**WxBox - 测试与使用**

**(v0.1.0)**

**修订历史记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 修改日期 | 版本  （指的是文档的版本号） | 说明 | 作者 |
| 2022/03/03 | v0.1.0 | 写个测试用的手册，随便把用法和一些说明也补充进这里 | 胡智钦 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[1 测试前说明 6](#_Toc97261594)

[1.1 注意事项 6](#_Toc97261595)

[1.1.1 Plugin相关的注意事项 6](#_Toc97261596)

[1.2 已知可能出现的异常情况 6](#_Toc97261597)

[1.3 WxBox目录结构说明 8](#_Toc97261598)

[1.4 WxBox主要功能 10](#_Toc97261599)

[1.5 Plugin说明 10](#_Toc97261600)

[1.5.1 wxbox模块部分API说明 10](#_Toc97261601)

[1.5.2 Plugin事件 11](#_Toc97261602)

[1.5.3 Plugin Command语法 12](#_Toc97261603)

[1.6 WxBox界面说明 13](#_Toc97261604)

[1.6.1 工具栏 14](#_Toc97261605)

[1.6.2 Client状态TableView 17](#_Toc97261606)

[1.6.3 Command 23](#_Toc97261607)

[1.6.4 WxBox托盘菜单 24](#_Toc97261608)

[1.6.5 主界面快捷键 25](#_Toc97261609)

[1.7 崩溃处理与crashdumper 25](#_Toc97261610)

[1.8 interact 25](#_Toc97261611)

[2 WxBox测试 26](#_Toc97261612)

[2.1 【WBTC1-启动WeChat新实例】 26](#_Toc97261613)

[2.2 【WBTC2-给未注入WxBot的WeChat进程注入WxBot模块】 30](#_Toc97261614)

[2.3 【WBTC3-查看定位特征】 32](#_Toc97261615)

[2.4 【WBTC4-查看所有联系人】 35](#_Toc97261616)

[2.5 【WBTC5-测试切到前台】 37](#_Toc97261617)

[2.6 【WBTC6-测试防止消息撤回】 38](#_Toc97261618)

[2.7 【WBTC7-退出登录】 39](#_Toc97261619)

[3 Plugin预定义的wxbox模块API与WxBot测试 39](#_Toc97261620)

[3.1 【WBPTC1-查看wxbox版本号】 40](#_Toc97261621)

[3.2 【WBPTC2-查看help文档】 41](#_Toc97261622)

[3.3 【WBPTC3-截图】 43](#_Toc97261623)

[3.4 【WBPTC4-只截取主屏幕】 43](#_Toc97261624)

[3.5 【WBPTC5-给WxBox发送不同Level的日志】 43](#_Toc97261625)

[3.6 【WBPTC6-清空WxBox的日志输出】 44](#_Toc97261626)

[3.7 【WBPTC7-使用shell执行关机命令】 44](#_Toc97261627)

[3.8 【WBPTC8-锁定PC屏幕】 45](#_Toc97261628)

[3.9 【WBPTC9-让PC弹窗】 46](#_Toc97261629)

[3.10 【WBPTC10-让PC播放一段语音】 47](#_Toc97261630)

[3.11 【WBPTC11-列出所有盘符以及某个文件夹内的所有文件】 47](#_Toc97261631)

[3.12 【WBPTC12-下载文件】 49](#_Toc97261632)

[3.13 【WBPTC13-手机控制WxBot模块Uninject】 49](#_Toc97261633)

[3.14 【WBPTC14-手机控制WxBox退出】 49](#_Toc97261634)

[3.15 【WBPTC15-判断是否登录】 49](#_Toc97261635)

[3.16 【WBPTC16-触发PC退出登录】 49](#_Toc97261636)

[3.17 【WBPTC17-查看登录用户的Profile】 49](#_Toc97261637)

[3.18 【WBPTC18-nickname、wxid、wxnumber之间的互相转换】 49](#_Toc97261638)

[3.19 【WBPTC19-获取群聊的wxid】 49](#_Toc97261639)

[3.20 【WBPTC20-向手机文件传输助手发送文本、图片和文件】 50](#_Toc97261640)

[3.21 【WBPTC21-向指定wxid联系人发送文本、图片和文件】 50](#_Toc97261641)

[3.22 【WBPTC22-向指定wxnumber联系人发送文本、图片和文件】 50](#_Toc97261642)

[3.23 【WBPTC23-向指定群聊发送图片和文件】 50](#_Toc97261643)

[3.24 【WBPTC24-向指定群聊发送文本，可选带@列表】 50](#_Toc97261644)

[3.25 【WBPTC25-在指定群聊中@一部分人】 50](#_Toc97261645)

[3.26 【WBPTC26-在指定群聊中@所有人】 51](#_Toc97261646)

[4 Plugin测试 51](#_Toc97261647)

[4.1 【PTC1-测试启用提供的example样本扩展】 51](#_Toc97261648)

[4.2 【PTC2-测试example样本扩展的long\_time\_task\_test方法】 52](#_Toc97261649)

[4.3 【PTC3-测试example样本扩展调用二进制模块】 53](#_Toc97261650)

[4.4 【PTC4-测试example样本扩展调用第三方lua库】 53](#_Toc97261651)

[4.5 【PTC5-观察example样本扩展是否能够捕获WeChat声明周期】 54](#_Toc97261652)

[4.6 【PTC6-测试捕获原始消息】 55](#_Toc97261653)

[4.7 【PTC7-测试消息过滤】 57](#_Toc97261654)

[4.8 【PTC8-测试消息替换】 59](#_Toc97261655)

[4.9 【PTC9-测试捕获处理过的文本消息】 59](#_Toc97261656)

[4.10 【PTC10-测试捕获处理过的非文本消息】 59](#_Toc97261657)

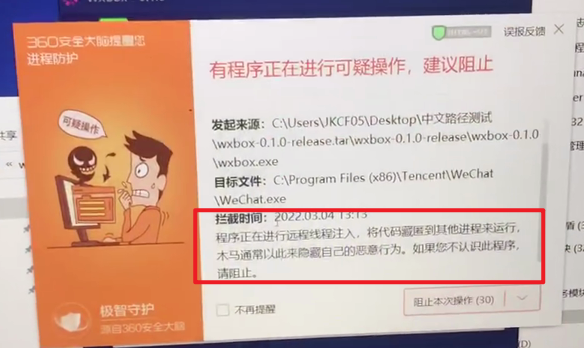
[4.11 【PTC11-测试捕获发送文本消息】 60](#_Toc97261658)

[4.12 【PTC12-测试替换发送文本消息】 61](#_Toc97261659)

# 测试前说明

## 注意事项

* 支持Windows 7 ~ Windows 10，Windows 11没测试过，**最好跑Windows10**；
* 先启动一次WxBox.exe，看看会不会**缺少什么依赖的运行时**或者运行时版本不匹配；
* 最好把火绒、360之类的**安全软件关掉**，或者给WxBox放行，因为WxBox的注入还有跨进程WriteProcessMemory等操作会被安全软件给拦截掉；



【360提示】

* 测试的时候，最好把WxBox解压到一个**中文路径**，这是为了测试是否存在编码问题，但正常使用时最好还是使用英文路径；
* 现在经过测试的**WeChat版本**是**3.4.5.27**和最新的**3.5.0.46**，因为WxBox是通过“特征”来定位到需要的Hook点还有WeChat某些接口的，其它版本没有提取特征，用现有的特征不一定能正常使用；
* 如果没有匹配微信版本的特征，可以尝试更新特征库；
* 要想测试3.4.5.27版本，可以到[这里下载](https://developer.lanzoug.com/file/?)，然后把exe安装包给解压出来，别直接安装，可以用7zip解压；
* **测试前必须核对下微信版本**，最好是**3.5.0.46**最新版本的WeChat，如果版本不对先更新；



【更新WeChat版本】

* WxBox不屏蔽掉WeChat自己的更新，因为在WeChat上有选项可以选择要不要自动更新的，所以没必要；
* WxBot模块注入WeChat进程后，WxBox可以关闭掉，不影响WxBot模块，等WxBox再次启动后，WxBot会重新连接WxBox。

### Plugin相关的注意事项

* WxBox **Plugin支持热更新**，WxBot会监控Plugins中lua文件的变动，所以在注入了WxBot可以直接修改、替换、新增Plugin；
* **在编写Plugin时，最好不要有耗时长的任务**，WxBot默认会把执行时长已经超过5秒的调用“**让出**”【也就是强制结束掉，不过如果是进入了本机调用或者线程I/O阻塞了，这种情况不能让出】；
* 耗时长的任务最好封装成二进制的Plugin，在线程中跑；
* **可以执行Plugin命令的地方**有两个，一个是WxBox的**Command**框，另一个是手机端的“**文件传输助手**”；
* 另外要注意命令中标点符号，**'**和**"**都是**英文半角标点符号**。
* 命令中字符串可以用**'你好'**， **"你好"**， **'hello"你好", "你好"'**

## 已知可能出现的异常情况

WxBox的功能是通过把WxBot.dll模块注入进WeChat的进程实现的，所以不能够保证注入后，WeChat能保持稳定运行。下面列出几种可能出现的小概率异常：

