佐罗软件跨浏览器的原生小程序系统

“PluginOK 中间件”开发者手册

版本	V2.2.16.8
版权	成都佐罗软件有限公司
日期	2024 年 11 月
技术支持热线电话	**4006831589/18081958957**
技术支持微信号	**ZorroSoft**
技术交流 QQ 群	**23126938**

索引

一、概述

PluginOK 中间件(以下简称“PluginOK”)，原名 WebRunLocal(本网通)，是一个实现浏览器与桌面程序之间可双向调用的低成本、强兼容、高性能、安全可控、轻量级、易集成、可扩展、跨浏览器的原生小程序系统，可顺畅支持 Windows XP 及以上版本的桌面系统和 Windows Server 2008 及以上版本的服务系统原生 Win32 程序直接内嵌运行到当前主流的浏览器网页中。PluginOK 及其之上运行的各种小程序还可在网页中通过脚本请求自动静默升级，无需用户干预。PluginOK 主程序默认以 Windows 系统服务的形式运行，从而确保中间件及小程序随时在线可用，也可配置为桌面进程形式运行。PluginOK 采用国际标准 HTML5 中的 Web Socket 技术作为中间件的通信基础，可确保本技术方案不会过时，而功能实现及传递的参数采用 JSON 封装包来实现开发语言的无关性和扩展性。这样的技术方案也可为不同系统或软件之间进行统一通讯的基础中间件，在网页中可实现多个不同小程序之间的互操作和协同，从而解决各软件或系统之间紧密协作的难题。

基于 PluginOK 的 SDK(二次开发包)接口可开发本地硬件 DLL 驱动模块的封装小程序，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；也可以将 Windows 本地 API 函数(如证书验证、文件读写及上传下载等)功能封装成小程序后供网页端调用；还可以把大量 ActiveX 控件(如 VLC 媒体播放)及常用自动化程序(如微软 Office、金山 WPS、永中 Office、AutoCAD、Solidworks 等)封装为独立进程的弹窗或内嵌网页小程序，从而可彻底解决它们在 Chrome、Edge、Firefox、360、IE、Opera、Electron、Vivaldi、Brave、QQ、华为、齐安信、红莲花等高版本浏览器中不能兼容运行的问题。

自互联网诞生以来，想要在浏览器中实现网页和本地程序双向调用，IE 中一般使用的是 ActiveX 控件技术，而在 Chrome、Firefox、Opera 中则使用的是 NPAPI 插件技术。但因安全隐患、稳定性等诸多问题，目前最主流的 Chromium 内核浏览器从 45 版本开始 NPAPI 插件被彻底禁用，微软也官方宣布 2022 年 6 月正式淘汰 IE 浏览器，当前最流行的前端框架 Vue 等也都不再支持 IE，这就导致跨浏览器的插件框架 FireBreath 也成了摆设。因此最近这些年里，那些依赖 ActiveX 或 NPAPI 实现的信息化系统无法在 Chrome、Firefox 等高版本浏览器中继续使用，给企业用户体验及相关业务的开展带来了巨大障碍，数年来一直没有一个完美的技术解决方案。PluginOK 作为 FireBreath 的最佳替代升级方案，是成都佐罗软件有限公司长期技术积累和持续技术攻关成功开发出来的成熟基础中间件，即使是美欧日等发达国家也没有同类产品问世，在跨浏览器的功能扩展领域，PluginOK 已处于世界领先的技术水平。需要特别说明的是，PluginOK 高级版提供的桌面程序窗口可真正内嵌网页使用的核心技术方案，2019 年申请了国家软件发明专利保护，并于 2023 年 5 月正式获得发明专利证书，专利号：**ZL 2019 1 1323165.1**，所有模仿、抄袭、破解等行为都存在巨大的法律风险，侵权必究。

有点热门的 Node.js 是一个基于 Chrome V8 引擎的 JavaScript 运行环境和新兴的前后端框架，只能实现在网页中调用符合标准的 DLL 函数用于扩展 JS 的功能，不能实现在 DLL 中反向调用前端服务，也不支持 ActiveX 控件，仅适用于无界面交互的操作场景。而且需要针对不同的系统和浏览器分别独立部署对应的版本，安装包上百 MB，不利于分发部署。最麻烦的是要改变用户使用原有浏览器

的习惯，是无法作为 B/S 业务系统的通用方案给客户使用的。市场上针对具体需求有一些零星的解决方案，比如把需要的 DLL 库功能封装为一个 HTTP 协议的 WebService 来供网页中调用。但由于 HTTP 协议是无状态的，无法做及时的回调处理，与本地有界面的程序也无法进行交互，而且大多数的实现未提供对 HTTPS 协议支持。然而因为安全原因，主流的 WEB 站点已切换到 HTTPS 协议，Chrome 等浏览器会自动标记 HTTP 网站是不安全的。再说在企业信息化/数字化系统中，往往要对接多个硬件设备或软件系统，这时候就必须有一个统一的技术解决方案，用 PluginOK 这样的通用解决方案无疑是最佳选择。

在 Windows 系统中 PluginOK 采用 COM 组件作为小程序开发的接口技术标准，所以只要是支持 COM 组件开发的编程语言，都可用于开发 PluginOK 之上运行的小程序，包括并不限于 C、C++、C#、VB、Delphi 等。SDK 包中包含了 4 个小程序的实现范例、打包发布工具及相关开发文档。为了支持小程序的安全可控调用，PluginOK 同时实现了小程序调用方的验证机制，在请求指定小程序服务时，需要传递指定规则的加密授权参数 tk，也确保不会被第三方未授权的调用。随着 PluginOK 上的小程序数量增加和功能的不断完善，将为开发网页应用(Web App)提供最强大的支撑，可彻底解决困扰众多 B/S 业务系统无法访问本地软硬件功能或纯脚本实现导致运行低效的问题。

尽快采用 PluginOK 中间件技术方案，您将获得以下好处：

1、技术领先于竞争对手，可最大程度的满足客户的多浏览器兼容使用需求；

2、大幅降低产品/项目的开发成本，从而提高公司的整体效益；

3、产品/项目的交付周期可大大缩短，从而可抢占市场先机；

4、提供最专业的开发和维护支持，使您能获得持续改进的产品；

5、对各种操作系统及各版本浏览器有最好的兼容，小程序的安装及升级便捷容易。

PluginOK 最低可兼容运行在 Windows XP 操作系统中，也支持新版的 Windows 10 及 11 桌面系统，也支持 Windows Server 2008 及以上 32/64 位的服务系统。

中间件标准版可兼容运行在所有支持 WebSoket 的浏览器中，包含 IE6 及以上版本。中间件高级版相对特殊一点，需要针对性开发后方可支持，高级版兼容一下浏览器：

1、IE 8 及以上版本；

2、Chrome 41 及以上版本；

3、FireFox 50 及以上版本；

4、Edge(Chrome 内核) 80 及以上版本；

5、Opera 36 及以上版本；

6、Brave 浏览器(英文版)；

7、Vivaldi 浏览器(英文版)；

8、Electron 桌面浏览器；

9、360 极速浏览器(X) 9.5 及以上版本(中文版)；

10、360 安全及企业安全浏览器(中文版)；

11、QQ 浏览器 10 及以上版本(中文版)；

12、搜狗浏览器(中文版)；

13、华为浏览器(中文版)；

14、微信网页窗口(中文版，需要配置后开启)；
15、齐安信浏览器(中文版)；
16、海泰红莲花浏览器(中文版)；
17、联想浏览器(中文版)...

针对不同的业务使用场景，PluginOK 支持三种类型的小程序：

A、DLL 小程序，主要实现为进程内的 COM 组件，也可以实现为进程外的 COM 组件(好处是小程序里如果崩溃不会影响主服务进程)，主要用于封装各种硬件设备的驱动库及无界面交互的 Win32 API 等。此类型小程序默认运行在 PluginOK 服务进程中，拥有较高的系统访问权限，不支持弹窗及其它界面交互操作。如需访问当前桌面登录用户的注册表及相关配置，则需配置 PluginOK 程序已非服务方式运行，这时为了确保前端可随时调用，就需要设置登录到桌面时自动启动运行，以非系统服务方式运行时，此类小程序也可支持有交互的弹窗界面场景；

B、EXE 弹窗小程序，此类型小程序是普通的 Win32 进程，只不过启动时需要添加指定的命令行参数，主要是封装打印机、扫描仪、高拍仪等的驱动程序功能给前端调用，以弹出类似网页新窗口的形式运行，权限和普通桌面应用一样；

C、EXE 内嵌网页小程序，此类型小程序实质也是 Win32 进程，除了启动需要指定的命令行参数之外，还需要集成 PluginOK 实现网页内嵌的 SDK，主要是封装有持续界面交互如视频播放 ActiveX 控件、办公软件及 CAD 软件的窗口以真正内嵌网页的形式运行，体验比原 ActiveX 更好，与 NPAPI 实现的效果类似。

二、中间件安装

PluginOK 的安装包非常小，同时支持 32 和 64 位系统及多个浏览器版本核心程序总共不到 15MB，而且不再依赖其它第三方运行库，提供支持命令行参数的绿色版安装方式，最大化方便客户的自定义打包安装。下面是如何进行配置安装：

在线体验请访问网站 http://local.zorrosoft.com 并下载中间件安装包执行安装。也可以从绿色版程序包下载并解压到非操作系统所在的某个磁盘目录，如果是绿色版程序包，找到 InstallWrl.bat 文件，然后选中此文件右键执行“以管理员身份运行”即可完成 PluginOK 的安装，PluginOK 主程序会自动以系统服务方式启动运行。卸载 PluginOK 是执行 RemoveWrl.bat 批文件，停止服务请执行 StopWrl.bat 批处理文件。

PluginOK 的所有小程序存放在主程序所在的 Plugins 子目录，每个小程序一个子目录，小程序目录名称即是此小程序的 PID，目录结构类似这样：PluginOK 安装路径\Plugins\PID\小程序文件，这样可以把你的小程序文件和 PluginOK 的程序文件按这个规则打包到一起，可以实现 PluginOK 安装的同时也把自己的小程序一起部署完成。

启动 Chrome 等浏览器，打开网站 http://local.zorrosoft.com，您将看到如下的界面：

在上述网页界面点击下载网络高级版安装包进行安装完成，会提示您的电脑需要审批后方可在线体验，网页中已经提供了 RTSP 网页中低延迟播放及 Office 文档在线编辑的体验入口，如您需要体验的功能不是这两个，请参考程序安装目录的文档“体验测试说明.txt”进行。

网络版 HTTP 网站默认的 WS 侦听端口是 83(单机版可能默认是 80)，HTTPS 网站默认的 WSS 侦听端口是 453(单机版默认是 443)，在实际的项目中可根据自己的需要修改配置文件 ZbaConfig.json(单机版是 WrlConfig.json)中 WS 及 WSS 的默认端口，修改保存后请重启服务(命令行管理员权限执行 net start ZbaService(单机版是 WrlService))以生效，实际生效的配置文件在 Data 子目录，此文件不存在时自动从主程序目录复制一份来使用。点击按钮“连接”，在网页中即可与 PluginOK 主程序及安装的小程序进行通讯。如遇前端连接提示失败，请到主程序所在的 Data 子目录下查看 ZbaService.txt(单机版是 WrlService.txt)中输出的实际侦听端口并修改前端连接的 WS 地址。配置文件中 NoService 可设置为 1，代表中间件主程序以普通桌面进程运行，默认是系统服务方式运行；Udisk 设置为 1 代表监控可热插拔的 U 盘；Mac 设置为 1 代表允许前端获取本机 MAC 地址；Localhost 设置为 1 代表用 localhost 建立侦听，否则为本机实际 IP 侦听，方便局域网其它主机来请求服务。

PluginOK 提供了查询本地安装的 PluginOK 中间件及小程序的版本等信息功能，也支持 PluginOK 本身的升级、小程序的安装及升级命令，便于您通过网页前端直接分发或升级自己的小程序而无需用户端确认。标准版更详细的功能测试说明请参考文档 TestWrl.txt。开发接口说明请参考下一节内容。

三、中间件接口

PluginOK 中间件进程支持并发请求 Web Socket 连接，网页中可同时请求多个小程序(部分小程序本身不支持并发启动的除外)的功能，小程序一般也支持同

时建立多个连接请求其功能服务。连接请求主要有三类：
A、请求 PluginOK 主程序服务，通过此连接可获得本系统的版本及小程序信息，系统的升级、小程序的安装和升级等请求都在此连接中进行；
B、请求 DLL 小程序的服务，此小程序内部可实现多个组件功能服务，连接参数中必须携带对应组件的参数 pid(实际使用的是 PluginConfig.json 中的 ProgID)；
C、请求 EXE 弹窗小程序的服务，连接参数中必须携带对应小程序的参数 pid(小程序唯一标识)，执行后会直接启动小程序的执行程序，并获得此程序建立的侦听端口，然后再通过此端口的连接从而可实现前端和小程序的相互通讯或调用。
D、请求 EXE 内嵌网页小程序的服务，需在 PluginOK 主程序中携带对应小程序的请求名称，执行后先启动小程序进程，并获得此程序建立的 WS 侦听端口，然后再通过此端口的连接从而可实现前端和小程序的相互通讯或调用。

