wxautox调用文档

一、环境配置

1. 微信版本

本项目支持 3.9.8+ 以上版本的微信客户端，不支持4.0测试版本

2. 安装

```
pip install wxautox-[指定版本号]-py3-none-any.whl
```

> 运行命令的路径需跟whl文件所在路径相同，否则需要制定whl绝对路径

二、微信相关

1. 获取实例

```
# 导入
>>> from wxautox import WeChat

# 获取微信窗口对象
>>> wx = WeChat()
初始化成功，获取到已登录窗口：xxxx
```

> 上面定义了wx变量，下述文档不再重复定义和解释wx变量

2. 发送消息相关

2.1 发送文字消息（普通模式）

wx.SendMsg

方法说明：

给指定人员（群组）发送消息

参数：

参数名	类型	默认值	说明
msg	str	/	要发送的文字内容
who	str	None	要发送给谁，默认则发送给当前打开的页面
clear	bool	True	是否清除原本聊天编辑框的内容
at	list,str	None	要@的人，可以是一个人或多个人，格式为str或list，例如："张三"或["张三", "李四"]
exact	bool	False	who参数是否精确匹配，默认False

返回值：

如果发送成功则返回True，否则为None

示例：

```
# 给“文件传输助手”发送消息
>>> who = '文件传输助手'
>>> msg = '''这是一条消息
这是第二行
这是第三行
'''
>>> wx.SendMsg(msg, who=who)
```

2.2 发送文字消息（打字机模式）

wx.SendTypingText

参数：

参数名	类型	默认值	说明
msg	str	/	要发送的文字内容
who	str	None	要发送给谁，默认则发送给当前打开的页面
clear	bool	True	是否清除原本聊天编辑框的内容
at	list,str	None	要@的人，可以是一个人或多个人，格式为str或list，例如："张三"或["张三", "李四"]
exact	bool	False	who参数是否精确匹配，默认False

返回值：无

示例：

```
# 发送消息
>>> who = '文件传输助手'
>>> msg = '''这是一条消息
这是第二行
这是第三行'''
wx.SendTypingText(msg, who=who)

# 换行及@功能
who = '工作群'
msg = '''各位下午好
{@张三}负责xxx
{@李四}负责xxxx'''
wx.SendTypingText(msg, who=who)
```

2.3 发送文件、图片、视频消息

wx.SendFiles

参数：

参数名	类型	默认值	说明
filepath	str | list	/	指定文件路径，单个文件str，多个文件list
who	str	None	要发送给谁，默认则发送给当前打开的页面
exact	bool	False	who参数是否精确匹配，默认False

返回值(bool):

如果发送成功则返回True，否则为False

示例:

```
# 给“文件传输助手”发送文件（图片同理）
>>> who = '文件传输助手'
# 指定文件路径（绝对路径）
>>> files = ['D:/test/test1.txt', 'D:/test/test2.txt', 'D:/test/test3.txt']
>>> wx.SendFiles(files, who=who)
```

2.4 发送@所有人消息

wx.AtAll

参数:

参数名	类型	默认值	说明
msg	str	None	要发送的文字内容，可为空
who	str	None	群名，默认则发送给当前打开的页面
exact	bool	False	who参数是否精确匹配，默认False

返回值：无

示例:

```
>>> who = '工作群'
>>> msg = '通知：XXXXXXXXXX'
>>> wx.AtAll(msg=msg, who=who)
```

2.5 发送自定义表情包

wx.SendEmotion

参数:

参数名	类型	默认值	说明
emotion_index	int	/	表情索引，从0开始计数
who	str	None	群名，默认则发送给当前打开的页面
exact	bool	False	who参数是否精确匹配，默认False

示例:

```
>>> who = '文件传输助手'
>>> emotion_index = 25   # 要发送第25个自定义表情
>>> wx.SendEmotion(emotion_index, who=who)
```

返回值(bool):

如果发送成功则返回True，否则为False

2.6 发送URL卡片

wx.SendUrlCard

参数:

参数名	类型	默认值	说明
url	str	/	要发送的url地址
friends	str|list	/	要发送给哪些人，str或者list

示例:

```
friends = ['张三', '李四']
url = "https://docs.wxauto.org"
wx.SendUrlCard(url, friends)
```

返回值（bool）：

如果发送成功则返回True，否则为False

3. 获取消息

3.1 获取历史消息

3.1.1 获取微信主窗口当前聊天窗口UI框架已加载所有聊天记录

wx.GetAllMessage

参数:

