象) = @.modemIrqThread(function(port){\n var sport = sio.port(port); __/*当硬件线路 (CTS, DSR, CD, RI) 的电压发生变化时触发事件*\n})
 breakIrqThread(回调函数, owner 对象) = @.breakIrqThread(function(port){\n var sport = sio.port(port); __/*接收到中断信号时触发事件*\n})
 txEmptyIrqThread(回调函数, owner 对象) = @.txEmptyIrqThread(function(port){\n var sport = sio.port(port); __/*输出缓冲区最后一个字符发送后触发此事件*\n})
 transmitAscii(文件路径, 进度回调函数) = @.transmitAscii("文件路径", \n
 function(length, bufferSize, buffer, totalLength){\n __/*传输文件协议: ASCII\nlength 为当前传输大小, totalLength 为总大小, 其他参数仅用于调试*\n } \n)
 receiveAscii(超时秒数, 文件路径, 进度回调函数) = @.transmitAscii("文件路径", \n
 function(length, bufferSize, buffer, totalLength){\n __/*接收文件协议: ASCII\nlength 为当前传输大小, totalLength 为总大小, 其他参数仅用于调试*\n } \n)
 transmitKermit(文件路径, 进度回调函数) = @.transmitKermit("文件路径", \n
 function(length, bufferSize, buffer, totalLength){\n __/*传输文件协议: Kermit\nlength 为当前传输大小, totalLength 为总大小, 其他参数仅用于调试*\n } \n)
 receiveKermit(超时秒数, 文件路径, 进度回调函数) = @.receiveKermit("文件路径", \n
 function(length, bufferSize, buffer, totalLength){\n __/*接收文件协议: Kermit\nlength 为当前传输大小, totalLength 为总大小, 其他参数仅用于调试*\n } \n)
 transmitYmodem(文件路径, 进度回调函数) = @.transmitYmodem("文件路径", \n
 function(length, bufferSize, buffer, totalLength){\n __/*传输文件协议: Ymodem\nlength 为当前传输大小, totalLength 为总大小, 其他参数仅用于调试*\n } \n)
 receiveYmodem(超时秒数, 文件路径, 进度回调函数) = @.receiveYmodem("文件路径", \n
 function(length, bufferSize, buffer, totalLength){\n __/*接收文件协议: Ymodem\nlength 为当前传输大小, totalLength 为总大小, 其他参数仅用于调试*\n } \n)
 transmitZmodem(文件路径, 进度回调函数) = @.transmitZmodem("文件路径", \n
 function(length, bufferSize, buffer, totalLength){\n __/*传输文件协议: Zmodem\nlength 为当前传输大小, totalLength 为总大小, 其他参数仅用于调试*\n } \n)
 receiveZmodem(超时秒数, 文件路径, 进度回调函数) = @.receiveZmodem("文件路径", \n
 function(length, bufferSize, buffer, totalLength){\n __/*接收文件协议: Zmodem\nlength 为

```

当前传输大小,totalLength 为总大小,其他参数仅用于调试*\n\n)
transmitXmodem1kCrc(文件路径,进度回调函数)= @.transmitXmodem1kCrc("文件路径",\n
    function(length,bufferSize,buffer,totalLength){\n        __/*传输文件协议:XMODEM, 1K block
size, 16 bit CRC\nlength 为当前传输大小,totalLength 为总大小,其他参数仅用于调试*\n    }\n)
receiveXmodem1kCrc(超时秒数,文件路径,进度回调函数)= @.receiveXmodem1kCrc("文件路径",\n
    function(length,bufferSize,buffer,totalLength){\n        __/*接收文件协议:XMODEM, 1K block
size, 16 bit CRC\nlength 为当前传输大小,totalLength 为总大小,其他参数仅用于调试*\n    }\n)
transmitXmodemCheckSum(文件路径,进度回调函数)= @.transmitXmodemCheckSum("文件路径
",\n function(length,bufferSize,buffer,totalLength){\n        __/*传输文件协
议:XMODEM,CHECKSUM\nlength 为当前传输大小,totalLength 为总大小,其他参数仅用于调试*\n
    }\n)
receiveXmodemCheckSum(超时秒数,文件路径,进度回调函数)= @.receiveXmodemCheckSum("文件
路径",\n    function(length,bufferSize,buffer,totalLength){\n        __/*接收文件协
议:XMODEM,CHECKSUM\nlength 为当前传输大小,totalLength 为总大小,其他参数仅用于调试*\n
    }\n)
transmitXmodemCrc(文件路径,进度回调函数)= @.transmitXmodemCrc("文件路径",\n
    function(length,bufferSize,buffer,totalLength){\n        __/*传输文件协议:XMODEM, 16 bit
CRC\nlength 为当前传输大小,totalLength 为总大小,其他参数仅用于调试*\n    }\n)
receiveXmodemCrc(超时秒数,文件路径,进度回调函数)= @.receiveXmodemCrc("文件路径",\n
    function(length,bufferSize,buffer,totalLength){\n        __/*接收文件协议:XMODEM, 16 bit
CRC\nlength 为当前传输大小,totalLength 为总大小,其他参数仅用于调试*\n    }\n)

```

4.4.4 串口 mscomm32 控件

comm32 效能略低,新版的 MSComm32.OCX 中存在一个影响传输二进制数据的 Bug。
使用 IE 控件打开网页最简单。

```

{"emit":"open","Obj":"comm","AppType":2,"AppShow":false,"par":{"objName":"mscomm1","Port":
"COM1","Settings":"9600,N,8,1"},"par0":{"Recv_AsciiHex":"Ascii"}}
发送数据
{"emit":"send","Obj":"comm","AppType":2,"AppId":1,"data":"from mscomm1"}
{"emit":"runCmd","Obj":"comm","AppType":2,"AppId":1,"codeStr":"AppJsObject.mscomm1.output=
mscomm1 write ascii"}

```

名称	设置	含义	说明
emit	open	必需。打开串口事件请求。	
Obj	comm	必需。	
AppId	键值	必需。未定义时默认为 1	一个页面打开多个应用,应用的 AppId 必须不同。
AppType	2	必需。 1 Pcomm 2 comm	有默认的右键菜单。 若为负数-1,则是浮动窗口
pos	1	0 无界面不需要 1 有界面,位置自动识别	
data		非必需。	启动后发送给串口的数据
AppShow	false	必需。空时默认窗口不可见。 有界面必需设置 true。	

AppStatus	1/0	系统自动反馈 1 打开 0 关闭	接收的数据
par			
objName	mscomm1	字符串变量，字母开头、数字、下划线、	用来在 js 调用变量
CommPort	com1/1	设置串口	
Settings	"9600,N,8,1"	设置波特率等参 波特率，校验位，数据位，停止位	
InBufferSize	1024	设置输入缓冲区大小	
OutBufferSize	1024	设置输出缓冲区大小	
RThreshold	1	设置收到多少个字符后触发 OnComm 事件	
InputMode	1	设置输入方式 二进制方式 1 文本模式 0	
InputLen	0	设置当前接收长度为 0	
DataBits	8		
StopBits	1		
BaudRate	9600		
DTREnable	1		
Handshaking	0		
NullDiscard	0		
ParityReplace	"?"		
RTSEnable	1		
SThreshold	2		
EOFEEnable	0		
par0			
toString	true/false	默认 true false 反馈为 buffer	反馈的数据默认由 buffer 转为 string
Send_AsciiHex	Ascii/Hex	Ascii Hex	全局写入方式
Recv_AsciiHex	Ascii/Hex	Ascii Hex	全局接收格式
toFormat	Ascii/Hex	Ascii Hex	对本次发送的 data 转换格式。
backNoJson	true/false	默认 false,反馈为 json true,调用的 dll 或者 ocx 反馈的数据直接发送，不处理编码或者 json 转换	runCmd 命令反馈的数据处理

4.4.5 USB Hid 通信接口信息

{"emit": "hid"} (后续删除改函数)

名称	设置	含义	说明
----	----	----	----

emit	hid	必需。获得 hid 通信接口信息。	

```
{"emit":"open", "Obj":"usbhid", "AppShow":false, "par":{}, "par0":{"objName":"usb", "getUsbHids":1}}
```

同时可以使用 runCmd 来运行代码，或者 runCall 执行函数

Return AppJsObject.usbHids

或者 AppJsObject.getUsbHids()

hid = USB HID 通信接口

hid.device(.(vendorId,productId,serialNumber) = 打开设备返回句柄

hid.device(.(path) = 打开设备返回句柄,参数为 each 迭代器返回的 deviceInfo.path

hid.device() = !stdhiddevice.

setNonblocking(true) = 启用非阻塞模式

errorMessage() = 返回错误信息

read() = 读数据,可选使用参数@1 指定缓冲区长度

readTimeout(__) = 读数据,参数@1 指定毫秒单位的超时值,\n 可选使用参数@1 指定缓冲区长度

write(.(数据,长度,报告 ID) = 写数据,除参数@1 以外其他参数为可选参数

getFeatureReport() = 读功能报告,可选使用参数@1 指定缓冲区长度

sendFeatureReport(.(数据,长度,报告 ID) = 写功能报告,除参数@1 以外其他参数为可选参数

getManufacturerString() = 读制造商字符串,可选使用参数@1 指定缓冲区长度

getProductString() = 获取产品字符串,可选使用参数@1 指定缓冲区长度

getSerialNumberString() = 获取序列号字符串,可选使用参数@1 指定缓冲区长度

getIndexedString() = 获取索引字符串,可选使用参数@1 指定缓冲区长度

close() = 关闭对象

path = 设备路径

vendorId = 厂商 ID

productId = 产品 ID

serialNumber = 序列号

releaseNumber = 设备版本号

manufacturerString = 制造商,字符串

productString = 产品,字符串

usagePage = 使用页

usage = 使用 ID

interfaceNumber = 接口编号

4.4.6 USB 热拔事件

```
{"emit":"open", "Obj":"driveNotify", "AppId":"notifyAppId1", "AppType":2, "AppShow":false, "par0":{"tr
```

```
ayInsert":{"msg":"U 盘打开","title":"监控设备","timeOut":3000}}}
```

名称	设置	含义	说明
emit	open	必需。监控热拔请求。	
Obj	driveNotify		
AppId	notifyAppId 1	键值。最好设置为特殊的	
AppType	数值或者数组 2 , [1,2],	1 DRIVE_NO_ROOT_DIR 说明无效的 2 DRIVE_REMOVABLE 可移动磁盘 3 DRIVE_FIXED 固定磁盘 4 DRIVE_REMOTE 网络磁盘 5 DRIVE_CDROM 光驱 6 DRIVE_RAMDISK 为 RAM 磁盘	
AppShow	false	无界面	
par0			
tray	true	默认托盘提示 false 关闭托盘提示	
trayInsert	U 盘打开提示设置	默认{"msg":"U 盘打开","title":" 监控设备","timeOut":3000}	
trayRemove	U 盘关闭提示设置	默认{"msg":"U 盘关闭","title":" 监控设备","timeOut":3000}	

插入 U 盘

```
Obj: "driveNotify"
data: {deviceName: "\\Device\\HarddiskVolume16\\", drive: "G:", driveType: 2, eventType: "insert", flags: 0}
eventType: "insert"
insertCnt: 1
rep: 0
sid: "1"
```

拔出 U 盘反馈

```
Obj: "driveNotify"
data: {drive: "G:", driveType: 1, eventType: "remove", flags: 0}
eventType: "remove"
removeCnt: 1
rep: 0
sid: "1"
```

关闭

```
{"emit":"close","Obj":"driveNotify","AppId":"notifyAppId1"}
```

4.5 用户间通话

打开 demo 下的 index.html,模拟不同 sid 打开浏览器。

连接 Appemit 授权后,在 sid=1 下发送命令。

```
{"emit":"msg","Obj":"sid","toSids":["2"],"toGids":[1,2],"data":"hi, I am Tom."}
```

在客户 cid 全集下,通过唯一的 sid 对话,可以一对一,或者一对多通话。

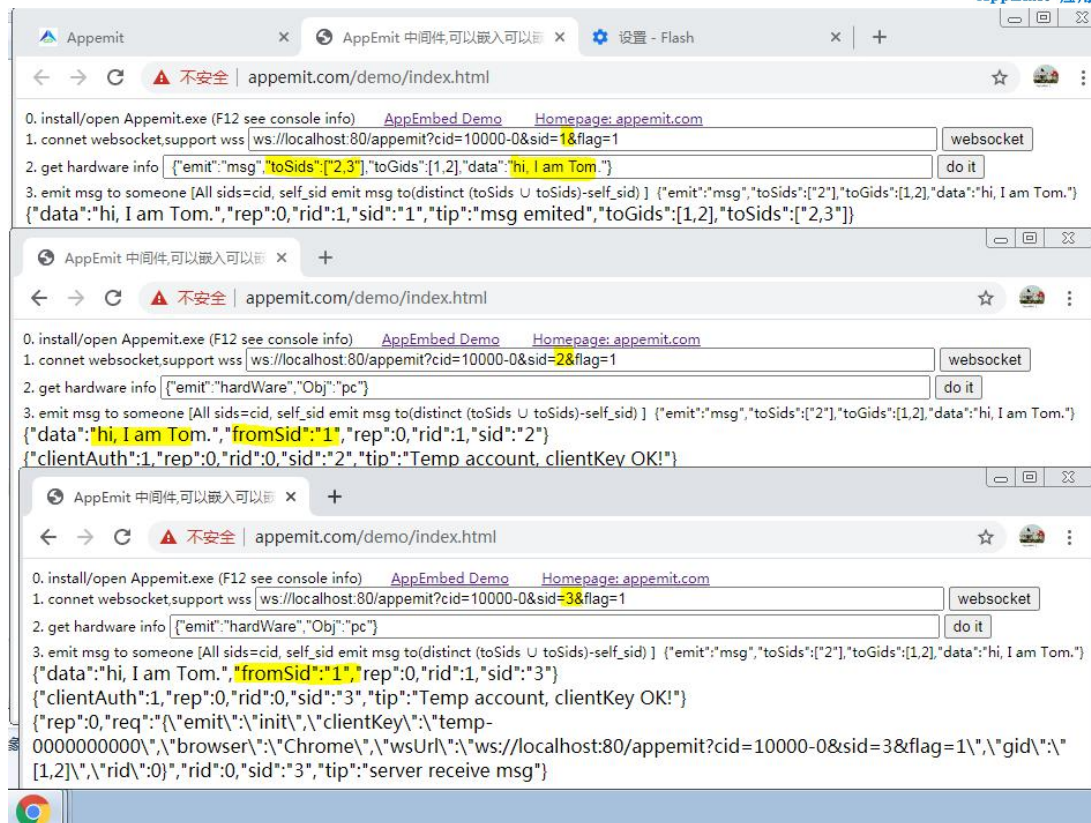


图 1 对 2 和 3 通话。

另外还可以设置不同群 gid，一个 sid 可以加入不同的 gid。

发送消息时，在 cid 全集下，所有的 toSids 和 toGids 取对应的 sid 交集剔除，并排除自身。

```
{"emit":"msg", "Obj":"sid", "toSids":["2"], "toGids":[1,2], "data":"hi, I am Tom."}
```

名称	设置	含义	说明
emit	msg	必需。通信事件请求。	
Obj	sid	用户自行设定的 sid	
toSids	必需要有一个	非必需。可以是数组。	All sids=cid, self_sid emit msg to(distinct (toSids ∪ toSids)-self_sid) 发送的全集是：发送的用户和发送所在的群（频道、小组）的用户的全集剔除后，排除自身
toGids		非必需。可以是数组。	
data		必需。	

4.6 打开 APP

参数格式如下

名称	设置	含义	说明
emit	open	必需。打开控件 APP 通信事件请求。	
Obj		必需。 flash 默认 word 后续支持 excel 后续支持	

		CAD 后续支持	
AppType	$\pm 1 \pm 2 \pm 3 \pm 4$	正数 表示嵌入 负数 浮动窗口 在 Obj 不同时，AppType 的含义不一样。	当 AppType 负数浮动窗口时，默认 APP 为置顶。 取消置顶 { "emit": "setPar", "Obj": "flash", "topMost": false }
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
par0			注意不同 APP 可能不同
attach	1	附着 APP 方式	wps 最好采用此方式。 如果程序出现未知错误，可采用此方式
header		头部	
userAgent		代理	只写属性，不可读。
crossDomain	bool	默认 true True false	是否跨域
rightMenu	右键菜单	-1 不处理，控件原有状态 0/null 禁用菜单 1 自定义简单菜单	
show_UpdateTool	显示下载插件等待窗口	默认 true false 不显示	需要下载更新包显示。可以直接在 https://github.com/appemit/appemit/plugins 下载全集的插件

4.6.1 打开文件

以系统默认的程序，打开文件。

为了安全，不支持直接打开的文件类型包括

"exe"; "msi"; "cmd"; "js"; "jar"; "inf"; "com"; "scr"; "reg"; "bat"; "vbs"; "py"

发送命令打开文件，以独立的方式打开，和浏览器进程独立，没有联系。

{ "emit": "open", "Obj": "file", "AppShow": false, "src": ["/demo/htmlDemo/file/d1.docx", null] }

src: EXE 文件路径, 启动参数="", 操作类型 = "open", 显示属性 _SW_, 工作目录=""

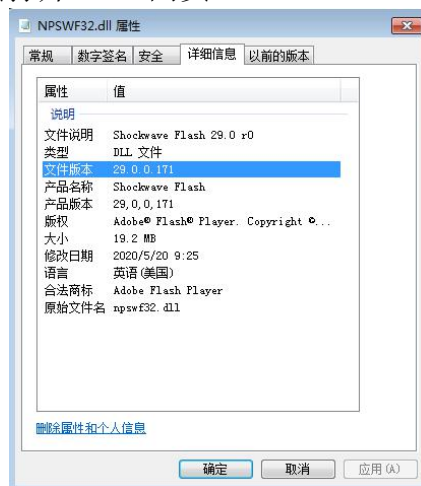
名称	设置	含义	说明
emit	open	必需。打开控件 APP 通信事件请求。	
Obj	file	必需。	

AppId	1	必需。	
src		必需。	[file,参数,startInfo]
pos	-1		
AppShow	false	必需。	不发送 pos 位置等数据
par0			
UWP	1、0、null	null 默认。以传统方式打开可执行文件，若失败尝试 UWP 文件打开 0 只以传统方式打开可执行文件 1 只打开 UWP 文件	

4.6.2 flash

两种方法，主要四种形式实现场景

- 1、使用 Appemit 程序自带的插件 plugins/Flash32.ocx
- 2、使用 Appemit 程序自带的插件 plugins/NPSWF32.dll
- 3、使用 IE 或者 webkit 内核打开 flash 网页



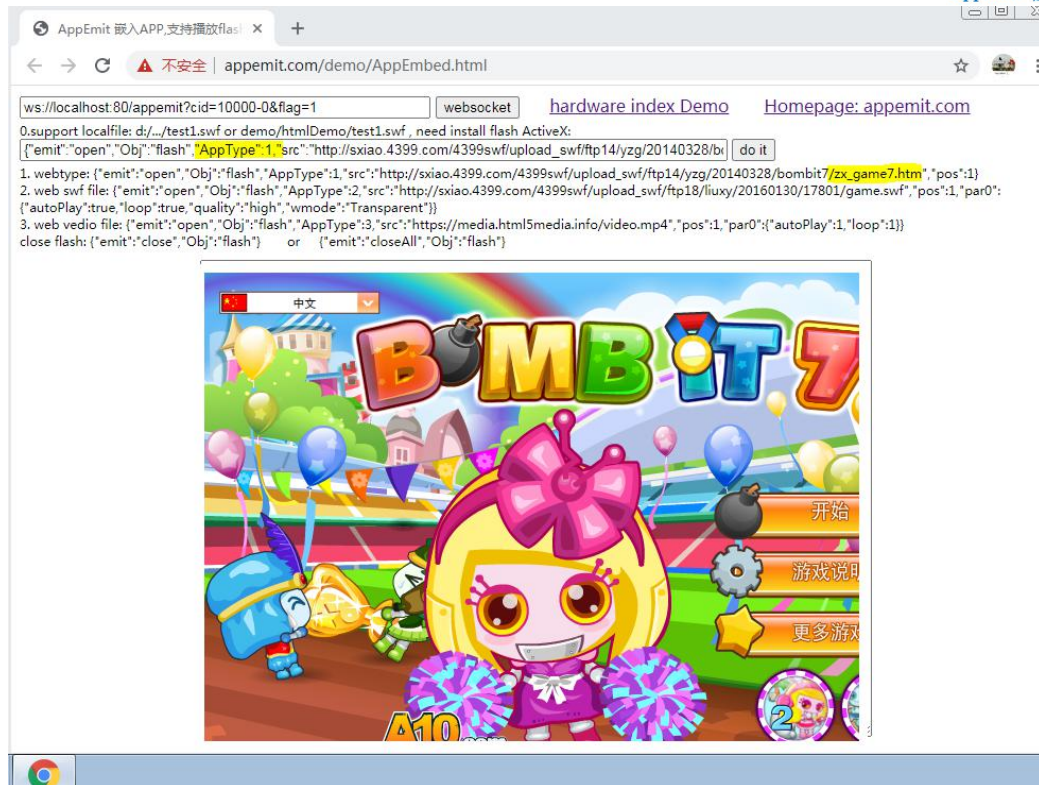
4.6.2.1 NPAPI 打开 flash 网页 webkit 内核

能打开常用网页，目前的插件不支持 html5 的媒体特性。如有需要，可以使用 node 或者 electron 插件。

使用 webkit 内核使用 Appemit 程序自带的插件 NPSWF32.dll，能打开嵌有 flash 的网页，不默认具有右键菜单。

连接授权后，发送命令"AppType":1 的形式。

```
{"emit":"open","Obj":"flash","AppType":1,"src":"http://sxiao.4399.com/4399swf/upload_swf/ftp14/yzg/20140328/bombit7/zx_game7.htm","pos":1}
```



```
{
  "emit": "open",
  "Obj": "flash",
  "AppType": 1,
  "src": "http://sxiao.4399.com/4399swf/upload_swf/ftp14/yzg/20140328/bombit7/zx_game7.htm",
  "pos": 1,
  "par0": {
    "header": null,
    "userAgent": null,
    "crossDomain": true
  }
}
```

名称	设置	含义	说明
emit	open	必需。打开控件 APP 通信事件请求。	
Obj	flash	必需。	
AppType	1	必需。 1 web 2 web flash 文件 3 web 媒体文件 4 ActiveX	若为负数-1,则是浮动窗口
src		必需。	
pos	{ "left": 372, "top": 203, "width": 606, "height": 406 }	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器，自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
Par0		可选。	
header		头部	
userAgent		代理	只写属性，不可读。