1、连接 PluginOK 服务：
在网页中使用 JS 脚本进行 Web Socket 连接的地址：
ws://wrl.zorrosoft.com:83?sid=123&flag=1&lang=CHS (如果是 https 网站，请使用安全连接：wss://wrl.zorrosoft.com:453?sid=123&flag=1&lang=CHS)。

说明：ws 或 wss 为 Web Socket 默认的访问协议标识，其中 wrl.zorrosoft.com 为方便测试 HTTPS 网站而使用的域名(您在具体的实施项目中可修改为指定的域名，http 网页可直接用 localhost 及 127.0.0.1)，使用 80、453 或您自己配置的通讯端口，请务必确保没有被防火墙等软件禁用。连接时一般需要传递以下参数：
A、sid 为本次连接的唯一会话标识 SessionID，可用数字或字符串类型，只需确保同时在线的每个连接是唯一的即可，需要注意的是，sid 不能包含& = ? ” :等特殊字符；
B、flag 为连接标识掩码，设置为 1 表示输出调试日志，设置为 2 表示通讯采用 GZip 压缩(如设置了压缩，前端收发数据包需要自行进行压缩或解压缩)；
C、lang 为请求语言版本，默认为简体中文 CHS，国内版只支持简体中文版，海外版支持英文版 ENG。

连接成功后，即可在输入框中填写对应的 JSON 组合命令包后点击发送，通讯记录中即可接收 PluginOK 给出的同样是 JSON 组合的响应内容。所有中间件系统内置的协议名称都是以 Wrl_开头的。需要注意的是：请求的 JSON 数据包，请确保是 UTF-8 的编码。

1) {"req":"Wrl_Version","rid":1,"para":{"Mac" : 1,"More" : 1,"Config" : 1,"Router" : 1,"IP" : 1}} 获取 PluginOK 版本及当前系统等信息，其中 Wrl_Version 为功能请求标识，协议中的 rid 代表此请求序号(长整型)，请求返回的结果 json 中也会带这个参数，因为 ws 请求和返回是异步执行的，这样可以确保请求和结果的一一对应关系，下同。Mac 指定是否获取本机 MAC 地址等信息，1 代表获取(如 WrlConfig.json 中配置 Mac 为 0 代表禁用此功能)；Router 为 1 时代表获取当前电脑所在网络网关的 MAC 地址，IP 为 1 时获取当前网络路由器 IP 地址和本机 IP 地址；More 取更多信息(当前登录用户、键盘和鼠标空闲时间、当前电脑制造商及型号等) ，Config 为 1 时获取中间件自身的一些配置参数。执行后在网页中将收到类似如下内容：

{"ret":0,"rid":1,"data":{"Version":"2.2.16.8","ValidDay":"2025-12-31

12:00","AuthName":" 成 都 佐 罗 软 件 有 限 公 司 ","Type":8,"OS":"6.2.9200.2","Win64":1,"UniID":"AF4D1A1623D03399E3E2F63449CF9B18","Time":9182568,"TickCount":163797640,"RunPath":"W:/WRL/","Mac":[{"Card":"Qualcomm QCA9377 802.11ac Wireless Adapter","Mac":"AC:6C:68:89:EC:9B"}],"Router":"GH:6Y:88:09:E6:3B","IP":[{"LIP":"192.168.1.18","RIP":"192.168.1.1"}],"Config":{"NoService":0,"Localhost":1,"Monitor":60,"Mac":1,"UDisk":0,"ThreadSize":3,"MaxThread":6,"DownSpeed":8,"NetOverTime":30,"Lang":"CHS"},"More":{"ComputerName":"hello","Manufacturer":"Dell Inc.","ProductName":" 3590","LoginUser":"test" }}}

ret 为请求返回值，0 代表成功，大于 0 的值表示请求发生错误，在 data 中会输出具体的错误描述，rid 为对应请求的序列号，客户端可通过对应此序号识别是哪个请求返回的内容。所有请求的数据信息，都会在 data 数据节点中返回，如果是数组，会以[]这样的节点返回(下同)。这里返回的 UniID，请反馈给客服人员以便开通试用权限。

Version 为当前 PluginOK 的版本号，Lang 为默认支持的语言版本，Valid 为当前授权有效期，AuthName 为授权的客户名称，Type 为授权类型，OS 为当前系统的版本，UniID 为当前终端电脑的唯一标识，Time 为当前系统的时间。如果有需要，可以返回更多的内容，比如当前系统的硬件配置等。

2) 请求更新中间件授权文件：

单机版需要分别指定授权文件在服务器上的地址及文件 MD5 校验码、文件大小和下载验证权限 Auth 及 Cookie，其中 Cookie 和 Auth 可为空，根据自己网站情况设置，网络版 para 中不需要任何参数，如果终端电脑和授权服务器网络通畅，也会每天同步一次使用授权。举例：

{"req":"Wrl_UpdateAuth","rid":2,"para":{"Url" : "http://local.zorrosoft.com/Files/Update/wrlauth.pid", "MD5":"8BBCD7EAD95EFC034B724C4D8A961C03","Size":262144,"Cookie" : "","Auth" : ""}}

说明：连接 PluginOK 服务建立的 WS 侦听端口后可执行，可下载 Url 指定的授权文件并更新到本地，实现不升级程序而更新中间件的授权；

3) 指定浏览器打开指定 Url：

分别指定浏览器类型Type(1代表IE、2代表Chrome、4代表Firefox、5代表Brave、8代表Opera、9代表Vivaldi、16代表Edge、20代表Electron、32代表360极速浏览器、33代表360安全浏览器、34代表360企业安全浏览器、40代表联想浏览器、50代表QQ浏览器、51代表微信网页浏览器、55代表奇安信可信浏览器、57代表红莲花安全浏览器、60代表搜狗浏览器、70代表华为浏览器)和Url参数；

Flag意义：IE中是指定浏览器版本，Chrome等浏览器0代表标签页打开，1代表新窗口打开；

Show显示窗口类型，默认1正常显示，2最小化显示 3最大化显示。

Url中有中文等特殊字符时，需要先进行UrlEncode编码。

举例 1：

强制用 IE9 兼容模式内核浏览器打开 http://www.zorrosoft.com 网站，9999 是 IE9 标准模式

{"req":"Wrl_OpenUrl","rid":3,"para":{"Type":"1","Url":

"http://www.zorrosoft.com","Flag":"9000"}}

举例 2：

Chrome 浏览器新窗口打开 http://www.zorrosoft.com

{"req":"Wrl_OpenUrl","rid":3,"para":{"Type":"2","Url":
"http://www.zorrosoft.com","Flag":"1"}}

4) 请求使用关联程序打开指定文件：

指定本地文件全路径，使用本地电脑默认关联的程序打开，路径用双斜杠\\或反斜杠/。如是中文及包含特殊字符的路径，需要先进行 UrlEncode 编码。

为安全起见，这些扩展名文件不支持：*.exe *.msi *.js *.jar *.inf *.com *.bat *.vbs *.py，如实在需要支持这些文件的启动，请购买单独发行的文件操作小程序授权，具体请微信 ZorroSoft 咨询客服。

举例：

{"req":"Wrl_OpenFile","rid":4,"para":{"File" : "E:/WRL/Ver/TestWrl.txt"}}

说明：连接中间件服务后可执行；

5) {"req":"Wrl_Restart","rid":5,"para":{"TK":"8EE0F6A89E301225DDD556DAEE460111F59EA39EB08EC5DC985725C33AFEACAB7076097D786164208D501EDD95F1CBDCADF174B6B005ECFB852BF6502A8CCAD1A8EC8C43A02DDF0B8932C4AB9F4D9E87F98ADAEA09AB5C4048FDDF4CF1E827CB4348B749470EE671E1B49251BFD6FCD4B490B7F828F3844C2EBED529588C90482A0E6E07E21137986153E9E0AECE68F3D61FC1F21639D97E11A0B94B3F5B2B23F16C40A6B17568B6BD2716AB7323D34BBC941CD80A833527320AA79E2F79A57659AADB7AB7B26D28593B6F415C1EF0EE033B35B3D723F98BECF0C334BB8C05DFF81104F84F2514D0E878BC8936D1962504939618EE43474438E5AC72E1E998D2"}} 请求重启 PluginOK 主程序，需要验证 TK 信息，TK 由打包工具生成，默认 24 小时内有效。

6) {"req":"Wrl_Remove","rid":5,"para":{"TK":"630035EFB20C2E5433722C36BCBD2F11A5658041E0782837C9E7D9417186A20E8079EF9E70D536E47FC14AA9FE8E5726B7F4E5EB685AED4DC5C0D519ACE64BA2037D0569FFCDCA8AE31718D9B193C049581077898C6B7A3A8467EED0D44A7F02390E22AADBB82C8907EB3E4E1ACAC056A7C35A8444497B7A2E742627C084D99A5EB98AEB5164B237584B6DAAED75D64FBB5A1DEC8A832CCDCC589EA94012BF55B4AA327B7B09E72106A52281D2C6030CD3B3C4621C20270B33FA9B6738ED5042410ED94B3148B835BA19619438422B00EF2F4B3944142751D7F94A477C33FB42170CE337F09D80AA0021110CA70F588959BD2ED4F9240CB07E3A456D29945B52"}} 请求卸载 PluginOK 中间件，需要验证 TK 信息，TK 由打包工具生成，默认 24 小时内有效。

7) 请求重新注册整个中间件或单个小程序：

如指定 PID 代表重新注册指定的小程序，否则执行重新执行整个中间件的安装(连接会断开)，用于修复 PlguinOK 及相关小程序的功能。

举例：

{"req":"Wrl_RegCom","rid":7,"para":{"PID" :
"A22E18F1-95F8-4FDB-99D2-188E5FB12B23"}}

或者 {"req":"Wrl_RegCom","rid":15,"para":{"NoService" : "1" ,"Localhost" : "0","NeedAdmin" : "0"}} NoService 为 1 代表以非系统服务方式重新安装中间件 Localhost 为 0 代表用本机实际 IP 来侦听，方便局域网其他电脑访问 NeedAdmin 为 1 代表设置个别电脑高级版需要管理员权限才能正常运行。中间件重新注册时需要 TK 校验，生成规则和重启服务请求一致，可由小程序打包工具生成。
说明：连接中间件服务后可执行；

8) {"req":"Plugin_List","rid":8,"para":{"Detail" : 0}} 获取当前已安装的所有小程序列表信息，Detail 为 1 时返回小程序全路径，返回内容举例如下：

{"ret":0,"rid":10,"req":"Plugin_List","data":[{"Type":2,"PID":"18BDC030-AF37-50E1-B0AE-E5EF336BE281","Version":"1.5.16.2","Name":"IE 标签小程序","ValidDay":"2025-07-17"},{"Type":2,"PID":"18BDC030-AF37-50E1-B0AE-E5EF336BE282","Version":"1.5.16.2","Name":"IE 内核内嵌小程序","ValidDay":"2025-07-17"},{"Type":2,"PID":"99225C6D-B7A3-441c-AEFB-3EE23ACA2209","Version":"1.5.16.2","Name":" 弹窗小程序范例(C# 语言)","ValidDay":"2025-12-31"},{"Type":1,"PID":"A22E18F1-95F8-4FDB-99D2-188E5FB12B23","Version":"1.5.16.2","Name":"DLL 小程序范例(C++ 语言)","ValidDay":"2025-12-31"},{"Type":1,"PID":"C0B01CD6-7DD9-4D3C-B668-04168D5236FC","Version":"1.5.16.2","Name":" 串口通信 Chrome 小程序","ValidDay":"2025-12-31"},{"Type":1,"PID":"D10495F4-DF0F-44FA-8647-91275347214A","Version":"1.5.16.2","Name":"DLL 小程序范例(C# 语言)","ValidDay":"2025-12-31"},{"Type":2,"PID":"E7C7BDA6-C828-46F1-A7BA-B4C572A01100","Version":"1.5.16.2","Name":"Flash Player 小程序","ValidDay":"2025-07-17"},{"Type":2,"PID":"F90B1CF0-8485-40ec-B4E8-B87598AAB35D","Version":"1.5.16.2","Name":" 弹窗小程序范例(C++ 语言)","ValidDay":"2025-12-31"}]}

