

VIP版本wxautox补充文档

VIP版本 wxautox 补充文档

一、概述

二、文档

(一) 微信WeChat

1. 微信对象

1.1 发送消息 - 打字机模式 `SendTypingText`

1.2 获取群聊列表 `GetAllRecentGroups`

1.3 发送消息 - 发送自定义表情包 `SendEmotion`

1.4 消息免打扰 `MuteNotifications`

1.5 邀请入群 `AddGroupMembers`

1.6 优化 `ChatWith` 方法

1.7 修改群聊名、备注、群公告、我在本群的昵称 `ManageGroup` (20241211新增)

1.8 修改好友备注、增加标签 `ManageFriend` (20241211新增)

1.9 自动保存消息内的卡片链接 `parseurl` 参数 (20241211新增)

1.10 获取当前聊天窗口详情 `CurrentChat` 方法增加 `details` 参数 (20241211新增)

2. 消息对象

2.1 `FriendMessage` 新增 `add_friend` 方法

2.2 `quote` 方法添加 `at` 参数

2.3 通过消息对象获取好友（群好友）信息 `sender_info` 方法

2.4 通过消息对象获取当前聊天页面详情 `details` 属性

2.5 解析卡片链接 `parse_url` 方法 (20241211新增)

3. 登录窗口对象

(二) 朋友圈WeChatMoments

1. 朋友圈窗口对象

1.1 获取朋友圈内容 `GetMoments`

1.2 刷新朋友圈

1.3 关闭朋友圈

2. 朋友圈对象

2.1 获取朋友圈内容

2.2 获取朋友圈图片

2.3 获取好友信息

三、BUG修复

常规BUG修复

四、用户协议

使用许可及限制

1. 合法用途

2. 禁止行为

3. 风险与责任

一、概述

该版本为 wxauto 的plus版本 wxautox，在保留 wxauto 的所有功能的基础上，修复了 wxauto 中的诸多BUG，增加了一些功能，且无需移动鼠标，效率更高、更稳定。

该文档为 wxauto 的补充文档，<https://wxauto.loux.cc>文档中已存在的内容将不会出现在该文档中，仅供VIP用户参考

该版本兼容开源版本 wxauto，代码无需修改，原有的 wxauto 项目只需将

```
1 | from wxauto import WeChat
```

改为

```
1 | from wxautox import WeChat
```

即可完成迁移。

二、文档

(一) 微信WeChat

1. 微信对象

微信对象指的是由以下代码获取到的对象，后续文档内不再重复定义

```
1 | from wxautox import WeChat
2 |
3 | wx = WeChat() # 获取微信对象
```

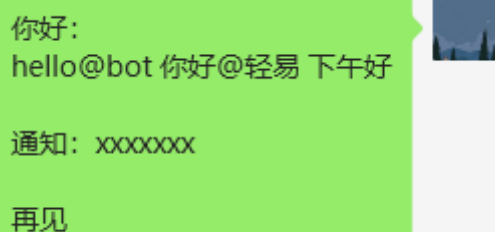
微信对象新增几个参数：

参数	类型	默认值	说明
nickname	str	None	指定微信窗口，用于区分不同微信客户端，如张三、李四
mouse_move	bool	False	是否移动鼠标来进行操作，默认为不移动
myinfo	bool	False	是否在初始化时获取本人微信号等信息，默认为不获取

1.1 发送消息 - 打字机模式 SendTypingText

该方法可模拟打字机模式逐字输入，行为更贴近人类。该方法还支持消息内@群好友，不必像原来一样@内容只能固定在文字内容前面，效果如下：

```
1 | text = '''你好：
2 | hello{@bot}你好{@轻易}下午好
3 |
4 | 通知：xxxxxxx
5 |
6 | 再见'''
7 | wx.SendTypingText(text)
```



1.2 获取群聊列表 GetAllRecentGroups

该方法可以获取通讯录中最近的群聊名称，以方便对群的操作，返回值格式为list，列表元素为元组，格式为('群聊名','人数')

参数	类型	默认值	说明
speed	int	1	动速度，数值越大滚动越快，但是太快可能导致遗漏，建议速度1-3之间
wait	float	0.05	滚动等待时间，建议和speed一起调整，直至适合你电脑配置和微信群数量达到平衡，不遗漏数据

```
1 groups = wx.GetAllRecentGroups()
2 # [
3 #     ('工作群', '500')
4 #     ('街坊群', '456')
5 #     ('八卦群', '123')
6 #     ...
7 # ]
```