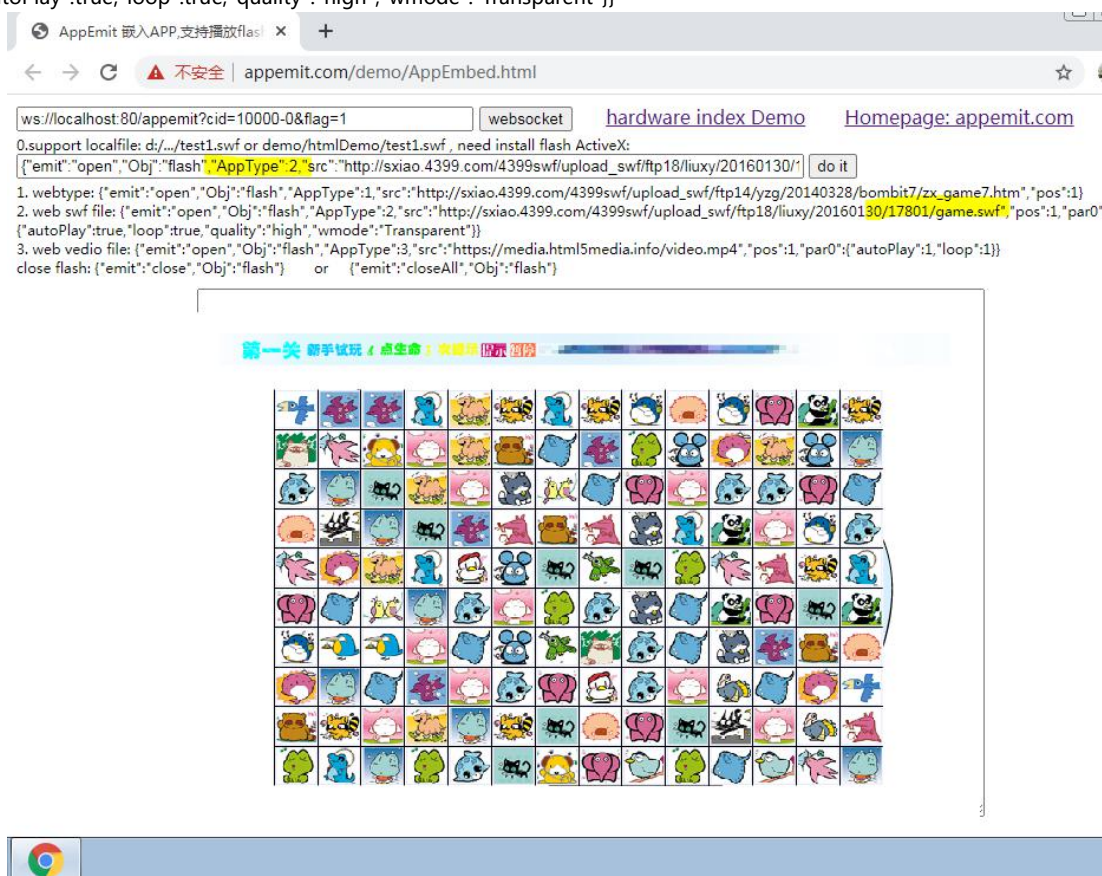
crossDomain	bool	默认 true True false	是否跨域
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	

4.6.2. 2NPAPI 打开网络 flash 文件

使用 Appemit 程序自带的插件 NPSWF32.dll，打开网络 flash 文件。

连接授权后，发送命令"AppType":2 的形式。

```
{
  "emit": "open",
  "Obj": "flash",
  "AppType": 2,
  "src": "http://sxiao.4399.com/4399swf/upload_swf/ftp18/liuxy/20160130/17801/game.swf",
  "pos": 1,
  "par0": {
    "autoplay": true,
    "loop": true,
    "quality": "high",
    "wmode": "Transparent"
  }
}
```



```
{
  "emit": "open",
  "Obj": "flash",
  "AppType": 2,
  "src": "http://sxiao.4399.com/4399swf/upload_swf/ftp18/liuxy/20160130/17801/game.swf",
  "pos": 1,
  "par0": {
    "autoplay": true,
    "loop": true,
    "quality": "high",
    "wmode": "Transparent",
    "header": null,
    "userAgent": null,
    "crossDomain": true
  }
}
```

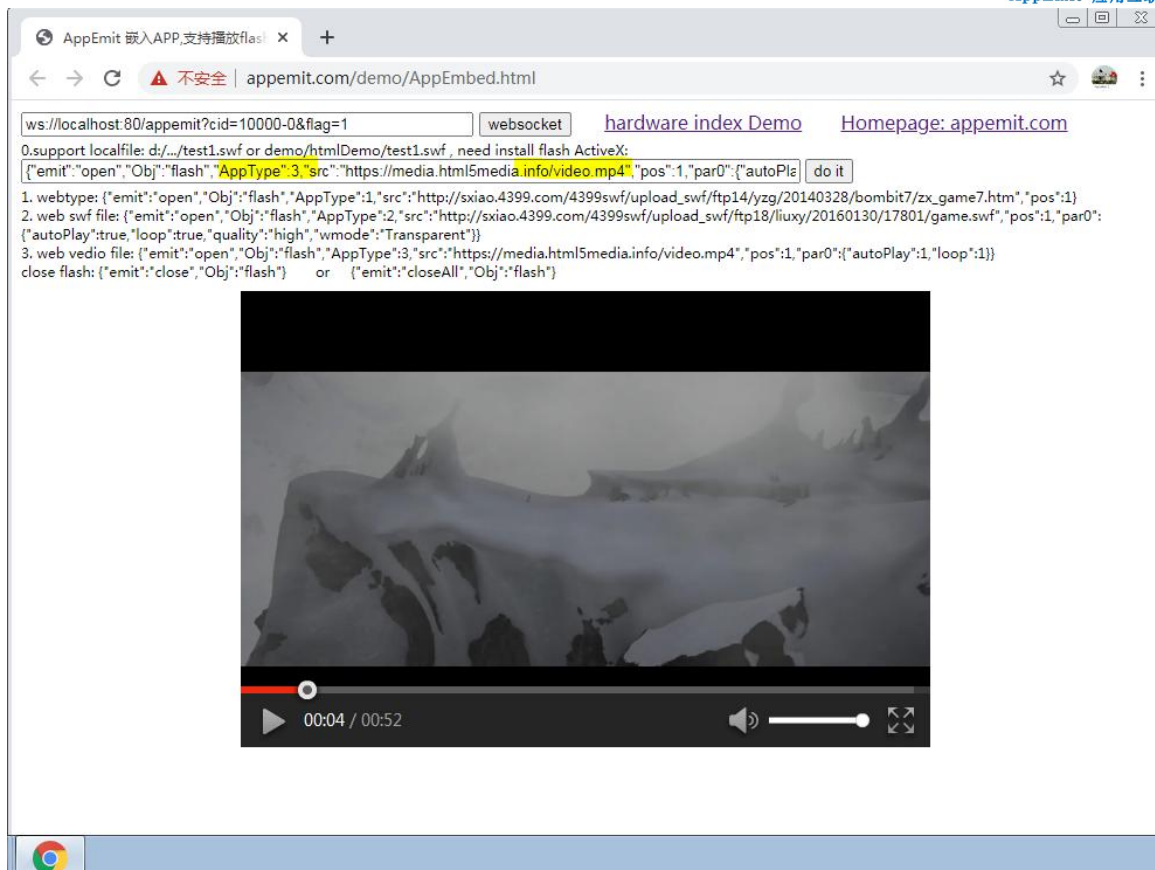
名称	设置	含义	说明
emit	open	必需。打开控件 APP 通信事件请求。	
Obj	flash	必需。	
AppType	2	必需。 1 web 2 web flash 文件 3 web 媒体文件 4 ActiveX	使用 webkit 内核打开，没有默认的右键菜单。 若为负数-2,则是浮动窗口

src		必需。	
pos	{"left":372,"top":203,"width":606,"height":406}	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器，自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
par0	autoPlay	可选。默认 true	参考 flash 官方默认参数。
	loop	可选。默认 true	参考 flash 官方默认参数。
	quality	可选。默认 high	参考 flash 官方默认参数。
	wmode	可选。默认 Transparent	参考 flash 官方默认参数。
header		头部	
userAgent		代理	只写属性，不可读。
crossDomain	bool	默认 true True false	是否跨域
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	

4.6.2. 3NPAPI 打开 flash rtmp 等媒体文件

使用 Appemit 程序自带的插件 NPSWF32.dll，打开网络媒体文件，包括 flv,mp4, rtmp 等。连接授权后，发送命令"AppType":3 的形式。

```
{"emit":"open","Obj":"flash","AppType":3,"src":"https://media.html5media.info/video.mp4","pos":1,"par0":{"autoplay":1,"loop":1}}
```



```
{
  "emit": "open",
  "Obj": "flash",
  "AppType": 3,
  "src": "https://cdn.jsdelivr.net/gh/appemit/appemit/docs/video/x1.flv",
  "pos": 1,
  "par0": {
    "autoplay": 1,
    "loop": 1,
    "header": null,
    "userAgent": null,
    "crossDomain": true
  }
}
```

名称	设置	含义	说明
emit	open	必需。打开控件 APP 通信事件请求。	
Obj	flash	必需。	
AppType	3	必需。 1 web 2 web flash 文件 3 web 媒体文件 4 ActiveX	若为负数-3,则是浮动窗口
src		必需。	视频格式: mp4、flv、m3u8、rtmp 视频编码: H.264 音频编码: AAC、MP3 音频格式: MP3
pos	{ "left": 372, "top": 203, "width": 606, "height": 406 }	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器, 自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false

AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
Par0	autoPlay	可选。默认 1	参考 https://player.alicdn.com/aliplayer/setting/setting.html
	loop	可选。默认 1	
header		头部	
userAgent		代理	只写属性，不可读。
crossDomain	bool	默认 true True false	是否跨域
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	

4.6.2.4ActiveX 打开 flash 文件

Flash_ocx 首次使用，而本机未安装则自动安装，win7 需要管理员权限，有黑窗提示。

1) 打开网络 flash 文件

打开 demo 下的 AppEmbed.html,连接授权后，发送使用 ActiveX ("AppType":4) 打开网络 flash 文件命令，参数如下。

```
{"emit":"open","Obj":"flash","AppType":4,"src":"http://img1.yo4399.com/swf/00/0ff035e0e96584c07df65ab3636f72.swf","pos":1,"par0":{"autoplay":1,"toolbar":0,"rightMenu":0,"hitCaption":0,"hideStop":0,"loop":1,"volumeMute":0,"flashVars":"a=0&b=0&c=SetInSrc"}}
```

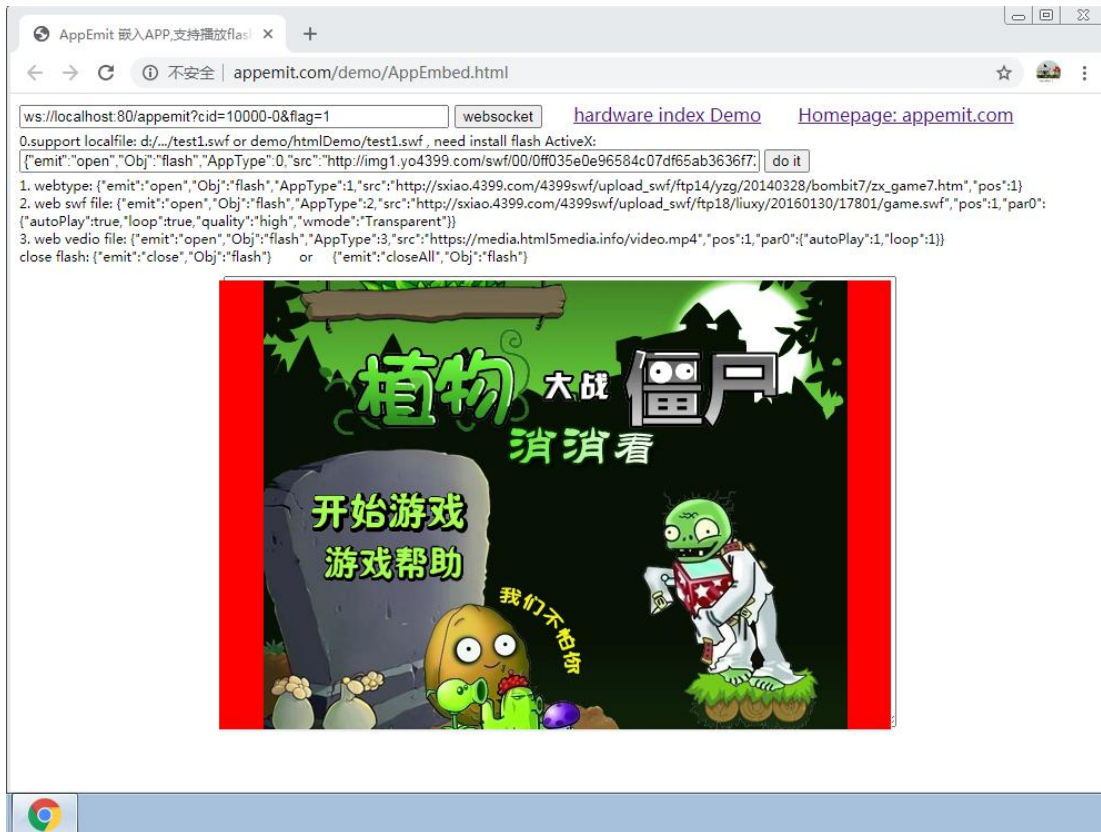
注意事项：

在客户端需要下载安装 flash player ActiveX。

路径是 / 或许 \\

flashVars 可以设置在 src 中

AppType 正数为嵌入 APP，负数为浮动窗口，后同。

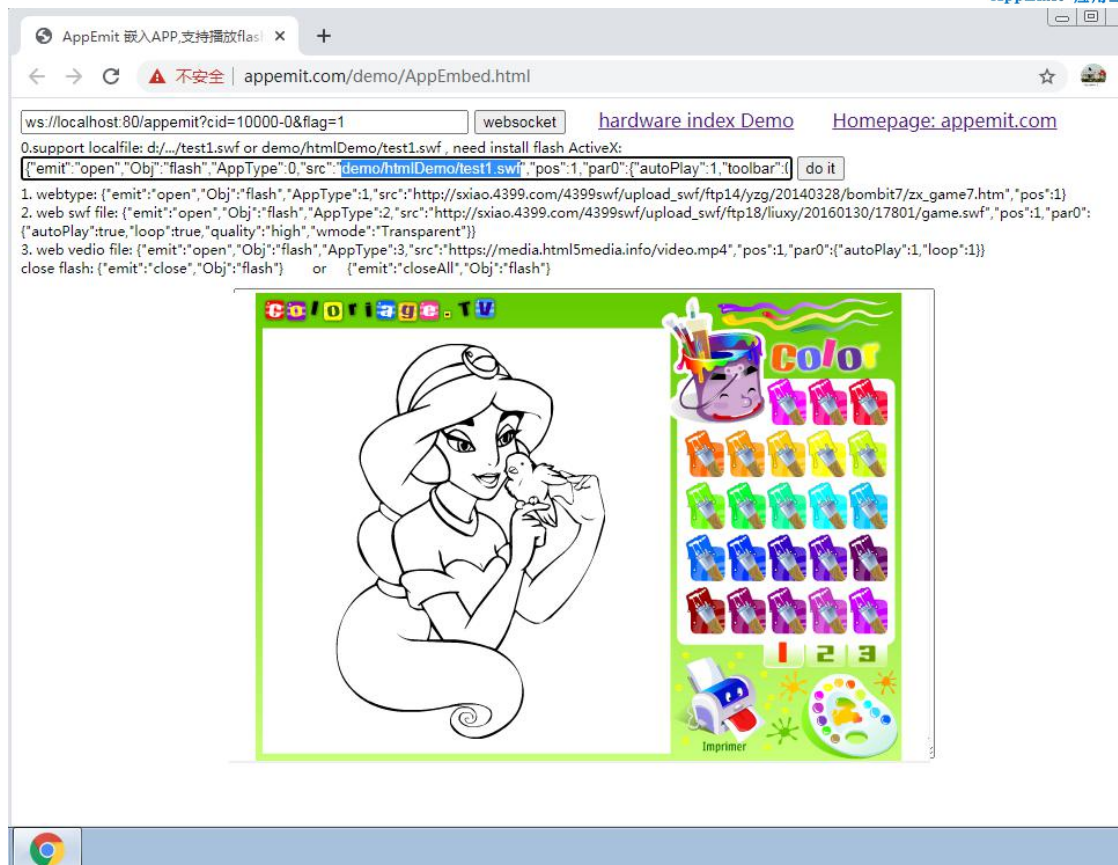


刷新即可关闭 flash

2) 打开本地 flash 文件

可以是绝对或者相对路径，相对于 AppEmit.exe 的路径: "demo/htmlDemo/test1.swf"。

```
{"emit":"open","Obj":"flash","AppType":4,"src":"demo/htmlDemo/test1.swf","pos":1,"par0":{"autoPlay":1,"toolbar":0,"rightMenu":0,"hitCaption":0,"hideStop":0,"loop":1,"volumeMute":0,"flashVars":{"a=0&b=0&c=SetInSrc"}}}
```



```
{
  "emit": "open",
  "Obj": "flash",
  "AppType": 4,
  "src": "http://img1.yo4399.com/swf/00/0ff035e0e96584c07df65ab3636f72.swf",
  "pos": 1,
  "par0": {
    "autoplay": 1,
    "toolbar": 0,
    "rightMenu": 0,
    "hitCaption": 0,
    "hideSt": 0,
    "loop": 1,
    "volumeMute": 0,
    "flashVars": "a=0&b=0&c=SetInSrc"
  }
}
```

名称	设置	含义	说明
emit	open	必需。打开控件 APP 通信事件请求。	
Obj	flash	必需。 flash 默认 word 后续支持 excel 后续支持 CAD 后续支持	
AppType	4	必需。 1 web 2 web flash 文件 3 web 媒体文件 4 ActiveX	设为 4 时, 如何本地没有安装 ActiveX 则默认使用方式 2 打开 若为负数-4,则是浮动窗口
src		必需。	
pos	{ "left": 372, "top": 203, "width": 606, "height": 406 }	必需。 1 默认使用代码自动识别的位置。	
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false

AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
par0	autoPlay	可选。 0 不自动播放 1 自动播放，默认	
	toolbar	可选。 0 没有控制条，默认 1 有控制条	
	rightMenu	可选。 0 1	
	hitCaption	可选。 0 左按鼠标不能拖动，默认 1 左按鼠标能拖动	
	hideStop	可选。 0 隐藏不见时不停止播放，默认 1 隐藏不见时停止播放	
	loop	可选。 0 不自动循环播放， 1 循环播放，默认	
	volumeMute	可选。 0 不静音，默认 1 静音	
	flashVars	可选。	可以设置在 src 里面
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	

4.6.3 打开网页

推荐使用 <https://httpbin.org/anything> 做测试。

含有 Object 对象的，一般需要注册才能使用。

下面举例为不含有 Object 对象的，和已经注册 NPSWF\vlc 的了。

可以加载前设置 header 来设置 Set-Cookie 等参数，便于浏览器反馈到服务器做 session 使用。

加载后设置 cookie 使用 cookieSet。

4.6.3.1 提交登陆命令

```
{"emit": "func", "Obj": "postForm", "par": {"loginSite": "https://oa.tongda2000.com/logincheck.php", "userpwd": {"UNAME": "lijia", "encode_type": 1}, "form": {"TO_ID": "912", "TO_NAME": "王", "SUBJECT": "}}
```

```
测试邮件标题"},"submitSite":"https://oa.tongda2000.com/general/email/new/submit.php"}}
```

4.6.3.2 可以定节点执行任务

可在打开网页前，后，关闭前后执行对应的操作。

```
par0={"AppStep":{"init":"AppJs.msgbox('测试')", "destroy":null, "closed":null,"loaded":"return {111}"}}}
```

4.6.3.3 在 index.html 调用 embed.html 里的 js 执行命令

可以使用 htmlStr，直接在 index.html 处理命令也可以，这样就不需要 embed.html。

直接操作 IE 内核对象,执行 js 赋值， AppJsObject.web.doScript 或者 AppJsObject.web.eval, 执行反馈

或者 AppJsObject.web.script 能直接获取变量和方法

```
var jscode1="" var a=3;var c='c';var b=a+1;alert(b);var myadd=function(a,b){return a+b;}"
```

```
ReqPar6= { "emit": "runCmd", "Obj": "web", "AppId": 1, "codeStr": '
AppJsObject.web.doScript('+jscode1+'); ++"    ++AppJsObject.web.script.c++"
"++AppJsObject.web.script.myadd(3,6);' };
```

直接操作 AppJsObject.web 调用

```
{"emit": "runCmd", "Obj": "web", "AppId": 1, "codeStr":
'AppJsObject.web.document.cookie="newID=4"; return AppJsObject.web.document.cookie;'} }
```

//直接操作 IE 内核对象, Jquery 反馈 元素,支持 Jquery js html css 操作
AppJsFun.log(AppJsObject.web.jQuery("#testWebObj")); //调试显示数据

//在 Jquery 中,用\$("#id")来获得页面的 input 元素,其相当于 document.getElementById("element")
但是,该获取的是一个 Jquery 对象,而不是一个 dom element 对象,加[0]

```
ReqPar8= { "emit": "runCmd", "Obj": "web", "AppId": 1, "codeStr": `return
AppJsObject.web.jQuery("#IDOpenPdf")[0].value;` };
```

//直接操作 IE 内核对象, getEle 反馈 元素,
ReqPar81= { "emit": "runCmd", "Obj": "web", "AppId": 1, "codeStr": `return {mark=1;
value=AppJsObject.web.getEle("IDOpenPdf").value;}` };

4.6.3.4 被嵌入的网页 embed.html 回调数据到 index.html

使用 external 接口，反馈 json 字符串数据

```
external.emitBack(JSON.stringify({a:56,b:"sdfsd"}))
```

在这里调用 IE 网页内核的方法 ,通过 external 调用

```
go(url);// headers 为追加 IE url,headers,target,flags , kit bkink(url)
goBack()
goForward()
```

```
doScript(code,frame) //IE(jscode,frame) kit bkink(jscode)
post(url,postdata,headers,target,flags) IE (url,postdata,headers,target,flags) , kit
blink(url,postdata)
refresh() 仅 IE 刷新页面\n 如果服务器未更新则不会重新下载,类似在浏览器中按 F5 的效果
refresh2(lev=0) 仅 IE 重新下载页面\n 可在参数中指定级别 Refresh20
refresh3(lev=3) 仅 IE 重新下载最新页面\n 添加 Pragma:no-cache 请求头,强制服务端刷新\n
类似在浏览器中按下 Ctrl+F5 的效果 Refresh23
reload() 重新下载最新页面= refresh3
stop()
document()
script()
getDoc() 可以包含 document 以及 iframe
click()
```

4.6.3.5 支持网页 Object 对象

具体例子参考 demo/htmlDemo/web.html

1 打开网页前单独注册控件，具体见 3.5.2.5 注册控件

```
{"emit":"func","Obj":"regsvr32","par":regPar};
```

2 直接在{"emit":"open","Obj":"web", "embed_object": [Par]} 设置 embed_object 即可，私有的需要验证。

```
Par={ //多个 object 可以设置 embed_object 为数组
  "objName":"ReportXl"
  , "pid":"ReportX" //使用公共插件 pid 名称, pid 名称是 AE 共有的, 则调用共有的
  插件
  , "reg":true //null 不执行注册 1 强制操作 true 未注册则注册 false 卸载注
  册 (重启 AppEmit 后, 就不能加载 dll 了)
  , "asAdmin":0 //1 需要管理员权限注册 0 不需要 , win7 管理员身份会有提示确认窗口
  //等 ReportX.ocx 下载下来可以拷贝到下面的私有目录测试
  // , "pid":"292A53CD-DA8F-46A5-808D-B286F2759C37" //使用私有插件, 则需要增加下
  列信息 如果 dll/ocx 文件改变了, 需要更新 appemit 登陆窗口中我的应用里面的 sha1 值

  , "dllFile":"/plugins/private/292A53CD-DA8F-46A5-808D-B286F2759C37/ReportX.ocx"
  //为了安全, 限制为 AppEmit 文件夹, 不能注册或者注销其它文件夹下面的 ocx
  , "CLASSES_ROOT":"HKEY_CLASSES_ROOT\\ReportProj1.ReportX\\CLSID" //可选, reg=
  true 检查是否已经注册, 一定要是\\分隔
  , "clsId":"A5DA6E97-1D4C-4708-B705-84A45716B4DD" //
  可选, reg= true 检查是否已经注册
  , "AuthKey":"A1-ZneY-2qGoXRfc7h6GZZxBB2gceORBwyhoxsA6GK5agLtIAwLhh6BnK61W8fORVN
  v"
}
```