Type 为小程序的类型(1 代表 DLL 小程序，2 代表弹窗小程序 8 代表内嵌网页小程序)，PID 是小程序唯一标识，Version 是小程序版本，Name 为小程序名称。

9) {"req":"Plugin_Exist","rid":9,"para":{"PID":"F90B1CF0-8485-40ec-B4E8-B87598AAB35D"}} 判断指定的小程序是否存在，参数 PID 为小程序唯一标识，如果小程序存在，正常返回类似如下内容：

{"ret":0,"rid":9,"data":{"Exist":1,"Version":"1.5.16.2","Name":"Windows 程序小程序范例","ValidDay":"2025-01-01"}}，Exist 为 1 标识存在。

10) {"req":"Plugin_Install","rid":1,"para":{"Name":"DLL 小程序范例(C++ 语言)","PID":"A22E18F1-95F8-4FDB-99D2-188E5FB12B23","Date":"2024-07-29","Desc":"无窗小程序(C++语言)-读写 TXT 记事本及获取当前系统信息","DownAddr":"http://local.zorrosoft.com/Files/Plugin/PluginComDll.pid","MD5":"DBED2F821A041B8BE40BBE8531CF8E37","Version":"1.5.16.2","Size":589824,"HideIns":0,"Type":1,"Cookie":"","Auth":"","TK":"7A1D56A154D25C6735A7BE41A1DA50711803BE2C655F5424A6F6A07F63BB42664C584F3BE3A22B826410975C540396E28CF4644456DBA8D4C103BE2945ADD2C10E21275F47347D445591626974B40E1A277D250E95680247BC1D9B5F087E531D6BB7F0CA64B876EB74ABEB4

BDB2C9284367C3BA1A6C1C02CC6A0F63FC26EEAF17105F9DE9CED1E59AEC77D
C3E3345E6F380F2A8424747630A918759D5C9FC66ABAFFB22C6741C7BC0BBE01
CCD849E3F77183010C0F0C685E2BCD53AF890C2183E7358AA6D2AC3E176F3925
B11CC5C375221E8563A241A80B3CB712643BD41850D54AEA4DC41DEB9B24496
600ACE629780675575815E1289DB4114D7A58B1FD1B"}}

请求安装指定的小程序，安装参数信息由打包工具自动生成，只需要编辑小程序包实际的下载地址，提供下载的服务器需要开通 pid 这个文件类型的下载。

11) {"req":"Plugin_Update","rid":1,"para":{"Name":"DLL 小程序范例(C++语言)","PID":"A22E18F1-95F8-4FDB-99D2-188E5FB12B23","Date":"2024-07-29","Desc":"无窗小程序(C++语言)-读写 TXT 记事本及获取当前系统信息","DownAddr":"http://local.zorrosoft.com/Files/Plugin/PluginComDll_Update.pid","MD5":"B9B6E667AD5171076C03116ADE8F872D","Version":"1.5.16.2","Size":589824,"HideIns":0,"Type":1,"Cookie":"","Auth":"","TK":"C37F3B9B929EB915E6D81F8D5708E6940B30D1E848848A11CF58E61471E57F8FEE89A05B2E7F108F6F84D779CC3F5DCBA7AE6DD55E5DF473EBF098707EADA7AC2E2ED17AF6B80A37529148D4FCCCE58B847D73599D76A791F8A3DE3ECEEF62CE1EE61D15CEF5D8EB944B52162E5DA60175CFE2716D0C78AC9AE35A266E278B69DC6764A79A19EE0869412F78CA3FC60BD17BAD653037ED9C3486F583EA3AFF4C04AE5DD271D58AA69A0078AA62498638931DF70F297B718AA2AB11C36BA1EBB1FBAABC0D7B1CA9B7701554BD84836C71F0BEAE0B27263B946A78E56F5AB561AFD2F112495FD8CC7ED2873E8EE7DC41105BFA1CB954A842AA5650DFC953B1FE9E"}}

请求升级指定的小程序，安装参数信息同样由打包工具自动生成，只需要编辑小程序包实际的下载地址，可用于小程序的增量更新。

12) {"req":"Plugin_Remove","rid":8,"para":{"PID":"99225C6D-B7A3-441c-AEFB-3EE23ACA2209","Type":1,"TK":""}} 卸载指定 PID 的小程序，需要同时指定小程序的类型及安全校验的 TokenKey，TK 可由小程序打包工具生成。

13) PluginOK 自身的升级请求举例：

{"req":"Wrl_Update","rid":1,"para":{"Name":"PluginOK 中间件单机版升级包","Date":"2024-10-13","Desc":"优化中间件高级版启动内嵌网页小程序处理过程，解决个别情况启动失败问题，优化内嵌网页小程序的窗口激活及输入焦点处理，优化 WS 连接释放过程...","DownAddr":"http://local.zorrosoft.com/Files/Update/Wrl_Update.pid","MD5":"B5E4E23E4E0F0C42CFC68747F2B1CD73","Version":"1.5.16.7","Size":13205504,"HideIns":0,"Cookie":"","Auth":"","Open":"","TK":"BD361D3E30D56C841CC18411CA9E3877950D9E88CCDD6271FA9352733A810FDC378E38741B780E096E7C9729FEEE5A58D7AADFF8D5A5D9DA5740B6B55DAE62510F51EF53D0AA1D25A98C57F7E2F7A1BFDE44005F1E95880F5D22C5D9A2FB03E06E44BAA5CF20555EF04FC871002C431E0BD794993848A55285AE141115257660111197F2CBC9BC8E210508D219F3A22D93435E48F0FE635044AE4D293D95D00969EDE6EA8D0960360E1A328915F27E44533F0A85F9C4B8CC8C66707CC7A332A052AA0A1C64C100DBDC575CA2F7B2BED1194419D2FD2AB8FB5A218CE3435511B58AE33AA6522247757BC5F41A72E3E8E3A1309F85B3791B157718CAF5432

13F58"}} 此升级请求可在 PluginOK 打包工具中制作生成。

14) 请求刷新指定小程序
使用场景：不用重启 PluginOK 服务即可扫描新增的小程序使用；如小程序开发调试时，程序及配置更新需要重新加载时 举例：
{"req":"Plugin_Refresh","rid":18,"para":{"PID" : "A22E18F1-95F8-4FDB-99D2-188E5FB12B23"}}
说明：连接中间件服务后可执行，PID 为小程序唯一标识；

15)、如中间件服务没有启动，前端可请求启动中间件，网页中点击以下链接，可直接启动中间件。
点击这里启动 PluginOK 单机版
点击这里启动 PluginOK 网络版

16)、移除本机中间件授权并卸载程序(网络版有效)：
如指定 PID 代表重新注册指定的小程序，否则执行重新执行中间件安装批处理(连接会断开)，可能因为一些工具或手工误删除了注册表等，用于修复本中间件使用配置。正式版需要添加 TK 校验，具体规则请参考 SDK 包中文档“中间件安全解决方案.pdf”
举例：{"req":"Wrl_RemoveAuth","rid":23,"para":{}}
说明：连接中间件服务后可执行，某终端用户不在使用时调用此接口卸载程序并释放占用的授权数量。

17)、前端请求配置参数
UDisk 是否监控 U 盘动态
Adjust 矫正参数，个别电脑小程序运行时位置错误，需要单独配置
Monitor 配置服务无响应超时时间，一旦超过服务会自动重启
以上参数可同时配置，也可以只配置一个参数
举例：{"req":"Wrl_Config","rid":21,"para":{"Monitor" : 30,"Adjust" : 0,"UDisk" : 1}}
说明：连接中间件主服务后可执行。

18)、针对 HTTPS 网站，需要搭配 SSL 证书使用，而 SSL 证书是有期限的，需要定期导入 SSL 新证书到授权文件中使用：
{"req":"Wrl_UpdateSslCert","rid":3,"para":{"Path" : "G:/SSL"}}
如果是网络版的话，请在授权服务器上打开测试网页进行，并且修改连接按钮上方连接地址中的端口参数，默认 83 改为 800 后连接成功再发送。
执行前需要把新的 SSL 证书放到单独的一个子目录，然后执行此命令，成功时返回 SSL 证书绑定的域名。授权文件更新后请替换到终端电脑需要安装的软件包中使用。

那么如何在前端实现 PluginOK 及相关小程序的下载或升级安装呢？您首先需要在网页中请求连接 PluginOK 服务的 Web Socket 侦听端口，如果无法连接，可提示用户下载并安装 PluginOK 的程序包，如果用户已安装无法连接，可尝试前端请求启动中间件中的方法，另外请确认连接端口是否正确、并提示重启电脑

再试及检查程序是否被 360 等杀毒或安全防护软件拦截运行。连接上后，即可通过上面的接口提供的信息来判断小程序是否存在及是否需要升级，获得指定小程序的安装信息，如果有必要，可提示用户安装或升级小程序。

PluginOK 还实现了 U 盘当前已加载和热插拔事件通知，一旦服务启动完成连接，会自动通知当前插入的 U 盘信息，U 盘的热插拔动作就会被捕获听通知给存在的客户端连接，比如可能会收到类似如下通知内容：
{"event":"Wrl_UsbEvent","data":{"Driver":"H","DID":"VID_0951&PID_1642\001CC0EC3519FB71A71823A3","Type":32768}} 这是标准的事件通知协议格式，event 代表事件协议，Wrl_UsbEvent 标识 U 盘热插拔事件。Driver 为本次插入的 U 盘对应的盘符，DID 为此设备的唯一 ID，Type 标识是插入还是拔出。

PluginOK 同时实现了小程序的安装及升级过程中的事件通知，这种网页中可随时输出安装及升级状态。

2、连接请求指定的小程序服务，协议举例如下：
ws://wrl.zorrosoft.com:83?sid=321&flag=1&lang=CHS&pid=C38672FA-B5C8-4D9D-89B5-2D71F0760661&cid=zorrosoft&tk=

和请求 PluginOK 中间件服务相比多了几个请求参数，其中 pid 为请求的小程序配置文件中的 PID(如是无界面交互的 DLL 小程序，是 Objects 下面的某个组件 ProgID)，cid 为请求的客户标识，由本中间件开发商负责分配(需要时请微信联系 ZorroSoft)，tk 为调用验证权限的加密字符串，具体规则由另外的文档进行说明。小程序授权为开发类型时 tk 为空也可以通过，方便开发阶段的调试。如果请求的小程序不存在、tk 加密字符串不合法或授权过期等，都会返回请求的错误描述。

一旦连接小程序服务成功后，即可调用小程序模块实现的具体功能了，具体请参考各小程序实现的接口说明文档。关于小程序的功能接口协议，可参考提供的 4 个范例，下面即将介绍如何开发实现在 PluginOK 中间件上的小程序。

四、小程序开发

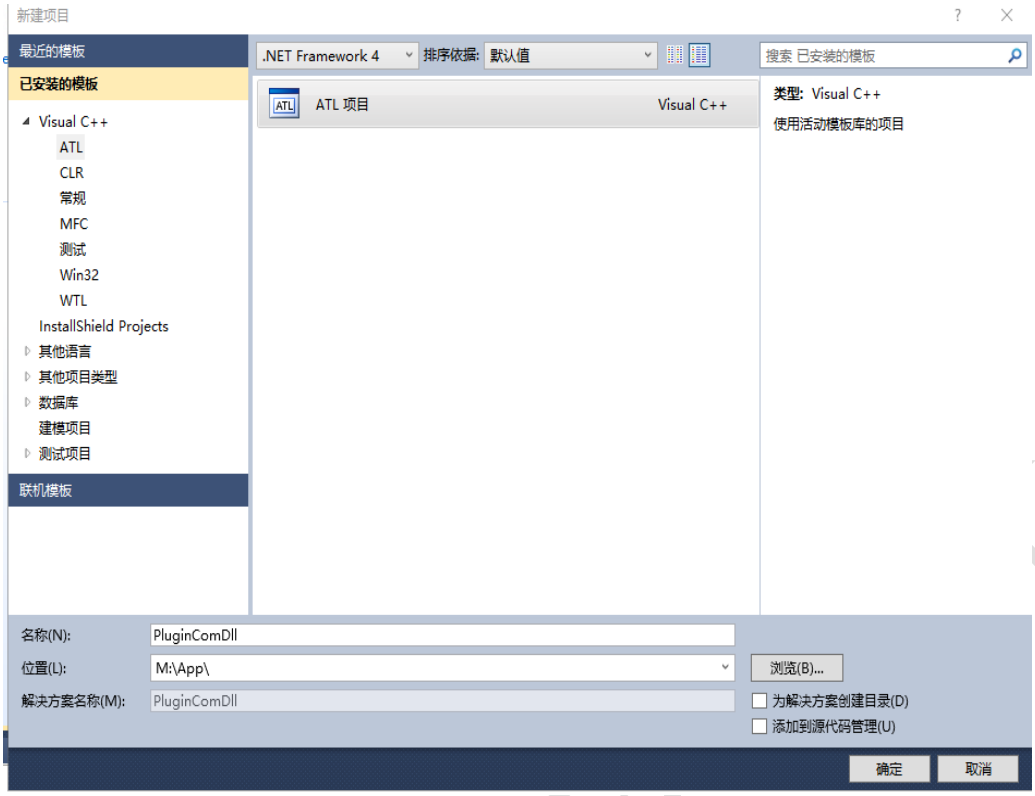
上面已经介绍了 PluginOK 中间件的诸多功能，您是不是急切的想知道如何开发并实现一个自己的小程序呢，^_^。在开发自己的小程序之前，请先熟悉一下中间件为您准备的诸多强大接口支持，参考文档“PluginOK 核心组件接口说明.xls”，这里除了小程序本身的接口之外，还提供了 Web Socket、JSON、Sqlite 等支持库的 COM 组件支持，也提供了在系统服务中启动一个桌面进程的支持。下面主要以开发 DLL 小程序和弹窗小程序为例说明。