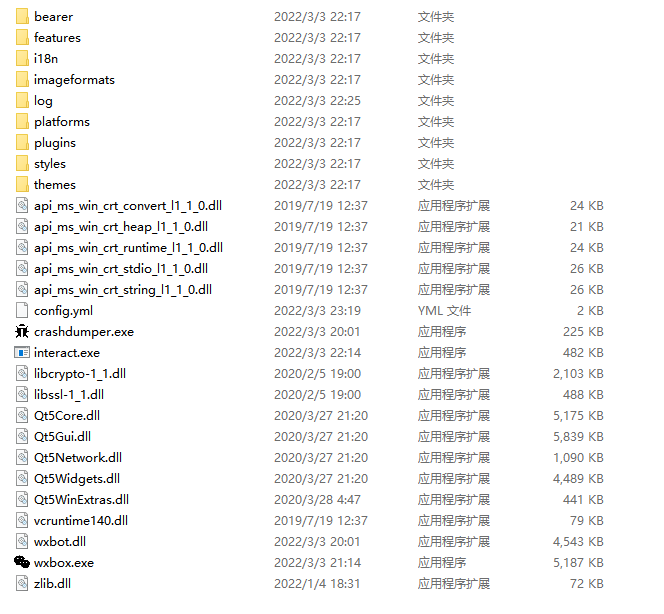
* 卸载WxBot模块时WeChat进程小概率崩溃【卸载WxBot模块时，会把WxBot.dll以及依赖的模块从WeChat进程中释放掉，但是有可能会漏掉某些已经进入了WxBot.dll线程还没有执行完的情况，这时候释放掉WxBot模块就会导致WeChat进程崩溃。在执行WxBot卸载时，虽然已经做了以下操作尽量避免这种情况：把所有WeChat其它线程Suspend掉，然后检查它们的整个调用堆栈，如果有进入了WxBot模块内存空间的，会睡眠一段时间，然后重新执行检查。不过由于Windows在执行StackWalk时并不能保证每次都返回每个线程正确完整的调用堆栈，所以会出现漏掉的情况，一旦漏掉，那么就有概率导致WeChat进程崩溃。不过由于这个检查的存在，这种情况在测试时基本没有出现过】
* 通过Plugin或者WxBox界面上触发WeChat退出登录时，有可能导致WeChat网络无法连接？除非重启WeChat，否则无法再次登录。【这种问题原因暂时不知道，有可能是WxBot触发Logout的方法并没有让WeChat的某些上下文信息得到重置？】
* 在一个旧版本的WeChat的环境出现过，先注入启动WeChat，然后更新WeChat重启，Client条目状态不断的闪烁，再次重启WeChat就好了【暂时没有复现出现】

还有一个不算是异常但会造成界面卡顿的情况，如果启用了WxBox的Hook Raw WeChat Message特性，也就是启用了Hook WeChat接收到的Raw（原始）消息，每个都会转发给所有的Plugin，当WeChat未登录并且收到大量消息时，在PC上登录WeChat，由于Hook Raw WeChat Message是阻塞处理的，这时候就会造成WeChat登录成功后，整个黑掉然后卡顿一小会，现象如下图：



【启用hook raw message和WeChat有大量未接收消息情况下登录出现的现象】

## WxBox目录结构说明



【WxBox根目录】



【WxBox目录说明】

还有一些上面的图没有列出来说明的目录和文件都是第三方的依赖。

## WxBox主要功能

* WeChat进程实例多开；
* 消息防撤回；
* 监控账户生命周期【登录、注销、退出PC客户端】；
* 检查PC微信是否处于登录状态；
* Plugin触发PC端退出登录；
* 获取登录账户的Profile；
* 获取包括群聊在内的所有联系人信息【只获取昵称、备注、wxnumber和wxid】；
* 昵称、wxnumber和wxid之间的相互转换；
* 接管WeChat原始消息和处理过的消息；
* 接管WeChat发送文本消息；
* Plugin主动执行发送文本消息、文件、图片，对于群聊可以指定@列表。可以指定发送的目标和内容；
* 过滤接收到的消息【原始消息拦截事件中】；
* 替换接收到的消息【原始消息拦截事件中】，可以替换发送者、消息内容，对于群聊可以替换talker；
* 自动回复；
* 通过WxBox或者手机端WeChat的“文件传输助手”向PC WeChat客户端发出执行Plugin命令的指令；
* 自定义纯文本和二进制类型的Plugin，延伸功能。

## Plugin说明

WxBox的核心功能在Plugin上，上面的所有功能都可以在自定义的Plugin中调用，而Plugin可以通过WxBox和filehelper来调用。

WxBot里面预定义了一个wxbox模块，它提供了一些API接口，这些API包含了上面还有之外的功能，完整的wxbox模块API可以看plugins目录下的**wxbox\_apis.txt**文件，下面列出部分。

### wxbox模块部分API说明

* >>wxbox.version查看wxbox的版本
* >>wxbox.help查看wxbox命令帮助，它会给文件传输助手发送一个wxbox\_apis.txt文件
* >>wxbox.snap向filehelper发送一个桌面系统所有显示器拼接的“**载图**”
* >>wxbox.snap\_main\_monitor向filehelper发送一个桌面主Monitor的“**载图**”
* >>wxbox.info、>>wxbox.warning、>>wxbox.error向WxBox发送不同Level的日志
* >>wxbox.shell在PC系统上执行有权限执行的所有类型的**shell命令**【\*】，可以通过它执行关机、打开程序、图片等指令
* >>wxbox.msgbox给PC弹出一个对话框，用来提示消息的
* >>wxbox.speak让PC播放一段指定文本的**语音**
* >>wxbox.lock\_screen让PC锁屏
* >>wxbox.list\_drivers列出PC系统上所有的硬盘驱动器盘符
* >>wxbox.list\_files列出PC系统上指定文件夹目录下的所有文件和文件夹
* >>wxbox.download把PC上指定的文件通过filehelper传输到手机上，大小限制跟WeChat发送文件大小的限制一样，听说最大是1GB，但是没测试过
* >>wxbox.uninject\_wxbot卸载PC上当前用户的WxBot模块
* >>wxbox.exit\_wxbox关闭PC上的WxBox
* >>wxbox.logout让PC上的WeChat退出登录
* >>wxbox.nickname\_to\_wxid昵称转为wxid
* >>wxbox.wxid\_to\_wxnumber从wxid转为wxnumber
* >>wxbox.chatroom\_wxid获取群聊的roomWxid
* >>wxbox.get\_all\_contacts获取所有联系人信息，这不是给filehelper或者WxBox调用的，这个在Plugin中调用
* >>wxbox.send\_text\_to\_filehelper向filehelper发送文本消息
* >>wxbox.send\_picture\_to\_filehelper向filehelper发送图片
* >>wxbox.send\_file\_to\_filehelper向filehelper发送文件
* >>wxbox.send\_text向指定wxid联系人发送文本消息
* >>wxbox.send\_picture向指定wxid联系人发送图片
* >>wxbox.send\_file向指定wxid联系人发送文件
* >>wxbox.send\_text\_with\_wxnumber向指定wxnumber联系人发送文本消息
* >>wxbox.send\_picture\_with\_wxnumber向指定wxnumber联系人发送图片
* >>wxbox.send\_file\_with\_wxnumber向指定wxnumber联系人发送文件
* >>wxbox.send\_text\_to\_chatroom向指定群聊发送文本消息，消息后面可选带@列表
* >>wxbox.send\_picture\_to\_chatroom向群聊发送图片
* >>wxbox.send\_file\_to\_chatroom向群聊发送文件
* >>wxbox.chatroom\_notiry在群聊上@一部分人
* >>wxbox.chatroom\_notiry\_all在群聊上@所有人

### Plugin事件

Plugin生命周期事件：

* + load，返回true表示Plugin初始化成功，返回false的话表示Plugin初始化失败，Plugin会被释放掉
  + prereload
  + unload

WeChat关联事件：

* + receive\_raw\_message
  + receive\_message
  + receive\_text\_message
  + send\_text\_message
  + login\_wechat\_event
  + logout\_wechat\_event
  + exit\_wechat\_event

#### WeChat关联事件对象

例如receive\_raw\_message事件，它们会接收一个event对象

function example.receive\_raw\_message(event)

    -- event:filter\_message()

    -- event:substitute\_wxid('filehelper')

    -- event:substitute\_message('你好')

    wxbox.info('example.receive\_raw\_message : ' .. event:message())

end

这个对象提供的数据Getter如下：

* + event:type()
  + event:message\_type()
  + event:wxid()
  + event:message()
  + event:chatroom\_talker\_wxid()
  + event:pdata1
  + event:pdata2

event对象提供的方法如下：

* + event:filter\_message()，一般只在receive\_raw\_message事件中使用，调用后该消息就会被过滤掉，PC上的WeChat客户端就接受不了这条消息
  + event:substitute\_wxid(new\_wxid)，替换wxid
  + event:substitute\_message(new\_message)，替换消息文本
  + event:substitute\_chatroom\_talker\_wxid(new\_chatroom\_talker\_wxid)，替换chatroom的talker

注意上面的四条方法，最好只在receive\_raw\_message和send\_text\_message中使用，因为这两条消息是阻塞处理的，像receive\_text\_message是投放到plugin virtual machine的事件队列中排队处理的，所以不会起作用，并且可能导致指针访问异常。

### Plugin Command语法

@syntax：

    statement ::= prefixexp methodcall

    prefixexp ::= ">>"

    methodcall ::= method | method ":" arglist

    method ::= methodname | modulename "." methodname

    arglist ::= expr {"," expr}

    expr ::= false | true | Numeral | LiteralString

调用例子：

@example:

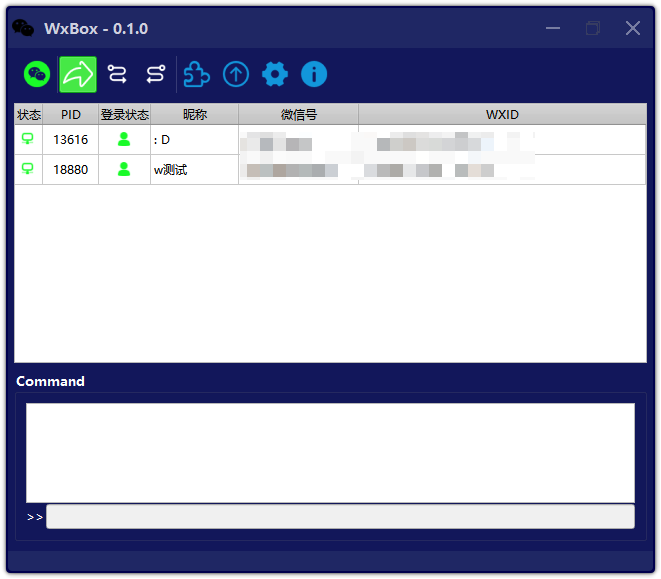
    >>wxbox.help

    >>wxbox.version

    >>wxbox.send\_text: 'filehelper', 'hello world'

    >>wxbox.set\_config: 'avoid\_revoke', true

## WxBox界面说明



【WxBox主界面】

WxBox整个界面分成三部分：1）工具栏；2）WxBot Client状态TableView；3）命令日志框以及命令执行文本框。

### 工具栏



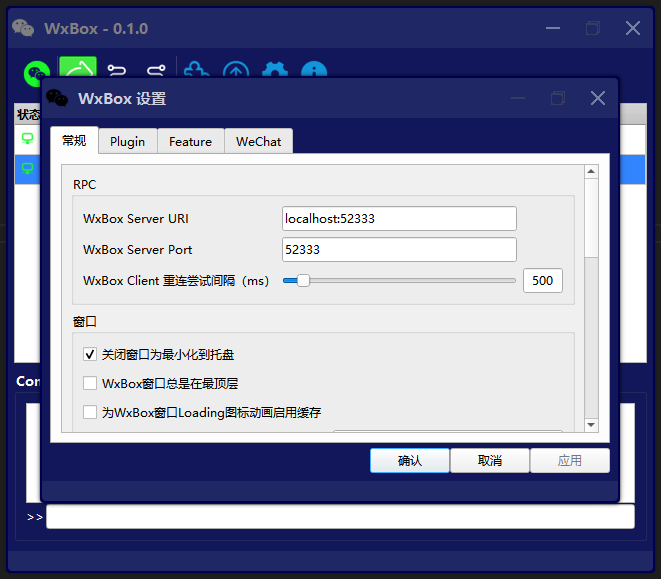
工具栏分为三块，功能、特性和辅助配置。

|  |  |
| --- | --- |
| **功能按钮** | |
| 按钮图标 | 说明 |
|  | 启动WeChat新进程实例 |

|  |  |
| --- | --- |
| **特性开关【**背景变为**绿色**说明该特性是**启用状态】** | |
| 按钮图标 | 说明 |
|  | 是否启用 “防止WeChat消息撤回功能” |
|  | 是否启用“Hook WeChat Raw Message的Plugin事件”【即是否接管接收的原始消息】 |
| C:\Users\hutingting\Desktop\未标题-1.png | 是否启用“Hook WeChat Send Text Message的Plugin事件”【即是否接管发送文本消息】 |

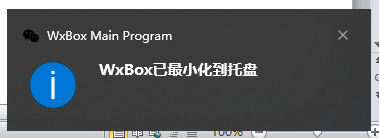
|  |  |
| --- | --- |
| **辅助配置按钮** | |
| 按钮图标 | 说明 |
|  | 打开Plugins根目录 |
|  | 更新特征库 |
|  | 打开设置对话框 |
|  | 打开About对话框 |

#### Setting（设置）对话框



【设置对话框】

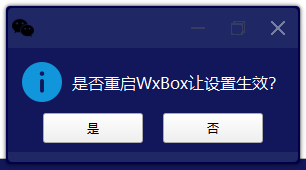
全部**保持默认**设置就可以了，只有一个地方需要注意的，**“关闭窗口为最小化到托盘”**。如果讨厌关闭窗口是最小化到托盘，而不是直接关闭，并且右下角有一个提示的话，可以在这里关闭掉。



【最小化到托盘的提示】

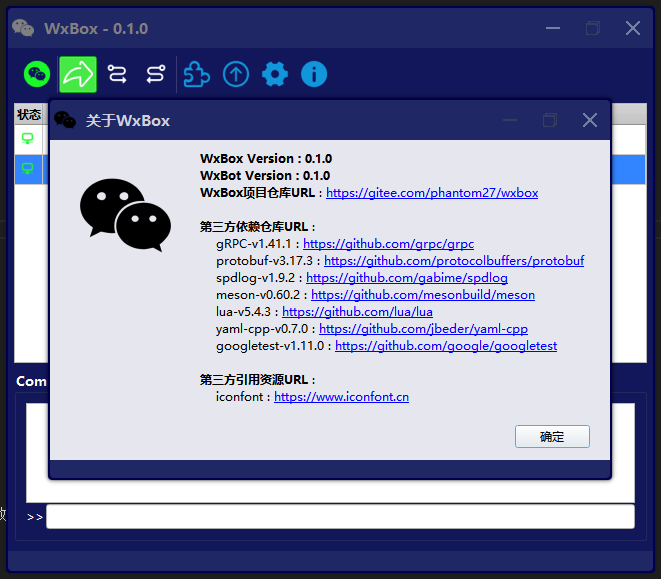
另外如果想改变端口或者跟日志有关的一些设置的话，需要重启WxBox，它会提示并且询问是否直接重启的的。