4.6.3.6 IE 内核打开网页

使用 IE 打开网页默认加执行了一次刷新(noRefresh=1 可以取消)。

```
{"emit":"open","Obj":"web","AppType":1,"pos":1,"par":{"htmlStr":null,"HttpServer_startUrl":null,"UR
```

```
L":{"http://www.appemit.com"},"par0":{"header":null,"noScriptErr":true,"UIFLAG":null,"DLCTL":null,"userAgent":null,"crossDomain":true,"rightMenu":null}}
```

名称	设置	含义	说明
emit	open	必需。打开网页事件请求。	
Obj	web	必需。	
AppType	1	必需。 1 IE 内核 2 webkit 内核 3 blink 内核	有默认的右键菜单。 若为负数-1,则是浮动窗口
pos	{"left":372,"top":203,"width":606,"height":406}	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器，自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
par		必需。优先级依次下降	三个参数必须有一个不是空。
htmlStr		Html 代码	
HttpServer_startUrl		以服务器形式打开本地 html 文件路径，可以是绝对或者相对路径。/ 为分隔符。	
URL	http://www.appemit.com 或者 /demo/html Demo/html.html	支持网页地址或者本地 html 文件路径。	
Par0		可选。	
emulation	若不指定，默认最新的 IE 兼容版本	11001 页面始终以 IE11 模式显示，而不考虑 <!DOCTYPE> 指令 11000 包含基于标准的 <!DOCTYPE> 指令的页面将以 IE11 模式显示 10001 页面始终以 IE10 模式显示，而不考虑 <!DOCTYPE> 指令 10000 包含基于标准的 <!DOCTYPE> 指令的页面将以 IE10 模式显示 9999 页面始终以 IE9 模式显示，而不考虑 <!DOCTYPE> 指令 9000 包含基于标准的 <!DOCTYPE> 指令的页面将以 IE9 模式显示 8888 页面始终以 IE8 模式显示，而不考虑 <!DOCTYPE> 指令 8000 包含基于标准的 <!DOCTYPE> 指令的页面将以 IE8 模式显示 7000 包含基于标准的 <!DOCTYPE>	参考

		指令的页面将以 IE7 模式显示	
noRefresh	0	默认 0 在等待 doc 加载完执行一次 refresh() 1 则在等待 doc 加载完不执行 refresh()	用户中途更改了网页，可以使用默认 0，这样可以刷新。或者使用右键用户自己刷新，或者发送 runCmd 的 refresh() 命令
noScriptErr	bool	默认 true True false	
header		所有的头部	对象或字符串 可包含 session "header": { "Accept-Encoding": "gzip", "Cookie": "a=1" } "header": "Accept-Encodin g: gzip, deflate\r\nAccept: image/jpeg \r\nAccept-Language: zh-CN"
go		go(URL, headers, target, flags)	使用 AppObject.web.go 命令可以对每次的加载细化参数。
wait	2000	null 默认 2000 毫秒 0 则一直等待完全打开 或者具体的毫秒数	如果有 cookieSet, 则等待 wait 尝试输入 cookie
userAgent		代理	默认 null
crossDomain	bool	默认 true True false	
cookieSet		"name=value;expires=..GMT;"	本地浏览器中的 cookie 多个记录 \r\n, \r, \n 换行, 默认 "wait": 2000 毫秒等待网页尝试打开赋值 cookieSet, 如果网页没有在 wait 秒内打开可能赋值不上。还可以发送 runCmd AppJsObject.web.document.cookie="a=5;expires=..." 一条一条赋值。本地浏览器设置
rightMenu	右键菜单	-1 不处理, 控件原有状态 0/null 禁用菜单 1 自定义简单菜单	各个含义不同, 请注意。 -1 不处理。默认有详细菜单。通常需要使用 flash 右键使用此值。 对于 flash 右键功能请测试。

UIFLAG	<pre> _UIFLAG_DIALOG=@0x1/*_UIFLAG_DIALOG*/ _UIFLAG_DISABLE_HELP_MENU=@0x2/*_UIFLAG_DISABLE_HELP_MENU*/ _UIFLAG_NO3DBORDER=@0x4/*_UIFLAG_NO3DBORDER*/ _UIFLAG_SCROLL_NO=@0x8/*_UIFLAG_SCROLL_NO*/ _UIFLAG_DISABLE_SCRIPT_INACTIVE=@0x10/*_UIFLAG_DISABLE_SCRIPT_INACTIVE*/ _UIFLAG_OPENNEWWIN=@0x20/*_UIFLAG_OPENNEWWIN*/ _UIFLAG_DISABLE_OFFSCREEN=@0x40/*_UIFLAG_DISABLE_OFFSCREEN*/ _UIFLAG_FLAT_SCROLLBAR=@0x80/*_UIFLAG_FLAT_SCROLLBAR*/ _UIFLAG_DIV_BLOCKDEFAULT=@0x100/*_UIFLAG_DIV_BLOCKDEFAULT*/ _UIFLAG_ACTIVATE_CLIENTHIT_ONLY=@0x200/*_UIFLAG_ACTIVATE_CLIENTHIT_ONLY*/ _UIFLAG_OVERRIDEBEHAVIORFACTORY=@0x400/*_UIFLAG_OVERRIDEBEHAVIORFACTORY*/ _UIFLAG_CODEPAGELINKEDFONTS=@0x800/*_UIFLAG_CODEPAGELINKEDFONTS*/ _UIFLAG_URL_ENCODING_DISABLE_UTF8=@0x1000/*_UIFLAG_URL_ENCODING_DISABLE_UTF8*/ _UIFLAG_URL_ENCODING_ENABLE_UTF8=@0x2000/*_UIFLAG_URL_ENCODING_ENABLE_UTF8*/ _UIFLAG_ENABLE_FORMS_AUTOCOMplete=@0x4000/*_UIFLAG_ENABLE_FORMS_AUTOCOMplete*/ _UIFLAG_ENABLE_INPLACE_NAVIGATION=@0x10000/*_UIFLAG_ENABLE_INPLACE_NAVIGATION*/ _UIFLAG_IME_ENABLE_RECONVERSION=@0x20000/*_UIFLAG_IME_ENABLE_RECONVERSION*/ _UIFLAG_THEME=@0x40000/*_UIFLAG_THEME*/ _UIFLAG_NOTHEME=@0x80000/*_UIFLAG_NOTHEME*/ </pre>	<p>可以使用一个或多个 _UIFLAG_ 前缀的常量自定义外观,多个常量之间用位或操作符() 连接。</p> <p>LOG 禁止选中文本(用于 web ui)</p> <p>_UIFLAG_SCROLL_NO 禁用滚动条</p> <p>_UIFLAG_NO3DBORDER 禁用所有窗口 3D 边框</p> <p>_FLAG_NO3DOUTERBORDER 禁用顶层窗口 3D 边框</p> <p>_UIFLAG_DISABLE_HELP_MENU 在菜单中移除帮助菜单</p> <p>_UIFLAG_DISABLE_SCRIPT_INACTIVE 窗口激活以前不运行网页脚本</p> <p>_UIFLAG_OPENNEWWIN 在新窗口打开链接</p> <p>_UIFLAG_FLAT_SCROLLBAR 显示平面滚动条</p> <p>_UIFLAG_ACTIVATE_CLIENTHIT_ONLY 仅在用户点击客户区时激活(非客户区指滚动条等位置)</p> <p>_UIFLAG_URL_ENCODING_DISABLE_UTF8 禁用 UTF8 发送 URL</p> <p>_UIFLAG_URL_ENCODING_ENABLE_UTF8 使用 UTF8 发送 URL</p> <p>_UIFLAG_ENABLE_FORMS_AUTOCOMplete 允许表单自动完成</p> <p>_UIFLAG_ENABLE_INPLACE_NAVIGATION 在点击邮件等链接时, 打开相关应用程序, 而不是新窗口</p> <p>_UIFLAG_NOTHEME 使用主题</p> <p>_UIFLAG_THEME 禁用主题</p> <p>_UIFLAG_NOPICS 禁用内容分级</p> <p>_UIFLAG_DIV_BLOCKDEFAULT 编辑模式回车输入 div</p> <p>_UIFLAG_DISABLE_EDIT_NS_FIXUP 编辑模式禁用名字空间修正</p> <p>_UIFLAG_LOCAL_MACHINE_ACCESS_CHECK 防止远程网页导航到本地计算机</p> <p>_UIFLAG_DISABLE_UNTRUSTEDPROTOCOL 禁止非信任协议,包含 ms-its, ms-itss,</p>
--------	---	--

		_UIFLAG_NOPICS=@0x100000/*_UI FLAG_NOPICS*/ _UIFLAG_NO3DOUTERBORDER=@0x 200000/*_UIFLAG_NO3DOUTERBOR DER*/ _UIFLAG_DISABLE_EDIT_NS_FIXUP= @0x400000/*_UIFLAG_DISABLE_EDI T_NS_FIXUP*/ _UIFLAG_LOCAL_MACHINE_ACCESS _CHECK=@0x800000/*_UIFLAG_LOC AL_MACHINE_ACCESS_CHECK*/ _UIFLAG_DISABLE_UNTRUSTEDPROT OCOL=@0x1000000/*_UIFLAG_DISA BLE_UNTRUSTEDPROTOCOL*/ _UIFLAG_HOST_NAVIGATES=@0x20 00000/*_UIFLAG_HOST_NAVIGATES */ _UIFLAG_ENABLE_REDIRECT_NOTIFI CATION=@0x4000000/*_UIFLAG_EN ABLE_REDIRECT_NOTIFICATION*/ _UIFLAG_USE_WINDOWLESS_SELEC TCONTROL=@0x8000000/*_UIFLAG _USE_WINDOWLESS_SELECTCONTR OL*/ _UIFLAG_USE_WINDOWED_SELECTC ONTROL=@0x10000000/*_UIFLAG_ USE_WINDOWED_SELECTCONTROL* / _UIFLAG_ENABLE_ACTIVEX_INACTIV ATE_MODE=@0x20000000/*_UIFLA G_ENABLE_ACTIVEX_INACTIVATE_M ODE*/ _UIFLAG_DPI_AWARE=@0x4000000 0/*_UIFLAG_DPI_AWARE*/	its,mk:@msitstore
DLCTL		_DLCTL_DLIMAGES=@0x10/*_DLCTL _DLIMAGES*/ _DLCTL_VIDEOS=@0x20/*_DLCTL_VI DEOS*/ _DLCTL_BGSOUNDS=@0x40/*_DLCT L_BGSOUNDS*/ _DLCTL_NO_SCRIPTS=@0x80/*_DLC TL_NO_SCRIPTS*/ _DLCTL_NO_JAVA=@0x100/*_DLCTL _NO_JAVA*/ _DLCTL_NO_RUNACTIVEXCTLS=@0x 200/*_DLCTL_NO_RUNACTIVEXCTLS */ _DLCTL_NO_DLACTIVEXCTLS=@0x40 0/*_DLCTL_NO_DLACTIVEXCTLS*/	DLCTL_前缀的常量以控制下 载行为,多个常量之间用位或 操作符() 连接。 _DLCTL_DLIMAGES 允许从服 务器下载图片,如果指定了第 三个参数,未指定此标志,则 网页不下载任何图片。 _DLCTL_VIDEOS 允许从服 务器下载视频片断,如果指定 了第三个参数,未指定此标 志,则网页不下载任何视频片 断。 _DLCTL_BGSOUNDS 允许 播放文档指定的背景声音 _DLCTL_NO_SCRIPTS web 窗体不执行任何页面脚本(指

		<pre> _DLCTL_DOWNLOADONLY=@0x800/ *_DLCTL_DOWNLOADONLY*/ _DLCTL_NO_FRAMEDOWNLOAD=@ 0x1000/*_DLCTL_NO_FRAMEDOWN LOAD*/ _DLCTL_RESYNCHRONIZE=@0x2000/ *_DLCTL_RESYNCHRONIZE*/ _DLCTL_PRAGMA_NO_CACHE=@0x 4000/*_DLCTL_PRAGMA_NO_CACH E*/ _DLCTL_NO_BEHAVIORS=@0x8000/ *_DLCTL_NO_BEHAVIORS*/ _DLCTL_NO_METACHARSET=@0x10 000/*_DLCTL_NO_METACHARSET*/ _DLCTL_URL_ENCODING_DISABLE_U TF8=@0x20000/*_DLCTL_URL_ENC ODING_DISABLE_UTF8*/ _DLCTL_URL_ENCODING_ENABLE_U TF8=@0x40000/*_DLCTL_URL_ENC ODING_ENABLE_UTF8*/ _DLCTL_NOFRAMES=@0x80000/*_D LCTL_NOFRAMES*/ _DLCTL_FORCEOFFLINE=@0x100000 00/*_DLCTL_FORCEOFFLINE*/ _DLCTL_NO_CLIENTPULL=@0x20000 000/*_DLCTL_NO_CLIENTPULL*/ _DLCTL_SILENT=@0x40000000/*_DL CTL_SILENT*/ _DLCTL_OFFLINEIFNOTCONNECTED= @0x80000000/*_DLCTL_OFFLINEIFN OTCONNECTED*/ _DLCTL_OFFLINE=@0x80000000/*_ DLCTL_OFFLINE*/ </pre>	<p>javascript 等)</p> <p>_DLCTL_NO_JAVA web 窗体不执行任何 Java applet</p> <p>_DLCTL_NO_RUNACTIVEXCTLS web 窗体不执行文档中的任何 ActiveX 控件;</p> <p>_DLCTL_NO_DLACTIVEXCTLS web 窗体不下载文档中的任何 ActiveX 控件;</p> <p>_DLCTL_DOWNLOADONLY web 窗体下载网页,但不显示</p> <p>_DLCTL_NO_FRAMEDOWNLOAD web 窗体对包含框架的页面进行语法分析但不下载任何帧, 同时忽略框架,</p> <p>_DLCTL_RESYNCHRONIZE web 窗体忽略缓存中的数据并向服务器请求更新</p> <p>_DLCTL_PRAGMA_NO_CACHE 迫使请求发送给服务器并忽略代理(这里一般指服务端缓存), 即使代理指明数据是最新的也是如此。</p> <p>_DLCTL_NO_METACHARSET 隐藏文档中的 META 元素指示的字符集;</p> <p>_DLCTL_URL_ENCODING_DISABLE_UTF8 禁止 UTF-8 编码</p> <p>_DLCTL_URL_ENCODING_ENABLE_UTF8 允许 UTF-8 编码</p> <p>_DLCTL_NOFRAMES 禁止框架</p> <p>_DLCTL_FORCEOFFLINE web 窗体工作在脱机方式</p> <p>_DLCTL_NO_CLIENTPULL web 窗体不执行任何客户端的 pull 操作</p> <p>_DLCTL_SILENT 组件对话框、脚本错误对话框静默模式</p> <p>_DLCTL_OFFLINEIFNOTCONNECTED 如果未连接互联网, 浏览器组件将以脱机方式工作</p>
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	

4.6.3.7 IE 内核打开含有 flash_ocx 网页

Flash_ocx 首次使用，而本机未安装则自动安装，win7 需要管理员权限，有黑窗提示。

```
{"emit":"open","Obj":"web","AppType":1,"AppId":1,"pos":1,"par":{"URL":"http://sxiao.4399.com/4399swf/upload_swf/ftp14/yzg/20140328/bombit7/zx_game7.htm","plugins":"flash_ocx"},"par0":{"rightMenu":-1}}
```

增加"plugins":"flash_ocx"，即可引入。

4.6.3.8 IE 内核启动前设置 cookie 或者清除 clearCookie

//在启动浏览器 IE 内核前，可以清除 cookie "clearCookie":"baidu.com", (匹配，没有 http)，然后可以设置 cookie,必须有域名 cookieDomain="http://baidu.com"，默认 expires=7 天，或者标准 cookie 时间格式。

//inet_setCookie 格式为 a=1;b=1; c=1，默认以 cookie_split="<|>"来分隔

```
ReqPar421=
{"emit":"open","Obj":"web","AppType":1,"AppId":1,"pos":1,"par":{"URL":"http://www.baidu.com"},"par0":{"clearCookie":"baidu.com","cookieDomain":"http://baidu.com","inet_setCookie":"a=5;b=6;c=7","AppStep":{"init":null,"loaded":"AppJsObject.web.waitDoc(); return AppJsObject.web.document.cookie"}}}
```

4.6.3.9 IE 内核启动后设置 cookie 或者清除 clearCookie

必须等待 doc 加载完成

```
{"emit": "runCmd", "Obj": "web", "AppId": 1, "codeStr": '
AppJsFun.inet.clearCookie("baidu.com");AppJsObject.web.document.cookie="c=3";AppJsObject.web.document.cookie="b=6";return AppJsObject.web.document.cookie;'};
```

4.6.3.10 webkit 内核打开网页

```
{"emit":"open","Obj":"web","AppType":2,"pos":1,"par":{"htmlStr":null,"HttpServer_startUrl":null,"URL":"http://www.appemit.com"},"par0":{"header":null,"userAgent":null,"crossDomain":true,"rightMenu":1,"downLoad":1}}
```

名称	设置	含义	说明
emit	open	必需。打开网页事件请求。	
Obj	web	必需。	
AppType	2	必需。 1 IE 内核 2 webkit 内核 3 blink 内核	有默认的右键菜单。 若为负数-2,则是浮动窗口
pos	{"left":372,"top":203,"width":606,"height":406}	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器，自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false

AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
par		必需。优先级别依次下降	三个参数必须有一个不是空。
htmlStr		Html 代码	
HttpServer_startUrl		以服务器形式打开本地 html 文件路径，可以是绝对或者相对路径。/ 为分隔符。	
URL	http://www.appemit.com 或者 /demo/html Demo/html.html	支持网页地址或者本地 html 文件路径。	
Par0		可选。	
header		头部	对象或字符串 "header": { "Accept-Encoding": "gzip", "Cookie": "a=1" } "header": "Accept-Encoding: gzip, deflate\r\nAccept: image/jpeg\r\nAccept-Language: zh-CN"
userAgent		代理	只写属性，不可读。
crossDomain	bool	默认 true ture false	是否跨域
cookieEnabled	true		默认 true
proxy		{ "proxy": { "type": "HTTP"; "hostname": "127.0.0.1"; "port": 8080; "username": ""; "password": ""; } }	
cookieSet		Js cookie 标准格式	
rightMenu	右键菜单	-1 不处理，控件原有状态 0/null 禁用菜单 1 自定义简单菜单	各个含义不同，请注意。 -1 不处理。默认没有菜单。通常需要使用 flash 右键使用此值。 对于 flash 右键功能请测试。

invoke	flase	true 支持跨线程打开 false 默认单线程打开	
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	
download	1	默认 true true false	支持连接下载功能

4.6.3.11 webkit 内核打开 flash_NP 网页

```
{"emit":"open","Obj":"web","AppType":2,"AppId":1,"pos":1,"par":{"URL":"http://sxiao.4399.com/4399swf/upload_swf/ftp14/yzg/20140328/bombit7/zx_game7.htm","plugins":["flash_NP"]},"par0":{"rightMenu":-1}}
```

增加"plugins":"flash_NP"，即可引入。

4.6.3.12 blink 内核打开网页

```
{"emit":"open","Obj":"web","AppType":3,"pos":1,"par":{"htmlStr":null,"HttpServer_startUrl":null,"URL":"http://www.appemit.com"},"par0":{"header":null,"userAgent":null,"crossDomain":true,"rightMenu":1,"download":1}}
```

名称	设置	含义	说明
emit	open	必需。打开网页事件请求。	
Obj	web	必需。	
AppType	3	必需。 1 IE 内核 2 webkit 内核 3 blink 内核	有默认的右键菜单。 若为负数-3,则是浮动窗口
pos	{"left":372,"top":203,"width":606,"height":406}	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器，自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
par		必需。优先级别依次下降	三个参数必须有一个不是空。
htmlStr		Html 代码	
HttpServer_s		以服务器形式打开本地 html 文件	

targetUrl		路径，可以是绝对或者相对路径。/ 为分隔符。	
URL	http://www.appemit.com 或者 /demo/html Demo/html.html	支持网页地址或者本地 html 文件路径。	
Par0		可选。	
header		头部	对象或字符串 "header": { "Accept-Encoding": "gzip", "Cookie": "a=1" } "header": "Accept-Encoding: gzip, deflate\r\nAccept: image/jpeg\r\nAccept-Language: zh-CN"
userAgent		代理	可读写。默认 Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3729.169 Safari/537.36
crossDomain	bool	默认 true true false	
Accept-Language	默认 zh-CN,zh;q=0.8	其中 q 是权重	
rightMenu	右键菜单	-1 不处理，控件原有状态 0/null 禁用菜单 1 自定义简单菜单	各个含义不同，请注意。 -1 不处理。默认有菜单。通常需要使用 flash 右键使用此值。 对于 flash 右键功能请测试。
cookieSet		Js cookie 标准格式	
invoke	false	true 支持跨线程打开 false 默认单线程打开	
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	
download	1	默认 true true false	支持连接下载功能

4.6.3.13 bink 内核打开 flash_NP 网页

```
{"emit":"open","Obj":"web","AppType":3,"AppId":1,"pos":1,"par":{"URL":"http://sxiao.4399.com/4399swf/upload_swf/ftp14/yzg/20140328/bombit7/zx_game7.htm","plugins":["flash_NP"]},"par0":{"rightMenu":-1}}
```

增加"plugins":"flash_NP"，即可引入。

4.6.4 打开媒体文件

4.6.4.1 视频

<https://github.com/videojs/video.js>

支持下列格式。其中 swf 格式见 flash 章节。

视频格式：mp4、flv、m3u8、rtmp

视频编码：H.264

音频编码：AAC、MP3

音频格式：MP3

```
{"emit":"open","Obj":"flash","AppType":3,"src":"https://media.html5media.info/video.mp4","pos":1,"par0":{"autoplay":1,"loop":1}}
```

4.6.4.2 libvlc 插件播放(支持 RTSP RTMP...)