1、DLL 小程序开发流程：

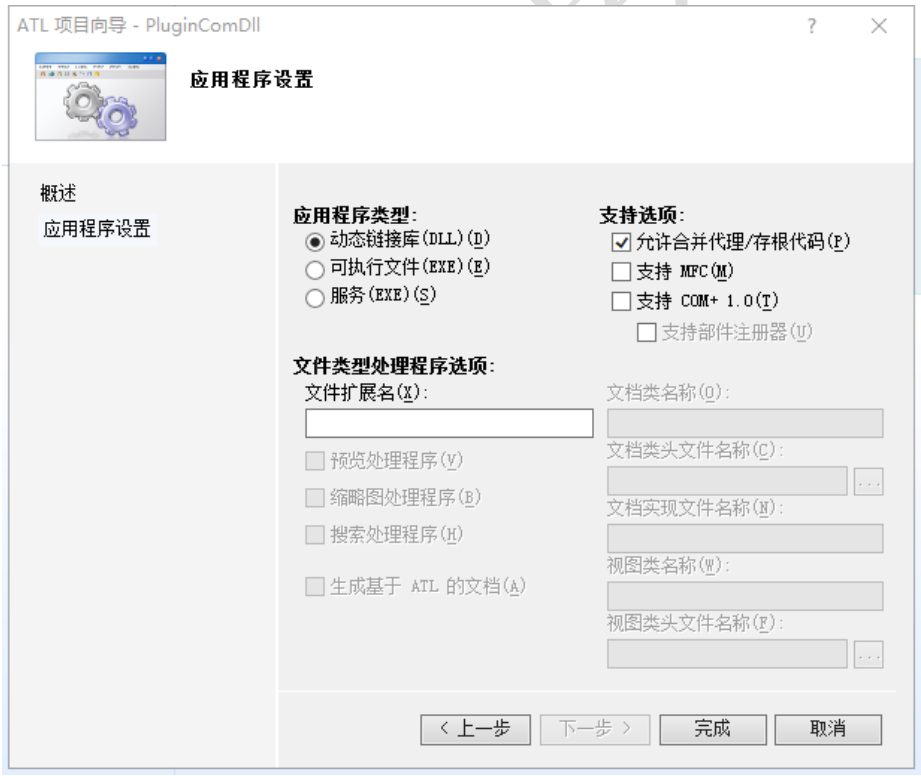
由于这种小程序主要是为解决在网页中操作本地硬件准备的，而硬件的驱动程序大多是 C/C++语言开发的，所以下面以 C++语言为例说明如何开发此类型的小程序。

1) 、小程序工程创建

启动 Visual Studio 开发环境，推荐使用 2010 版，然后点击新建项目，如下图所示：

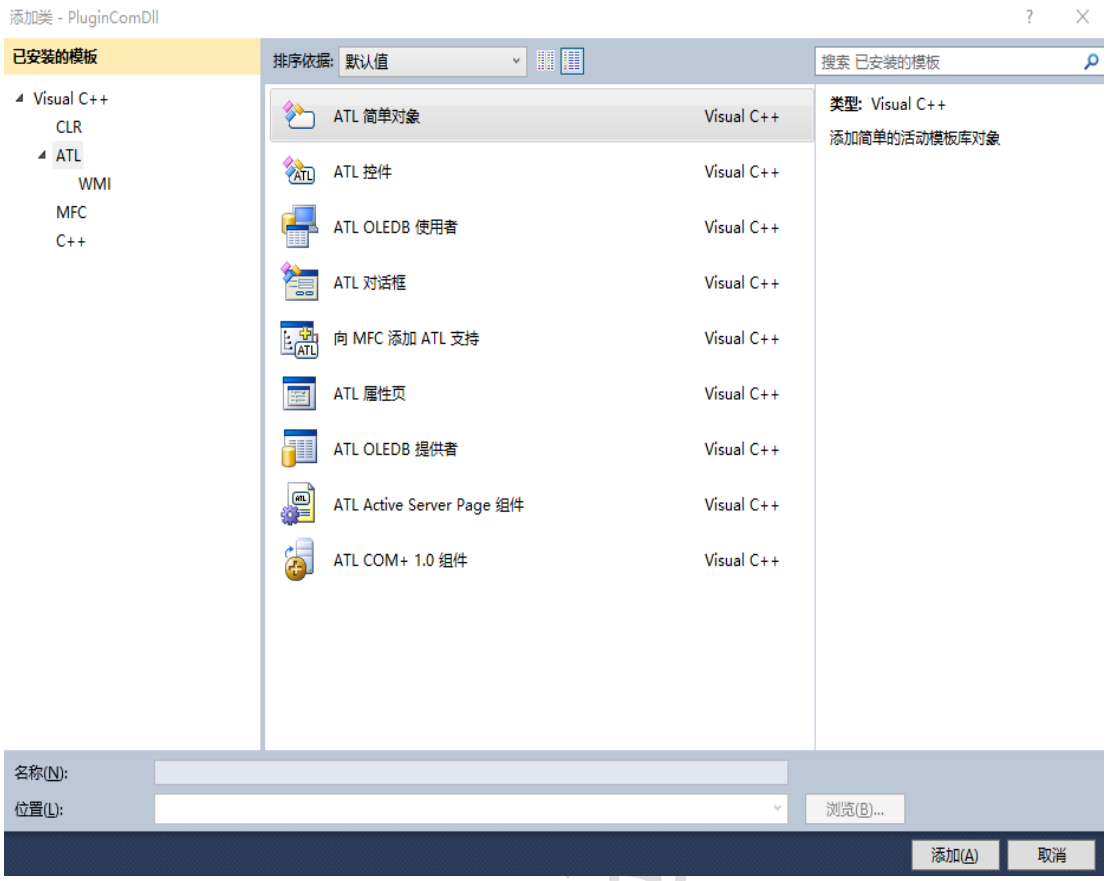


确定后点击下一步继续，界面如下：



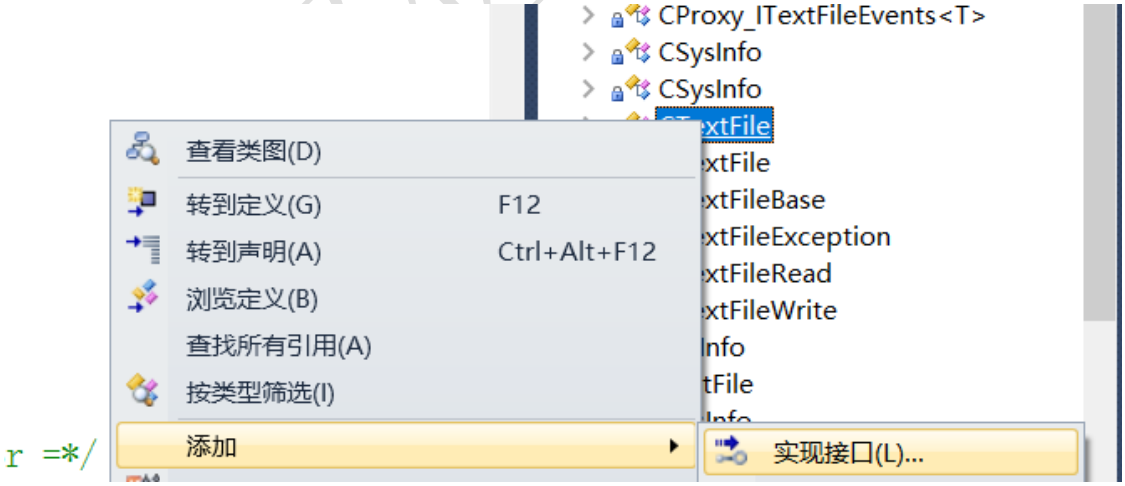
点击完成，工程就创建好了。

选中刚创建的工程，右键点击添加->类，弹出如下界面：

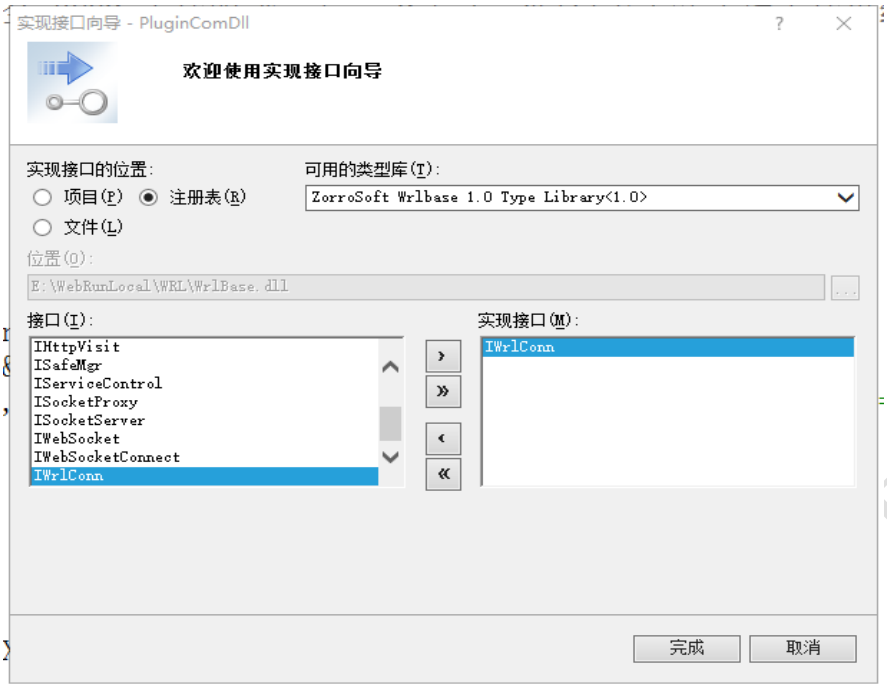


选中 ATL 中的 ATL 简单对象，点击添加，然后再输入一个 COM 组件接口名称如 TextFile，然后点击完成即可。这样 COM 组件的一个对象类就创建好了。

然后在解决方案资源管理器中，切换到类视图选中 CTextFile 类，右键点击添加中的实现接口功能，如下图所示：



然后在弹出的界面中，选择注册表可用的类型库找到网络版类型库 ZorroSoft ZbaBase 1.0 Type Library(单机版是 ZorroSoft WrlBase 1.0 Type Library)或直接勾选文件，找到中间件安装目录下的网络版 ZbaBase.dll(单机版是 WrlBase.dll)，在接口列表中找到 IWrlConn 并添加到右侧列表中后完成，如下图所示：



小程序接口所需要实现的方法都列出来了。请根据自己的需要进行实现。此范例的详细代码工程请参考 SDK 开发包中的 PluginComDll 工程。

2) 、DLL 小程序需要实现的接口说明：

STDMETHOD(Load)(LPDISPATCH piDispatch, BSTR bstrAuthInfo, BSTR bstrLang)

加载小程序，当网页中请求此小程序组件的服务时，PluginOK 服务会创建此小程序组件的实例，并调用此方法，传递一个 IWebSocketConnect 接口(通过查询接口得到)实例到小程序组件中，通过此接口可与网页进行实时通讯了，收发数据包及传输文件等操作。第二个参数 bstrAuthInfo 为小程序的 JSON 格式授权信息(包含小程序的授权类型、授权有效期、请求的客户 ID 及会话 ID 等)，bstrLang 为请求的语言版本。

STDMETHOD(Unload)(EWrlCloseConnType eCloseConnType, BSTR bstrReason)