【重启WxBox提示对话框】

#### About（关于）对话框



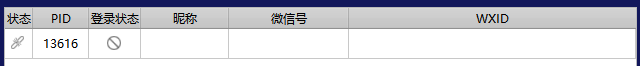
【About对话框给出一些第三方库的仓库地址，note：超链接可以直接点开】

### Client状态TableView

这个表会把所有的WeChat进程列出来，并且WxBot连接状态以及登录的用户Profile信息都会列出来。

#### 各状态图标表现例子

##### 未注入WxBot模块状态



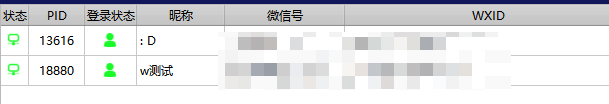
##### 已注入WxBot模块但未连接状态



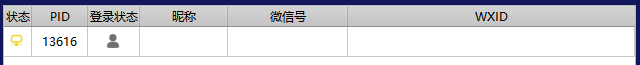
##### 连接正常但用户未登录状态



##### 连接正常并且用户处于登录状态

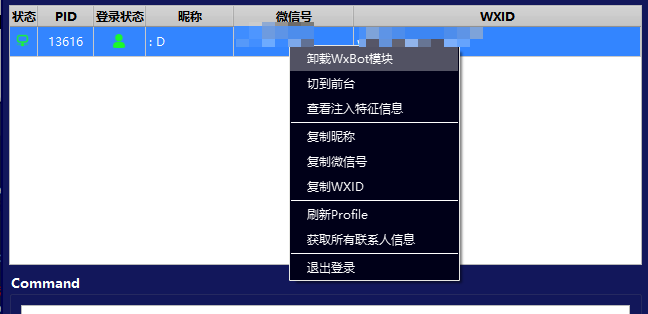


##### 已注入WxBot模块，但特征定位不完整状态



#### Client Item右键菜单

每个Client Item都支持右键菜单，提供一些额外的功能。



【Client Item右键菜单】

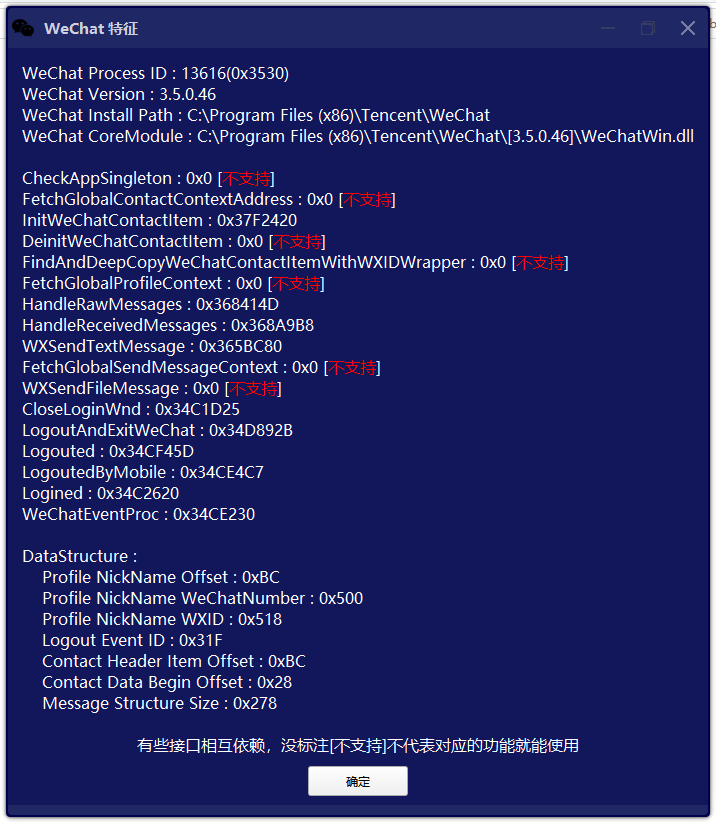
##### 切到前台

切到前台是指把该WeChat提升到桌面顶层上来

##### 查看定位特征



【特征完整的情况下，是全白的】



【定位不完整的话会把定位失败的部分标注为不支持】

##### 获取所有联系人信息



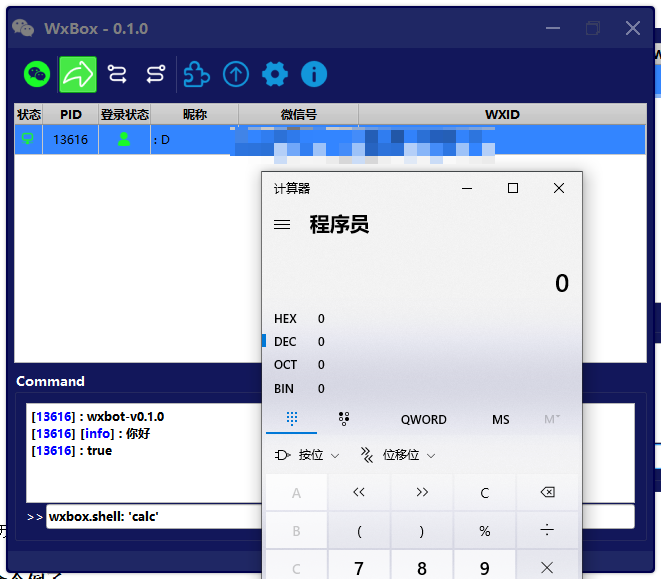
### Command



【Command区域说明】

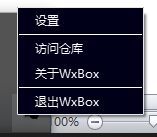
命令输入框支持一定数目的历史记录，用up和down键可以切换。

#### Command执行命令例子



**注意**：在WxBox上指令命令可以**省略>>前缀**，WxBox会自动补充

### WxBox托盘菜单



【Windows右下脚托盘菜单】

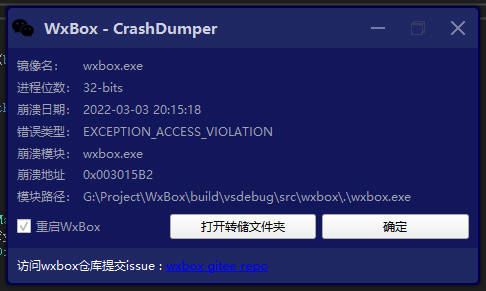
### 主界面快捷键

* Ctrl + O ：启动微信实例；
* Ctrl + , ：打开设置对话框

## 崩溃处理与crashdumper

WxBox.exe和WxBot.dll崩溃都能生成coredump，WxBox.exe由crashdumper生成，WxBot.dll的coredump由WeChat来生成，都会由对话框提示，可以把生成的coredump和日志直接发给我，或者到[WxBox代码仓库](https://gitee.com/phantom27/wxbox)上提交Issue。

crashdumper在WxBox的根目录下，界面长下面这样：

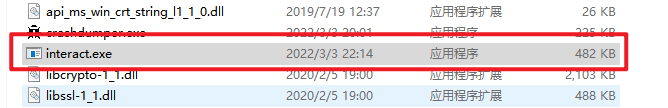


【CrashDumper主界面】

点击“打开转储文件夹”会自动选择最新的coredump，每个coredump有一个配对的txt，它是一个Call Stack、异常时线程上下文还有所有模块信息的集合。当WxBox崩溃时，会自动启动crashdumper，不需要去管它。

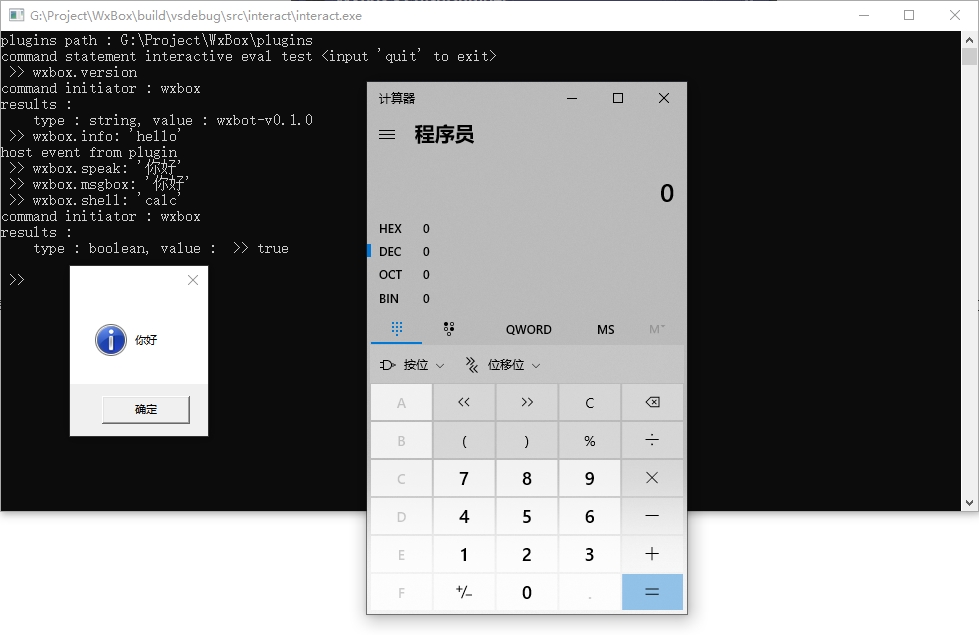
## interact

WxBox根目录还有个interact.exe：



【interact.exe】

它是用来测试Plugin命令的执行语法的

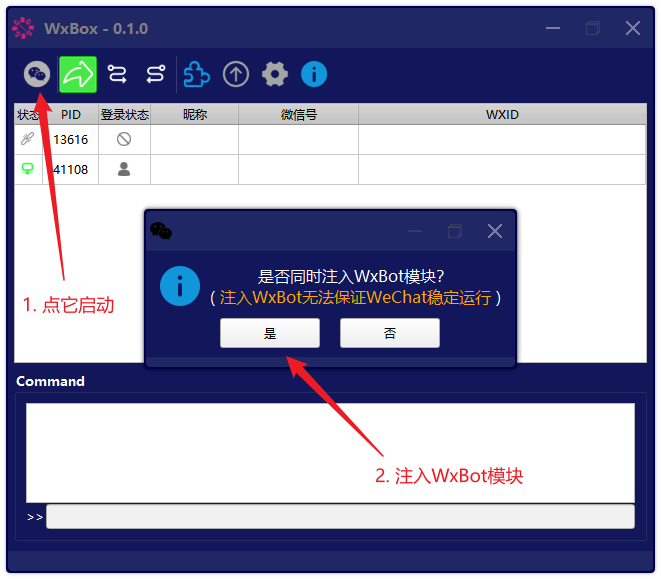


【interact演示】

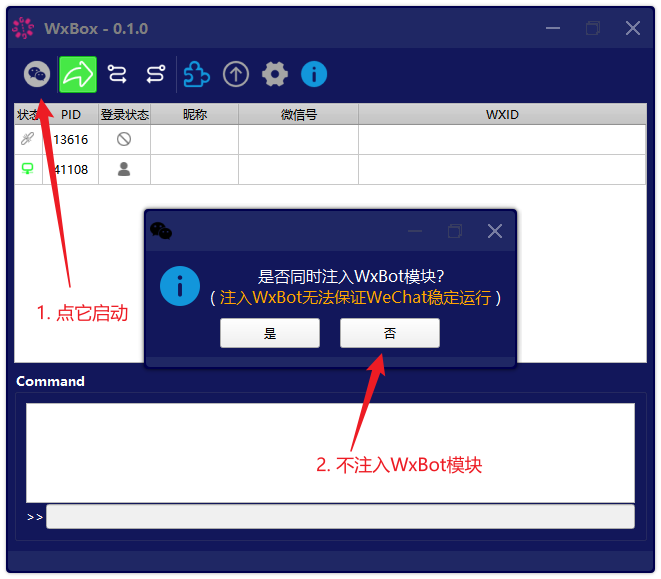
# WxBox测试

## 【WBTC1-启动WeChat新实例】

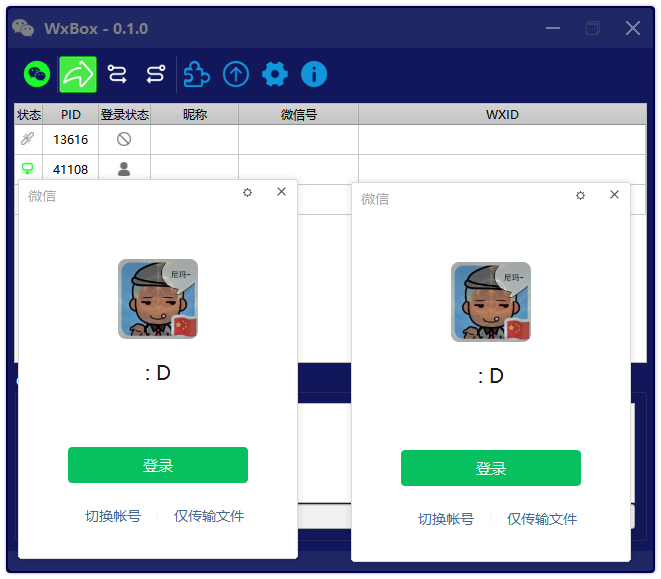
WxBox启动WeChat时如果正常安装了WeChat，那么不需要在Setting中配置路径，WxBox会自动定位，首先是从进程列表中定位，如果没有WeChat启动，在Windows下会到注册表中定位。