1.3 发送消息 - 发送自定义表情包 SendEmotion

参数	类型	默认值	说明
emotion_index	int	/	要发送的索引值，从0开始

如果index大于等于账号内自定义表情数量，则会发送失败，返回False，成功返回True

```
1 index = 0
2 wx.SendEmotion(emotion_index=index)
```

1.4 消息免打扰 MuteNotifications

参数	类型	默认值	说明
mute	bool	True	对当前聊天对象开启或关闭消息免打扰，True开启免打扰，False关闭免打扰

```
1 group = '工作群'
2 mute = True # True为开启免打扰，False为关闭免打扰
3
4 # 先打开指定聊天窗口，再执行免打扰操作
5 wx.ChatWith(group)
6 wx.MuteNotifications(mute=mute)
```

1.5 邀请入群 AddGroupMembers

参数	类型	说明
group	str	群名或者群备注名
members	list	成员列表，可以是昵称、备注名、微信号；最好是微信号或者唯一的备注名

```
1 from wxautox import weChat
2 import time
3
4 wx = weChat()
5
6 targets = [
7     '好友1',
8     '好友2',
9     '好友3'
10 ]
11 group = 'wxauto交流群'
12 wx.AddGroupMembers(group, targets)
```

1.6 优化 ChatWith 方法

优化了 chatwith 及各种发送消息的方法，增加exact参数，用于判断在搜索who的时候是否需要精准匹配，默认为False，改为True则需要一字不差才进行发送

参数	类型	默认值	说明
who	str	/	要打开的聊天框好友名; * 最好完整匹配，不完全匹配只会选取搜索框第一个
timeout	int	2	超时时间，默认2秒
exact	bool	False	是否精确匹配，默认False

1.7 修改群聊名、备注、群公告、我在本群的昵称 ManageGroup (20241211新增)

参数	类型	默认值	说明
name	str	None	修改群名称
remark	str	None	修改备注名
myname	str	None	修改我的群昵称
notice	str	None	修改群公告
quit	bool	False	是否退出群，当该项为True时，其他参数无效

```

1 remark = '工作群（不要发错）'
2 wx.ManageGroup(remark=remark)
3 # 返回值: dict
4 # {
5 #     'remark': True    # 如果未成功则为False
6 # }

```

1.8 修改好友备注、增加标签 ManageFriend (20241211新增)

参数	类型	默认值	说明
remark	str	None	修改备注名
tags	list	None	要增加的标签列表

```

1 remark = '张三'
2 tags = ['同事']
3 wx.ManageFriend(remark=remark, tags=tags)
4 # 返回值: bool, 是否成功修改备注名或标签

```

1.9 自动保存消息内的卡片链接 parseurl 参数 (20241211新增)

以下方法增加 parseurl 参数，用于自动解析消息内卡片消息的URL链接

- GetAllMessage
- GetListenMessage
- GetAllNewMessage
- GetNextNewMessage
- AddListenChat

```

1 msgs = wx.GetAllMessage(parseurl=True)
2 msgs
3 # 卡片链接解析后的格式为: "[wxauto卡片链接解析]https://xxxxxxx"
4 # [
5 #     ...
6 #     ['Time', '13:34'],
7 #     ['Self', '[wxauto卡片链接解
8 #         析]https://mp.weixin.qq.com/s/x8ilebSF5_KYd0PloyZm-Q']

```

1.10 获取当前聊天窗口详情 CurrentChat 方法增加 details 参数 (20241211新增)

```

1 wx.CurrentChat(details=True)
2 # {
3 #     'chat_type': 'group',
4 #     'chat_name': 'wxauto四群',
5 #     'group_member_count': 490
6 # }

```

2. 消息对象

有关消息对象的更多内容，可点击查看[消息对象 | wxauto](#)

消息对象指的是 `GetAllMessage`、`GetListenMessage` 等所有有关获取消息的方法返回的列表内的对象元素，后续文档内不再重复定义

```
1 msgs = wx.GetAllMessage()
2 msg = msgs[-1]    # 以最后一条消息作为消息对象，命名为msg，后续文档内不再重复定义
```

2.1 FriendMessage 新增 add_friend 方法

参数	类型	默认值	说明
addmsg	str	None	添加好友的消息，不填则微信默认
remark	str	None	备注名，不填则无
tags	list	None	标签列表，不填则无
permission	str	'朋友圈'	朋友圈权限，可选值：'朋友圈'，'仅聊天'

用于支持通过群消息快捷申请好友，以下代码即可发起好友请求

```
1 if msg.type == 'friend':    # 仅当消息类型为friend时可用该方法
2     msg.add_friend()
```