当网页关闭时，会请求释放小程序时调用此方法。

STDMETHOD(UsbChanged)(BSTR bstrDisk, unsigned long nStatus, BSTR bstrName)

当发生 USB 设备热插拔事件时会调用此方法。bstrDisk 为 USB 设备对应的盘符，nStatus 为状态插入还是拔除，bstrName 为 USB 设备唯一名称

STDMETHOD(SendJson)(BSTR bstrContent)

PluginOK 系统服务调用发送 JSON 数据包方式的命令请求，范例提供了默认实现，直接转调用接口 IWebSocketConnect 中的方法 AsynSendText 即可。

STDMETHOD(RecText)(BSTR bstrContent)

收到小程序调用方发来的普通文本信息。

STDMETHOD(RecJson)(unsigned long nReqID, BSTR bstrPushName, BSTR bstrContent)

收到小程序调用方发来的 JSON 数据包，nReqID 为请求序号，bstrPushName,为请求协议名称，bstrContent 为请求参数内容，小程序中可解析内容并根据请求协议名称执行相关代码。

STDMETHOD(RecByte)(BYTE *pContent, unsigned long nLength,VARIANT_BOOL bMoreFlag)
收到小程序调用方发来的二进制数据流，一般用于传递数据比较大的文件、图片或视频数据流
STDMETHOD(HttpRequst)(BSTR bstrUrl,BSTR bstrPara,BSTR *pVal)
执行 HTTP 请求，保留方法，暂时无用。
在小程序的每个组件中实现这些方法，就代表完成了一个小程序组件对象的开发。与小程序调用方通讯的接口 IWebSocketConnect 还有其它一些功能，以及自带的 JSON 解析器组件使用说明，将在另外的文档中进行介绍。

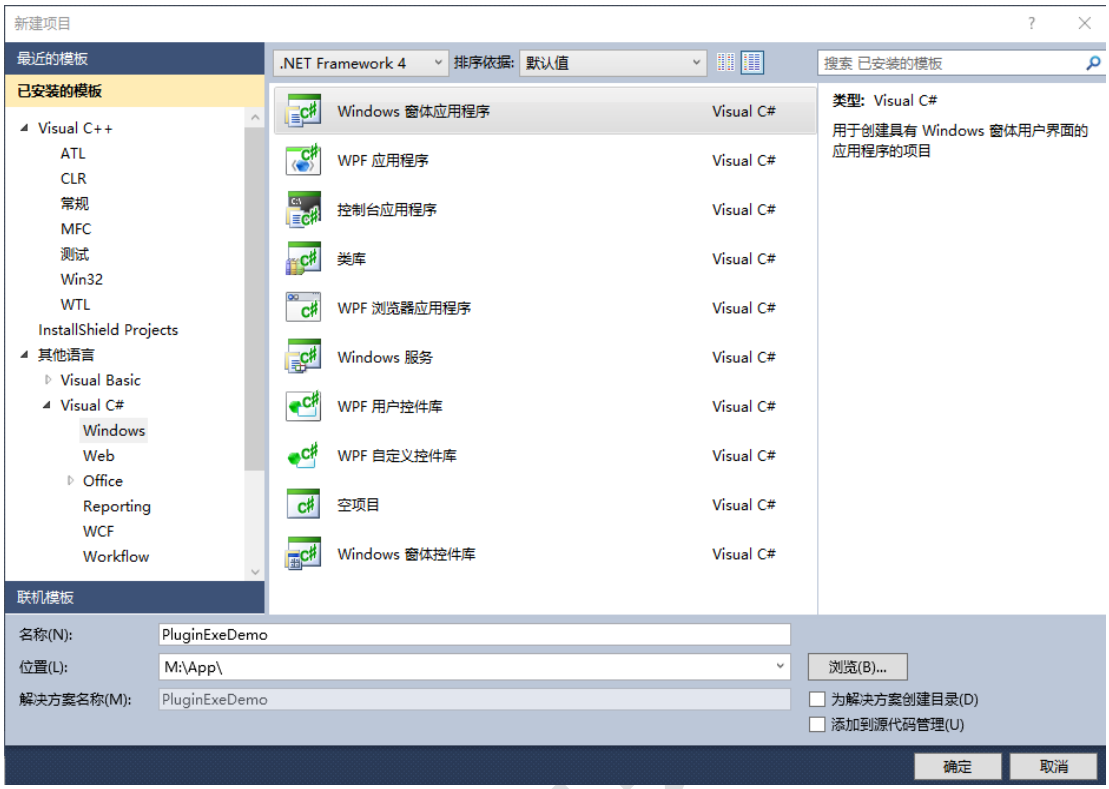
关于C#版DLL小程序开发，请按网络上介绍的方法，创建一个开发C#的COM组件工程：类库类型，然后修改[assembly: ComVisible(true)]为COM可见。添加一个接口并实现，接口和实现类分别定义一个GUID，分别对应小程序配置文件中的ProgID和ClassID，小程序PID用assembly: Guid。具体配置请参考范例PluginNetDll。C#版DLL小程序打包发布时，记得把小程序的DLL模块安装到PluginOK安装目录。

因为安全原因，通过打包工具安装的小程序是不能直接替换程序文件后运行的。而开发小程序过程中需要经常替换文件运行，所以需要用开发者模式(删除打包安装生成的文件PluginAuth.paf，用PluginConfig.json)来测试小程序，具体操作如下：

请在和我们沟通后确定好你的小程序PID(不能和其它人重复)，然后在中间件程序目录Plugins之下再建立PID名称的子目录，把小程序运行程序及其配置文件PluginConfig.json放进去(C#的程序需要放到中间件程序目录，配置文件还是在小程序子目录)。然后在中间件服务连接下，请求注册小程序：
{"req":"Wrl_RegCom","rid":4,"para":{"PID" :
"A22E18F1-95F8-4FDB-99D2-188E5FB12B23"}}，然后再请求
{"req":"Plugin_Refresh","rid":7,"para":{"PID" :
"A22E18F1-95F8-4FDB-99D2-188E5FB12B23"}} 即可在前端通过调用Plugin_Refresh扫描此小程序信息动态加载使用。已经加载的DLL小程序是无法替换的，需要停止服务(管理员权限运行StopWrl.bat)后替换后再启动服务(管理员权限运行InstallWrl.bat)。此类小程序在PluginOK中间件运行于系统服务模式时是无法直接调试的，只能通过输出日志来判断代码运行的正确性。

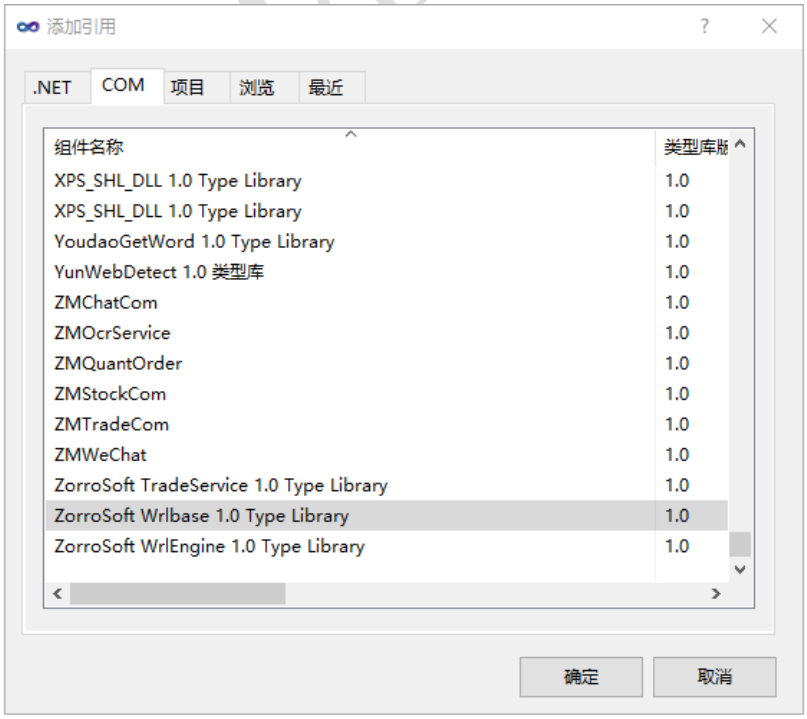
2、弹窗小程序开发流程：

下面以 C#语言为例演示如何开发此类小程序。
1)、小程序工程创建
启动 Visual Studio 开发环境，推荐使用 2010 版，然后点击新建项目，如下图所示：



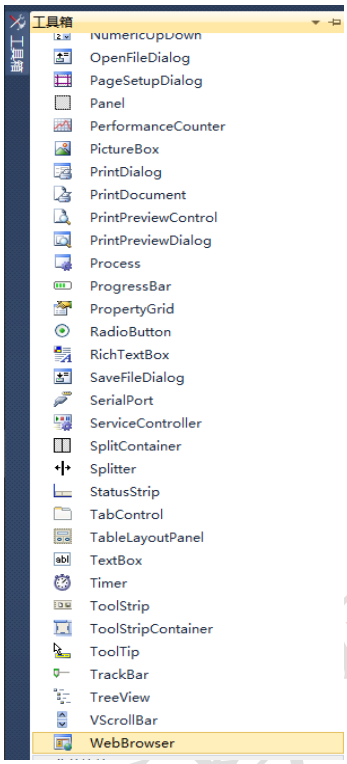
点击完成，工程就创建好了。

点击 VS 开发环境中的项目，然后添加引用，在如下界面选中网络版类型库 ZorroSoft ZbaEngine 1.0 Type Library 和 ZorroSoft ZbaBase 1.0 Type Library(单机版是 ZorroSoft WrlEngine 1.0 Type Library 和 ZorroSoft WrlBase 1.0 Type Library)两个组件并确定：