【启动新WeChat实例，并且注入WxBot模块】



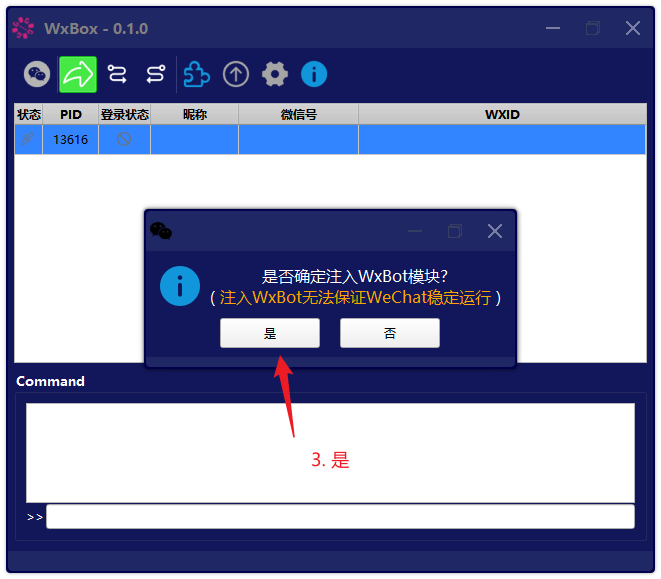
【启动新WeChat实例，但不注入WxBot模块】

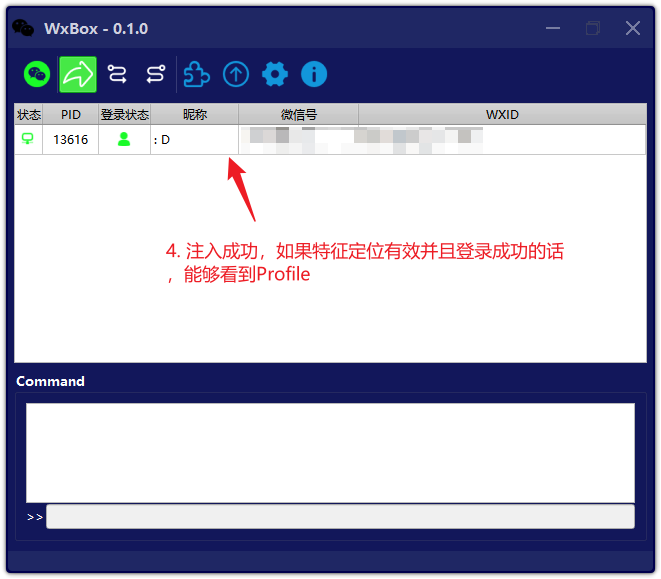


【启动成功的话能看到两个WeChat】

## 【WBTC2-给未注入WxBot的WeChat进程注入WxBot模块】



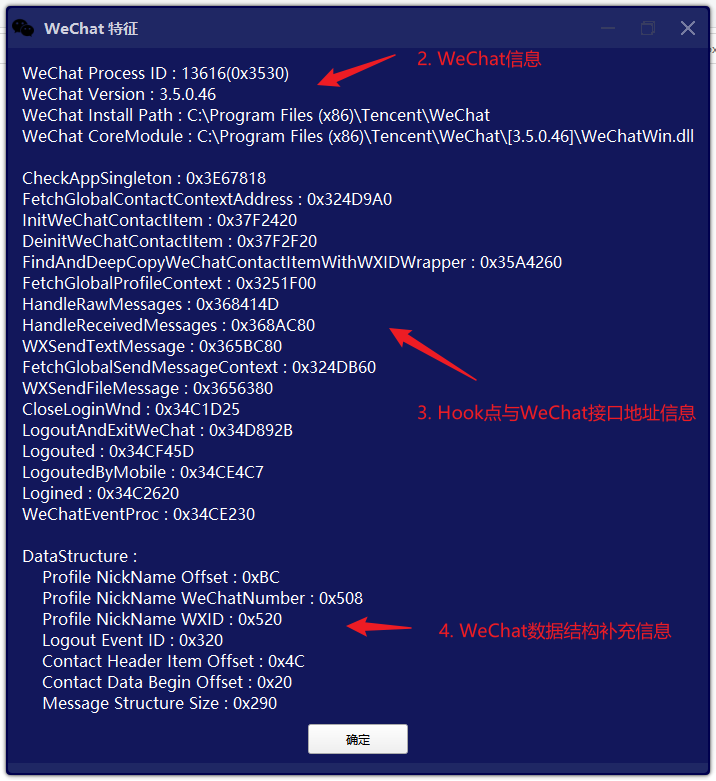




## 【WBTC3-查看定位特征】

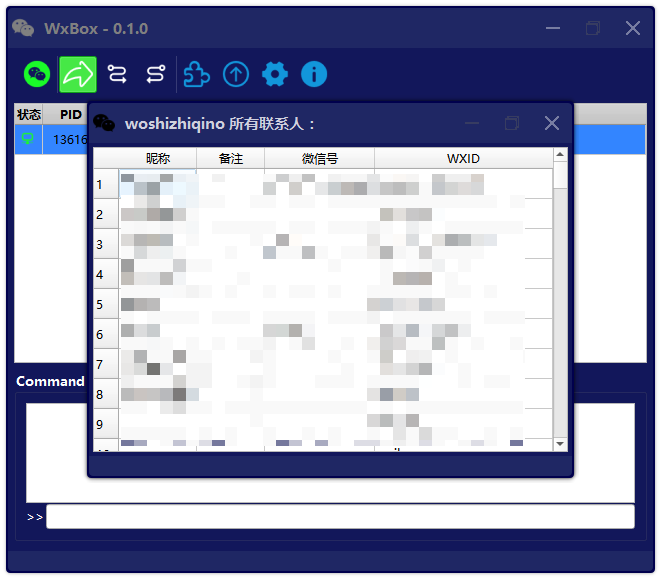
接着TC2，然后右键Client条目查看



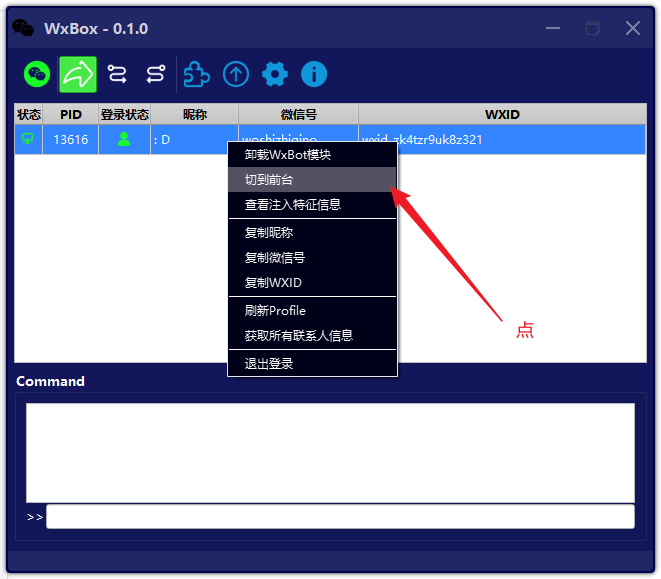


## 【WBTC4-查看所有联系人】



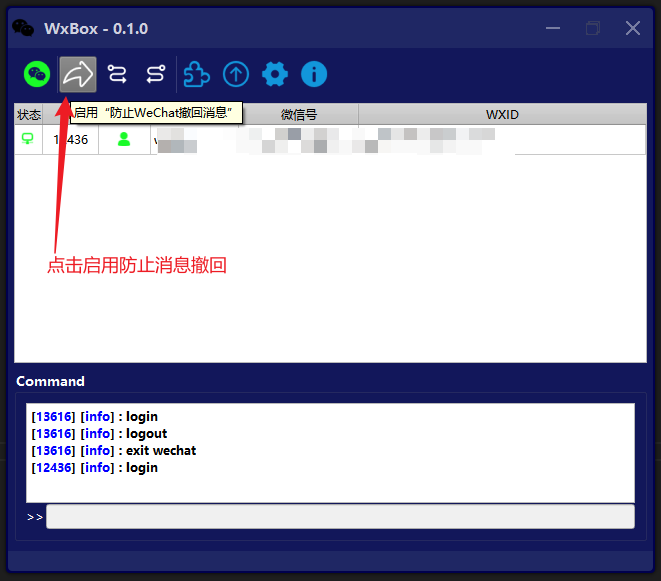


## 【WBTC5-测试切到前台】



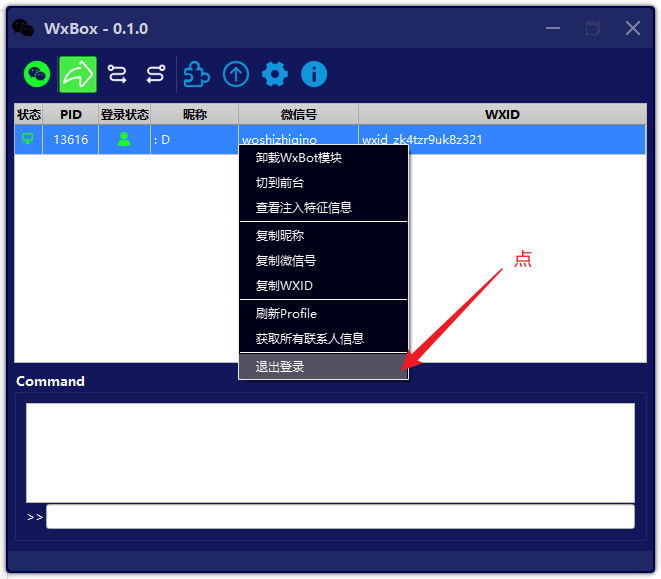
看看WeChat有没有被唤起来了

## 【WBTC6-测试防止消息撤回】



接着在手机上用别的WeChat给自己发送消息并撤回，观察是否能够防止PC上的WeChat消息撤回【给别人发也可以】。

## 【WBTC7-退出登录】

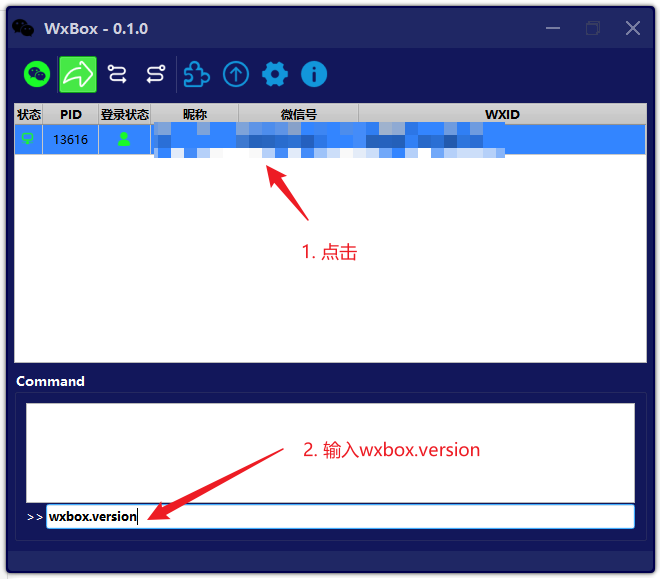


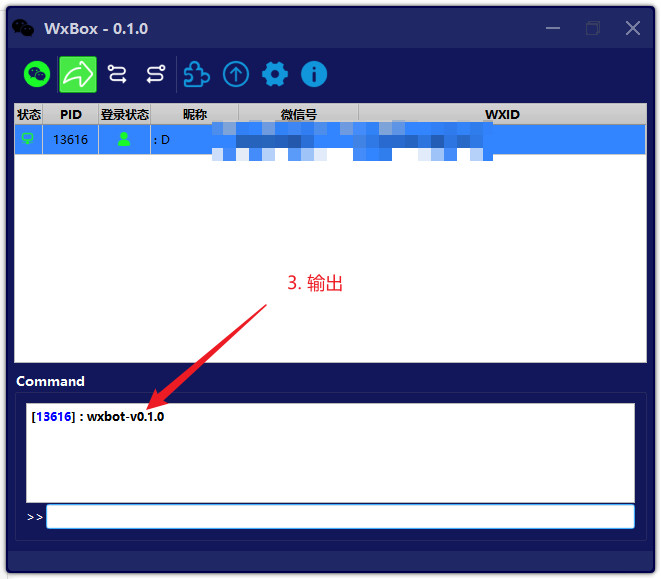
看看PC上的WeChat是否退出登录了

# Plugin预定义的wxbox模块API与WxBot测试

这里的api可以去看**plugins目录下的wxbox\_apis.txt**有完整的参数说明。可以通过WxBox的命令行框或者手机上的“文件传输助手”来测试。但是要**注意，在WxBox上执行的话，不要带>>前缀**。另外要注意标点符号，**'**和**"**都是**英文半角**的，还有下面的部分带参数的命令**不要直接复制使用**，因为下面并没有规范的处理标点符号，office docs文档也会自动处理标点符号的。

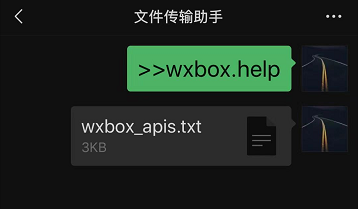
## 【WBPTC1-查看wxbox版本号】



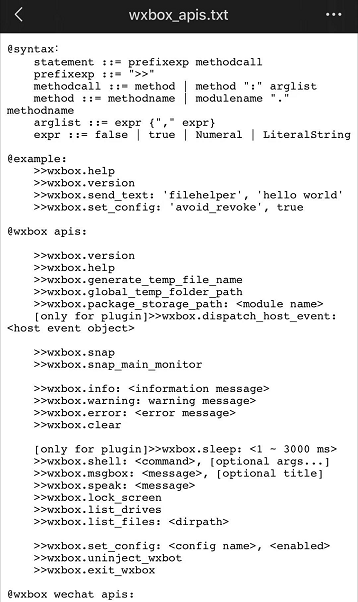


## 【WBPTC2-查看help文档】

这里改用手机的“文件传输助手”测试，打开手机WeChat的“文件传输助手”，然后发送“>>wxbox.help”