参数名	类型	默认值	说明
savepic	bool	False	是否自动保存聊天图片
savevideo	bool	False	是否自动保存聊天视频
savefile	bool	False	是否自动保存聊天文件
savevoice	bool	False	是否自动保存聊天语音转文字内容
parseurl	bool	False	是否自动获取卡片链接

示例:

```
savepic = False
savefile = False
savevoice = False
parseurl = False
msgs = wx.GetAllMessage(savepic, savefile, savevoice, parseurl)
```

返回值(List[Message]):

返回一个消息列表，列表元素为`Message`对象，详见 8.1 `Message`对象的相关说明

3.1.2 加载微信主窗口当前聊天窗口UI框架更多消息

wx.LoadMoreMessage

参数:

参数名	类型	默认值	说明
interval	float	0.3	滚动间隔时间，看自己电脑卡顿程度调整，默认0.3秒

```
>>> wx.LoadMoreMessage()
```

返回值(bool):

成功加载返回True，否则返回False

3.2 获取微信主窗口新消息

获取微信主窗口新消息需要切换微信主窗口的聊天对象窗口，原理是通过获取微信会话列表中，有哪些会话头像右上方有未读新消息数字角标，通过数字角标中的数字来判断新消息数量，仅未设置消息免打扰的会话才有数字角标。

3.2.1 获取微信主窗口一个未读消息内容（需未设置消息免打扰）

wx.GetNextNewMessage

参数:

参数名	类型	默认值	说明
savepic	bool	False	是否自动保存聊天图片
savevideo	bool	False	是否自动保存聊天视频
savefile	bool	False	是否自动保存聊天文件
savevoice	bool	False	是否自动保存聊天语音转文字内容
parseurl	bool	False	是否自动获取卡片链接

示例:

```
savepic = False
savevideo = False
savefile = False
savevoice = False
parseurl = False
msgs = wx.GetNextNewMessage(savepic, savevideo, savefile, savevoice,
parseurl)
```

返回值(Dict[str, List[Message]]):

返回一个字典，key为该消息群或好友名，str格式；value为消息内容，list格式，列表元素为`Message`对象，详见 8.1`Message`对象的相关说明

3.2.2 获取微信主窗口所有未读消息内容（需未设置消息免打扰）

wx.GetAllNewMessage

参数:

参数名	类型	默认值	说明
savepic	bool	False	是否自动保存聊天图片
savevideo	bool	False	是否自动保存聊天视频
savefile	bool	False	是否自动保存聊天文件
savevoice	bool	False	是否自动保存聊天语音转文字内容
parseurl	bool	False	是否自动获取卡片链接

示例:

```
savepic = False
savevideo = False
savefile = False
savevoice = False
parseurl = False
msgs = wx.GetAllNewMessage(savepic, savevideo, savefile, savevoice, parseurl)
```

返回值(Dict[str, List[Message]]):

返回一个字典，key为该消息群或好友名，str格式；value为消息内容，list格式，列表元素为`Message`对象，详见 8.1`Message`对象的相关说明

3.3 监听模式获取消息（微信子窗口模式，无需消息免打扰，效率最高）

监听模式指的是将指定聊天窗口独立出去，是为了避免微信主窗口因为切换聊天页面导致目标窗口中聊天消息的UI元素消失，从而失去判断哪些是历史消息哪些是新消息的依据。这种方式对新消息判断的准确率高、获取效率也高，缺点是内存占用大、独立窗口有数量限制

3.3.1 添加监听对象

wx.AddListenChat

参数:

参数名	类型	默认值	说明
who	str	/	要监听的群名或好友名
savepic	bool	False	是否自动保存聊天图片
savevideo	bool	False	是否自动保存聊天视频
savefile	bool	False	是否自动保存聊天文件
savevoice	bool	False	是否自动保存聊天语音转文字内容
parseurl	bool	False	是否自动获取卡片链接
exact	bool	False	who参数是否精确匹配，默认False

示例：

```
who = '工作群'
savepic = False
savevideo = False
savefile = False
savevoice = False
parseurl = False
wx.AddListenChat(who, savepic, savevideo, savefile, savevoice, parseurl)
```

返回值：无

3.3.2 获取监听消息

wx.GetListenMessage

参数：

参数名	类型	默认值	说明
who	str	None	已监听的群名或好友名，不填则获取所有监听对象的新消息

```
>>> msgs = wx.GetListenMessage()
```

返回值(Dict[ChatWnd, List[Message]])：

返回一个字典，key为一个微信子窗口 `ChatWnd` 对象，详见 8.2 `ChatWnd` 对象相关说明；value为消息内容，list格式，列表元素为 `Message` 对象，详见 8.1 `Message` 对象的相关说明