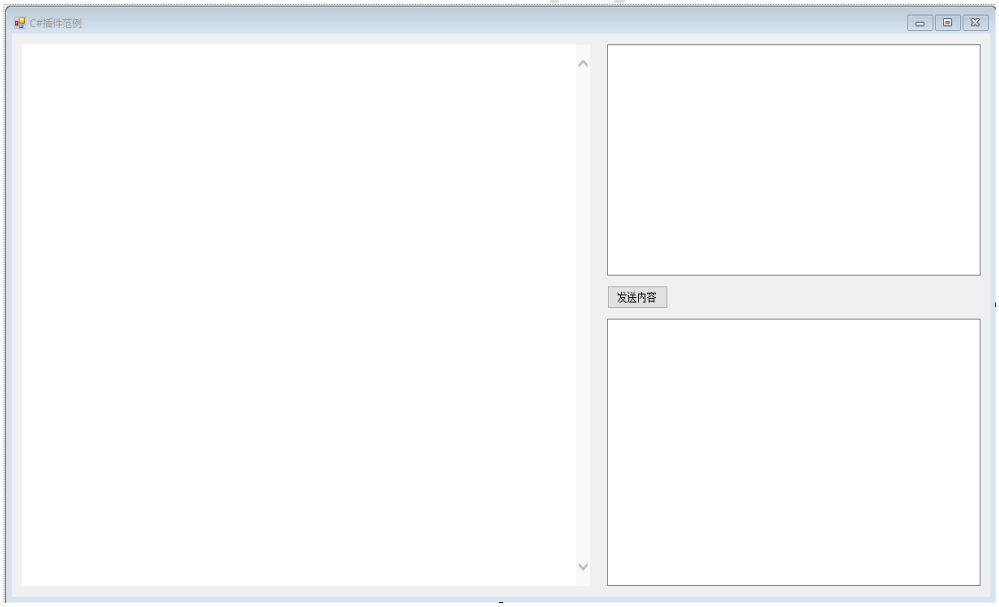


这样工程中就可以调用 PluginOK 中间件核心组件的服务了，工程详细代码请参考 PluginExeDemo。

选中刚创建的工程，在主窗口界面中添加一个 Web Browser 控件，如下图所示：



然后简单调整下窗口大小，添加 2 个文本编辑框和一个按钮，如下图所示：



在 Form1.cs 文件中添加核心组件的应用

网络版使用：

```
using ZbaEngine;
using ZbaBase;
```

单机版使用：

```
using WrlEngine;
using WrlBase;
```

实现 PluginOK 的 Web Socket 服务及事件处理，主要涉及 SocketProxyClass 和 WebSocketEventSink，范例详细代码如下：

```
public partial class Form1 : Form
{
    /// <summary>
    /// web socket 服务对象
    /// </summary>
    SocketProxyClass WebSocketServer = null;
    /// <summary>
    /// web socket 服务事件对象
    /// </summary>
    WebSocketEventSink WebSocketEvent = null;

    /// <summary>
    ///  命令行启动参数
    /// </summary>
    Dictionary<string, string> m_Para;

    public Form1(string strPara)
    {
        InitializeComponent();

        m_Para = new Dictionary<string, string>();

        ///  解析命令行参数
        string[] cmdArray = strPara.Split('&');
        foreach (string cmd in cmdArray)
        {
            string[] paraArray = cmd.Split('=');
            foreach (string para in paraArray)
            {
                m_Para.Add(paraArray[0], paraArray[1]);
                break;
            }
        }
    }

    public void Send(string strSID,string strContent)
    {
        WebSocketServer.AsynSendText(strSID,strContent);
    }

    public void OpenUrl(string strUrl)
    {
        this.IEBrowser.Navigate(strUrl);
```

```
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            WebSocketServer = new SocketProxyClass();
            if (null == WebSocketServer)
                return;
            WebSocketEvent = new WebSocketEventSink();
            if (null == WebSocketEvent)
                return;
            WebSocketEvent.SetForm(this);
            ushort nPort = ushort.Parse(m_Para["PORT"]);
            ushort nListenPort = WebSocketServer.Listen(nPort, m_Para["SID"],
m_Para["AI"]);

            /// 建立事件通知
            WebSocketServer.NewConnEvent += WebSocketEvent.NewConnEvent;
            WebSocketServer.RecMsgEvent += WebSocketEvent.RecMsgEvent;
            WebSocketServer.RecTextEvent += WebSocketEvent.RecTextEvent;
            WebSocketServer.ConnCloseEvent +=
WebSocketEvent.ConnCloseEvent;
        }

    private void Form1_Closed(object sender, EventArgs e)
        {
            /// 移除事件通知
            WebSocketServer.NewConnEvent -= WebSocketEvent.NewConnEvent;
            WebSocketServer.RecMsgEvent -= WebSocketEvent.RecMsgEvent;
            WebSocketServer.RecTextEvent -= WebSocketEvent.RecTextEvent;
            WebSocketServer.ConnCloseEvent -=
WebSocketEvent.ConnCloseEvent;
            /// 释放对象
            if (null != WebSocketServer)
            {
                WebSocketServer.Close();
                WebSocketServer = null;
            }

            WebSocketServer = null;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            string strLastSID = WebSocketEvent.GetLastSID();
```

```
        if(0 == strLastSID.Length)
        {
            MessageBox.Show("还未有来自网页的连接！");
            return;
        }
        Send(strLastSID, this.textBox1.Text);
    }

/// <summary>
/// WebSocket 服务事件通知
/// </summary>
public class WebSocketEventSink : _ISocketProxyEvents
{
    /// <summary>
    ///   主窗口
    /// </summary>

    Form1 m_Form;
    string m_strLastSID;

    public void SetForm(Form1 Form)
    {
        m_Form = Form;
    }

    public string GetLastSID()
    {
        return m_strLastSID;
    }

    /// <summary>
    /// 通知新连接
    /// </summary>
    /// <param name="bstrSID"></param>
    public void NewConnEvent(string bstrSID)
    {
        m_strLastSID = bstrSID;
        m_Form.textBox2.AppendText("收到新连接：");
        m_Form.textBox2.AppendText(bstrSID);
        m_Form.textBox2.AppendText("\r\n");
    }

    /// <summary>
    /// 通知连接收到 JSON 数据包
```

```
        /// </summary>
        /// <param name="bstrSID"></param>
        /// <param name="nReqID"></param>
        /// <param name="bstrReqName"></param>
        /// <param name="bstrContent"></param>
public void RecMsgEvent(string bstrSID, uint nReqID, string bstrPushName, string
bstrMsg)
        {
            m_Form.textBox2.AppendText("收到新数据包，请求序号：");
            m_Form.textBox2.AppendText(nReqID.ToString());
            m_Form.textBox2.AppendText("协议名：");
            m_Form.textBox2.AppendText(bstrPushName);
            m_Form.textBox2.AppendText("内容：");
            m_Form.textBox2.AppendText(bstrMsg);
            m_Form.textBox2.AppendText("\r\n");

            if (bstrPushName == "Demo_OpenUrl")
            {
                /// 获得打开 URL 地址，调用浏览器打开
                JsonServiceClass JsonService = new JsonServiceClass();
                JsonService.ParseString(bstrMsg);
                string strUrl = JsonService.GetStringValue("url");
                JsonService = null;
                m_Form.OpenUrl(strUrl);
                return;
            }
            /// 回传给网页内容
             m_Form.Send(bstrSID,"收到请求" + bstrPushName);
        }

        /// <summary>
        /// 通知连接收到文本内容
        /// </summary>
        /// <param name="bstrSID"></param>
        /// <param name="bstrText"></param>
        public void RecTextEvent(string bstrSID, string bstrText)
        {
             m_Form.textBox2.AppendText("收到文本内容：");
            m_Form.textBox2.AppendText(bstrText);
            m_Form.textBox2.AppendText("\r\n");
            /// 回传给网页内容
            m_Form.Send(bstrSID,"收到文本内容" + bstrText);
        }
        /// <param name="bstrSID"></param>
```

```
                /// <param name="Content"></param>
                /// <param name="nLen"></param>
                /// <param name="bMoreFlag"></param>
                public void RecByteEvent(string bstrSID, Object Content, uint nLen,bool
bMoreFlag)
                {
                }
                /// <summary>
                ///  通知关闭连接
                /// </summary>
                /// <param name="bstrSID"></param>
                public void ConnCloseEvent(string bstrSID)
                {
                }
               /// <summary>
                ///  通知HTTP侦听端口
               /// </summary>
               /// <param name="nPort"></param>
               public void HttpPortEvent(ushort nPort)
               {
               }
               /// <summary>
               ///  通知 HTTP 同步请求处理
               /// </summary>
               /// <param name="bstrSID"></param>
               /// <param name="bstrProtocol"></param>
               /// <param name="bstrUrl"></param>
               /// <param name="bstrPara"></param>
               /// <param name="pVal"></param>
               public void HttpReqEvent(string bstrSID, string bstrProtocol, string
               bstrUrl, string bstrPara, out string pVal)
               {
               }
               /// <summary>
               /// WS 连接请求中出现错误
               /// </summary>
               /// <param name="bstrSID"></param>
               /// <param name="nReqID"></param>
               /// <param name="bstrErrInfo"></param>
               public void RecErrEvent(string bstrSID, uint nReqID, string bstrErrInfo)
               {
               }
               /// <summary>
               /// WS 收到操作码，如收到文字、字节流等
```

```
        /// </summary>
        /// <param name="bstrSID"></param>
        /// <param name="nOpCode"></param>
        public void OpCodeEvent(string bstrSID, ushort nOpCode)
        {
        }
    }
```

在程序入口，需要接收 PluginOK 服务传递过来的命令行参数，实现代码如下：

```
        static void Main(string[] args)
        {
            if (args.Length <= 0)
            {
                MessageBox.Show("请输入启动参数");
                Application.Exit();
                return;
            }

            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1(args[0]));
        }
```

从范例代码实现的基本逻辑：PluginOK 中间件提供了一个 Web Socket 服务组件，通过 PluginOK 服务传递过来的参数创建此服务组件成功后，网页端即可与此服务组件进行通讯了。然后编译生成 X86 的程序即可。

此类小程序调试时，也是需要把程序 exe 目标生成到其对应小程序的目录，并放上 PluginConfig.json 配置文件，然后在程序调试运行的命令行配置上设置类似这样的参数：PORT=80&SID=111，其中第一个是小程序侦听端口号，SID 为前端传入的会话 ID 来启动调试。C++的弹窗小程序范例参考工程 PluginWinExe。

3、内嵌网页小程序开发流程：

1)、内嵌网页小程序的接口，主要实现在程序库 ZbaApplet.dll(网络版，单机版是 WrlApplet.dll)中提供，是本中间件高级版实现的最核心功能。和弹窗小程序开发相比，需要额外引入此程序库，接口说明请参考文档“PluginOK 核心组件接口说明.xls”。

内嵌网页小程序的开发，和弹窗小程序大体相似，不过是在弹窗小程序基础之上，需要额外调用内嵌网页小程序的接口，实现窗口小程序以真正内嵌网页的形式运行。已有 C++实现的 WTL 工程范例(演示 Flash Player ActiveX 控件的调用)和 C#的 WPF 工程范例，此范例工程代码如需要请微信(ZorroSoft)联系客服，获得源码的客户请严格遵守我们的保密规定，仅限于内部使用，不得外泄，否则一旦发现，本公司有权要求赔偿。

2)、前端请求启动内嵌网页小程序方法，PluginOK 官方发布的一些小程序，提供了单独的请求启动方法，具体参考对应小程序的文档。第三方开发的小程序，请用以下方法启动：

请求启动内嵌网页小程序通用方法 Wrl_AppletStart，前提：连接中间件服务后可执行，可用以下参数：

Type 为浏览器类型，传 0 自动判断(前提是当前浏览器已启动并显示在最前端)

Url：加载小程序所在的网页地址，必须包含小程序显示位置和大小信息，不建议使用了

Title：Url 所指向网页标题中的关键词

Web：打开配置(新增方式)，可代替 Url 使用，Flag 值+64 使用此配置，此命令中必须指定 Left、Top、Width、Height 的值，建议改用此参数配置启动

NodeName：小程序在 Html 网页中的元素唯一 ID

PID：小程序唯一 ID，对某个小程序来说是唯一的，一旦确定后不能再修改

Flag：掩码标记：1 指定新标签加载(1 和 16 都不指定时为当前页加载) 2 显示标题栏 4 不自动裁剪越界窗口 8 自动适配网页高度和宽度显示 64 启用 Web 参数 128 防截屏 256 强制显示到副屏 512 允许同一网页加载多实例 4096 调试模式，生成启动自己小程序进程的命令行参数，不实际启动

IframeX 和 IframeY 分别为 iframe 嵌套的横竖偏移坐标

BarW 和 BarH 分别是网页右侧和底部预留区域

小程序实际显示首先会基于网页中 margin 指定的坐标和大小，再根据 IframeX、IframeY、BarW、BarH 设定的值做修正

Open：小程序加载后自动打开的本地地址或远程地址

注意：Open 和 Url 如果有特殊字符或中文等，需要用 URL 编码处理后传递

举例，以启动 Flash Player 内嵌网页小程序为例：

打开测试网页 http://zorrosoft.com/flashframe.html 连接成功后，在发送上方的输入框中复制粘贴以下内容后发送：

{"req":"Wrl_AppletStart","rid":7,"para":{"Type":"0","Title":"Flash 播放小程序","NodeName":"FlashApplet","PID":"FlashWebPlayer","Flag":2,"Left":20,"Top":20,"Width":480,"Height":320,"IframeX":0,"IframeY":210,"BarW":0,"BarH":0,"Url":"http://zorrosoft.com/flashframe.html","Open":"http://zorrosoft.com/Files/test.swf "}}

如需要调试内嵌小程序，需要把启动小程序的 JSON 包中的 Flag 值在正常值之上加上 4096，这时启动小程序的命令行参数会生成到中间件 data 子目录下的 ZbaService.txt(单机版在 WrlService.txt)中，复制命令行参数并设置到 VS 调试命令行配置上，并把小程序的 EXE 生成目标设置到对应的目录下即可。详情参考文档“PluginOK 高级版中内嵌窗口小程序调试步骤.pdf”。内嵌窗口小程序，和弹窗小程序对比，给前端返回的通知里会多一个 Wrl_AppletOK 通知，代表小程序启动就绪，可以对此小程序窗口进行控制了。

五、小程序配置

PluginOK 系统服务对每个小程序的请求调用，是根据小程序的配置文件进行的。所以每个新开发的小程序，需要制作一个 JSON 格式的配置文件，文件名固定为 PluginConfig.json，并请务必确保文件格式为 **UTF-8 编码(无签名)**，用操作系统内置的记事本打开编辑即可。这个配置文件开发阶段可用，和你的小程序执行程序按规则(目录：PluginOK 安装路径\Plugins\PID\)放到一起即可。在需要发布版本时，使用 WrlSDK 包中的打包发布。

配置文件项目举例：


```
{
  "Type": 1,
  "OS": 1,
  "Control": 0,
  "Http": 0,
  "OSMinVer": "5.0",
  "Version": "1.5.16.2",
  "Name": "DLL 小程序范例",
  "Corp": "WRL",
  "Restart": 0,
  "Icon": "",
  "Date": "2024.07.29",
  "Desc": "DLL 版 COM 小程序范例-读写 TXT 记事本及获取当前系统信息",
  "Home": "http://zorrosoft.com/WRL",
  "Down": "http://zorrosoft.com/WRL",
  "Module": "PluginComDll.dll",
  "PID": "A22E18F1-95F8-4FDB-99D2-188E5FB12B23",
  "Objects": [
    {
      "ProgID": "C38672FA-B5C8-4D9D-89B5-2D71F0760661",
      "ClassID": "40613676-C8A5-4879-A59B-9CE6406476F6"
    },
    {
      "ProgID": "E46E1B5B-2891-414F-A21A-17B00231E650",
      "ClassID": "6D7595BD-BFB0-4D2C-906B-247028691A50"
    }
  ],
  "Updates": []
}
```