【手机文件传输助手会收到一个wxbox\_apis.txt文件】



【点开能查看wxbox的帮助文档】

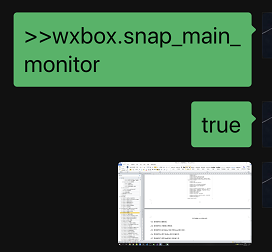
## 【WBPTC3-截图】

执行>>wxbox.snap命令，看文件传输助手能否接收到一张所有屏幕拼接的截图，**注意：**如果要**用WxBox执行**的话，**不要带>>**，下面所有的命令一样。



## 【WBPTC4-只截取主屏幕】

执行>>wxbox.snap\_main\_monitor命令，看文件传输助手能否接收到一张主屏幕的截图



## 【WBPTC5-给WxBox发送不同Level的日志】

手机执行

>>wxbox.info: "information信息"

>>wxbox.warning: "warning信息"

>>wxbox.error: "error信息"

看WxBox能否收到日志

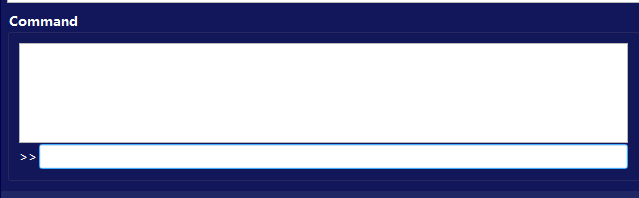


## 【WBPTC6-清空WxBox的日志输出】

手机执行>>wxbox.clear，看WxBox能否清空日志



【执行前】

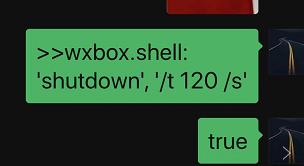


【执行后】

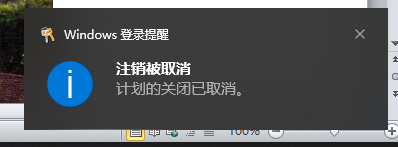
## 【WBPTC7-使用shell执行关机命令】

Windows环境的话

手机或WxBox执行>>wxbox.shell: 'shutdown', '/t 120 /s' 命令

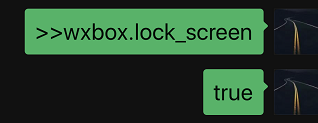


要想取消的话，打开powershell执行shutdown /a



## 【WBPTC8-锁定PC屏幕】

执行>>wxbox.lock\_screen，看PC能否锁屏

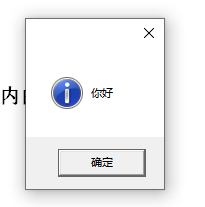




【Windows被锁屏了】

## 【WBPTC9-让PC弹窗】

执行>>wxbox.msgbox: "你好"

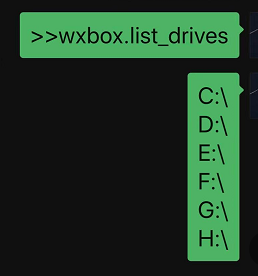


## 【WBPTC10-让PC播放一段语音】

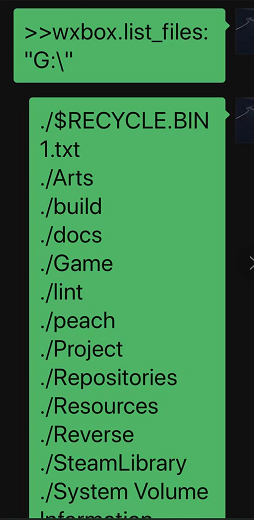
执行>> wxbox.speak: "你好"，听听Windows是否有播放你好语音