2.2 quote 方法添加 at 参数

参数	类型	默认值	说明
msg	str	/	要发送的消息内容
at	str list	None	要@的人的昵称或微信号，单人str多人list

用于在引用消息的同时，@指定人员

```
1 if msg.type in ['friend', 'self']:    # 仅当消息类型为friend或者self时可用该方法
2     msg.quote('xxx', at=['张三', '李四'])
```

2.3 通过消息对象获取好友（群好友）信息 sender_info 方法

```
1 from wxautox import weChat
2
3 wx = weChat()
4 msgs = wx.GetAllMessage()
5
6 # 获取最后一条消息，假设为好友发来的消息
7 msg = msgs[-1]
8
9 # 仅消息类型为`friend`的才有该方法
10 if msg.type == 'friend':
11     sender_info = msg.sender_info()
12
13 # {
14 #     'nickname': '张三',
15 #     'id': '123456',
16 #     'remark': '同事张三',
```

```
17 # 'tags': '同事',
18 # 'source': '通过扫一扫添加',
19 # 'signature': '张三的个性签名'
20 # }
```

参数名	说明
nickname	昵称
id	消息的 ui 控件提取到的 runtimeid，唯一，不用可忽略
remark	备注
tags	标签
source	来源
signature	个性签名

2.4 通过消息对象获取当前聊天页面详情 details 属性

```
1 ... # 省略msg对象的获取过程
2
3 print(msg.details)
4 # {
5 #     'id': '428532284143281',
6 #     'type': 'friend',
7 #     'sender': '张三',
8 #     'content': '哈哈',
9 #     'sender_remark': '同事张三',
10 #     'chat_type': 'group',
11 #     'chat_name': '工作群',
12 #     'group_member_count': 54
13 # }
```

参数名	说明
id	消息的 ui 控件提取到的 runtimeid，唯一，不用可忽略
type	消息类型，friend其他人发的消息、time时间消息、sys系统消息、self自己发的消息
sender	消息发送人的昵称
content	消息内容
sender_remark	消息发送人的备注，没有则为None
chat_type	聊天类型，group为群聊、friend为好友聊天、official为公众号
chat_name	当天聊天对象名，群名或好友名...
group_member_count	群聊人数，如果是群消息则有该参数

2.5 解析卡片链接 parse_url 方法 (20241211新增)

3. 登录窗口对象

登录窗口对象指的是登录时的窗口对象元素，用于自动登录或者获取二维码等操作

1. 获取登录二维码

```
1 from wxautox.elements import Loginwnd
2
3 # 当客户端掉线时
4
5 ## 1. 获取二维码
6 wxlogin = Loginwnd()
7 if wxlogin.UiaAPI.exists(3):
8     wxlogin.reopen() # 重新打开微信登录窗口
9     qrcode = wxlogin.get_qrcode() # 保存并返回二维码图片地址
10    ... # 拿到的二维码自行处理
```

2. 自动登录

```
1 ## 2. 自动登录（如果是被踢下线则需要重新扫码登录）
2 wxlogin = Loginwnd()
3 if wxlogin.UiaAPI.exists(3):
4     wxlogin.reopen() # 重新打开微信登录窗口
5     login_result = wxlogin.login() # 当login_result为True时，登录成功，None则
    登录失败需要扫码
```

（二）朋友圈WeChatMoments

1. 朋友圈窗口对象

朋友圈窗口对象指的是朋友圈的窗口对象，提供对朋友圈窗口的各种操作，如获取朋友圈内容、刷新、关闭等功能。

获取朋友圈对象

```
1 from wxautox import weChat
2
3 wx = WeChat()
4 pyq = wx.Moments() # 打开朋友圈并获取朋友圈窗口对象（如果为None则说明你没开启朋友圈，
    需要在手机端设置）
```




1.1 获取朋友圈内容 GetMoments

参数	类型	说明	说明
next_page	bool	False	是否翻页后再获取
speed1	int	3	翻页时的滚动速度，根据自己的情况进行调整，建议3-10自行调整
speed2	int	1	翻页最后时的速度，避免翻页过多导致遗漏所以一般比speed1慢，建议1-3

```

1 # 获取当前页面的朋友圈内容
2 moments = pyq.GetMoments()
3
4 # 通过`next_page`参数获取下一页的朋友圈内容
5 moments = pyq.GetMoments(next_page=True)

```

这里获取到的moments是一个列表，列表中每个元素都是一个**朋友圈对象**。

1.2 刷新朋友圈

```

1 # 刷新朋友圈
2 pyq.Refresh()