说明：Type为小程序类型(重要)，1为DLL小程序，2为弹窗小程序，8为内嵌网页小程序。OS为适用系统，1为32位Windows，2为64位Windows。Control为控制掩码是保留字段，Http为是否支持HTTP同步请求，如果是C#开发的Dll小程序，记得添加一行："Asm": 1，而C#的Exe小程序不需要。OSMinVer为最低要求系统版本，Version为小程序版本，Name为小程序名称，Corp为小程序开发公司，Icon为小程序展示图下载地址，Date为小程序发布日期，Desc为小程序功能描述，Home为小程序介绍网页URL地址(备用)，Down为小程序下载地址(备用)，Module为小程序的执行文件名(重要)，PID为小程序唯一标识，Objects为小程序中实现的多个组件信息。ProgID和ClassID分别对应COM组件接口的实际ID，Updates是升级信息(备用)。如果是弹窗或内嵌网页的小程序，还需要指定此小程序所用的默认侦听端口，如果端口被占用，端口号会自动+1再进行尝试，侦听成功时会返回当前所用端口并通知到前端。为了解决个别DLL或OCX控件前端调用后资源释放存在的问题，特别增加了一个选项Restart，设置为1时代表前端断开WS连接要求自动重启中间件服务，只适用于DLL小程序。

注意：DLL 小程序，配置文件中的 PID 为小程序实现的 COM 组件类型库 ID，而弹窗或内嵌网页小程序的 PID，可通过工具创建任意一个 GUID 来作为 PID 使用即可，一旦确定后提交给我们申请正式授权，就不能再更改。由于 DLL 小程序可实现多个 COM 组件，在请求其中一个组件服务时，PID 直接传此组件的 ProgID，而不是传小程序的 PID。

每个小程序，还可通过独立的 Config.json 配置文件来配置小程序中哪些模块需要注册使用，如果是弹窗或内嵌网页的小程序类型可配置独立的 WS 和 HTTP 侦听端口。举例：{"PORT": 12900, "HTTP": 12903,"COM": "YourOcx1,YourOcx2"}

开发测试小程序时，找到 PluginOK 程序所在目录，在 Plugins 子目录创建你的小程序目录，然后把小程序相关的所有文件及小程序的配置文件都放到这个子目录中，重启 PluginOK 服务即可。DLL 小程序默认只支持 32 位程序，如有对 64 位小程序的需求，请微信联系 ZorroSoft 定制对应的技术方案。

六、小程序打包

找到 PluginOK 中间件 ZbaSDK(单机版是 WrlSDK)程序包 Package 目录中的打包工具程序 ZbaPackage.exe(单机版是 WrlPackage.exe)并启动，不勾选小程序包开关时，代表制作 PluginOK 自身的升级包，这时只需要设置中间件安装和打包输出目录，并确保最新的 PluginOK 程序已经放到中间件安装目录下。

填写相关参数或设置相关目录，如下图所示：

勾选小程序包开关时，可选择打小程序的完整包还是打升级包，必须设置小程序文件目录、小程序授权文件、打包输出目录，并确保小程序文件目录已经放

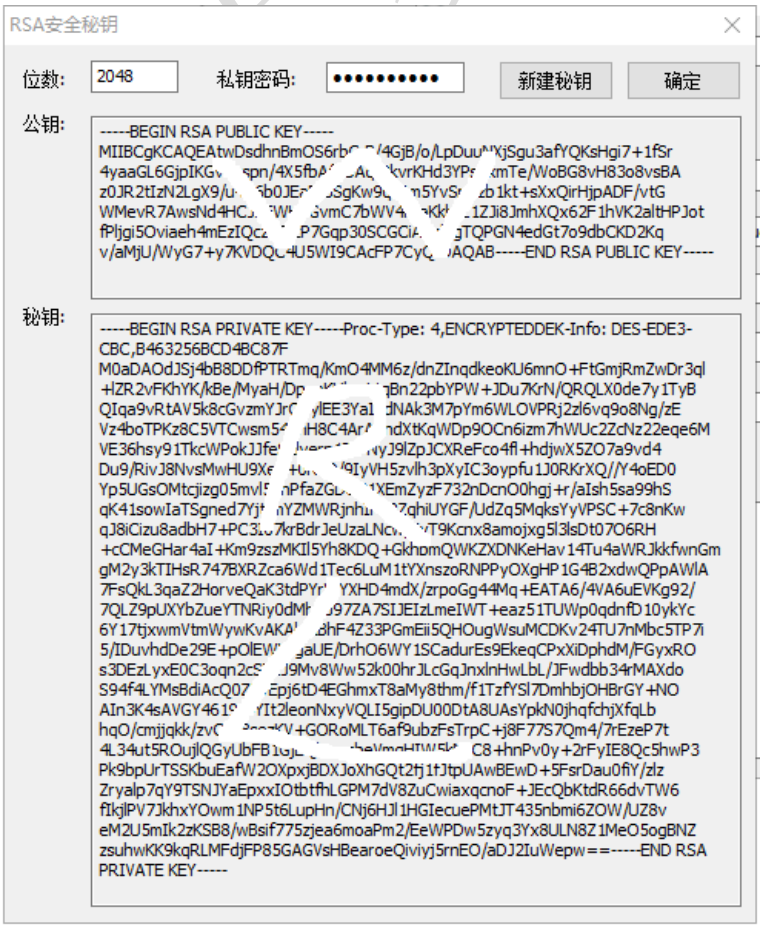
入其配置文件。如果你的小程序有些支持库或其他配置文件需要放到操作系统的 System32 目录下，请设置系统 System32 目录并放置对应的文件，同理如有支持库需要放置到 PluginOK 程序主目录时，请设置中间件安装目录并放入对应的程序文件，如果没有就不用设置，ProgramFiles 目录同理，切不可直接设置操作系统的相关目录直接进行打包。上述设置的几个磁盘目录，请确保不要是相互嵌套的目录结构，也无其它无关的文件，以避免打包后的文件过大。存放小程序的目录等都支持子目录结构，子目录下的文件也都会自动打包进入最终的程序包中。需要注意的是，虽然 PluginOK 授权支持自定义显示产品名称，但其执行程序、运行库 DLL 及配置文件的名称等是不能修改的，否则会导致整个系统运行异常。

配置好后点击执行打包后请等待打包结束。打包完成时复制打包结果的 JSON 信息，DownAddr 中添加实际的下载路径后用于安装或升级配置。在打包输出目录下，会创建对应版本的小程序包目录，可以找到相关的 JSON 配置打包信息和最终的小程序包 pid 文件。

小程序打包中需要的小程序授权文件，需要提供贵公司信息、联系人的信息及小程序的用途及配置文件进行申请，具体请联系客服(QQ：188872589 微信:ZorroSoft)进行。

PluginOK 中间件小程序开发交流 QQ 群 23126938，加入请在浏览器中打开此链接：https://jq.qq.com/?_wv=1027&k=5RfNDiB 申请入群暗号：PluginOK

第一次运行打包时，需要创建小程序自身所用的 RSA 公钥及秘钥信息，点击小程序安全设置，在弹出的界面设置 RSA 的长度及私钥的密码，然后点击新建秘钥后确定即可，如下图所示：



创建小程序的 RSA 安全秘钥后在打包程序的 data 子目录会同时输出小程序的公钥证书文件 PluginPubKey.crt，作为小程序的调用方需要用此公钥证书进行加密操作，所以请妥善保管此公钥信息避免泄露，调用时，建议在网页脚本中调用后台方法直接生成加密信息。如发现小程序的安全秘钥泄露，可通过重新创建并升级小程序来解决。

开发阶段小程序的授权类型为开发版，是不校验小程序调用安全机制的，但会有些弹窗提示，不保证每次请求都成功。当集成了 PluginOK 后需要正式发布时，请微信(ZorroSoft)联系客服购买 PluginOK 的正式版授权，以避免弹窗和请求失败，客服热线电话：**4006831589/18081958957**。更多信息，请访问本公司官方网站：http://www.zorrosoft.com，谢谢！

七、常见问题

1、 如何判断 PluginOK 中间件是否已安装？

答：首先请确认自己配置的 WS 侦听端口是多少，单机版默认配置的是 80 端口，网络版默认配置的是 83 端口，可自行在 ZbaConfig.json(单机版是 WrlConfig.json)中修改。如果这些端口已被占用，中间件会自动把端口号+1 后进行侦听。只要前端连接不上配置端口或自动+1 的端口就可以认为还未安装，如果中间件被手工停止了服务，前端可通过自定义协议的 URL 进行启动，启动网络版：点击这里启动 PluginOK 网络版，启动单机版：点击这里启动 PluginOK 单机版

2、 如何在前端实现 PluginOK 中间件升级？

答：网络版中间件升级的请求 JSON 包类似这样：

{"req":"Wrl_Update","rid":1,"para":{"Name":"BrowserApplet 中间件网络版升级包","Date":"2024-11-10","Desc":"1、优化中间件高级版浏览器兼容处理，解决多个独立窗口释放内嵌小程序后再启动故障问题；2、解决大文件二进制流接收在某些电脑上的兼容使用问题；3、PageHiOffice 组件增加样式支持，插入内容时支持设置样式名称，选中内容可设置样式 …","DownAddr":"http://local.zorrosoft.com/Files/Update/Zba_Update.pid","MD5":"B5EB73651DE2143560DFA44D78206AD6","Version":"2.2.16.8","Size":14123008,"HideIns":0,"Cookie":"","Auth":"","Open":"","TK":"263BD3512D9B2A243EB365411CEA812ED2DD4DC4102489CE38AE56AAD97EAC6E8C5348FFCD9AFC62ED7EC8305D7249A9123FF23F4B91098515338921746CFD6D7A825A0FDC55009CDA5683203C835273BD546AFEA03FA65E1CC909AF72A1516AAB2D25F4EB35296A8B5DD00292601603F0FDB919265BB136E5C03F4E89BB7346435EDEFDFA52FEF84FE3CCA8C885929C3325BCFA3140550CB4CF6E6C7EAED16E3BDCCEF3C8BB92218E6BF17D4D9A50BB5E92D82E80856E9E40201EE8D5D731D4944340F4ACF1AA0B3DBD5C411733B9E13C3CDD59F731E18BFE1203E89B8ABDBC2B30DB9A922E1F3924634286251D9FB0FCC35EC077CB243D08902DE5B938FB64"}} 此 JSON 可通过 SDK 包中的打包工具生成，需要指定中间件文件需要升级的目录，打包时需要注意开发版和正式版的打包工具配置文件是不一样的。关于如何打包请参考 SDK 包中的文档“PluginOK 开发者手册.pdf”。中间件本身也提供了判断是否

有新版的接口，请参考“TestWrl.txt”文档中的协议 Wrl_Version 说明。

3、 如何实现 PluginOK 小程序安装和升级？

答：小程序安装或升级的协议请参考安装后的文件 TestWrl.txt，每个小程序都可以通过 SDK 包中的打包工具生成安装包和升级包及对应的 JSON 请求，打包时同样需要注意开发版和正式版的打包工具配置文件是不一样的。然后把生成的安装或升级包文件放到您的 WEB 服务器上，确保无限制可通过 HTTP 协议下载，然后修改对应的 JSON 请求，写上 DownAddr 中实际的下载地址。然后在前端通过 WS 连接到 PluginOK 中间件上后执行，将自动启动小程序的安装和升级。

4、 前端 WS 连接不上如何办？

答：前端 WS 连接不上时，首先需要确定中间件是否已经安装并启动，其次是查看中间件程序所在的 data 子目录下 ZbaService.txt(单机版是 WrlService.txt)运行日志，看实际侦听的 IP 和端口是多少。如果侦听端口和你连接指定的端口不匹配肯定是连接不上的，检查本机防火墙设置是否允许指定端口侦听，再检查是否本机启动了网络代理程序，如已启动先退出。另外需要确保中间件程序运行未被某些安全软件或杀毒软件拦截。PluginOK 中间件及相关小程序，我们是进行了数字签名的，并提交到 360 做了白名单，如还遇到拦截，请手工添加整个目录程序进入其白名单程序库中再试。如果还不行，请重新安装操作系统或更换电脑进行体验。