使用 vlc 播放，几乎所有格式，包括本地媒体。

```
[{"emit":"open","Obj":"libvlc","AppType":1,"pos":1,"AppId":1,"par":{"mrl":"rtsp://wowzaec2demo.steamlock.net/vod/mp4:BigBuckBunny_115k.mov","localFile":0},"par0":{"fullscreen":true,"volume":70,"autoplay":true,"controls":true,"snapPath":"/file/snap/","snapFormat": ".bmp","snapWidth":0,"snapHeight":0}}, {"emit":"open","Obj":"libvlc","AppType":1,"pos":1,"AppId":2,"par":{"mrl":"rtsp://@202.69.69.180:443/webcast/bshdlive-pc","localFile":0},"par0":{"fullscreen":1,"volume":70,"autoplay":1,"controls":0}}]
```

左键双击全屏，右键双击截图。

4.6.4.3 Libvlc 播放监控事件

```
//
"event":["click","Rclick","resume","play","pause","stop","end","volume","mute","fullscreen","Error"]
//监控左右单击事件，右键双击截图 左键双击全屏
//可以通过回调函数来处理对应的注册事件
//先定义回调函数
callback1= function(Jdata,inPar){
    console.log("call balc 1",Jdata,inPar) //业务处理，play 要考虑 resume 之后也会有一个 play
    if (Jdata.data && Jdata.data.first) {
        console.log("第一次播放成功")
    }
}

function regEvent(){
```

```

var
ReqPar11={"emit":"open","Obj":"libvlc","AppType":1,"pos":1,"AppId":1,"par":{"mrl":"/demo/htmlDe
mo/file/h1.mp4","localFile":1},"par0":{"volume":70,"autoplay":1,"controls":1,"event":["click","Rclick",
"resume","play","pause","stop","end","volume","mute","fullscreen","Error"]}}

    AE.callbackFunc= [{"equ": { "Obj":"libvlc",   "event": "play" ,"rep": 0},"func":callback1,
"inPar":"inPar 111"}] ;

    AE.OpenApp(ReqPar11);
}

```

4.6.5 直播 RTSP、RTMP

4.6.5.1 RTMP

```

rtmp
{"emit":"open","Obj":"flash","AppType":3,"src":"rtmp://202.69.69.180:443/webcast/bshdlive-pc","po
s":1,"par0":{"autoPlay":1}}

```

4.6.5.2 RTSP

可采用 vlc 的 libvlc 库插件播放，默认 vlc 插件为 2.2.5.1，vlc 的 NPAPIwebkit 网页播放，rtsp 转化为 webRTC 浏览器直接播放，rtsp 转化为 ogg 浏览器直接播放。

4.6.5.2.1 libvlc 插件播放 RTSP，支持多开

支持多开，使用网页的 vlc 插件直接打开 Rtsp 流，见案例 demo\htmlDemo\rtsp.html

```

{"emit":"open","Obj":"libvlc","AppType":1,"pos":1,"AppId":1,"par":{"mrl":"rtsp://wowzaec2demo.st
reamlock.net/vod/mp4:BigBuckBunny_115k.mov","localFile":0},"par0":{"fullscreen":1,"volume":70,
"autoplay":1,"controls":0}}

```

4.6.5.2.2 NPAPI 插件播放 RTSP,在一个 html 设置多个 Object 就可实现多开

使用 webkit 网页的 vlc 插件直接打开 Rtsp 流，可以使用 npvlc.dll 即 NPAPI 技术 或者 axvlc.dll
见案例 demo\htmlDemo\rtsp.html

更多参考 <https://wiki.videolan.org/Documentation:WebPlugin/>

1. 设置 mrl 等效 实现 html，插件直接播放

```

{"emit":"open","Obj":"media","AppType":1,"pos":1,"AppId":1,
"par":{"mrl":"rtsp://@202.69.69.180:443/webcast/bshdlive-pc","htmlStr":null,"HttpServer_startUrl":
null,"kernel":2},"par0":{"fullscreen":true,"volume":70,"autoplay":true,"controls":true}}

```

2. mrl 其等效实现 htmlStr 相同的代码。

```

var    htmlStr=AE.txt2code(function(){/*
<object type='application/x-vlc-plugin' id='vlc' events='True' width='100%' height='300'>
    <param name='mrl'
value='rtsp://wowzaec2demo.streamlock.net/vod/mp4:BigBuckBunny_115k.mov' />
    <param name='volume' value='50' />
    <param name='autoplay' value='true' />
    <param name='loop' value='false' />
    <param name='fullscreen' value='false' />
</object>
*/});

```

```
{"emit":"open","Obj":"media","AppType":1,"pos":1,"AppId":1,"par":{"htmlStr":htmlStr,"HttpServer_startUrl":null}} }
```

3. 或者将 htmlStr 保存到本地文件，定义

```
"HttpServer_startUrl":"demo/htmlDemo/rtsp_HttpServer_startUrl.html"
ReqPar=
{"emit":"open","Obj":"media","AppType":1,"pos":1,"AppId":1,"par":{"htmlStr":null,"HttpServer_startUrl":"demo/htmlDemo/rtsp_HttpServer_startUrl.html"}} }
```

如果同时定义 3 个变量， 优先级为 htmlStr > HttpServer_startUrl > mrl

使用 NPAPI，直接在 html 定义多个 rtsp 就可以实现多开。

4.6.5.2.3 rtsp 转化为 webRTC 浏览器直接播放-原生播放

只能开播放一个 RTSP，能切换。

支持个格式有要求 audio codec ALAW or MULAW

将 RTSP 转为 webRTC，，适用延迟精度要求较高的场所，这样只是使用了 webRTC 后台服务而没有浏览器插件。

见案例 demo\htmlDemo\rtsp2WebRTC.html

force=1 表示强制关闭已经打开的 App，重新打开

```
Var ReqPar=
{"emit":"open","Obj":"rtsp2webRTC","AppShow":0,"AppFollow":0,"AppRuntime":1,"par":{"webRTC_cfg":"webRTC_cfg","pid":"pid","webRTC_dos":0,"AuthKey":"pidAuthKey","par0":{"force":1}};
```

```
AE.OpenApp(ReqPar );
```

4.6.5.2.4 rtsp 转化为 ogg 浏览器直接播放-原生播放

使用 Vlc 提供服务，转化编码，有延迟，适用精度要求不高的场所，这样只是使用了后台服务而没有插件。

见案例 demo\htmlDemo\rtsp_ogg.html

force=1 表示强制关闭已经打开的 App，重新打开

举例：将 avi 转化为 rtsp(如果有 rtsp 直接使用),在将 rtsp 转化为 ogg，在 html5 的播放器播放

```
{"emit":"open","Obj":"media","AppType":1,"AppId":"AppId_lvcSvr1","AppShow":0,"par":{"gui":0,"cmds":["-vvv \\"file:///\\$\\{dir_Cur}\\demo\\htmlDemo\\file\\h1.mp4\\" --loop --sout \\"#transcode{vcodec=h264,acodec=mpga,ab=128,channels=2,samplerate=44100}:rtp{sdp=rtsp://:8554/vlc}\\",\\"-vvv \\"rtsp://:8554/vlc\\" --loop :sout=#transcode{vcodec=theo,vb=800,acodec=vorb,ab=128,channels=2,samplerate=44100}:http{mux=ogg,dst=:8080/stream}:sout-all :sout-keep\\""]} }
```

{"emit":"getPar","Obj":"app","par":["intranetIP"]} 获得内网 IP 192.168.1.105

示例如下

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title></title>
```

```

</head>
<body>
  <video src="http://192.168.1.105:8080/stream" type="video/ogg" width="600"
height="450" autoplay="autoplay" controls="controls" loop="loop">
Your browser does not support the video tag.

</video>
</body>
</html>

```

4.6.5.2.5 其它功能

4.6.5.2.5.1 转码直接播放

```

{"emit":"open","Obj":"media","AppType":1,"pos":1,
"par":{"gui":0,"cmds":"-vvv  \file:///${dir_Cur}/demo/htmlDemo/file/h1.mp4\" --loop --sout
\"#transcode{vcodec=h264,acodec=mpga,ab=128,channels=2,samplerate=44100}:rtp{sdp=rtsp://:8554
/vlc}\" },
"mrl":"rtsp://:8554/vlc","htmlStr":null,"kernel":2},"par0":{"
"fullscreen":true,"volume":70,"autoplay":true,"controls":true }}

```

4.6.5.2.5.2 转码等服务

启动服务，可以多开，执行转码等服务，可再播放，最好使用命令组发送，确保前一条命令执行完。

```

[{"emit":"open","Obj":"media","AppType":1,"AppId":"AppId_lvcSvr1","AppShow":0,"par":{"gui":0,
"cmds":"-vvv  \file:///${dir_Cur}/demo/htmlDemo/file/h1.mp4\" --loop --sout
\"#transcode{vcodec=h264,acodec=mpga,ab=128,channels=2,samplerate=44100}:rtp{sdp=rtsp://:8554
/vlc}\" }},
{"emit":"open","Obj":"media","AppType":1,"pos":1,"AppId":1,
"par":{"mrl":"rtsp://:8554/vlc","htmlStr":null,"kernel":2},"par0":{"
"fullscreen":true,"volume":70,"autoplay":true,"controls":true }}}]

```

4.6.5.2.5.3 其它说明

可用的替换路径变量

1. `${dir_AppData_Local}`: "C:/Users/*/AppData/Local"
2. `${dir_AppData_Roaming}`: "C:/Users/*/AppData/Roaming"
3. `${dir_DESKTOP}`: "F:/Users/desk"
4. `${dir_DOCUMENTS}`: "C:/Users/Public/Documents"
5. `${dir_FAVORITES}`: "C:/Users/*/Favorites"
6. `${dir_Temp}`: "C:/Users/*/AppData/Local/Temp/"
7. `${dir_Cur}`: "D:/*/AppEmit"

4.6.6 打开 PDF 文件

打开阅读 PDF 文件，支持本地和网络 PDF

```
{ "emit": "open", "Obj": "web", "AppType": 4, "pos": 1, "par": { "URL": "https://cdn.jsdelivr.net/gh/appemit/appemitweb@master/docs/demo/AppEmit_help.pdf", "par0": { "rightMenu": 1 } }
```

名称	设置	含义	说明
emit	open	必需。打开 PDF 事件请求。	
Obj	web	必需。	
AppType	4	必需。 1 IE 内核 2 webkit 内核 3 blink 内核 4 PDF 文件	有默认的右键菜单。 若为负数-4,则是浮动窗口
pos	{ "left": 372, "top": 203, "width": 606, "height": 406 }	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器，自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
par		必需。	
URL	https://cdn.jsdelivr.net/gh/appemit/appemitweb@master/docs/demo/AppEmit_help.pdf 或者 /demo/AppEmit_help.pdf	支持网页 pdf 地址,或本地文件绝对或相对路径。	
Par0		可选。	
rightMenu	右键菜单	-1 不处理，控件原有状态 0/null 禁用菜单 1 自定义简单菜单	各个含义不同，请注意。
rightMenu	右键菜单	-1 不处理，控件原有状态 0/null 禁用菜单 1 自定义简单菜单	各个含义不同，请注意。 -1 不处理。默认有菜单。通常需要使用 flash 右键使用此值。 对于 flash 右键功能请测试。

show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	

4.6.7 Office

创建、打开阅读、编辑 Office 文件，支持本地和网络文件，包括 word\excel\PPT 文件
更多 VBA 官方参考 <https://docs.microsoft.com/zh-cn/office/vba/api/overview/word>
或者录制宏，编辑参考代码。

4.6.7.1 msOffice

```
{
  "emit": "open",
  "Obj": "office",
  "AppType": 1,
  "src": "/demo/htmlDemo/file/d1.docx",
  "pos": 1,
  "par0": {
    "Titlebar": 0,
    "ProtectDoc": 0,
    "Menubar": 0,
    "Toolbars": 0,
    "btnFile": 0,
    "HttpPost": {
      "URL": "http://eu.httpbin.org/post",
      "valid": 0,
      "fileName": "s1.docx"
    },
    "MNU_NEWBLANK": 0,
    "MNU_NEW": 0,
    "MNU_OPEN": 0,
    "MNU_CLOSE": 0,
    "MNU_SAVE": 0,
    "MNU_SAVEAS": 0,
    "MNU_PGSETUP": 0,
    "MNU_PRINT": 0,
    "MNU_PROPS": 0,
    "show_UpdateTool": true
  }
}
```

名称	设置	含义	说明
emit	open	必需。打开 office 事件请求。	
Obj	office	必需。	
AppType	1	必需。 1 word 2 excel 3 powerpoint	有默认的右键菜单。 若为负数-1,则是浮动窗口 对于特殊 wps，通常采用 attach 方式打开。
src	字符串； path 为 [path, openPar]	path 为必需，字符串网址或者本地相对或者绝对路径 ,或者数组（防止网页 src 里面没有文件名称或者后缀名称需要修改，则使用 fileType）。 openPar: 具体不同。 "/demo/htmlDemo/file/x1.xls" ["/demo/htmlDemo/file/x1.xls",openPar] [[http://.../demo/htmlDemo/file/x1.xls,"x2.xlsx"],openPar]	Word 系列文件 Excel 系列文件 PPT 系列文件 不同应用的 openPar 不同，详细见后
pos	{"left":372,"top":203,"width":606,"height":406}	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器，自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时，App 随浏览器移动变形隐藏显示等事件
Par0		可选。	
progId	[]	默认为 MS office	采用不同自动化服务器

		["Word.Application","Kwps.Applicati on","WPS.Application"] ["Excel.Application","Ket.Application ","ET.Application"] ["PowerPoint.Application","WPP.App lication"]	打开文档、表格、幻灯片， 按顺序依次尝试。 默认打开微软 office\金山 wps
attach	1	附着 APP 方式	wps 最好采用此方式。 如果程序出现未知错误， 可采用此方式
noClose	true	默认 true 屏蔽关闭功能 false 不屏蔽关闭功能	
noMinMax	1	1 禁用关闭 最小 最大化按钮 0/null 可最小化 关闭	遮盖关闭 最小 最大化 按钮
openPar			不同应用不一样，详细见 后
Titlebar	0	默认 0 不显示标题栏 1 显示标题栏 0 不显示	
Menubar	1	默认 1 显示菜单栏 1 显示标题栏 0 不显示	
Statusbar	1	默认 1 显示状态栏 1 显示状态栏 0 不显示	
Protect	1		不同应用不一样，详细见 后
Toolbars	1	默认 1 ， 显示工具栏 1 显示 0 不显示	需要手动点击
btnFile	1	工具栏的文件 File 按钮 1/true 默认显示 0	Toolbars 为真时有效
ViewType	1		不同应用不一样，详细见 后
HttpPost	{ "URL": "url" , "valid": 1, "method": " POST" "username" : "name" "username" : "*" " "cookie": nul l }	URL 上传的地址，必需 method 上传的方法 POST\GET,默 认 POST cookie	如果上传，URL 必 需,valid=1 设为有效。
header		头部	以下参数用于下载上传 使用

userAgent		代理	默认 null
cookies			
proxy			
proxyBypass			
flags			
crossDomain	bool	默认 true True false	
MNU_UPLOAD	0	默认 0，工具栏上传命令菜单 1 显示上传提交 0 不显示	Toolbars、btnFile、HttpPost["URL"] 同时为真时有效
MNU_NEWBLANK	0	默认 0，工具栏 FILE 命令菜单 1 显示新建空白 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_NEW	0	默认 0，工具栏 FILE 命令菜单 1 显示新建 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_OPEN	0	默认 0，工具栏 FILE 命令菜单 1 显示打开 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_CLOSE	0	默认 0，工具栏 FILE 命令菜单 1 显示关闭 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_SAVE	0	默认 0，工具栏 FILE 命令菜单 1 显示保存 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_SAVEAS	0	默认 0，工具栏 FILE 命令菜单 1 显示另存 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_PGSETUP	0	默认 0，工具栏 FILE 命令菜单 1 显示页面设置 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_PRINT	0	默认 0，工具栏 FILE 命令菜单 1 显示打印 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_PROPS	0	默认 0，工具栏 FILE 命令菜单 1 显示属性 0 不显示	Toolbars、btnFile 同时为真时有效
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	

4.6.7.2 Word 其它参数

更多 <https://docs.microsoft.com/zh-cn/office/vba/api/overview/word>

名称	设置	含义	说明
Par0		可选。	

src	[path,0,1,0]	ConfirmConversions=false, _ ReadOnly=false, AddToRecentFiles=false, PasswordDocument="", _ PasswordTemplate="", Revert=false, WritePasswordDocument="", _ WritePasswordTemplate="", Format=wdOpenFormatAuto, XMLTransform=""	按顺序																		
ViewType	1	wdMasterView 5 主控视图。 wdNormalView 1 普通视图。 wdOutlineView 2 大纲视图。 wdPrintPreview 4 打印预览视图。 wdPrintView3 页面视图。 wdReadingView 7 阅读视图。 wdWebView 6 Web 视图。																			
Protect	[Type,NoReset,Password,UseIRM, EnforceStyle Lock]	Type 必需 wdAllowOnlyComments=1 只允许将批注添加到文档中。 wdAllowOnlyFormFields=2 只允许通过窗体域将内容添加到文档中。 wdAllowOnlyReading=3 允许对文档进行只读访问。 wdAllowOnlyRevisions=0 只允许对现有内容进行修订。 wdNoProtection=-1 不对文档应用保护。 <table><tr><td>名称</td><td>必需/可选</td><td>说明</td></tr><tr><td>Type</td><td>必需</td><td>wdallowOnlyComments=1 只允许将批注添加到文档中。 wdallowOnlyFormFields=2 只允许通过窗体域将内容添加到文档中。 wdallowOnlyReading=3 允许对文档进行只读访问。 wdallowOnlyRevisions=0 只允许对现有内容进行修订。 wdNoProtection=-1 不对文档应用保护。</td></tr><tr><td>NoReset</td><td>可选</td><td>如果为 False , 则将窗体域重置为其默认值;如果为 True , 则在文档受保护时保留当前的窗体字段值。 如果 _Type_ 不是 wdallowOnlyFormFields, 则 NoReset 将被忽略。</td></tr><tr><td>Password</td><td>可选</td><td>如果提供, 则密码能够编辑文档, 或者更改或删除保护。</td></tr><tr><td>UseIRM</td><td>可选</td><td>指定在保护文档不受更改时是否使用信息权限管理 (IRM)。</td></tr><tr><td>EnforceStyleLock</td><td>可选</td><td>指定是否对受保护的文档强制执行格式设置限制。</td></tr></table>	名称	必需/可选	说明	Type	必需	wdallowOnlyComments=1 只允许将批注添加到文档中。 wdallowOnlyFormFields=2 只允许通过窗体域将内容添加到文档中。 wdallowOnlyReading=3 允许对文档进行只读访问。 wdallowOnlyRevisions=0 只允许对现有内容进行修订。 wdNoProtection=-1 不对文档应用保护。	NoReset	可选	如果为 False , 则将窗体域重置为其默认值;如果为 True , 则在文档受保护时保留当前的窗体字段值。 如果 _Type_ 不是 wdallowOnlyFormFields, 则 NoReset 将被忽略。	Password	可选	如果提供, 则密码能够编辑文档, 或者更改或删除保护。	UseIRM	可选	指定在保护文档不受更改时是否使用信息权限管理 (IRM)。	EnforceStyleLock	可选	指定是否对受保护的文档强制执行格式设置限制。	更改为自行在 officeJs 中控制。后面同 对非保护的文档保护。 网络文件, 不设置则默认为保护文档, [3,null,"random"], 产生随机密码, 密码不可获取, 请慎重处理。 不保护设置为[-1] 对本地文件不默认处理
名称	必需/可选	说明																			
Type	必需	wdallowOnlyComments=1 只允许将批注添加到文档中。 wdallowOnlyFormFields=2 只允许通过窗体域将内容添加到文档中。 wdallowOnlyReading=3 允许对文档进行只读访问。 wdallowOnlyRevisions=0 只允许对现有内容进行修订。 wdNoProtection=-1 不对文档应用保护。																			
NoReset	可选	如果为 False , 则将窗体域重置为其默认值;如果为 True , 则在文档受保护时保留当前的窗体字段值。 如果 _Type_ 不是 wdallowOnlyFormFields, 则 NoReset 将被忽略。																			
Password	可选	如果提供, 则密码能够编辑文档, 或者更改或删除保护。																			
UseIRM	可选	指定在保护文档不受更改时是否使用信息权限管理 (IRM)。																			
EnforceStyleLock	可选	指定是否对受保护的文档强制执行格式设置限制。																			
Unprotect	pwd	文档解锁密码	对保护的文档(type!=-1)输入解锁密码 Unprotect 不保护。																		

4.6.7.3Excel 其它参数

更多 <https://docs.microsoft.com/zh-cn/office/vba/api/overview/excel>

名称	设置	含义	说明
Par0		可选。	

src	[path, openPar]	(UpdateLinks、ReadOnly、Format、Password、WriteResPassword、IgnoreReadOnlyRecommended、原产地、定界符、可编辑、通知、转换器、AddToMru、Local、CorruptLoad)	["/demo/htmlDemo/file/x1.xlsx",null,1] [[https://cdn.jsdelivr.net/g/h/appemit/appemitweb@master/docs/demo/file/x1.tmp,"x11.xlsx"]] https://docs.microsoft.com/zh-cn/office/vba/api/excel.workbooks.open
ViewType	1	xlNormalView 1 正常。 xlPageBreakPreview 双面 分页预览。 xlPageLayoutView 第三章 页面视图	
Protect	数组[]	(Password、DrawingObjects、内容、场景、UserInterfaceOnly、AllowFormattingCells、AllowFormattingColumns、AllowFormattingRows、AllowInsertingColumns、AllowInsertingRows、AllowInsertingHyperlinks、AllowDeletingColumns、AllowDeletingRows、AllowSorting、AllowFiltering、AllowUsingPivotTables)	对所有已经已存工作表保护 注意 AllowFiltering 参数 https://docs.microsoft.com/zh-cn/office/vba/api/excel.worksheet.protect
Unprotect	pwd	文档解锁密码	输入解锁密码 Unprotect 不保护。
Activate_ID	1,数值或者字符串	工作表的序号(左边第一为 1), 或者名称.默认打开的第几个工作表,或者为该名称的工作表	暂未启用

4.6.7.4 PowerPoint 其它参数

更多 <https://docs.microsoft.com/zh-cn/office/vba/api/overview/powerpoint>

名称	设置	含义	说明
Par0		可选。	
src	[path, openPar]	(ReadOnly、Untitled 无标题、 WithWindow)	https://docs.microsoft.com/zh-cn/office/vba/api/powerpoint.presentations.open
ViewType	1	ppViewHandoutMaster '讲义母版 4 ppViewMasterThumbnails '缩略图母版 12 ppViewNormal '普通视图 9 ppViewNotesMaster '备注母版 5 ppViewNotesPage '备注页 3 ppViewOutline '大纲视图 6	ppt ppViewType 类型

		ppViewPrintPreview '打印预览 10 ppViewSlide '幻灯片视图 1 ppViewSlideMaster '幻灯片母版 2 ppViewSlideSorter '幻灯片浏览 7 ppViewThumbnails '缩略图视图 11 ppViewTitleMaster '标题母版 8	

4.6.7.5 疑难

1. 要确保单独使用 word 打开, 文件格式不报错, 否则打不开。
2. 比如文档格式不对, 需要重新另存打开, 否则会出现等未知报错。
3. 选择保存时如果时间过长, 会出现服务器正在运行中, 需要在任务管理器关闭对应的 word\excel\ppt 的进程, 重新启动。

4. Office 自动化初始化

Office 服务器 CLSID 项

Access.Application {73A4C9C1-D68D-11D0-98BF-00A0C90DC8D9}

Excel.Application {00024500-0000-0000-C000-000000000046}

FrontPage.Application {04DF1015-7007-11D1-83BC-006097ABE675}

Outlook.Application {0006F03A-0000-0000-C000-000000000046}

Application.Application {91493441-5A91-11CF-8700-00AA0060263B}

Word.Application {000209FF-0000-0000-C000-000000000046}

在运行中

D:\PROGRA~1\MICROS~1\Office15\WINWORD.EXE /RegServer

检查 HKEY_CLASSES_ROOT\CLSID CLSID 项的 LocalServer32

D:\PROGRA~1\MICROS~1\Office15\EXCEL.EXE /automation

D:\PROGRA~1\MICROS~1\Office15\WINWORD.EXE /Automation

D:\PROGRA~1\MICROS~1\Office15\POWERPNT.EXE /AUTOMATION

4.6.8 WPS

创建、打开阅读、编辑金山 Office 文件, 支持本地和网络文件, 包括 wps\et\wpp 文件

更多 VBA 官方参考 <https://open.wps.cn/docs/office> wps 客户端开发

和 office 基本一致, 主要设置 progId 参数和增加了 attach 参数

<https://docs.microsoft.com/zh-cn/office/vba/api/overview/word>

或者录制宏, 编辑参考代码。

```
{"emit":"open","Obj":"office","AppType":1,"src":"/demo/htmlDemo/file/d1.docx","pos":1,"par0":{"attach":1,"progId":
```