```

1.3 关闭朋友圈

```

1 # 关闭朋友圈
2 pyq.Close()

```

2. 朋友圈对象

朋友圈对象指的是朋友圈中的每一条朋友圈，提供对朋友圈的各种操作，如获取朋友圈内容、点赞、评论等功能。

```

1 # 获取朋友圈对象
2 moments = pyq.GetMoments()
3
4 # 获取第一条朋友圈
5 moment = moments[0]

```



2.1 获取朋友圈内容

```

1 # 获取朋友圈内容
2 info = moment.info
3 # {
4 #     'type': 'moment',          # 类型，分为`朋友圈`和`广告`
5 #     'id': '4236572776458165', # ID
6 #     'sender': '天天鲜花2号客服', # 发送者

```

```

7   #      'content': '客订花束',          # 内容，就是朋友圈的文字内容，如果没有文字内容则
    为空字符串
8   #      'time': '4分钟前',             # 发送时间
9   #      'img_count': 3,                 # 图片数量
10  #      'comments': [],                 # 评论
11  #      'addr': '',                     # 发送位置
12  #      'likes': []                     # 点赞
13  # }
14
15  moment.sender
16  # '天天鲜花2号客服'
17
18  moment.content
19  # '客订花束'
20
21  moment.time
22  # '4分钟前'
23
24  # info中所有的键值对都可以通过对象的属性来获取，就不一一列举了
25  ...

```

2.2 获取朋友圈图片

- SaveImages(): 保存朋友圈图片到本地

参数	类型	默认值	说明
save_index	int list	None	保存图片的索引，可以是一个整数或者一个列表，如果为None则保存所有图片
savepath	str	None	绝对路径，包括文件名和后缀，例如："D:/Images/微信图片_xxxxxx.jpg"，如果为None则保存到默认路径

```

1  # 获取朋友圈图片
2  images = moment.SaveImages()
3  # [
4  #     'D:/Images/微信图片_xxxxxx1.jpg',
5  #     'D:/Images/微信图片_xxxxxx2.jpg',
6  #     'D:/Images/微信图片_xxxxxx3.jpg',
7  #     ...
8  # ]

```

2.3 获取好友信息

```
1 # 获取好友信息
2 moment.sender_info()
3 # {
4 #     'nickname': None,
5 #     'id': 'xxxxxxx',
6 #     'remark': None,
7 #     'tags': None,
8 #     'source': '通过搜索手机号添加',
9 #     'signature': None
10 # }
```

三、BUG修复

常规BUG修复

- 修复 `getNextNewMessage` 在窗口未置顶时无法判断新消息的bug
- 接收监听消息时可能获取到历史消息的问题
- `SendMsg` 方法中的 `clear` 参数失效问题
- 引用方法不稳定的bug
- 添加好友时因好友过多导致卡在选择权限报错的问题
- 修复了已登录微信但是窗口处于关闭状态时初始化报错的问题
- 有部分设备或系统在自动保存图片的时候会无法成功修改保存路径，增加一个复制粘贴修改保存路径的方法，如果正常使用则无序修改，否则可将 `wxautox.elements` 中的 `WxParam.SAVE_PATH_METHOD` 改为2，尝试第二种方法进行修改路径的操作
- 优化 `GetAllFriends` 方法，解决重复、遗漏的问题
- 修复消息对象调用转发功能时会打开非文本类消息内容的BUG

四、用户协议

用户协议

最后更新日期：2024年12月7日

感谢您使用 `wxauto(x)`（以下简称“本项目”）。为明确用户责任，特制定本用户协议（以下简称“协议”）。请在使用前仔细阅读并同意以下条款。您使用本项目即视为您已接受并同意遵守本协议。

使用许可及限制

1. 合法用途

用户应仅将本项目用于合法用途，包括但不限于：

- 个人学习和研究。
- 在不违反适用法律法规及第三方协议（如[微信用户协议](#)）的情况下个人使用。

2. 禁止行为

不得私自删除该协议中任何内容。

用户不得将本项目用于以下用途：

- 开发、分发或使用任何违反法律法规的工具或服务。
- 开发、分发或使用任何违反第三方平台规则（如[微信用户协议](#)）的工具或服务。
- 从事任何危害他人权益、平台安全或公共利益的行为。

3. 风险与责任

用户在使用本项目时，须自行确保其行为的合法性及合规性。
任何因使用本项目而产生的法律风险、责任及后果，由用户自行承担。