5、 PluginOK 中间件重启后会自动运行吗？

答：中间件主程序默认是以系统服务方式启动运行，当操作系统启动后服务就会自动启动，所以当您登录到系统桌面时，前端是可以随时调用相关功能的。如因一些特殊情况不能以系统服务方式运行时需要修改配置文件中的参数 NoService 为 1，这种运行模式下登录到系统桌面时，中间件才会启动运行，请确保自动运行的注册表项目不会被设置为禁止启动，或被安全或杀毒软件拦截。

6、 PluginOK 中间件安装和卸载时提示需要管理员权限确认如何处理？

答：这种一般是在 Windows 7 及以上版本系统中出现，在安装 PluginOK 中间件、非系统服务模式运行登录到桌面运行时可能会出现此提示，请选择是(Y)，否则会导致功能运行异常。

7、 PluginOK 中间件提示需要管理员权限确认时显示产品及公司名称如何修改？

答：需要用自己公司的签名证书重新签名，签名后的所有 EXE 及 DLL 文件需要重新提交到 360 做白名单，否则可能在运行时被 360 安全卫士等拦截。

8、 PluginOK 中间件是否完全开源？

答：本中间件是浏览器之外的一个通用外接系统，如果开源的话，会导致被人滥用从而威胁客户电脑的安全。所以正常情况下我们只提供前端调用的代码完全开源，如您需要本中间件的源码用于安全审核或备案等，需另行协商。

9、 PluginOK 小程序是否完全开源？

答：PluginOK 之上运行的小程序，如您需要，可以付费获得相关代码用于定制开发，满足系统自主可控要求。

10、 PluginOK 中间件是否收费？

答：本中间件及相关小程序如用于商业用途是收费的，具体收费政策请联系本公司客服获取最新报价信息。如用于公益用途，需相关公益机构出具相关证明。

11、 PluginOK 网络版和单机版有啥差异？

答：不管网络版还是单机版，都可以用于企业内网，无需外网可运行。简单说网络版适用于项目，单机版适用于产品，网络版有终端电脑使用数量和使用权限限制，单机版没有，因此单机版报价会比网络版贵一些。由于网络版有集中的授权管理端，所以在更新 SSL 证书后的授权文件升级时非常方便，只需要更新服务端的授权文件即可完成所有终端的自动授权。

12、 PluginOK 中间件会对浏览器做改动吗？

答：不会对浏览器的任何程序及运行中的内存数据做修改，也不会修改浏览器的主页等配置，技术实现也完全不依赖任何浏览器提供的开发接口，所以浏览器的升级也不会导致技术方案失效，可避免浏览器升级可能导致插件无法运行的问题。

13、 PluginOK 中间件如何保证安全？

答：PluginOK 中间件及相关小程序，我们承诺不存在任何木马和病毒行为，如确实需安全审查或备案等，可在签订相关协议后付费获取相关源代码来进行验证，也可以把软件包提供给相关专业机构进行检测。关于安全调用及使用方面的详细信息，请参考 SDK 包中文档“中间件安全解决方案.pdf”。

14、 如何关闭前端获取本机 Mac 地址功能？

答：请修改 ZbaConfig.json(单机版是 WrlConfig.json)中 Mac 配置为 0 后保存。如是制作安装包，请修改程序目录的配置文件，如已安装实际生效的是 Data 子目录下的配置文件。除了 Mac 地址可唯一标识终端电脑外，中间件在协议 Wrl_Version 实现中会返回对应终端电脑的唯一序号，可以替代 MAC 地址用于识别终端电脑。

15、 PluginOK 中间件标准版和高级版有啥差异？

答：高级版最显著的特征是支持内嵌网页小程序的运行，比如支持在网页中直接原生播放 RTSP 视频、在网页中直接打开 Office 文档实现在线编辑、直接打开 doc、dwg 等图纸进行在线编辑、查看和审阅等。高级版会比标准版报价贵一些，详细报价文档请微信联系客服获取。

16、 PluginOK 中间件可直接进行二次开发吗？

答：如您的需求不需要高级版的功能，可直接在 http://local.zorrosoft.com 上下载 SDK 包和安装中间件后，可参考 SDK 包中的范例工程进行开发，小程序的配置可先直接沿用范例工程的 PluginConfig.json，等需要实际打包安装发布时，再提交自己的小程序配置文件 PluginConfig.json 和公司相关信息申请对应的授权文件进行测试。如是需要高级版的功能，请先联系客服签订软件试用协议，测试验证高级版功能是否满足贵公司需求，如满足可先行购买一年的网络高级版授权进行二次开发，我们会提供 Flash Player 的内嵌网页小程序源代码工程(C++)以供参考。另外二次开发接口，请参考 SDK 包中的文档“PluginOK 核心组件接口说明.xls”。

17、 高级版启动小程序时 Url 参数的作用？

答：用于获取初始化显示内嵌网页小程序窗口的大小、位置和其它启动参数，一般这个参数指向您实际加载内嵌小程序的网页地址，如中间件获取不到 url 指向网页中的相关小程序参数，会导致实际显示的小程序位置与大小可能与需要的不符。新版本提供了不需要 Url 参数启动的方法，不建议使用了，需另外配置 Web 参数，具体参考测试网页的样例。

18、 线上开发版和正式版有何差异？

答：线上开发版主要用于集成测试验证功能，PluginOK 中间件会在启动时弹出定时消失的相关提示，部分请求可能随机性的调用失败，如需给客户现场部署使用，请联系客服购买正式授权，另外开发版授权一般是短期的，过期后自动转为试用版，更多功能会受到使用限制。不管是开发版还是正式版，网络版或单机版，前端集成时的调用接口都是一样的，所以在开发版上验证满足需求后，很容易的切换到正式版来使用。如果需要更多的安全特性，正式版中部分接口调用需要验证权限，详细请参考 SDK 包中文档“中间件安全解决方案.pdf”的说明。

19、 购买 PluginOK 中间件是否有正规发票？

答：PluginOK 中间件及相关小程序，由成都佐罗软件有限公司研发并销售，可提供正规的增值税普通发票或增值税专用发票，一般税点是 1%-3%，开票名称一般为本软件的产品名称，如有特殊要求请联系我们确认。

20、 PluginOK 中间件支持信创系统吗？

答：信创系统的版本还在研发中，请耐心等待。

21、 PluginOK 中间件有 Linux 和 Mac 系统的版本吗？

答：目前暂时只支持 Windows 系统，如您有其它系统版本的需求，请微信联系我们客服沟通。

22、 PluginOK 中间件是否对其它软件或模块有依赖？

答：本软件发行版已经包含所有需要的 DLL 及 EXE 程序，无需任何.NET 或 C++运行库等的额外支持，除非您的小程序需要依赖的相关支持库。具体到相关的小程序，可能需要安装对应的软件，比如需要打开在线编辑、查看或审核 doc 文档，需要安装微软 Office 软件或金山 WPS 办公软件。

23、 PluginOK 中间件及相关小程序是否可以代理销售？

答：我们欢迎各种互惠互利平等的合作，如您有客户资源或销售能力，可随时联系我们做进一步沟通，即使您不是公司，也可以成为我们的业务合伙人，只要有销售业绩就有相应的回报。

24、 PluginOK 中间件会一直持续更新并可免费获取升级版吗？

答：本中间件从 2018 年立项投入开发至今，我们一直在做持续的维护升级，2019 年 5 月发布第一个标准版，2020 年 7 月发布第一个高级版，之后基本上保持一个月 1-2 个升级版的节奏持续更新，给中间件赋能，并基于中间件开发了 PageHiPlayer、PageHiOffice、PageHiCAD 等多个典型应用产品，证明了 PluginOK 的强大和生命力。我们一直在努力，持续根据您的需求完善产品和提高稳定性及使用体验。本产品是成都佐罗软件有限公司最核心的拳头产品，会持续获得最大成都的资源进行升级和维护，只要购买了官方发布版本授权的客户，在授权期内，都可免费获得

最新的升级程序包。

25、 网络版授权服务端程序是否可以运行在虚拟机或云主机中？

答：网络版授权服务端程序，目前政策是可以运行在虚拟机或云主机中，支持 Windows 7 及以上的桌面系统，或 Windows Server 2008 及以上服务系统。

26、 在一个 WS 连接中可以启动多个内嵌小程序吗？

答：每个 WS 连接需要确保是唯一的会话 SID，同时一个 WS 连接里尽量确保只启动一个小程序，这样关闭连接也只关闭这个小程序，不会影响其它小程序的运行，当然中间件本身并没有做这个限制，完全可以在同一个 WS 连接中启动多个小程序的实例。

27、 安装时 360 安全卫士或其它杀毒软件弹出告警如何解决？

答：本软件每个正式版都会提交到 360 做程序白名单，如还遇到告警，请手工把程序目录所有 EXE 和 DLL 等添加到本地白名单中，其它安全类软件也是这样操作，本软件承诺不包含任何病毒及木马功能，所有 DLL 和 EXE 程序都有本公司的签名证书，请使用时确认签名证书是否被破坏。可把本软件在线提交到：https://www.virscan.org 进行扫描确认是否安全。

28、 如何屏蔽中间件和播放小程序中的运行日志？

答：在前端请求 WS 连接的参数 flag 掩码中，去掉 1 即可，测试网页中默认设置的 1，所以会输出比较多的日志信息，一旦系统运行稳定，可以屏蔽日志输出，提升系统运行速度。

29、 如何在同一个网页中启动内嵌小程序的多个实例？

答：默认情况下，在不同的网页中可以分别启动一个内嵌小程序的实例。如您需要在同一个网页中启动播放小程序的多个实例，方法如下：A、WS 连接中间件的参数 flag 值，在原来基础上+512，比如原来是 1，那就写成 513；B、请求启动播放小程序的 Url 指定的网页中，不要设置多路播放的源和显示位置信息，统一改为启动播放的 JSON 参数中指定；C、启动每一个播放小程序实例，都应该在单独的 WS 连接中进行，就是说不能放在同一个到中间件的 WS 连接中启动；D、启动多个实例时应该按先后顺序启动，不能同时启动，就是等一个启动后再启动下一个。

30、 是否支持 HTTPS 网站使用？

答：是支持的。默认授权是开通 HTTP 网站的，前端 WS 连接地址用 IP 即可，如果您的 B/S 是 HTTPS 的，请提前和我们沟通说明，需要您在 HTTPS 网站主域名下申请一个其它地方不会用到的二级域名，然后用这个二级域名申请其 SSL 证书并下载证书文件包发给我们，需要下载给 Apache 服务器使用的，我们将用这个证书包给您制作专用的授权文件。如果有支持绑定域名的通用 SSL 证书也可使用。

31、 绑定 HTTPS 网站的 SSL 证书到期如何处理？

答：如果已经到期，需要按第 30 条的方式，重新申请新的证书文件包发给我们做授权文件。如果证书还没到期，最好是提前几天申请新的证书，然后自己可以根据 testwrl.txt 文档中的说明自行更新证书信息到授权文件，具体指令是 Wrl_UpdateSslCert，获得新的授权文件后，还可以根据文档中的指令 Wrl_UpdateAuth 让终端强制更新授权文件，或者制作一个中间件的升级包强制终端抓紧更新一下，否则证书到期后前端就

无法连接到中间件服务端口，影响使用。更多细节请参考文档“HTTPS 网站中配置 WSS 连接说明书.pdf”。

32、 PluginOK 中间件中的 WebSocket 连接是否支持二进制数据流传递？

答：在最新版中是支持的，在小程序开发接口中也提供了对应的事件通知支持。一般建议是在传递二进制数据流之前，先发一个 JSON 格式的内容描述下二进制数据流的相关信息，让对方接收时更容易进行处理。

33、 PluginOK 中间件支持服务器文件的多线程下载吗？

答：在网络版 2.2.15.1(单机版 1.5.15.1)及以上版本中是支持的，在中间件的配置文件中，涉及多线程下载有这样几个配置：

DownSpeed 下载速度控制倍数，默认 8，如果网速比较快，可以适当调大，当前默认配置适用于网速偏慢情况，设置 1 代表一次获取 16Kb 数据；

NetOverTime 网络连接及请求数据超时时间，默认 30 秒，可根据自己需要调整；

MaxThread 多线程下载时允许的最大线程数，如果网速低或电脑配置低，建议使用较小的数值，网速快且电脑配置高时，可适当调大数值；

ThreadSize 启用多线程的最低文件大小，MB 为单位，MaxThread 即使设置了大于 1 的数值，如果当前下载的文件小于 ThreadSize 的值，也不会启动多线程下载。

34、 PluginOK 中间件支持异步文件上传吗？

答：在网络版 2.2.15.1(单机版 1.5.15.1)及以上版本中是支持的，只要中间件还在继续运行，文件异步上传就会继续执行直到完成，前提是购买并部署了文件操作小程序并在自己的小程序中请求指定了异步文件上传参数。