```
"KWPS.Application","noMinMax":1,"ViewType":1,"Titlebar":0,"Menubar":1,"Toolbars":1,"btnFile":0}}
```

名称	设置	含义	说明
emit	open	必需。打开 office 事件请求。	
Obj	office	必需。	
AppType	1	必需。 4 word 5 excel 6 powerpoint	有默认的右键菜单。 若为负数-1,则是浮动窗口
src	字符串; path 为 [path,	path 为必需, 字符串网址或者本地 相对或者绝对路径 ,或者数组 (防止网页 src 里面没有	Word 系列文件 Excel 系列文件 PPT 系列文件

	openPar]	文件名称或者后缀名称需要修改, 则使用 fileType)。 openPar: 具体不同。 "/demo/htmlDemo/file/x1.xls" ["/demo/htmlDemo/file/x1.xls", openPar] [[http://.../demo/htmlDemo/file/x1.xls, "x2.xlsx"], openPar]	不同应用的 openPar 不同, 详细见后
pos	{"left":372,"top":203,"width":606,"height":406}	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器, 自动识别的位置需要优化
AppShow		必需。 true App 可见 false App 不可见	通常空时默认为 true 调用 comm 为 false
AppFollow		必需。空时默认为 true 1 跟随变化 0 不变化	AppShow 为 true 时, App 随浏览器移动变形隐藏显示等事件
Par0		可选。	
proglId	"KWPS.Application" 或者 []	默认为 MS office ["KWPS.Application", "WPS.Application"] ["KET.Application", "ET.Application"] ["WPP.Application", "WPP.Application"]	采用不同自动化服务器打开文档、表格、幻灯片, 按顺序依次尝试。
attach	1	附着 APP 方式	wps 最好采用此方式
noMinMax	1	1 禁用关闭 最小 最大化按钮 0/null 可最小化 关闭	遮盖

其它参数见 Office 章节, 注意金山 wps 的 VBA 代码和微软的不一定相同。

4.6.9 Office javascript

4.6.9.1 嵌入网页 JS 控制

在前章节的基础上, 增加一个 javascript 文件或者 html 文件, 在 officeJs 中添加代码控制 office, 支持 word excel ppt。更多参考 VBA 和宏录制。

更多 <https://docs.microsoft.com/zh-cn/office/vba/api/overview/word>

```
{"emit":"open","Obj":"office","AppType":1,"src":"/demo/htmlDemo/file/d1.docx","pos":1,"par":{"htmlStr":null,"HttpServer_startUrl":null,"URL":"/demo/htmlDemo/js/officeJs.html","webJs":1},"par0":{"Titlebar":0,"Menubar":0,"Toolbars":0,"btnFile":0,"Statusbar":0}}
```

在 websocket 中打开 d1.docx, 实际在 par 的文件/demo/htmlDemo/ officeJs.html 中的 js 修改打开 d2.doc, 并且增加输入文字功能。

名称	设置	含义	说明
emit	open	必需。打开 office 事件请求。	
Obj	office	必需。	
AppType	1	必需。	有默认的右键菜单。

		1 word 2 excel 3 powerpoint	若为负数-1,则是浮动窗口
src	字符串; path 为 [path, openPar]	path 为必需, 字符串网址或者本地相对或者绝对路径 ,或者数组 (防止网页 src 里面没有文件名称或者后缀名称需要修改, 则使用 fileType))。 openPar: 具体不同。 "/demo/htmlDemo/file/x1.xls" ["/demo/htmlDemo/file/x1.xls",openPar] [[http://.../demo/htmlDemo/file/x1.xls,"x2.xlsx"],openPar]	Word 系列文件 Excel 系列文件 PPT 系列文件 不同应用的 openPar 不同, 详见后
par			下面三个参数必须有一个。
htmlStr		Html 代码	
HttpServer_startUrl		以服务器形式打开本地 html 文件路径, 可以是绝对或者相对路径。/为分隔符。	
URL	http://www.appemit.com.../officeJs.html 或者 /demo/htmlDemo/js/officeJs.html	支持网页地址或者本地 html 文件路径。	
webJs	1	启用 par 的设置的嵌入网页 JS 1 IE 2 webkit 3 blink 0 关闭	或者设置为 0, 把 officeJs 的 src 参数设置在 src 里面。 使用 runCmd 执行输入命令。

在 officeJs.html 中也可以设置 src 参数。

webJs0	0	0 关闭本 js 的控制, 即使 webJs 大于 0 注释后启用。	系统默认执行了 4 个过程接口 AppJs_init AppJs_loaded AppJs_destroy AppJs_closed
--------	---	---------------------------------------	--

➤ AppJs_init=function(AppType)

如果在浏览器网页中没有设置 src, 可以在这里设置。

webJs0, 0 关闭本 js 的控制

objName:"word" 为程序对象名称, 可不设置, 自动根据 AppType 判断。如果修改了为 word1, 则后面的 AppJsObject.word.Selection, 应该修改为 AppJsObject.word1.Selection

- AppJs_loaded=function(AppType){
//系统启动 APP,产生 AppJsObject.objName 对象后执行
 定义了全局变量 AppJsObject.word
 Selection=AppJsObject.word.Selection;Application=AppJsObject.word.Application;' //注意
 结尾一定要有; 或者\r\n
- AppJs_destroy=function(){ //AppJsObject.objName 销毁前执行
- AppJs_closed=function(status){ //AppJsObject.objName 关闭后执行。已经没有
 AppJsObject.objName 对象了

4.6.9.2 浏览器网页执行代码命令控制

Var Req={"emit":"runCmd","Obj":"office","codeStr": "Selection.TypeParagraph();Selection.TypeText("控制插入一句话-runCmd"); } "

EmitReq(Req);

通过前面 officeJs.html 定义的 AppJsObject.objName，可以直接控制 office。当然也可以不需要 officeJs.html 定义，能在浏览器网页中 websocket 发送使用"emit":"runCmd"命令发送 AppJs_init、AppJs_loaded，代码也可以。

可以添加 js 文件或者 html，在 html 还能动态控制 office，更多功能请联系我们。

4.6.10 Office DsoFramer

创建、打开阅读、编辑 Office 文件，支持本地和网络文件，包括 word\excel\PPT 文件

```
{
  "emit": "open",
  "Obj": "office",
  "AppType": 8,
  "src": "/demo/htmlDemo/file/x1.xls",
  "pos": 1,
  "par0": {
    "Caption": true,
    "ProtectDoc": 0,
    "Toolbars": true,
    "btnFile": true,
    "MNU_NEWBLANK": 0,
    "MNU_NEW": 0,
    "MNU_OPEN": 0,
    "MNU_CLOSE": 0,
    "MNU_SAVE": 0,
    "MNU_SAVEAS": 0,
    "MNU_PGSETUP": 0,
    "MNU_PRINT": 0,
    "MNU_PROPS": 0,
    "show_UpdateTool": true
  }
}
```

名称	设置	含义	说明
emit	open	必需。打开 office 事件请求。	
Obj	office	必需。	

AppType	8	必需。 8 DsoFramer	有默认的右键菜单。 若为负数-1,则是浮动窗口
src	字符串 src 网址或者本地相对或者绝对路径 或者数组 [src,flag, ProgId ,user name,pwd,fileType]	src 为必需。 flag:true/false ProgId: Word.Document Excel.Sheet PowerPoint.Show Visio.Drawing filename:temp.doc 等防止 src 里面没有文件名称或者后缀名称不对, 则使用 fileType; {"/demo/htmlDemo/file/x1.xls",true, "Excel.Sheet","name","PWD"}	Word 系列文件 Excel 系列文件 PPT 系列文件 fileType
pos	{"left":372,"top":203,"width":606,"height":406}	必需。 1 默认使用代码自动识别的位置。	对不同的浏览器,自动识别的位置需要优化
Par0		可选。	
Caption	0	默认 false True 显示路径 string 显示具体 string 0/false/null	AppType 为负数时候显示标题栏
ProtectDoc	0	默认 0 , 保护文档 1 肯定, 解锁密码为 随机字符串, 不可获得。 String, 解锁密码为具体 String 0 可编辑文档	
Toolbars	1	默认 1 , 显示工具栏 1 显示 0 不显示	
btnFile	1	工具栏的文件 File 按钮 1 /true 默认显示 0	Toolbars 为真时有效
ShowView	1	wdNormalView = 1, wdOutlineView = 2, wdPrintView = 3, wdPrintPreview = 4, wdMasterView = 5, //这个是大纲 wdWebview = 6	
HttpPost	{"URL":"url", "method":"POST"}	URL 上传的地址, 必需 method 上传的方法 POST\GET,默认 POST	如果上传, URL 必需。

	"username": "name" "username": "*" } "cookie": null }	cookie	
MNU_UPLOAD	0	默认 0，工具栏上传命令菜单 1 显示上传提交 0 不显示	Toolbars、btnFile、HttpPost["URL"] 同时为真时有效
MNU_NEWBLANK	0	默认 0，工具栏 FILE 命令菜单 1 显示新建空白 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_NEW	0	默认 0，工具栏 FILE 命令菜单 1 显示新建 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_OPEN	0	默认 0，工具栏 FILE 命令菜单 1 显示打开 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_CLOSE	0	默认 0，工具栏 FILE 命令菜单 1 显示关闭 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_SAVE	0	默认 0，工具栏 FILE 命令菜单 1 显示保存 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_SAVEAS	0	默认 0，工具栏 FILE 命令菜单 1 显示另存 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_PGSETUP	0	默认 0，工具栏 FILE 命令菜单 1 显示页面设置 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_PRINT	0	默认 0，工具栏 FILE 命令菜单 1 显示打印 0 不显示	Toolbars、btnFile 同时为真时有效
MNU_PROPS	0	默认 0，工具栏 FILE 命令菜单 1 显示属性 0 不显示	Toolbars、btnFile 同时为真时有效
show_UpdateTool	0	在配置文件设置默认值 0 1 显示更新文件下载窗口 0 不显示	

4.6.10.1 更多示例

1. 打开本地 txt,只读，不显示标题栏，显示工具栏，不上传

```
{
  "emit": "open",
  "Obj": "office",
  "AppType": 1,
  "src": "/demo/htmlDemo/file/t1.txt",
  "pos": 1,
  "par0": {
    "openPar": [null, true],
    "Titlebar": 0,
    "ProtectDoc": 0,
    "Menubar": 0,
    "Toolbars": 0,
    "btnFile": 0,
    "HttpPost": {
      "URL": "http://eu.httpbin.org/post",
      "valid": 0,
      "fileName": "s1.docx"
    },
    "MNU_NEWBLANK": 0,
    "MNU_NEW": 0,
    "MNU_OPEN": 0,
    "MNU_CLOSE": 0,
    "MNU_SAVE": 0,
    "MNU_SAVEAS": 0,
    "MNU_PGSETUP": 0,
    "MNU_PRINT": 0,
    "MNU_PROPS": 0,
    "show_UpdateTool": true
  }
}
```

2. 打开网络 excel, 重命名, 不显示标题栏, 显示工具栏, 不上传

```
{"emit":"open","Obj":"office","AppType":2,"src":["https://cdn.jsdelivr.net/gh/appemit/appemitweb@master/docs/demo/file/x1.tmp","a1.xlsx"],"pos":1,"par0":{"Titlebar":0,"Menubar":1,"Toolbars":1,"btnFile":0,"HttpPost":{"URL":"http://eu.httpbin.org/post","valid":0},"MNU_NEWBLANK":0,"MNU_NEW":0,"MNU_OPEN":0,"MNU_CLOSE":0,"MNU_SAVE":0,"MNU_SAVEAS":0,"MNU_PGSETUP":0,"MNU_PRINT":0,"MNU_PROPS":0,"show_UpdateTool":true}}
```

3. 打开网络 ppt, 重命名, 幻灯片视图, 不显示标题栏, 显示工具栏, 不上传

```
{"emit":"open","Obj":"office","AppType":3,"src":["https://cdn.jsdelivr.net/gh/appemit/appemitweb@master/docs/demo/file/p1.tmp","p1.pptx"],"pos":1,"par0":{"ViewType":1,"Titlebar":0,"Menubar":1,"Toolbars":1,"btnFile":0,"HttpPost":{"URL":"http://eu.httpbin.org/post","valid":0},"MNU_NEWBLANK":0,"MNU_NEW":0,"MNU_OPEN":0,"MNU_CLOSE":0,"MNU_SAVE":0,"MNU_SAVEAS":0,"MNU_PGSETUP":0,"MNU_PRINT":0,"MNU_PROPS":0,"show_UpdateTool":true}}
```

4. 6. 10. 2 更多示例

4. 打开网络 docx, 不显示工具栏, 保护文档

```
{"emit":"open","Obj":"office","AppType":8,"src":["https://cdn.jsdelivr.net/gh/appemit/appemitweb@master/docs/demo/file/d1.tmp",false,"Word.Document","userName","PWD","tmp.docx"],"pos":1,"par0":{"Caption":true,"ProtectDoc":0,"Toolbars":0,"btnFile":true,"MNU_NEWBLANK":0,"MNU_NEW":0,"MNU_OPEN":0,"MNU_CLOSE":0,"MNU_SAVE":0,"MNU_SAVEAS":0,"MNU_PGSETUP":0,"MNU_PRINT":0,"MNU_PROPS":0,"show_UpdateTool":true}}
```

5. 打开网络 xlsx, 显示工具栏, 显示文件按钮, 上传文件(提交地址需要修改)

```
{"emit":"open","Obj":"office","AppType":8,"src":["https://cdn.jsdelivr.net/gh/appemit/appemitweb@master/docs/demo/file/x1.tmp",false,"Excel.Sheet","userName","PWD","a1.xlsx"],"pos":1,"par0":{"Caption":true,"ProtectDoc":0,"Toolbars":1,"btnFile":true,"HttpPost":{"URL":"http://eu.httpbin.org/post","fileName":"s1.xlsx"},"MNU_UPLOAD":1,"MNU_NEWBLANK":1,"MNU_NEW":0,"MNU_OPEN":0,"MNU_CLOSE":0,"MNU_SAVE":0,"MNU_SAVEAS":0,"MNU_PGSETUP":0,"MNU_PRINT":0,"MNU_PROPS":0,"show_UpdateTool":true}}
```

6. 打开本地 pptx, 显示工具栏, 不显示文件按钮

```
{"emit":"open","Obj":"office","AppType":8,"src":["/demo/htmlDemo/file/p1.pptx",false,"PowerPoint.Show","name","PWD"],"pos":1,"par0":{"Caption":0,"ProtectDoc":1,"Toolbars":1,"btnFile":0,"MNU_NEWBLANK":0,"MNU_NEW":0,"MNU_OPEN":0,"MNU_CLOSE":0,"MNU_SAVE":0,"MNU_SAVEAS":0,"MNU_PGSETUP":0,"MNU_PRINT":0,"MNU_PROPS":0,"show_UpdateTool":true}}
```

4. 6. 10. 3 其它参数

文档另存为

HRESULT SaveAs([in] VARIANT strFileName, [in] VARIANT dwFileFormat, [out,retval] long* pbool);

参数:

strFileName: 文件本地路径, 如 c:\\11.doc

dwFileFormat: 文件格式

Excel: Type enum XlFileFormat	Word: Type enum WdSaveFormat	PPT: enum PpSaveAsFileType
xlAddIn = 18	wdFormatDocument = 0	ppSaveAsPresentation = 1
xlCSV = 6	wdFormatTemplate = 1	ppSaveAsPowerPoint7 = 2
xlCSVMac = 22	wdFormatText = 2	ppSaveAsPowerPoint4 = 3
xlCSVMSDOS = 24	wdFormatTextLineBreaks = 3	ppSaveAsPowerPoint3 = 4
xlCSVWindows = 23	wdFormatDOSText = 4	ppSaveAsTemplate = 5
xlDBF2 = 7	wdFormatDOSTextLineBreaks	ppSaveAsRTF = 6

xlDBF3 = 8 xlDBF4 = 11 xlDIF = 9 xlExcel2 = 16 xlExcel2FarEast = 27 xlExcel3 = 29 xlExcel4 = 33 xlExcel5 = 39 xlExcel7 = 39 xlExcel9795 = 43 xlExcel4Workbook = 35 xlIntlAddIn = 26 xlIntlMacro = 25 xlWorkbookNormal = -4143 xlSYLK = 2 xlTemplate = 17 xlCurrentPlatformText = -4158 xlTextMac = 19 xlTextMSDOS = 21 xlTextPrinter = 36 xlTextWindows = 20 xlWJ2WD1 = 14 xlWK1 = 5 xlWK1ALL = 31 xlWK1FMT = 30 xlWK3 = 15 xlWK4 = 38 xlWK3FM3 = 32 xlWKS = 4 xlWorks2FarEast = 28 xlWQ1 = 34 xlWJ3 = 40 xlWJ3FJ3 = 41 xlUnicodeText = 42 xlHtml = 44	= 5 wdFormatRTF = 6 wdFormatUnicodeText = 7 wdFormatEncodedText = 7 wdFormatHTML = 8	ppSaveAsShow = 7 ppSaveAsAddIn = 8 ppSaveAsPowerPoint4FarEast = 10 ppSaveAsDefault = 11 ppSaveAsHTML = 12 ppSaveAsHTMLv3 = 13 ppSaveAsHTMLDual = 14 ppSaveAsMetaFile = 15 ppSaveAsGIF = 16 ppSaveAsJPG = 17 ppSaveAsPNG = 18 ppSaveAsBMP = 19
---	--	---

4. 6. 10. 4 疑难

DsoFramer 是微软的开源控件，现在已经不更新，容易出现 bug。

比如文档格式不对，需要重新保存在打开，否则会出现两个窗口等未知报错。

选择保存时如果时间过长，会出现服务器正在运行中，需要在任务管理器关闭对应的 word\excel\ppt 的进程，重新启动。

4. 6. 11 网页直接注册并调用 DLL OCX

主要有两种方法来实现调用 DLL 或者 OCX。

- 1、通过打开 IE 内核，操作 Object 调用 ActiveX，一般调用 ActiveX 需要已经注册。
- 2、JS 控制 AppEmit 调用 DLL 接口，互动信息，有些 ActiveX 可以免注册，见 4.6.12 公共 DLL\组件调用开发。

可以调用标准的 COM 组件、ActiveX 以及 DLL，支持可视化 ocx，同时可以开发互动的 dll，能在浏览器中控制调用的组件或者程序。

4.6.11.1 使用 IE 内核打开 ActiveX

使用 IE 内核打开网页,在非 IE 浏览器中调用 activeX 控件,及使用原有的 html 的代码即可。这种方法通常需要注册对应的 activeX

对于已经安装 flash ActiveX，则可以直接使用 IE 内核打开含有 flash 的网页。

```
{"emit":"open","Obj":"web","AppType":1,"AppId":1,"pos":1,"par":{"URL":"http://sxiao.4399.com/4399swf/upload_swf/ftp14/yzg/20140328/bombit7/zx_game7.htm"}}
```

4.6.11.2 使用 webkit 内核打开 NPAPI

在 chrome 浏览器中调用 object 控件，插件中已经包含了 NPSWF，可以直接打开含有 flash 的网页。

```
{"emit":"open","Obj":"web","AppType":2,"AppId":1,"pos":1,"par":{"URL":"http://sxiao.4399.com/4399swf/upload_swf/ftp14/yzg/20140328/bombit7/zx_game7.htm"}}
```

4.6.11.3 自动注册 ocx、dll 再使用 IE 内核或者 webkit 打开

需要注册才能打开 object，AppType=1 IE

NPAPI 则需要设置 AppType=2 webkit AppType=3 blink，使用 webkit 内核打开。

下面的例子是使用 IE 内核打开 ReportX.ocx

```
var htmlStr=AE.txt2code(function() {/**
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <div>
        reportX ，需要注册才能使用。
        <OBJECT ID='ReportX1' CLASSID='CLSID:A5DA6E97-1D4C-4708-B705-84A45716B4DD' >
            <PARAM NAME="Visible" VALUE="1">
            <PARAM NAME="AutoScroll" VALUE="1">
            <PARAM NAME="AutoSize" VALUE="1">
        </OBJECT>
    </div>
    **/});

ReqPar=
{"emit":"open","Obj":"web","AppType":1,"pos":1,"AppId":1,"par":{"URL":null,"htmlStr":htmlStr,"HttpServer_startUrl":null
    ,"embed_object" : {
        "objName":"ReportX1"
        ,"admin":0    //1 需要管理员权限注册 0 不需要
        ,"reg":true    //null 不注册 true 注册 false 卸载注册
        //    ,"pid":"ReportX"    //使用公共插件
        ,"pid":"292A53CD-DA8F-46A5-808D-B286F2759C37" //使用私有插件,
```

则需要增加下列信息

```
    ,"AuthKey":"A1-S0N5E1i-ya9KMYgpVm9eLAKYgLn-BB3t_Ut2ZnhWmiBReUv3u_eKl34oJisqX
```



```

2TM"
        , "dllFile": "/plugins/private/292A53CD-DA8F-46A5-808D-B286F
2759C37/ReportX.ocx"
        , "sha1": "234703F3A90202BF275EA76F0AAE73666644561B"
    }
}
} ;

```

4.6.12 公共 DLL\组件调用开发

通过 DLL 文件调用，主要区分 GUI 和非 GUI，采取不同的调用方式，主要如下：