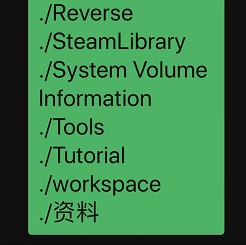
## 【WBPTC11-列出所有盘符以及某个文件夹内的所有文件】

执行>>wxbox.list\_drives列出所有盘符



执行>> wxbox.list\_files: 'G:\' 列出G盘所有文件





## 【WBPTC12-下载文件】

执行>> wxbox.download: "G:\你好.jpg" 观察文件传输助手是否收到该文件

## 【WBPTC13-手机控制WxBot模块Uninject】

执行>>wxbox.uninject\_wxbox 看看PC WeChat是否把WxBot模块给卸载了

## 【WBPTC14-手机控制WxBox退出】

执行>>wxbox.exit\_wxbox 看看WxBox是否退出程序

## 【WBPTC15-判断是否登录】

执行>>wxbox.is\_logined

## 【WBPTC16-触发PC退出登录】

执行>>wxbox.logout

## 【WBPTC17-查看登录用户的Profile】

执行>>wxbox.profile\_wxid 获取自己的wxid

执行>>wxbox.profile\_wxnumber 获取自己的wxnumber

执行>>wxbox.profile\_nickname 获取自己的昵称

## 【WBPTC18-nickname、wxid、wxnumber之间的互相转换】

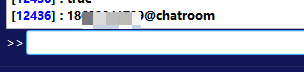
执行>>wxbox.nickname\_to\_wxid: ': D' 通过nickname获取wxid

执行>>wxbox.wxnumber\_to\_wxid: 'woshishui' 通过wxnumber获取wxid

执行>>wxbox.wxid\_to\_wxnumber: 'wxid\_ss10x8dxxxx' 通过wxid获取wxnumber

## 【WBPTC19-获取群聊的wxid】

执行>> wxbox.chatroom\_wxid: '群聊的名字'



## 【WBPTC20-向手机文件传输助手发送文本、图片和文件】

执行>>wxbox.send\_text\_to\_filehelper: "这是一条文本消息" 向文件传输助手发送文本消息

执行>>wxbox.send\_picture\_to\_filehelper: "G:\你好.jpg" 向文件传输助手发送图片

执行>>wxbox.send\_file\_to\_filehelper: "G:\你好.jpg" 向文件传输助手发送文件

## 【WBPTC21-向指定wxid联系人发送文本、图片和文件】

执行>>wxbox.send\_text: "wxid\_ss10x8dxxxx", "这是一条文本消息" 向指定wxid联系人发送文本消息

执行>>wxbox.send\_picture: "wxid\_ss10x8dxxxx", "G:\你好.jpg" 向指定wxid联系人发送图片

执行>>wxbox.send\_file: "wxid\_ss10x8dxxxx", "G:\你好.jpg" 向指定wxid联系人发送文件

## 【WBPTC22-向指定wxnumber联系人发送文本、图片和文件】

执行>>wxbox.send\_text\_with\_wxnumber: " woshishui", "这是一条文本消息" 向指定wxnumber联系人发送文本消息

执行>>wxbox.send\_picture\_with\_wxnumber: " woshishui", "G:\你好.jpg" 向指定wxnumber联系人发送图片

执行>>wxbox.send\_file\_with\_wxnumber: "woshishui", "G:\你好.jpg" 向指定wxnumber联系人发送文件

## 【WBPTC23-向指定群聊发送图片和文件】

执行>>wxbox.send\_picture\_to\_chatroom: 'xxxxxx群聊的wxid@chatroom',"G:\你好.jpg" 发送图片

执行>>wxbox.send\_file\_to\_chatroom: 'xxxxxx群聊的wxid@chatroom',"G:\你好.jpg" 发送文件

## 【WBPTC24-向指定群聊发送文本，可选带@列表】

执行>> wxbox.send\_text\_to\_chatroom: 'xxxxxx群聊的wxid@chatroom','这是一条文本消息','wxid\_8yxaroxxxdo12','wxid\_8yxaroxxxdo12'

文本消息的后面接的是要@的人的wxid，可选并且不定数目

## 【WBPTC25-在指定群聊中@一部分人】

执行>> wxbox.chatroom\_notify: 'xxxxxx群聊的wxid@chatroom','wxid\_xxxrokryddo12','wxid\_8yxaroxxxdo12'

群聊wxid后接的是要@的人的wxid，可选并且不定数目

## 【WBPTC26-在指定群聊中@所有人】

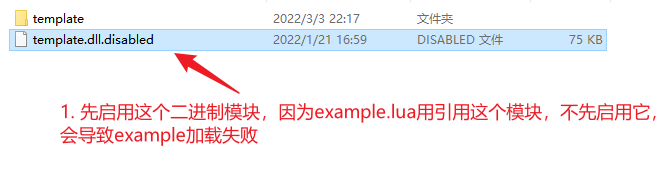
执行>> wxbox.chatroom\_notify\_all: 'xxxxxx群聊的wxid@chatroom'

# Plugin测试

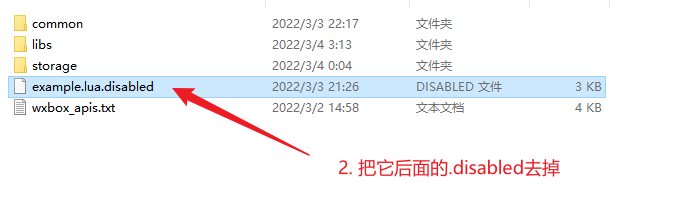
可以通过WxBox的命令行框或者手机上的“文件传输助手”来测试，最好在PC上用WxBox测，因为输入会方便的多。下面在提供的example扩展中执行测试。

## 【PTC1-测试启用提供的example样本扩展】

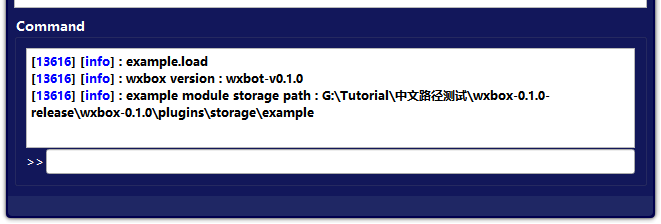
**注意：**WxBox的Plugin支持热更新，修改Plugin内容后直接保存就可以了，WxBot会自动刷新内容。



【启动二进制的template扩展，**把文件名后面的.disabled去掉**】



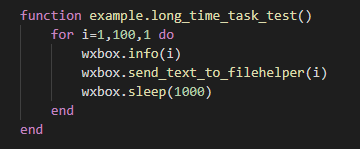
【启动example扩展】



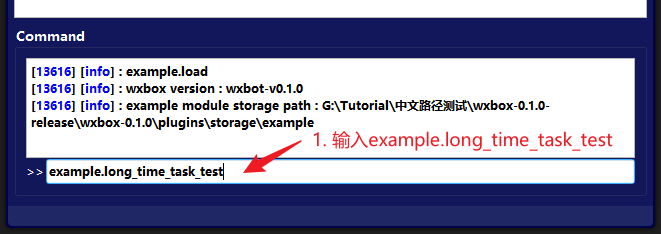
【可以看到example扩展的加载事件打出的日志】

## 【PTC2-测试example样本扩展的long\_time\_task\_test方法】

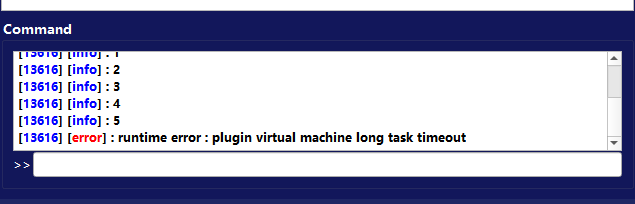
example的long\_time\_task\_test方法，里面执行很长的耗时任务，观察日志，看看执行后，WxBot有没有把任务给中断掉