```

createEmbedEx    createEmbed(comCarrier,clsId,iid)    //可视化的控件
    createObject(clsId,iid)
createInstance(clsId,itface)
createUnknown(clsId,iid)
dotNet    //应用程序域

```

直接系统 com 组件调用
设置 ComDll:"Com"

主要用来处理非标准的链接库，或者需要增加其它函数等操作，可以在 js 里面操作 object 对象与 dll 联系通信。

4.6.12.1 调用方法

1 使用 IE 或者 webkit blink 浏览器内核通过 activeX 或者 NPApi 来用 object 调用 dll，即使用原有的代码就可以。

2 如果需要使用 js 直接调用 dll 的方法，则要对 dll 注册或者不注册，再对接口函数需要声明，才能调用。方法 2 可以参考 demo 里面的开发工具来转换 API 函数，但是转换的函数仅供参考，需要测试。

下面介绍接口函数 API 声明。

4.6.12.1.1 API 数据类型声明

注意 API 数据类型声明严格区分大小写，数据类型大写表示对类型有更严格的限制条件。

其中数值类型小写表示允许负数，大写表示无符号数据类型(没有负数，仅有正整数)。

而对于支持指针的类型(string,pointer)，小写表示允许 null 值并允许自动转换（例如字符串转换为指针），大写表示不接受 null 实参。

注意 C/C++/WINAPI 的数据类型名有一些乱，同一个类型有无数的不同名字，而同一个类型名字可以处理为不同的类型，当然在 AppJsObject 里静态类型已经尽可能地规范和简化，所以学习起来并不难，只要掌握几个类型就可以了。

例如句柄类型，有时候作为无符号数，有时候作为有符号数，有时候又作为指针类型处理，实际上你无论是使用指针类型、有符号数值类型、还是无符号数值类型，对于句柄来说内存中的值是相同的，例如 `topointer(-1) == topointer(0xFFFFFFFF)` 的结果是相等的。在 AppJsObject 标准库里，一般句柄类型都被处理为指针，只有窗口句柄(HWND)在 AppJsObject 中统一被声明为 `addr` 类型。

API 数据类型		长度	对应的 AppJsObject 数据类型	C 语言类型	WINAPI 类型	备注
	类型					
无符号 字节	BYTE	8 位	数值 例: { char chr = 'A'# }	unsigned char	BYTE	注意 C++ 中的 bool(小写)类型为

字节	byte			char,		一个字节(8 位),等价于 AppJsObject 中的 byte 类型.
无符号短整型	WORD	16 位	数值 例: { short n = 123 }	unsigned short	WORD	
短整型	word			short		
无符号整型	INT	32 位	数值 例: { int n = 123 }	unsigned int,unsigned long	DWORD	winapi 中 H 前缀通常表示 32 位数。
整型	int			int,long	LRESULT, LPARAM, WPARAM	
无符号长整型	LONG64	64 位	math.size64()长整数,或普通数值 例: { LONG a = 123; LONG b = math.size64(456) }	unsigned long long,unsigned long __int64		<p>可缩写为 LONG,</p> <p>1、API 函数返回值中 LONG 类型返回为 math.size64 对象</p> <p>2、在 API 回调函数中, LONG 类型回调参数为 math.size64 对象</p> <p>3、在结构体中 LONG 类型字段值为 math.size64 对象时,AppJsObject 始终</p> <p>保持该对象的类型以及地址不变,反之则处理为 64 位浮点数值</p>
长整型	long64			long long,__int64		可缩写为 long

指针地址	ADDR	32位/64位	数值 例： { ADDR n = 123 }	void*	UINT_PTR,ULONG_PTR	注意使用数值表示地址时,为保持更好的兼容性,请使用此类型,而不要使用固定位长的 int,INT,long,LONG 等类型替代,
	addr				,DWORD_PTR,HWND INT_PTR,LONG_PTR,HWND	
浮点数	float	32位	数值 例： { float n = 123 }	float	FLOAT	
双精度浮点数	double	64位	数值 例： { double n = 123 }	double		
布尔值	bool	32位	true,false	int	BOOL	静态类型中的布尔值应明确指定 true 或 false,而不应当象 AppJsObject 类型那样自由的使用其他类型来替代.
					int	注意 C++中的 bool(小写)类型一般实现为 1 个字节(8 位),在 AppJsObject 中应使用 byte 类型表示 而 AppJsObject 中的 bool 类型,在 C++中表示为 BOOL(大写).

指针	pointer				PVOID,	此类型可缩写为 ptr ,或 PTR ,也可以使用所有 p 开头的自定义类型名字表示指针。
	ptr	32 位 / 64 位	pointer null		LPVOID,	大写的数据类型表示参数不接受 null 指针(空指针为 null , 而不是 0), 不能转换字符串为常量指针。
					LPCVOID	
					HANDLE,	
				void*		
	POINTER	32 位 / 64 位	pointer		HINSTANCE	指针值必须是一个 pointer 类型或 null 值, 在 AppJsObject 中空指针为 null 而不是 0, table 或 cdata 对象可以通过元表中的 _topointer 成员返回一个有效指针(不能为 null)或指针数值, _topointer 可以是一个值(元数据)或者一个函数(元方法)。注意对于结构体 AppJsObject 将忽略元表中的 _topointer

	PTR					
字符串	string	32 位	string pointer null	const char*	LPCSTR,	二进制字符串，也可以接受 pointer 指针值，大写则表示参数不接受 null 指针。在 API 回调函数或结构体中 API 返回 0~0xFFFF 或 -1 的指针地址 AppJsObject 将转换为指针而不是字符串
	STRING	32 位	string pointer		LPSTR,	
文本字符串	str	32 位	string pinter null	const char *		在普通 API 中表示 '\0' 结束的文本，在 Unioode API 中等价于 ustring 。在 API 回调函数或结构体中 API 返回 0~0xFFFF 或 -1 的指针地址 AppJsObject 将转换为指针而不是字符串

字节数组	pointer	32位 / 64位	buffer null	char *, unsigned char *		<p>字节数组实际上就是字符串、或二进制字符串，不同的是可以让API写入数据</p> <p>(AppJsObject 中普通字符串所占用的内存是只读的，修改字符串会创建新的字符串对象)，这种可以让API写入数据的字符串在AppJsObject 中对应的就是 buffer 类型，使用 raw.buffer() 函数创建 buffer</p>
Unicode 字符串	ustring	32位	ustring pointer null	const wchar_t*	LPCWSTR,	<p>Unicode(UTF16) 字符串，AppJsObject 会自动做双向编码转换，传给 API 时是 UTF16，从 API 返回时转换为 UTF8。类型名大写时禁用 null 值。在 API 回调函数或结构体中 API 返回 0~0xFFFF 或 -1 的指针地址</p> <p>AppJsObject 将转换为指针而不是字符串</p>
	USTRING	32位	string pointer		LPWSTR,	

结构体	struct	32 位	table	struct	可以在 API 参数中使用空表 {} 表示 C/C++ 中的 null 结构体指针 注意在 API 参数中,struct 被转换为指针按引用传递,如果是按值传递的结构体,需要展开所有成员为普通参数.
联合	union	32 位	table	union	u = { union value = { char c=8; short s=123; } }
	void			void	标识函数返回值为空

声明的数据类型必须保持绝对正确，使用错误的类型会导致内存读写错误并导致程序崩溃。使用 api 函数大部分的导致崩溃的错误原因在于数据类型定义错误。

4.6.12.1.2 API 数组(API Array)

静态数组必须写在一个结构体里面。

```
array = {
byte buffer[256]={1;2;3}
}
```


必须在中括号内指定一个有效的数值表示数组长度。

如果不指定任何数值,则表示一个弹性数组(变长数组),AppJsObject 将会在运行时检测获取实际的数组长度,弹性数组长度不能为 0。

```
array = {
byte buffer[]
}
```

array.buffer={1;2;3} //根据运行时的数组长度确定静态类型数组长度

CTYPE		示例	备注
无符号字节	BYTE []	<pre>array = { BYTE str[2] = { 'A' #, 'B' # } }</pre>	填充至定长 string ,
		或者	或者 table 数组(元素是数值或 null 值)
		<pre>array = { BYTE str[2] = "AB" }</pre>	如果将字段指定为 null 或字符串,则 AppJsObject 将该字段存储为字符串对象。
		或者	如果指定为 table 字节数组,则需要调用 string.pack 函数将数组转换为字符串。
		<pre>array = { BYTE str[2] = {0x41;0x42} }</pre>	也可以使用 raw.buffer 创建的缓冲区作为参数

		也可以不指定数组长度， AppJsObject 根据运行时值自动获取数组长度， 例如：	
字节	byte []	array = { BYTE str[] = "AB" }	
无符号短整型	WORD []	array = { WORD arr[2] = { 'A'#, 'B'# }	如果是一个 table 数组，元素必须是数值或 null 值，如果值为空，或者是一个字符串，AppJsObject 将该字段作为一个 UTF16 编码的 Unicode 字符串处理，在调用 API、以及输出参数时 AppJsObject 会自动做 UTF16 到 UTF8 的双向自动转换（ AppJsObject 字符串的默认编码为 UTF8 ）
短整型	word []	}	
无符号整型	INT []	array = { INT arr[2] = { 'A'#, 'B'# }	数组元素必须是数值或 null 值
整型	int []	}	
无符号长整型	LONG64 []	array = { LONG arr[2] = { 'A'#, 'B'# }	数组元素必须是数值或 null 值
长整型	long64 []	}	
无符号指针地址	ADDR []		
指针地址	addr []		
浮点数	float []		数组元素必须是数值或 null 值
双精度浮点数	double []		数组元素必须是数值或 null 值
布尔值	bool []		数组元素必须是布尔值或 null 值
指针	pointer [] POINTER []		数组元素必须是 pointer 或 null 值
字符串	string []		数组元素必须是 string 或 null 值

	STRING []		
结构体	struct []		至少要声明第一个数组元素

如果是一个 **struct** 组成的数组，则至少显示的声明数组中的第一个元素为有效结构体，其他已声明长度的数组赋值可指定任意个数的元素,也可以不指定值。未声明长度的数组必须赋值为非空数组以获取运行时数组长度。数组长度不能为 0。

4.6.12.1.3 输出参数(out Parameters)

在 **AppJsObject** 中，如果在数据类型以后添加&符号，表示这个参数的值允许被外部函数修改并且会返回修改后的值。

如果一个函数包含输出参数，那么传址参数会按原来的先后顺序附加在返回值后面返回。

AppJsObject 中函数是纯函数，函数数据只有唯一的入口(参数)，也只有唯一的出口(参数)，所以被修改的输出参数必须显示的从返回值输出。

```
apifunc = dllfile.api( "apifunc", " int ( int hWnd, string lpText,string &lpCaption ,INT uType )" )
```

```
result, /*输出参数追加在返回值后面*/lpCaption = apifunc(result,hWnd,lpText, lpCaption,uType);
```

输出参数			说明
数据类型	AppJsObject 类型	C 语言类型	
无符号字节	BYTE &	unsigned char *	
字节	byte &	char *	
无符号短整型	WORD &	unsigned short *	
短整型	word &	short *	
无符号整型	INT &	unsigned int *	
整型	int &	int *	
无符号长整型	LONG64 &	unsigned longlong *	该参数支持普通数值，以及 math.size64() 创建的长整数
长整型	long64 &	longlong *	
无符号指针地址	ADDR &	void**	
指针地址	addr &	void**	
浮点数	float &	float *	
双精度浮点数	double &	double *	
布尔值	bool &	bool * int *	
指针	pointer & POINTER &	void **	二级指针。

字符串	<p>string &</p> <p>STRING &</p>	<p>char *</p>	<p>string &类型的参数参数可以使用 <code>raw.buffer</code> 创建一段可写入的内存缓冲区（这时候对应的输出参数返回值为 <code>buffer</code> 类型，而非字符串）。</p> <p>如果参数是一个字符串，这时候 <code>AppJsObject</code> 会创建一个等长的临时的内存缓冲区并拷贝字符串到该内存，并将内存指针发送给 <code>API</code> 函数，在调用结束后增加相应的返回值返回新的字符串。</p>
-----	---	---------------	---

如果参数是一个指定缓冲区长度的数值（以字节为单位），**AppJsObject** 初始化缓冲区所有字节值为 0，并且在缓冲区尾部增加 2 个字节并写入 '\u0000'。使用参数 0 表示传递给缓冲区一个 null 指针(而不是使用 null 空参数)

这种 API 类型，也可以在 **AppJsObject** 中声明为 pointer 指针类型，然后用 `raw.buffer()` 函数创建一个可以让 API 写入数据的字节数组传过去。

	str &	char *	同上,但以'\0'为终结符返回文本字符串。如果参数使用 raw.buffer 创建一段可写入的内存缓冲区，这时候对应的输出参数返回值为字符串，而非 buffer 类型
宽字符串	ustring & USTRING &	wchar_t *	可以使用 raw.buffer 创建一段可写入的内存缓冲区（这时候对应的输出参数返回值为字符串，而非 buffer 类型）。

如果参数是一个字符串 (AppJsObject 会自动转换为 UTF16 编码再取长度)，这时候 AppJsObject 会创建一个等长的临时的内存缓冲区并拷贝字符串到该内存，并将内存指针发送给 API 函数，在调用结束后增加相应的返回值返回新的字符串。

如果参数是一个指定缓冲区长度的数值（以字符为单位，一个字符占 2 个字节），AppJsObject 初始化缓冲区所有字节值为 0，并且在缓冲区尾部增加 2 个字节并写入 '\u0000'。使用参数 0 表示传递给缓冲区一个 null 指针(而不是使用 null 空参数)

			<p>也可以使用 <code>raw.buffer()</code> 函数创建一个缓冲区作为参数，如果使用此缓冲区，输入值不转换编码，但返回值转换为 UTF8 编码，并转换为字符串类型作为返回值（不改变输入参数的类型）</p>
结构体	<code>struct &</code>	<code>void*</code>	<p>结构体按引用传递，具有副作用的，即使不接收对应的返回值，结构体仍然可被 API 函数修改值。</p> <p>可以在 API 参数中使用空结构体 <code>{}</code> 表示 C/C++ 中的 null 结构体指针</p>

对于数值类型、`pointer` 类型不能使用输出参数来代替输入参数需要重载 API 函数。

如果 API 形参定义为 `int`，那么你传递实参数值 123 时，API 函数接收到的是 123 这个数值。

如果 API 形参定义为 `int &`，那么你同样传递参数值 123 时，API 收到的是指向 123 的地址，是另外一个数值。

对于 `pointer` 类型、`pointe &` 引用类型，会有同样的问题。

而对于 `string` 类型、`struct` 类型，你在形参中加不加`&`，API 接收到的是相同的指针地址。唯一的区别是引用参数会返回修改后的值。字符串中的引用参数并不象其他参数类型那样 - 给 API 提供的是变量的地址，也就是你不能将 `string &` 理解为 C 中的 `const char **` 这样的二级指针用来接收一个字符串（`const char *`）的地址。如果需要接收这样的地址，应当改用一个结构体来获取。

4.6.12.2 可视化 Flash ocx 调用

能在网页中执行命令，控制 flash, 举例见 `demo/htmlDemo/dll.html`

```
{"emit":"open","Obj":"dll","AppType":1,"src":[],"pos":1,"par":{"htmlStr":null,"HttpServer_startUrl":null,"URL":"/demo/htmlDemo/js/dllJs.html","webJs":1}}
```

调用 `C:/Windows/SysWOW64/Macromed/Flash/Flash32_32_0_0_403.ocx` 控件，需要自行安装 flash.

可以在 `dllJs.html` 进一步设置参数 `AppType==1` 情况，和过程调用代码。同时也能通过浏览器网页发布命令。

名称	设置	含义	说明
emit	open	必需。打开 dll 事件请求。	
Obj	dll	必需。	
AppType	1	必需。 根据自己的代码确定。	有默认的右键菜单。 若为负数-1,则是浮动窗口
src	字符串数组	打开多个 DLL，只能一个可视化	
par			下面三个参数必须有一个。
plugins		Flash 默认自动	引入调用的插件，如果没有则会 自动下载已经上传插件库的插 件
htmlStr		Html 代码	
HttpServer_startUrl		以服务器形式打开本地 html 文件路径，可以是绝对或者相对路径。/为分隔符。	
URL	http://www.appemit.com.../dllJs.html 或者 /demo/htmlDemo/js/dllJs.html	支持网页地址或者本地 html 文件路径。	
webJs	1	启用 par 的设置的嵌入网页 JS 1 IE 2 webkit	或者设置为 0，把 dllJs 的 src 参数设置在 src 里面。 使用 runCmd 执行输入命令。

		3 blink 0 关闭	
par0			
AppMethod	null	默认非必需。 POST。同步呼叫的控件，应该设置为 POST，采取 websocket 相同的异步通话	

在 dllJs.html 中也可以设置 src 参数。

webJs0	0	0 关闭本 js 的控制，即使 webJs 大于 0 注释后启用。	系统默认执行了 4 个过程接口 AppJs_init AppJs_loaded AppJs_destroy AppJs_closed
--------	---	-----------------------------------	--

4.6.12.2.1 嵌入网页 JS 控制

➤ AppJs_init=function(AppType)

如果在浏览器网页中没有设置 src，可以在这里设置。

webJs0，0 关闭本 js 的控制

```

createType:"createEmbed" //可视化的控件 createEmbed(comCarrier,clsId,iid),只
能有一个,createObject(clsId,iid) createInstance(clsId,itface) createUnknown(clsId,iid)
,libName:"flash"

//ocx

,objName:"shockwave"
//,comCarrier:"custom" // createEmbed 可视
化 picture(图片控件设置)、static、winform 默认推荐 custom //不同类型组件
采取不要载体

,dllFile:"C:/Windows/SysWOW64/Macromed/Flash/Flash32_32_0_0_403.ocx" // 必
须 \\ /

,clsId : "D27CDB6E-AE6D-11CF-96B8-444553540000"

//guid

,iid:null
,itface:null
,embedObject:null

```

➤ AppJs_loaded=function(AppType){

//系统启动 APP,产生 AppJsObject.shockwave 对象后执行

AppJsObject.shockwave.Movie = "/demo/htmlDemo/file/test1.swf"

➤ AppJs_destroy=function(){ //AppJsObject.shockwave 销毁前执行

➤ AppJs_closed=function(status){ //AppJsObject.shockwave 关闭后执行。已经没有 AppJsObject.shockwave 对象了

4.6.12.2.2 浏览器网页执行代码命令控制

```

if (cmdId=="stop" ){
    codeStr = txt2code(function()/*

```

```

AppJsObject.shockwave.stop();
return {status="stop"}
*/});
    } else if(cmdId=="play" ){
        codeStr = txt2code(function()){/*
AppJsObject.shockwave.play();
return {status="play"}
*/});
    }
Var Req={"emit":"runCmd","Obj":"dll","codeStr": codeStr}
EmitReq(Req);

```

通过前面 dllJs.html 定义的 AppJsObject.shockwave，可以直接控制 flash。当然也可以不需要 dllJs.html 定义，能在浏览器网页中 websocket 发送使用"emit":"runCmd"命令发送 AppJs_init、AppJs_loaded，代码也可以。

4.6.12.3 报表控件 reportX 调用

能在网页中执行命令，控制 reportX, 举例见 demo/htmlDemo/dll.html

```

{"emit":"open","Obj":"dll","AppType":2,"src":[],"pos":1,"par":{"htmlStr":null,"HttpServer_startUrl":null,"URL":"/demo/htmlDemo/js/dllJs.html","webJs":1},"par0":{"AppMethod":"msg Synchronization component need set: POST" }}

```

调用"/plugins/thirdparty/report/ReportX.ocx" 控件

可以在 dllJs.html 进一步设置参数 AppType==2 情况，和过程调用代码。

同时也能通过浏览器网页发布命令。

名称	设置	含义	说明
emit	open	必需。打开 dll 事件请求。	
Obj	dll	必需。	
AppType	2	必需。 根据自己的代码确定。	有默认的右键菜单。 若为负数-1,则是浮动窗口
src	字符串数组	打开多个 DLL，只能一个可视化	
par			下面三个参数必须有一个。
plugins	["ReportX"]	字符串或者数组	引入调用的插件，如果没有则会 自动下载已经上传插件库的插 件
htmlStr		Html 代码	
HttpServer_startUrl		以服务器形式打开本地 html 文件路径，可以是绝对或者相对路径。/为分隔符。	
URL	http://www.appemit.com.../dllJs.html 或者 /demo/htmlDemo/js/	支持网页地址或者本地 html 文件路径。	

	dllJs.html		
webJs	1	启用 par 的设置的嵌入网页 JS 1 IE 2 webkit 3 blink 0 关闭	或者设置为 0，把 dllJs 的 src 参数设置在 src 里面。 使用 runCmd 执行输入命令。
par0			
AppMethod	null	默认非必需。 POST。同步呼叫的控件，应该设置为 POST，采取 websocket 相同的异步通话	

在 dllJs.html 中也可以设置 src 参数。

webJs0	0	0 关闭本 js 的控制，即使 webJs 大于 0 注释后启用。	系统默认执行了 4 个过程接口 AppJs_init AppJs_loaded AppJs_destroy AppJs_closed
--------	---	--------------------------------------	--

4.6.12.3.1 嵌入网页 JS 控制

➤ AppJs_init=function(AppType)

如果在浏览器网页中没有设置 src，可以在这里设置。

```
src:[ {
    // webJs0:0, //只能设置为
    0 关闭本 js 的控制；或者注释 为默认打开控制
    createType:"createEmbed" //可视化的控件
    createEmbed(comCarrier,clsId,iid) , createObject(clsId,iid) createInstance(clsId,itface)
    createUnknown(clsId,iid)
    ,libName:"reportXOCX"
    ,objName:"reportX"
    // ,comCarrier:"custom" // createEmbed 可视
    化 picture(图片控件设置)、static、winform 默认 推荐 custom //不同类型组
    件采取不要载体
    ,dllFile:"/plugins/thirdparty/report/ReportX.ocx"
    ,clsId : "A5DA6E97-1D4C-4708-B705-84A45716B4DD"
    //guid
    ,iid:null
    ,itface:null
    ,embedObject:null
    } //暂时支持一个控件
]
```

➤ AppJs_loaded=function(AppType){

//系统启动 APP,产生 AppJsObject.reportX 对象后执行

➤ AppJs_destroy=function(){ //AppJsObject.reportX 销毁前执行

- AppJs_closed=function(status){ //AppJsObject.reportX 关闭后执行。已经没有 AppJsObject.reportX 对象了

4.6.12.3.2 浏览器网页执行代码命令控制

```
if(cmdId=="reportX_open" ){
    codeStr = txt2code(function(){/*
//AppJsObject.reportX.OpenReport(..io.fullpath("/demo/htmlDemo/file/rep1.rpxe"))
var reportXPath=AppJs.dlgOpen('rpxe|*.rpxe|所有文件|*.*|')
AppJsVar_reportXPath=reportXPath; //全局变量 ,没有 var
if reportXPath {AppJsObject.reportX.OpenReport(reportXPath)
    return {status="open reportX"; path=reportXPath}
}else {
    return null;
}
*/});

}else if(cmdId=="reportX_save" ){
    //方案 1 通过 websocket 交互信息.
    /*
    if ( !reportXPath) return;
    codeStr = 'AppJsObject.reportX.SaveReport("'" +reportXPath+"' ) \
//通过 websocket 交互信息，或者直接采取全局变量
    return {status="save reportX"} ;
    */

    // 方案 2 直接采取全局变量

    codeStr = 'AppJs.msgbox(AppJsVar_reportXPath);
AppJsObject.reportX.SaveReport(AppJsVar_reportXPath);'
    codeStr+= 'return {status="save reportX 2"}' ;
```

// 方案 3 另存为

```
// codeStr = "var savePath=AppJs.dlgSave('rpxe|*.rpxe|所有文件|*.*|','另存为') if
(savePath) AppJsObject.reportX.SaveReport(savePath);"
```

```
}Var Req={"emit":"runCmd","Obj":"dll","codeStr": codeStr}
EmitReq(Req);
```

通过前面 dllJs.html 定义的 AppJsObject.reportX，可以直接控制 reportX。当然也可以不需要 dllJs.html 定义，能在浏览器网页中 websocket 发送使用"emit":"runCmd"命令发送 AppJs_init、AppJs_loaded，代码也可以。

4.6.12.4 第三方 dll 调用

1. dll_demo1.dll 安装在/plugins/thirdparty/dll_demo1.dll

能在网页中执行命令，控制 dll_demo1,举例见 demo/htmlDemo/dll.html

```
{"emit":"open","Obj":"dll","AppType":4,"src":[],"AppShow":false,"par":{"htmlStr":null,"HttpServer_startUrl":null,"URL":"/demo/htmlDemo/js/dllJs.html","webJs":1},"par0":{"AppMethod":"msgSynchronization component need set: POST" }}
```

可以在 dllJs.html 进一步设置参数 AppType==4 情况，和过程调用代码。

同时也能通过浏览器网页发布命令。

名称	设置	含义	说明
emit	open	必需。打开 dll 事件请求。	
Obj	dll	必需。	
AppType	4	必需。 根据自己的代码确定。	有默认的右键菜单。 若为负数-1,则是浮动窗口
src	字符串数组	打开多个 DLL，只能一个可视化	
pos	0	无界面不需要	
AppShow	false	必需。窗口不可见。 无界面必需设置 false。	
par			
plugins		字符串或者数组	引入调用的插件，如果没有则会 自动下载已经上传插件库的插 件
htmlStr		Html 代码	下面三个参数必须有一个。
HttpServer_startUrl		以服务器形式打开本地 html 文件路径，可以是绝对或者相对路径。/为分隔符。	
URL	http://www.appemit.com.../dllJs.html 或者 /demo/htmlDemo/js/dllJs.html	支持网页地址或者本地 html 文件路径。	
webJs	1	启用 par 的设置的嵌入网页 JS 1 IE 2 webkit 3 blink 0 关闭	或者设置为 0，把 dllJs 的 src 参数设置在 src 里面。 使用 runCmd 执行输入命令。
par0			
AppMethod	null	默认非必需。 POST。同步呼叫的控件，应该设置为 POST，采取 websocket 相同的异步通话	
backNoJson	true/false	默认 false,反馈为 json true,调用的 dll 或者 ocx 反馈的数据直接发送，不处理编码或者 json 转换	反馈的数据处理
codePage	65001	65001 utf8 1200 UCS-2LE 1201 UCS-2BE	反馈的数据编码，默认 65001

		0 不处理	
在 dllJs.html 中也可以设置 src 参数。			
webJs0	0	0 关闭本 js 的控制，即使 webJs 大于 0 注释后启用。	系统默认执行了 4 个过程接口 AppJs_init AppJs_loaded AppJs_destroy AppJs_closed

4.6.12.4.1 嵌入网页 JS 控制

演示调用 dll 的 add 方法，主要是初始化后，声明 `dll.api("Add","int(int a,int b)","cdecl");` // 必须是 cdecl 声明

➤ AppJs_init=function(AppType)

如果在浏览器网页中没有设置 src，可以在这里设置。

```
src:[ {
    // webJs0:0,                                //只能设置为0 关闭本 js 的控制 ;
    //或者注释 为默认打开控制
    createType:"createDll"                        // dll 调用
    ,objName:"dll_demo1"

    ,dllFile:"/plugins/private/50FCF891-1B93-4AE5-8A66-AB26A3C03378/dll_demo1.dll"
// 必须 \\ / 放在/plugins/private/clsId/文件夹下面
    ,clsId:"50FCF891-1B93-4AE5-8A66-AB26A3C03378" //guid 发送给 dll 信息
    ,iid:null
    ,data:'{"msg":"init from JS","info":1}'        //发送给 dll 数据

} //暂时支持一个控件
]
,AppShow:false                                //整体不可见 必须设置，默认可见

// dll 接口_declspec(dllexport) int Add(int a, int b )
//自行声明接口，再调用。

AppJsObject.dll_demo1.myAdd =AppJsObject.dll_demo1.dll.api( "Add","int(int a,int
b)","cdecl" ); //必须是 cdecl 声明

var sum=AppJsObject.dll_demo1.myAdd(3,5);
//AppJs.log(sum);
AppJs.msgbox(sum);
return {data=sum};                            //结果在 external.out 中
```

4.6.12.4.2 浏览器网页执行代码命令控制

//或者浏览器网页 发布命令 runCmd,执行 AppDll_RevMsg 方法

```
codeStr=txt2code(function(){/*
var sum=AppJsObject.dll_demo1.myAdd(3,5);
```

```

return sum;
*   /});
    var Req={"emit":"runCmd","Obj":"dll","codeStr":codeStr }
    EmitReq(Req);

```

通过前面 dllJs.html 定义的 AppJsObject.dll_demo1，可以直接控制 dll_demo1。当然也可以不需要 dllJs.html 定义，能在浏览器网页中 websocket 发送使用"emit":"runCmd"命令发送 AppJs_init、AppJs_loaded，代码也可以。

4.6.12.4.3 浏览器网页执行代码命令控制，完整案例

```

var init_App= function(){

    var ReqPar = { "emit": "open",
        "Obj": "dll",
        "AppId": 1,
        "AppType": 1,
        "src": [{ //, 私有的必需,在 appemit 登录窗口，我的应用，插件中设置 sha1 pid
AuthKey 等数据
                                // webJs0:0,                                //只能设置为 0
关闭本 js 的控制 ; 或者注释 为默认打开控制
                                //  "asAdmin":0    //1 需要管理员权限注册 0 不需要
                                //  ,"reg":null    //使用 appemit 直接调用，一般可以不注册。null 不注册    true 注
注册 false 卸载注册
                                createType: "callDll", // dll 开发
                                objName: "dll_demo2", // 自行定义
//AppJsObject.dll_demo2 来访问此 dll
                                dllFile:
"/plugins/private/9E8FD996-7D60-430A-8CE1-6796416E3FE0/dll_demo2.dll", // 必须 \\ / 分隔,
个人的插件放在 uugins/private/clsId/文件夹下面
                                // stdcall_cdecl:"stdcall",
                                pid: "9E8FD996-7D60-430A-8CE1-6796416E3FE0",
                                AuthKey:
"A1-coHyj1YZZzOIYdT0cRbr3tri2b5GrCAIN_HFirBlqEBb6DcbyMzC5NpDjgQd40IM", // pid 的授权，
appemit 网站获取
                                // clsId: "9E8FD996-7D60-430A-8CE1-6796416E3FE0", // guid
自己自行生成的， 最好和 pid 不同。 dll 里面设置的一致
                                CLASSES_ROOT: null, // 自行开发 dll 不用注册，则不用设
置
// ,clsIdAuthKey:"A1-QG7YAE5aUc7CCPSswU6yfhz6ojjZKFu-MPJy6nWAotpxq9QXRBTxJQZehllpoAph"
/I 文件访问的授权 自行设置
                                ver: "1.0.0",
                                iid: null
                                //, data:{"msg":"init from JS","info":1}                                //发送给 dll 数据
                                } // 暂时支持一个控件
        ],

        "AppShow": false,
        "par": { "webJs": 0 },

```

```
"par0": { "AppMethod": "msg Synchronization component need set: POST" } }
```

AE.OpenApp(ReqPar); // 连接好了 dll_demo2, 生成了 AppJsObject.dll_demo2 对象

```

}
var codeStr0 = AE.txt2code(function() { /*
AppJsObject.dll_demo2.fun_call=AppJsLibrary.dll_demo2.dll.api( "fun_call","int(string strIn,string &
strOut)","stdcall" ); //cdecl
*/ });

// 初始化回调执行 init_App,再生成全局对象 AppJsObject.fun_call 的方法 fun_call
AE.callbackFunc= [    {"equ": {  "clientAuth":1,"rep": 0,"rid":0},"func":init_App }
                      ,{"equ": { "Obj":"dll" ,"AppStatus":1,"rep":
0,"rid":1},"func":function(){AE.OpenApp({"emit":"runCmd","Obj":"dll","AppId":1,"codeStr":codeStr0 })
}} //
                      , {"equ":
{"Obj":"dll" ,"rid":2},"func":function(){AE.callbackFun_cancel=true;}} // 后面可以取消回调，不再判
断执行
];

```

```
var init_mark= AE.InitApp("ws://localhost:80/appemit?cid=00000-1&sid=1&flag=1"); //flag=1
调试环境
```

//////////前面的代码是在加载网页的时候执行一次
//后续的 runCmd 可以使用 AE.OpenApp 多次调用。

```

var codeStr = AE.txt2code(function() { /*

var strOut = raw.buffer(4096);
var strIn={`{...}`;
var outInt=AppJsObject.dll_demo2.fun_call(strIn,strOut );
//AppJs.msgbox(outInt);

return  {outInt=outInt;strOut= web.json.tryParse( raw.str( strOut),false,936 )}; //false 没有转义
符 ,936 GBK, 65001 UTF-8  raw.str 纯粹字符串 raw.string /0 标识的字符串
*/ });

var runCmd=  function(){
    // 调用  demo2_call

    AE.OpenApp({ "emit": "runCmd", "Obj": "dll", "AppId": 1, "codeStr": codeStr })

}

```

4.6.12.5 系统 com 组件

调用系统 com 组件，不要指定 dll 路径。

```
//调用 Connection var Req3= [{"emit":"open","Obj":"dll","AppId": 1,
"src":[ {
ComDll:"Com" //com 组件, 不写默认为 dll
,createType:"createObject" // createEmbedEx
,clsId : "ADODB.Connection"
,ver:"1.0.0" } ]
,"AppShow":false //整体不可见 必须设置
,"par":{"webJs":0}
}
, { "emit":"runCmd","Obj":"dll", "AppId": 1, "codeStr": 'return
{"adChar"=AppJsObject.comTest1.adChar;doc=com.tlbDoc(AppJsObject.comTest1 )};' } ]
```

```
AE. OpenApp(Req3, 10, true);
```

4.6.13 自行 DLL 开发调用

可以通过 2 种方法来实现

1. 开发的 DLL 可以和 AppEmit 保持接口一致，有些 ActiveX 可以免注册。
2. 开发的标准的 DLL 或者 OCX，支持 Object，则使用 web 内核打开。

下面介绍第一种方法，自有开发的私人 DLL 或者组件，需要申请 Pid 后开发 dll, 安装在 /plugins/private/ 目录下，需要双方验证成功后才能调用
参考 demo/htmlDemo/dll/dll_demo1.c, 提供了接口规范。 使用最新版本 TCC 开发的 dll。

4.6.13.1 开发

自有开发的私人 DLL 或者组件，需要申请 Pid （pid>=guid=clsId）可使用 GUID 工具自行生成后开发 dll, 安装在 /plugins/private/ 目录下，需要双方验证成功后才能调用
参考 demo/htmlDemo/dll/dll_demo1.c, 提供了接口规范。 使用最新版本 TCC 开发的 dll。

主要是 5 个函数接口, App 开头的函数和变量名称不要改变

```
AppDll_init      dll 初始化，用户输入参数 认证
AppDll_loaded    dll 启动后执行
AppDll_destroy   dll 退出前执行
AppDll_RevMsg    互动时 接收 消息
AppDll_SendMsg   互动时 反馈 消息
```

模板如下, 使用了 cJSON 开源库。

```
#include <windows.h>
// #include <string.h>
#include "cJSON.h"  ////需要下载开源库 cJSON
#include "cJSON.c"

// 开发可以互动信息的动态链接库 DLL
// http://www.appemit.com
////生成 DLL 必须使用最新版 tcc 扩展库才能支持 UTF8, UTF16 字符串
/*
```

入口函数,该函数可以有也可以没有。

入口函数会自动加锁以保证线性调用,要避免在 DllMain 内调用下列函数:

- 1、调用 LoadLibrary 或其他可能加载 DLL 的 API 函数(CreateProcess 等)
- 2、可能再次触发 DllMain 的函数,例如 CreateThread,ExitThread
- 3、GetModuleFileName, GetModuleHandle 等其他可能触发系统锁的 API 函数

总之在 DllMain 最好不要调用 API 函数.

```
*/
```

```
/*
```

主要是 5 个函数接口,App 开头的函数和变量名称不要改变

AppDll_init dll 初始化, 用户输入参数 认证

AppDll_loaded dll 启动后执行

AppDll_destroy dll 退出前执行

AppDll_RevMsg 互动时 接收 消息

AppDll_SendMsg 互动时 反馈 消息

```
*/
```

char *mycid="00000-1"; //申请的公司产品 cid , 需要修改

char *clsId="50FCF891-1B93-4AE5-8A66-AB26A3C03378"; // pid>=guid=clsId 可使用 GUID 工

具自行生成

int dllThid; //本进程 ID

// sid; //websocket 里面设置的,一般为用户 ID 或者 sessionId, 唯一

// rid ; //第 rid 次调用

int AppAuth=0; //

//测试

char* joinStr(char *s1, char *s2, char *s3)

```
{
```

char *result = malloc(strlen(s1)+strlen(s2)+strlen(s3)+1);//+1 for the zero-terminator

if (result == NULL) exit (1);

strcpy(result, s1);

strcat(result, s2);

strcat(result, s3);

return result;

```
}
```

int __stdcall DllMain(void * hinstDLL, unsigned long fdwReason, void * lpvReserved) {

if (fdwReason == 1/*DLL_PROCESS_ATTACH*/){

```
}
```

return 1;

```
}
```

//__declspec(dllexport) 声明导出函数

__declspec(dllexport) int AppDll_RevMsg(HWND hwnd, char *ids,char *msg)

```
{
```

```

AppDll_SendMsg(hwnd,ids,msg);
    return 0;

}
__declspec(dllexport) int AppDll_SendMsg(HWND hwnd, char *ids,char *msg)
{
    //检查是否有 clsId
    cJSON * ids_json= cJSON_Parse(ids);
    if (!ids_json) {ids_json=cJSON_CreateObject();}
    if (!cJSON_GetObjectItem(ids_json,"clsId")) {
        cJSON_AddStringToObject(ids_json, "clsId",  clsId);    //必需字段
    }
    ids=cJSON_PrintUnformatted(ids_json);
    cJSON_Delete(ids_json);

    struct {char * ids;char * msg; } callBackMsg = {
        .ids=ids,
        .msg =msg,                                // 只会反馈 msg 里面 data 字段到 websocket

    };

    SendMessage(
        hwnd,0xACCE ,
        "AppOnMsg({string ids;string msg;})", //要调用的窗体函数名( 结构体原型声明 ); 结
        &callBackMsg //将前面定义的结构体作为调用参数
    );

    return 0;

    /*
    0xACCE=_WM_THREAD_CALLBACK 使所有回调安全的转发到 UI 线程。
    _WM_THREAD_CALLBACK 可以跨线程跨语言并且不需要创建回调线程,适用任何普通
    winform 对象。

    */
}
__declspec(dllexport) int AppDll_init(HWND hwnd, char *ids,char *msg)
{
    //判断获得 AppEmit 提供 ids(格式 json)里面 cid sid pid AuthKey 等数据
    //和 websocket 的自行 web 提供的 Json 里面的 data，判断验证来源是否正确
    // ids
    ="{\"cid\":\"00000-1\", \"sid\":\"f1s\", \"rid\":\"2333\", \"AuthKey\":\"000\", \"clsId\":\"50FCF891-1B93-4
    AE5-8A66-AB26A3C03378\"}";

    //具体数据和判断需要修改
    cJSON * ids_json= cJSON_Parse(ids);

```

```

// char * ids3= cJSON_PrintUnformatted(ids_json); //如果解析报错尝试使用 cJSON
// cJSON * ids_json2= cJSON_Parse(ids3);

if (!cJSON_GetObjectItem(ids_json,"clsId") || !cJSON_GetObjectItem(ids_json,"cid")
|| !cJSON_GetObjectItem(ids_json,"AuthKey")) { cJSON_Delete(ids_json);return -1;}

dllThid = GetCurrentThreadId();
cJSON_AddNumberToObject(ids_json, "dllThid", dllThid); //必需
cJSON_AddStringToObject(ids_json, "more", NULL); //备用字段
//必须反馈验证
if (strcmp(cJSON_GetObjectItem(ids_json,"cid")->valuelstring,mycid)==0 &&
strcmp(cJSON_GetObjectItem(ids_json,"AuthKey")->valuelstring,"000")==0 &&
strcmp(cJSON_GetObjectItem(ids_json,"clsId")->valuelstring,clsId)==0) {

    //msg 若反馈必须有  \clsId\ "AppAuth"。AppAuth 为 1 clsId 一致 才继续 // 支持\ 或者 '

    msg =
joinStr("{\"data\":{\"code\":200,\"cid\": \"00000-1\", \"sid\": \"123\", \"rid\": -1, \"rec\": \"\",msg, \"AppStep\": \"init\"}}");
    cJSON_AddNumberToObject(ids_json, "AppAuth", 1);

    char *ids2=cJSON_PrintUnformatted(ids_json);
    //必须反馈验证

    AppDll_SendMsg(hwnd,ids2,msg);

}else{
    cJSON_AddNumberToObject(ids_json, "AppAuth", 0);
    char *ids2=cJSON_PrintUnformatted(ids_json);
    // 反馈验证
    AppDll_SendMsg(hwnd,ids2,NULL);
}

cJSON_Delete(ids_json);

return 0;
}
__declspec(dllexport) int AppDll_loaded(HWND hwnd, char *ids,char *msg)
{
    //处理业务 //msg 若反馈必须有 data
    // msg =
joinStr("{\"data\":{\"code\":200,\"cid\": \"00000-1\", \"sid\": \"123\", \"rid\": -1, \"rec\": \"\",msg, \"AppStep\": \"loaded\"}}");

    //若反馈则
    AppDll_SendMsg(hwnd,ids,msg); //如果 msg 中没有 data，则不反馈到浏览器中

```

```

        return 0;
    }
    __declspec(dllexport) int AppDll_destroy(HWND hwnd, char *ids, char *msg)
    {
        //处理业务 msg 若反馈必须有 data
        //msg =
        joinStr("{\"data\":{\"code\":200,\"cid\":\"00000-1\",\"sid\":\"123\",\"rid\":-1,\"rec\":\"\",msg,\"AppStep\":\"destroy\"}}");

        //若反馈则
        AppDll_SendMsg(hwnd, ids, msg);

        return 0;
    }
    //测试
    __declspec(dllexport) int Add(int a, int b)
    {
        return a+b;
    }
    __declspec(dllexport) int AppDll_init(HWND hwnd, char *ids, char *Json)
    {
        //判断获得 AppEmit 提供 ids(格式 json)里面 cid sid pid Auth 等数据
        //和 websocket 的自行 web 提供的 Json 里面的 data，判断验证来源是否正确

        //msg 若反馈必须有 \"AppStep\", \"clsId\" \"AppAuth\"。AppAuth 为 true clsId 一致 才
继续
        // 支持\" 或者 ' 如果有 data，只发送 data 字段到浏览器 websocket 接收
        char *msg =
        joinStr(joinStr("{\"data\":{\"code\":200,\"cid\":\"00000-1\",\"sid\":\"123\",\"rid\":-1,\"rec\":\"\",Json,\"AppStep\":\"init\"},\"AppStep\":\"init\",\"AppAuth\":true,\"clsId\":\"\"),clsId,\"\"}");
        dllThid = GetCurrentThreadId();
        //必须反馈验证
        AppDll_OnMsg(hwnd, ids, msg);

        return 0;
    }
    __declspec(dllexport) int AppDll_loaded(HWND hwnd, char *ids, char *Json)
    {
        //处理业务 //msg 若反馈必须有 \"clsId\"。如果有 data，只发送 data 到浏览器 websocket
接收
        char *msg =
        joinStr(joinStr("{\"data\":{\"code\":200,\"cid\":\"00000-1\",\"sid\":\"123\",\"rid\":-1,\"rec\":\"\",Json,\"AppStep\":\"loaded\"},\"AppStep\":\"loaded\",\"clsId\":\"\"),clsId,\"\"}");

        //若反馈则 OnMsg
        AppDll_OnMsg(hwnd, ids, msg);

        return 0;
    }

```



```

}
__declspec(dllexport) int AppDll_destroy(HWND hwnd, char *ids, char *Json)
{
    //处理业务 msg 若反馈必须有 \"clsId\".如果有 data, 只发送 data 到浏览器 websocket 接收
    char *msg =
joinStr(joinStr("{\"data\":{\"code\":200,\"cid\":\"00000-1\", \"sid\":\"123\", \"rid\":-1, \"rec\":\"\", \"Json\", \"AppStep\":\"destroy\"}, \"AppStep\":\"destroy\", \"clsId\":\"\", clsId, \"\"}");

    //若反馈则 OnMsg
    AppDll_OnMsg(hwnd, ids, msg);

    return 0;
}
//测试
__declspec(dllexport) int Add(int a, int b )
{
    return a+b;
}

```

4.6.13.2 嵌入网页 JS 控制

调用 dll_demo1.dll, 能在嵌入网页中执行命令, 控制 dll_demo1, 举例见 demo/htmlDemo/dll.html

```

{"emit":"open","Obj":"dll","AppType":3,"src":[],"AppShow":false,"par":{"htmlStr":null,"HttpServer_startUrl":null,"URL":"/demo/htmlDemo/js/dllJs.html","webJs":1},"par0":{"AppMethod":"msg Synchronization component need set: POST"}}

```

调用/plugins/private/50FCF891-1B93-4AE5-8A66-AB26A3C03378/dll_demo1.dll
可以在 dllJs.html 进一步设置参数 AppType==3 情况, 和过程调用代码。
同时也能通过浏览器网页发布命令 runCmd。

下面为 dll_demo1.dll 安装在
/plugins/private/50FCF891-1B93-4AE5-8A66-AB26A3C03378/dll_demo1.dll"

能在网页中执行命令, 控制 dll_demo1, 举例见 demo/htmlDemo/dll.html

```

{"emit":"open","Obj":"dll","AppType":3,"src":[],"AppShow":false
"pos":0,"par":{"htmlStr":null,"HttpServer_startUrl":null,"URL":"/demo/htmlDemo/js/dllJs.html","web Js":1},"par0":{"AppMethod":"msg Synchronization component need set: POST"}}

```

可以在 dllJs.html 进一步设置参数 AppType==3 情况, 和过程调用代码。

同时也能通过浏览器网页发布命令。

名称	设置	含义	说明
emit	open	必需。打开 dll 事件请求。	
Obj	dll	必需。	
AppType	3	必需。 根据自己的代码确定。	有默认的右键菜单。 若为负数-1,则是浮动窗口
src	字符串数组	打开多个 DLL, 只能一个可视化	
pos	0	无界面不需要	

AppShow	false	必需。窗口不可见。 无界面必需设置 false。	
par			
plugins		字符串或者数组	引入调用的插件，如果没有则会 自动下载已经上传插件库的插 件
htmlStr		Html 代码	下面三个参数必须有一个。
HttpServer_startUrl		以服务器形式打开本地 html 文 件路径，可以是绝对或者相对路 径。/为分隔符。	
URL	http://www.appemit.com.../dllJs.html 或者 /demo/htmlDemo/js/dllJs.html	支持网页地址或者本地 html 文 件路径。	
webJs	1	启用 par 的设置的嵌入网页 JS 1 IE 2 webkit 3 blink 0 关闭	或者设置为 0，把 dllJs 的 src 参 数设置在 src 里面。 使用 runCmd 执行输入命令。
par0			
AppMethod	null	默认非必需。 POST。同步呼叫的控件，应该设 置为 POST，采取 websocket 相同 的异步通话	
codePage	65001	65001 utf8 1200 UCS-2LE 1201 UCS-2BE 0 不处理	反馈的数据编码，默认 65001
backNoJson	true/false	默认 false,反馈为 json true,调用的 dll 或者 ocx 反馈的数 据直接发送，不处理编码或者 json 转换	反馈的数据处理

在 dllJs.html 中也可以设置 src 参数。

webJs0	0	0 关闭本 js 的控制，即使 webJs 大于 0 注释后启用。	系统默认执行了 4 个过程接口 AppJs_init AppJs_loaded AppJs_destroy AppJs_closed
--------	---	---	--

➤ AppJs_init=function(AppType)

如果在浏览器网页中没有设置 src，可以在这里设置。

```

src:[ {
    // webJs0:0, //只能设置为
0 关闭本 js 的控制 ; 或者注释 为默认打开控制
    createType:"createDll" // dll 调用
    ,objName:"dll_demo1"

    ,dllFile:"/plugins/private/50FCF891-1B93-4AE5-8A66-AB26A3C03378/dll_demo1.dll" //
    必须 \\ / 放在/plugins/private/clsId/文件夹下面
    ,clsId : "50FCF891-1B93-4AE5-8A66-AB26A3C03378"

    //guid 发送给 dll 信息

    ,iid:null
    ,data: '{"msg": "init from JS", "info": 1}' //
    发送给 dll 数据

} //暂时支持一个控件
]
,AppShow:false //整体不可见 必须设置，默认

```

可见

```

➤ AppJs_loaded=function(AppType){
//系统启动 APP ,产生 AppJsObject. shockwave 对象后执行
➤ AppJsObject.dll_demo1.AppDll_loaded(AppDll_hwnd, "", {"msg": "loaded from
JS", "info": 2});

➤ AppJs_destroy=function(){ //AppJsObject.shockwave 销毁前执行
AppJsObject.dll_demo1.AppDll_destroy(AppDll_hwnd, "", {"msg": "destroy from JS", "info": 3});
➤ AppJs_closed=function(status){ //AppJsObject 关闭后执行。已

```

4. 6. 13. 3 浏览器网页执行代码命令控制