【long\_time\_task\_test方法的内容】



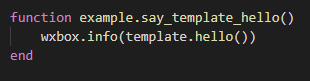
【输入：example.long\_time\_task\_test】

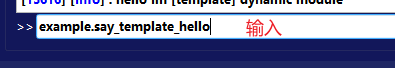


【默认超时时长是5秒，可以看到WxBot把这个耗时任务给结束掉了】

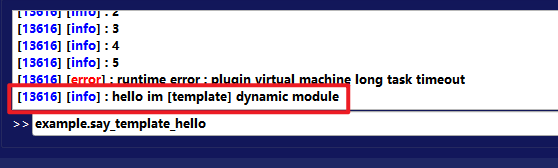
## 【PTC3-测试example样本扩展调用二进制模块】

say\_template\_hello方法调用了二进制template.dll模块中的方法





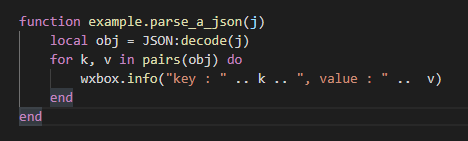
【输入example.say\_template\_hello】



【观察是否有这条返回消息】

## 【PTC4-测试example样本扩展调用第三方lua库】

example.parse\_a\_json方法调用了第三方的lua库来解析一个json





【输入example.parse\_a\_json: '{"name":"你好"}'】

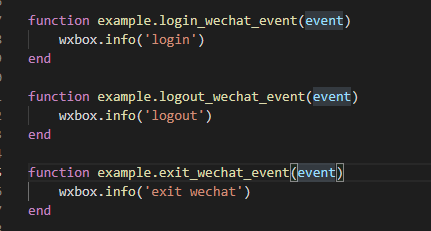


【返回结果】

## 【PTC5-观察example样本扩展是否能够捕获WeChat声明周期】

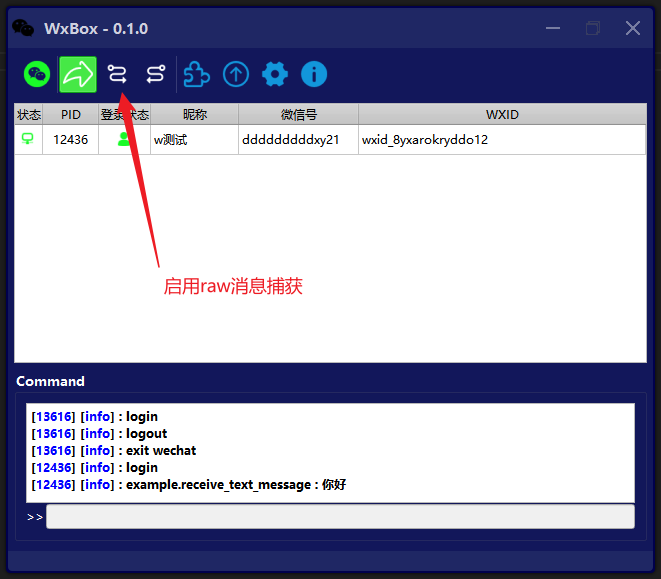
登录、注销然后关闭微信后，看WxBox是否有下面这些日志



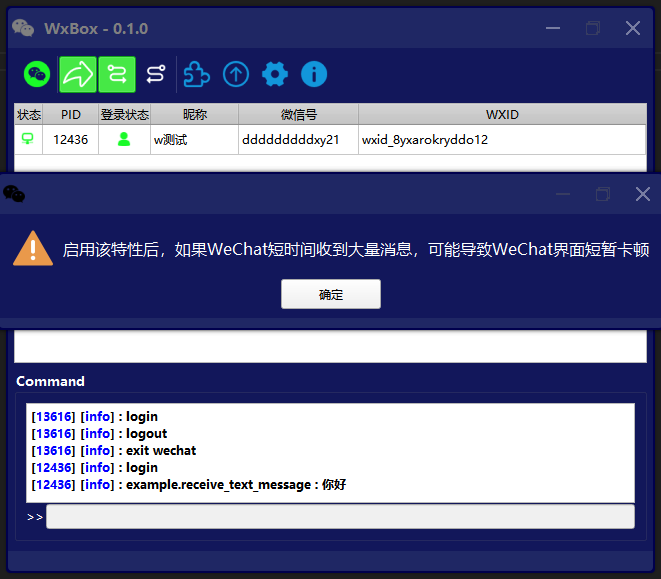


【example关于WeChat生命周期的事件处理方法内容】

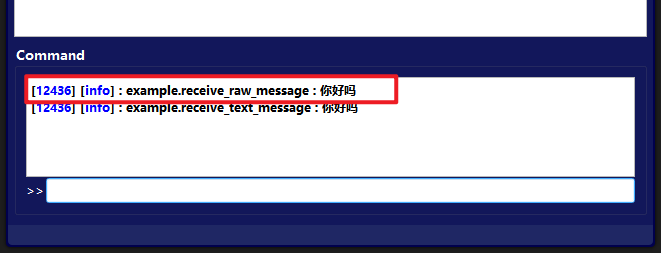
## 【PTC6-测试捕获原始消息】



【启用raw消息捕获】



【如同提示说的，短时间大量消息进来会导致WeChat界面短暂卡顿】

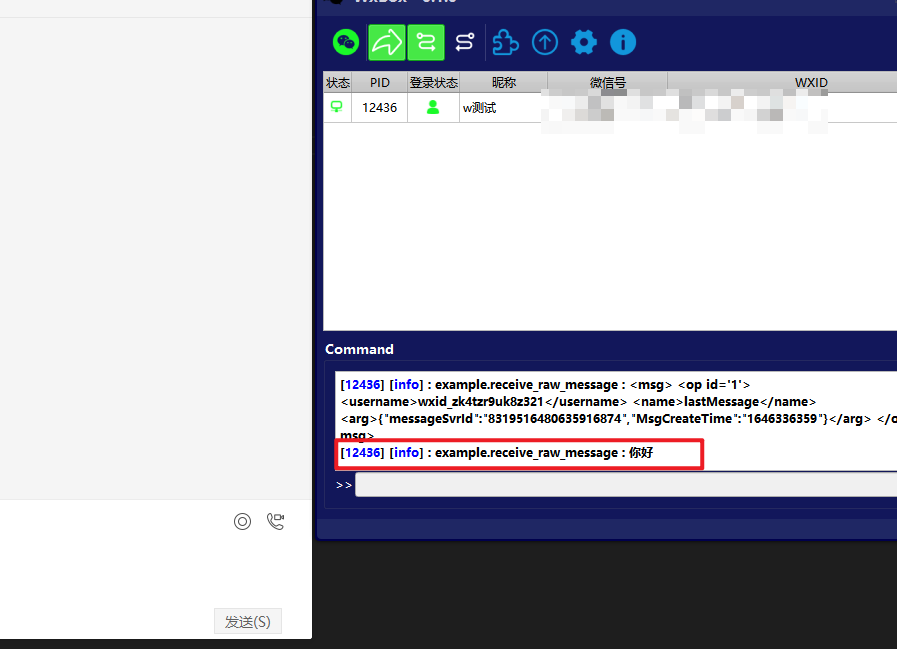


【用手机给自己发送一条消息，看看能否在日志上看到消息】

## 【PTC7-测试消息过滤】

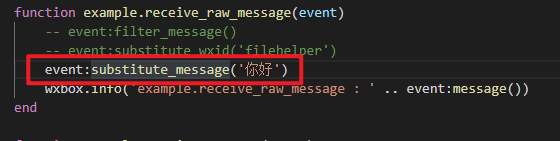


【把example.lua中receive\_raw\_message的第**一行注释解开**，这里是**过滤了所有的消息**】



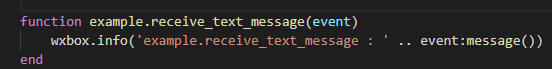
【如果功能正常的话，能看到WxBox能接收到消息日志，但是PC WeChat上接收不到】

## 【PTC8-测试消息替换】



【把receive\_raw\_message的第三行注释解开，观察是否接收的所有消息都变成了“你好”】

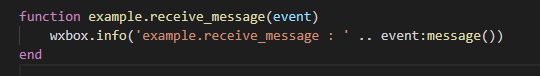
## 【PTC9-测试捕获处理过的文本消息】

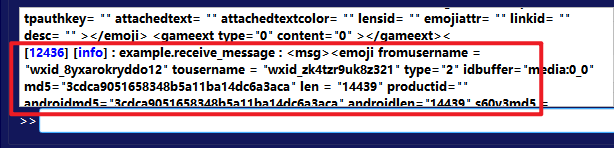




【观察消息进来后，能否接收到这条日志】

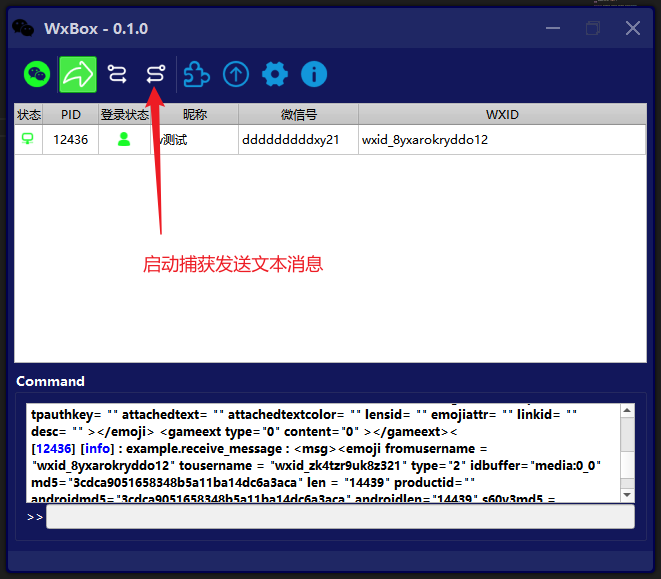
## 【PTC10-测试捕获处理过的非文本消息】



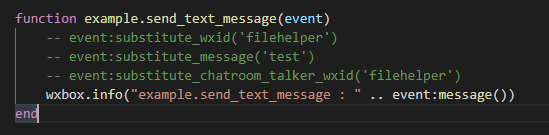


【发个表情包，看能否接收到这条日志】

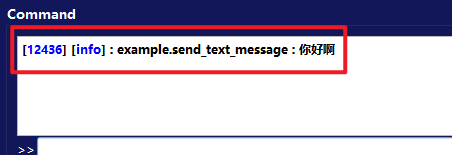
## 【PTC11-测试捕获发送文本消息】



【对于发送消息的捕获】

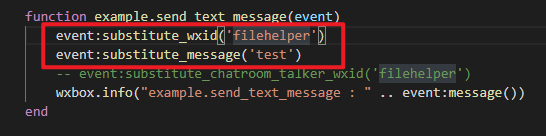


【example的send\_text\_message事件处理】

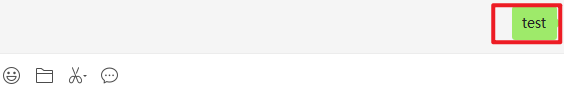


【在PC WeChat上给任意人发送一串文本消息看看能不能收到这条日志】

## 【PTC12-测试替换发送文本消息】

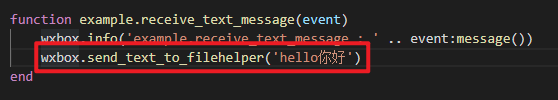


【把example.send\_text\_message的前两行的注释去掉】



【观察是否都替换为把test发送给了文件传输助手】

## 【PTC13-测试自动回复】



【给example.receive\_text\_message添加如上图一行】



【随便给自己发一条文本消息，看看能否给文件传输助手自动发送一条hello你好】