```

codeStr= txt2code(function(){/*
AppJsObject.dll_demo1.AppDll_RevMsg(AppDll_hwnd, "", {"data": "send from JS", "info": 3});
* /});
var Req={"emit": "runCmd", "Obj": "dll", "codeStr": codeStr }
EmitReq(Req);

```

通过前面 dllJs.html 定义的 AppJsObject.dll_demo1, 可以直接控制 dll_demo1。当然也可以不需要 dllJs.html 定义, 能在浏览器网页中 websocket 发送使用 "emit": "runCmd" 命令发送 AppJs_init、AppJs_loaded, 代码也可以。

4.7 关闭 APP

close 和 closeAll 都是强制关闭指令。如果未断开 websocket，指令 1、2 都可以关闭 runtime 运行服务 APP。

- 1) close 必须指定 AppId 和 Obj
{"emit":"close","Obj":"media","AppId":1}
刷新浏览器即默认可关闭非 runtime 运行服务类型的 APP。
- 2) 如果关闭该类型的全部，发送 closeAll 即可关闭 Obj 类别所有。
{"emit":"closeAll","Obj":"media"}
- 3) 如果没有关闭而重新连接，增加 AppRuntime 参数关闭上次开启的 runtime 运行服务 APP。
{"emit":"closeAll","Obj":"media","AppRuntime":1}

4.7.1 关闭 cid 下对应的 APP

{"emit":"close","Obj":"media","AppId":1}

名称	设置	含义	说明
emit	close	必需。关闭控件 APP 通信事件请求。	
AppId	1,2...	必需。具体的可调整。	
Obj	Flash,media	必需。	

4.7.2 关闭 cid 下所有 Obj 类型 APP

{"emit":"closeAll","Obj":"media"}

名称	设置	含义	说明
emit	closeAll	必需。关闭所有控件 APP 通信事件请求。	关闭在 cid 下运行的所有控件 APP
Obj	flash	必需。	

4.7.3 关闭重新连接前的 runtime 运行服务 APP

{"emit":"closeAll","Obj":"media","AppRuntime":1}

名称	设置	含义	说明
emit	closeAll	必需。关闭关闭重新连接前的 runtime 运行服务 APP 事件请求。	
Obj	flash	必需。	
AppRuntime	1	必需。	

4.8 控制 APP 窗口

4.8.1 显示

{"emit":"show","Obj":"flash"}

名称	设置	含义	说明
emit	show	必需。显示窗口请求。	
Obj	flash	必需。	
AppId			
AppShow	true/null	默认 null 可视化 false 不可见	
pos		可选	

par0			
debounce	300	默认 300 毫秒	防抖函数，在限定的延时间内仅保留最后一次调用，延时默认 300 毫秒

4.8.2 隐藏

```
{"emit":"hide","Obj":"flash"}
```

名称	设置	含义	说明
emit	hide	必需。隐藏窗口请求。	
Obj	flash	必需。	
Appld			

4.8.3 移动

```
{"emit":"move","Obj":"flash","pos":{"left":10,"top":6,"width":206,"height":106}}
```

名称	设置	含义	说明
emit	move	必需。移动窗口请求。	
Obj	flash	必需。	
Appld			
pos	屏幕绝对位置	必需。	
par0			
debounce	300	默认 300 毫秒	防抖函数，在限定的延时间内仅保留最后一次调用，延时默认 300 毫秒

4.9 appemit 操作

4.9.1 文件 config.ini

config.ini 只能设置 userSet 节

```
[userSet]
```

```
excludePorts=[2049]
```

```
wss=[443,7131,7132,7133,7366]
```

```
ws=[80,8617,8618,8619,8811]
```

```
autoUpdate =1
```

```
URLProtocol=1
```

```
autoRun=1
```

```
runAsAdmin=1
```

```
certFile=/bin/cert/local_AE_2021.crt
```

```
keyFile=/bin/cert/local_AE_2021.key
```

```
ETC_HOSTS={"local.appemit.com":"127.0.0.1"}
```

名称	设置	含义	说明
userSet			

ws	[80,8617,8618,8619,8811]	数值或者数组	
wss	[443,7131,7132,7133,7366]	数值或者数组	
excludePorts	[2049]	数值或者数组，大于 2000 的 chrome 不支持 websocket 的端口	
autoUpdate	0	1 0 默认更新	
URLProtocol	1	1/null 默认支持 web 端访问 protocol 0	需要管理员权限
autoRun	1	1/null 默认自动运行 0	最好不要更改，需要作为后台服务一直运行。
runAsAdmin	1	1/null 默认管理员身份运行 0	URLProtocol 和 runAsAdmin 同时为 0 才关闭管理员权限启动
certFile		/bin/cert/local_*_2021.crt	
keyFile		/bin/cert/local_*_2021.key	
ETC_HOSTS		ETC_HOSTS={"local.A.com":"127.0.0.1"}	断网运行 wss，需要设置，修改 hosts 来支持。

4.9.2 错误信息

{"emit":"lasterr"}

名称	设置	含义	说明
emit	lasterr	必需。获得最近错误请求。	"

4.9.3 重启

{"emit":"restart","Obj":"app"}

只有一个用户使用的时候才能重启程序。

名称	设置	含义	说明
emit	restart	必需。appemit 重启请求。	重启后 client 需要重新连接。
Obj	appemit	必需。	

4.9.4 更新

{"emit":"update","Obj":"app"}

名称	设置	含义	说明
emit	update	必需。询问 appemit 是否更新程序请求。	默认强制更新。如果 config.ini 里面设置 autoUpdate=0，则询问更

			新。
Obj	appemit	必需。	

4.9.5 关于

```
{"emit":"about","Obj":"app"}
```

名称	设置	含义	说明
emit	about	必需。获得关于请求。	返回 {"data":{"appName":"app", ,"url":"http://www.appemit.com/", ,"verDesc":"\u516C\u5171\u514D\u8D39\u7248(Public free Version)", ,"verType":0,"version":"0.3.5"}," }
Obj	appemit	必需。	

4.9.6 版本信息

```
{"emit":"version","Obj":"app"}
```

名称	设置	含义	说明
emit	version	必需。获得版本请求。	{"data":{"verDesc":"\u516C\u5171\u514D\u8D39\u7248(Public free Version)", ,"verType":0,"version":"0.3.5"}," }
Obj	appemit	必需。	

5 发布

5.1 注册设置

- 1、运行 appemit，在本地 demo/login.html 或者 www.appemit.com/login.html 注册，获得新的 UID
- 2、设置里面新增一个 cid，双击打开复制 cid 和 clientKey。

5.2 局域网设置

在登陆窗口，局域网用户付费成功后，会出现设置选项卡，

- 3、勾选局域网，
- 4、或者增加 service。期中 service 可以在 config.ini 里面设置，作为最终设置。
- 5、在该页面运行 cmd 按钮即可，显示结果提示为局域网，以及显示 privilege 和 service。
- 6、执行更新本地数据，这个时候\bin\cfg\多了一个独一的 aaset.stglan，更新版本替换次文件就好了。
- 7、确保 config.ini 里面的 lanUserCfg=1，重新运行 AppEmit.exe

8、如果修改了 service,则需要修改 appemit.min.js 里面的 initSet 中的 service 以及需要调用的 ws://localhost:80/新的 service?

这样会增加配置文件/bin/cfg/aeset.stglang,

需要分发的局域网用户程序应该含有现在修改后的配置文件/bin/cfg/aeset.stglang, 即打包开发者电脑的 AppEmit 程序包, 而不是下载官网的。

同时设置 Config.ini 启动参数 lanUserCfg=1 来控制。

还原为外网, 则清空输入数据运行 cmd 按钮, 显示提示为外网。或者直接修改 Config.ini 参数 lanUserCfg=0。配置文件/bin/cfg/aeset.stglang 可以删除。

5.3 设置账号和授权

- 1、替换 appemit.min.js 里面的 initSet 的 clientKey
- 2、修改 wsUrl 里面调用的 cid
- 3、登入 <https://obfuscator.io> 把 wsInit.initSet 代码混淆加密。

5.4 修改程序和服务名称

可以联系客服确定权限。

在登陆窗口, 局域网用户付费成功后, 可以修改 privilege 权限。

```
{"rename"={1;3;5;7;9;11;13;15/*改名*/}
;"service"={2;3;6;7;10;11;14;15/*service*/}
;"high"={4;5;6;7;12;13;14;15/*exe 高级功能*/}
;"ieSecurity"={8;9;10;11;12;13;14;15/*IE 安全*/}
};
```

修改程序名称, 可以避免, 在同一台电脑上, 会默认关闭已经打开的同名程序。

更改服务名称, 可以使用 websocket 连接自己的服务名称, 同时网页定义的 protocol 也是为自己的服务名+web。

可以修改端口, 避免 80 端口被多个程序占用。

要是除了访问网页, 还涉及访问本地相对路径的文件, 建议最周全的是同时修改程序名称和服务名称以及端口, 变为只能自己使用的程序。

修改了服务名称, 需要同时更新 AppEmit.min.js 里面的 service, wsInit 的 initSet 参数, 以及单独调用 InitApp 过程的 wsUrl。

5.5 更新本地数据

发布二次开发产品前, 先强制更新本地数据。

发布(publish)

账户(Account)

支付(Pay)

发布软件前

强制更新本地产品和插件数据

5.6 Js 文件

设置 appemit.min.js 的初始化参数 wslnit。

clientKey 这部分代码最好应该隐藏，可以使用混淆的办法（比如 <https://obfuscator.io>，可以勾选 split strings，5；选择 String Array Encoding RC4；如果禁止调试可 debug Protection）。或者使用 OTP 加密方法（需要有服务器）。

5.7 插件部署

在发布时，可把需要使用的插件全部下载（<https://github.com/appemit/appemitPluginsAll>），把此 AppEmit 插件目录文件和 AppEmit.exe 一起打包后发布，不用单独分发。

不使用的插件可以删除。

Demo 可以删除。

插件支持 zip、7z、lzma 格式打包。

名称	路径	作用
blink	/plugins/lib/web/blink/.dll/node.dll	浏览器内核
	/plugins/lib/web/blink/.dll/wkex.dll	
flash_ocx	/plugins/Flash32.ocx	ActiveX flash
flash_NP	/plugins/NPSWF32.dll	NP 版本 flash
	/plugins/plugin.vch	
aliplayer	/plugins/common/aliplayer	Aliplayer 播放器支持 视频格式：mp4、flv、m3u8、rtmp 视频编码：H.264 音频编码：AAC、MP3 音频格式：MP3
PDF	/plugins/common/pdfjs	浏览 PDF 文件
mscomm32	/plugins/common/comm/mscomm32.ocx	
Pcomm	/plugins/common/comm/Pcomm.dll	
vlc	/plugins/plugins /plugins/axvlc.dll /plugins/eventRouter.dll /plugins/libvlc.dll	默认 vlc 插件为 2.2.5.1

	/plugins/libvlc.dll /plugins/npvlc.dll /plugins/vlc.exe /plugins/vlc-cache-gen.exe	
ReportX	/plugins/thirdparty/report/ReportX.ocx	报表控件
pdfjs	/plugins/common/pdfjs	阅读 PDF 文档
rtsp2webRTC	/plugins/common/rtsp/rtsp2webRTC	RTSP 转化为 webRTC
officeDso	/plugins/common/office/dsoframer.ocx	Office 接口
yeyou	/plugins/common/yeyou	页游导航
ahk	/plugins/thirdparty/ahk	AutoHotkey_H dll，嵌入运行 ahk 代码
AutoHotkey	/plugins/thirdparty/AutoHotkeyTool/AutoHotkey	AutoHotkey
AutoGUI	/plugins/thirdparty/AutoHotkeyTool/AutoGUI	Ahk 的界面编辑器
MacroCreator	/plugins/thirdparty/AutoHotkeyTool/MacroCreator	Ahk 的动作宏记录

5.8 发布

发布必须包含的目录和文件

```
├bin
├plugins      （可以按需配置）
├AppEmit.exe
├config.ini   （可以按需配置）
└uninstall.exe
```

插件 plugins 可以按需配置。

文件 config.ini 只能修改 userSet 节。

; ini 文件应该保存为 ANSI 编码

:[userSet] 可以配置的参数 ;[softInfo] 程序的基本信息和[status] 程序运行状态 只读。

;agreementChecked=1 同意程序的用户使用协议

;autoRun=1 自启动,会在注册表注册当前用户自启动，不推荐 0 不自动运行

;allUsers=1 管理员身份启动时则注册系统所有用户自启动;allUsers=2 强行提权注册系统所有用户自启动,当前用户和系统同时自启动，保留后者。

;需要本机所有用户使用,设置 allUsers=2 自动提权(win7 黑窗)，或者 allUsers=1 管理员身份运行。推荐不要切换用户，应该是注销再登陆。

;使用 https,wss 需要设置 wss,以及 mkcert 或者 keyFile,certFile

;局域网用户可以设置 lanUserCfg=1 不记录 IP

;;;;;;;;;;使用 https,需要设置 wss;;;;;;;;;;

; 方案 1，使用 mkcert 设置本地的 CA 认证 CARoot=1，默认

```

certFile=/bin/cert/localhost202120y.crt, keyFile=/bin/cert/localhost202120y.key
; 方案 2, 自行提供 DNS 域名解析, 设置解析到 127.0.0.1 的域名 local.a.com 的 keyFile 和
certFile。
;keyFile=1 默认的 SSL 验证文件, 设置为用户的验证文件的相对路径
keyFile=/bin/cert/a.key
;certFile=1 默认的 SSL 验证文件, 设置为用户的验证文件的相对路径
certFile=/bin/cert/a.crt
;addSSL_File=[{"certFile":"a.crt","keyFile":"a.key",caFile:null}] 可以增加 SSL 授权文
件
;如果要求断网使用, 需要设置 ETC_HOSTS。
;ETC_HOSTS={"local.appemit.com":"127.0.0.1"} 默认 1, 使用 wss, 断网或者
局域网需要本地验证 SSL, 生产环境 C:\Windows\System32\drivers\etc\HOSTS 设置为客户自己的
域名 local.domain.com 解析 127.0.0.1。默认的为免费测试, 有时效问题。
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;URLProtocol=1 注册具体的私有协议名称 appemitweb,用来网页在没有用户在线时或者没有启
动时打开后台程序
;ws=[80,8617,8618,8619,8811] 为 0 时则关闭 websocket 端口 , 可以更改。
;wss=[443,7131,7132,7133,7366] 为 0 时则关闭 websockets 端口 , 可以更改。
;excludePorts=[2049] 排除端口

;lanUserCfg=1 或者具体路径, 局域网可不记录 IP, 0 普通外网用户

;autoUpdate=0 推荐不自动更新

;logMemo=1 注释为空或者 1 普通用户默认记录默认 10000 条登陆日志 格式
"cid00000-1|1|2021/11/06 20:22:10" ; 局域网用户设定 1 记录, 否则不记录。为 0 都不记录。
;ca_sha1=1 一般用户可以忽略, 生成 CA 使用。
;certName=App Local Websockets Root CA 一般用户可以忽略, 生成 CA 使用。。

```

5.9 安装

把**开发者电脑**的 appemit 的文件夹以及需要使用的额外文件打包, 复制到客户的 windows 电脑, 双击 appemit.exe 即可启动。

默认会设置开机自动启动。

如果用户安装了杀毒软件或者电脑卫士, 请设置白名单 appemit.exe。

5.9.1 附加安装

把 appemit.exe 和客户程序可以一起打包安装。

单用户 ws 版本直接安装运行 appemit.exe 就可以。

多用户以及 wss 版本, 推荐管理员身份运行 appemit.exe。

为避免 UAC 的黑框提示, 可以客户程序提前设置好注册表, 然后普通权限运行 appemit.exe。

5.9.1.1 局域网

登录 appemit.com/login.html，见前述章节，设置局域网版本，获得/bin/cfg/aeset.stglan 文件，同时会修改 config.ini 中的 URLProtocol=1

5.9.1.2 网页启动程序

以下 HKEY_LOCAL_MACHINE 的注册需要管理员权限修改。为服务名称+web 设置私有的协议 appemitweb，注册如下

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\appemitweb]
```

```
@="appemitweb"
```

```
"URL Protocol"="具体路径..\..\AppEmit.exe"
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\appemitweb\DefaultIcon]
```

```
@="具体路径..\..\AppEmit.exe"
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\appemitweb\shell]
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\appemitweb\shell\open]
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\appemitweb\shell\open\command]
```

```
@="\"具体路径..\..\AppEmit.exe\" \"%1\" "
```

5.9.1.3 多用户

设置系统本机自动运行。

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

```
AE_AppEmit "具体路径..\..\AppEmit.exe" -hlm
```

同时需要设置网页启动程序，才能切换用户使用。

为避免手动点击，推荐不切换用户，而是注销或者重启以另外的用户登陆。

5.9.1.4 支持 wss

一种方案是使用 mkcert 生成支持本地的 CA，设置改参数本地浏览器需要重启才有效。

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SystemCertificates\ROOT\Certificates\5202492F1C0067489F50D63D1C5654E4D0D81D83
```

```
BLOB=/bin/mkcertCA/rootCA.txt 的二进制数据
```

方案二是自行提供 DNS 域名 A 解析 local.a.com 到 127.0.0.1，在 config 设置解析到域名 local.a.com 的 SSL 证书 keyFile 和 certFile。证书的时间最好较长，要不后续更新证书麻烦。

如果要求断网使用，设置 ETC_HOSTS={"local.appemit.com":"127.0.0.1"}来修改 hosts 文件。

6 问题

1. 支持 linux mac?

目前版本不支持，使用在 windows 系统上,支持 32、64 位系统。

2. 收费吗?
版本分为免费版、基础版、企业版、旗舰版, 支持的功能不同。
3. 免费版有何限制条件?
初始时和每 20,40,80...分钟时消息发送有弹窗。收费版本没有限制, 包括支持局域网。
4. 测试点击连接, 为何没有反应?
首先要打开 AppEmit.exe 服务, 可以 F12 查看报错情况。重启系统后, AppEmit.exe 进程自动开启, 没有被关闭。
5. 为何 IE、edge 等其它浏览器正常, chrome 不显示 APP 窗口?
请确保 chrome 没有被托管(菜单里最后出现“由贵单位管理”)限制功能。
6. App 的位置如何确定?
在 html 里面设置节点 appembed1 的位置, appemit.js 自动识别, 并控制 App 的位置和节点 appembed1 一致。
7. App 窗口能否不需要跟随浏览改变大小?
设置 AE.AppFollow, 31 为 APP 和浏览器的所有变化跟随控制, 0 不变化, 或者设置对应的 div 为绝对位置和大小, 具体见 AE.AppFollow 参数。
8. AppEmit 的执行顺序如何?
Html 加载时候执行 AE.InitApp, 连接好 websocket, 获得反馈 clientAuth: 1, service: "appemit"后, 然后执行具体的 AE.OpenApp(ReqPar, interval, sync)APP 命令就好。
9. 大量文件如何加载减少代码改动?
可以参考 autoEmbed.html, 增加一个分流, 360 浏览器直接跳转, chrome edge 浏览器嵌入即可。
10. 被嵌入的网页 IE10 不支持 JSON.stringify
//引入 json2 解决 IE10 不支持 JSON.stringify

<https://github.com/douglascrockford/JSON-js/blob/master/json2.js>

```
if(typeof JSON!="object"){JSON={}}function(){function use strict;var rx_one=/^[\s]*$/;var rx_two=/^(?:"\\|\/|bfrnt|u[0-9a-fA-F]{4})/g;var rx_three=/^(?:"\\|\/|bfrnt|u[0-9a-fA-F]{4})/g;var rx_escapable=/^(?:"\\|\/|bfrnt|u[0-9a-fA-F]{4})/g;var rx_dangerous=/^(?:"\\|\/|bfrnt|u[0-9a-fA-F]{4})/g;function f(n){return(n<10)?"0"+n:n}function this_value(){return this.valueOf()}if(typeof Date.prototype.toJSON!="function"){Date.prototype.toJSON=function(){return isFinite(this.valueOf())?(this.getUTCFullYear()+"-"+f(this.getUTCMonth()+1)+"-"+f(this.getUTCDate()+1)+"T"+f(this.getUTCHours()+1)+"."+f(this.getUTCMinutes()+1)+"."+f(this.getUTCSeconds()+1)+"Z":null);Boolean.prototype.toJSON=this_value;Number.prototype.toJSON=this_value;String.prototype.toJSON=this_value}var gap;var indent;var meta;var rep;function quote(string){rx_escapable.lastIndex=0;return rx_escapable.test(string)?""+string.replace(rx_escapable,function(a){var c=meta[a];return typeof c=="string"?c:"\\u"+("0000"+a.charCodeAt(0).toString(16)).slice(-4)}+"\\":'"'+string+'"'}function str(key,holder){var i;var k;var v;var length;var mind=gap;var partial;var value=holder[key];if(value&&typeof value=="object"&&typeof value.toJSON=="function"){value=value.toJSON(key)}if(typeof rep=="function"){value=rep.call(holder,key,value)}switch(typeof value){case"string":return quote(value);case"number":return(isFinite(value))?String(value):"null";case"boolean":case"null":return String(value);case"object":if(!value){return"null"}gap+=indent;partial=[];if(Object.prototype.toString.apply(value)=="[object Array]"){length=value.length;for(i=0;i<length;i+=1){partial[i]=str(i,value)}if("null"jv=partial.length==0?"[]":gap?("[\n"+gap+partial.join(",\n")+gap)+"\n"+mind+"]"):"["+partial.join(",")+"]";gap=mind;return v}if(rep&&typeof rep=="object"){length=rep.length;for(i=0;i<length;i+=1){if(typeof rep[i]=="string"){k=rep[i];v=str(k,value)}if(v){partial.push(quote(k)+(gap)?":":"."+v)}}}else{for(k in value){if(Object.prototype.hasOwnProperty.call(value,k)){v=str(k,value);if(v){partial.push(quote(k)+(gap)?":":"."+v)}}}v=partial.length==0?"{}":gap?("{\n"+gap+partial.join(",\n")+gap)+"\n"+mind+"}":"{"+partial.join(",")+"}";gap=mind;return v}if(typeof JSON.stringify!="function"){meta={"b":"\\b","t":"\\t","n":"\\n","f":"\\f","r":"\\r","u":"\\u","l":"\\l","s":"\\s"};JSON.stringify=function(value, replacer, space){var i;gap="";indent="";if(typeof space=="number"){for(i=0;i<space;i+=1){indent+=" "}}else if(typeof space=="string"){indent=space;rep=replacer;if(replacer&&typeof replacer!="function"&&typeof replacer!="object"||typeof replacer.length!="number"){throw new Error("JSON.stringify");}return str("",{"":value})}if(typeof JSON.parse!="function"){JSON.parse=function(text,reviver){var j;function walk(holder,key){var k;var v;var value=holder[key];if(value&&typeof value=="object"){for(k in value){if(Object.prototype.hasOwnProperty.call(value,k)){v=walk(value,k);if(v!=undefined){value[k]=v}else{delete value[k]}}}}return reviver.call(holder,key,value)}text=String(text);rx_dangerous.lastIndex=0;if(rx_dangerous.test(text)){text=text.replace(rx_dangerous,function(a){return("\u"+("0000"+a.charCodeAt(0).toString(16)).slice(-4))});if(rx_one.test(text.replace(rx_two,"@").replace(rx_three,"").replace(rx_four,""))){j=eval("(("+text+")");return(typeof reviver=="function"?walk({"":j},""):j)}throw new SyntaxError("JSON.parse");}}}
```

11. 被嵌入网页如何调试?

IE, webkit 可以在被嵌入网页里面引入

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/firebug-lite/1.4.0/firebug-lite.js#startOpened"></script>
```

或者

```
<script
```

```
src="https://cdn.jsdelivr.net/gh/appemit/appemitweb@master/docs/firebug_1.4.js#startOpened"></script>
```

12. 可以系统所有用户使用吗？

可以，allUsers=1 管理员身份启动时则注册系统所有用户自启动;allUsers=2 强行提权注册系统所有用户自启动。

不推荐切换用户，应该是注销后登陆或者重启登陆。