3.3.3 移除监听对象

wx.RemoveListenChat

参数：

参数名	类型	默认值	说明
who	str	None	已监听的群名或好友名，不填则获取所有监听对象的新消息

返回值：无

4. 添加好友

4.1 发送好友申请

wx.AddNewFriend

参数：

参数名	类型	默认值	说明
keywords	str	/	微信号、手机号、QQ号
addmsg	str	'你好，我是xxxx'	添加好友的消息
remark	str	None	备注名

示例：

```
>>> keywords = '13800000000'       # 微信号、手机号、QQ号
>>> addmsg = '你好，我是xxxx'       # 添加好友的消息
>>> remark = '备注名字'             # 备注名
>>> tags = ['朋友', '同事']         # 标签列表
>>> wx.AddNewFriend(keywords, addmsg=addmsg, remark=remark, tags=tags)
```

返回值(Tuple[int, str])：

返回值是一个tuple格式，两个元素，分别是状态码和消息，规则如下：

{
  0: '未知原因添加失败',
  1: '发送请求成功',
  2: '已经是好友',
  3: '已被对方拉黑',
  4: '找不到相关账号或内容',
}

> 注：微信有一定的限制，如果频繁添加好友，可能会被限制添加好友的权限，所以请谨慎使用！！！

4.2 接受好友请求

4.2.1 获取新的好友申请对象列表

wx.GetNewFriends

参数：无

示例：

```
>>> new = wx.GetNewFriends()
>>> new
[<wxauto New Friends Element at 0x1e95fced080 (张三: 你好,我是xxx群的张三)>,
<wxauto New Friends Element at 0x1e95fced081 (李四: 你好,我是xxx群的李四)>]
```

返回值(List[NewFriendsElement])：

返回值是一个list，列表元素为`NewFriendsElement`，详见 8.3 `NewFriendsElement`相关说明

4.2.2 通过好友申请对象接受好友请求

示例：

```
# new的定义见 4.2.1
# 获取第一个可接受的新好友对象
>>> new_friend1 = new[0]
>>> print(new_friend1.name)  # 获取好友申请昵称
张三
>>> print(new_friend1.msg)  # 获取好友申请信息
你好,我是xxx群的张三

# 接受好友请求，并且添加备注“备注张三”、添加标签wxauto
>>> new_friend1.Accept(remark='备注张三', tags=['wxauto'])
```

> 注：该方法接受好友请求后，并不会自动切换回聊天页面，需要配合调用 7.3 `SwitchToChat` 切换至聊天页面，否则其他有关聊天页面的方法不可使用

5. 获取通讯录相关

5.1 获取通讯录好友

5.1.1 获取粗略信息

wx.GetAllFriends

参数：

参数名	类型	默认值	说明
keywords	str	None	搜索并筛选关键词

示例：

```
>>> wx.GetAllFriends()
[{'nickname': '张三', 'remark': '张总', 'tags': None},
 {'nickname': '李四', 'remark': None, 'tags': ['同事', '初中同学']},
 {'nickname': '王五', 'remark': None, 'tags': None},
...]
```

返回值(List[Dict[str, str]])：

返回值是一个list，列表元素为dict，每个dict包含了一个好友的粗略信息

5.1.2 获取详细信息

wx.GetFriendDetails

参数：

参数名	类型	默认值	说明
n	int	None	获取前n个好友详情信息, 默认为None，获取所有好友详情信息
tag	str	'A'	从哪个标签开始获取，A-Z其中一个
timeout	int	0xFFFFF	获取好友详情信息的超时时间，单位为秒

示例：

```
>>> wx.GetFriendDetails()
[{'微信号：': 'abc123456',
  '地区：': '上海 浦东新区',
  '备注': '',
  '标签': 'wxauto',
  '共同群聊': '1个',
  '来源': '通过扫一扫添加',
  '昵称': '张三'},
{'备注': '',
 '企业': '广州融创文旅城',
 '实名': '**莹',
 '官方商城': '🎟️购滑雪票入口🎟️',
 '通知': '回复时间为工作日9点-18点',
 '会员商城': '🎁热雪值兑换雪票🎁',
 '冰箱赞滑': '👬申请冰箱主理人👬',
 '全民滑雪': '购票赢黄金会籍',
 '共同群聊': '1个',
 '昵称': '广州大冰箱'},...]
```

返回值(List[Dict[str, str]]):

返回值是一个list，列表元素为dict，每个dict包含了一个好友的详细信息

5.2 获取群列表

wx.GetAllRecentGroups

参数：

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

示例：

```python
groups = wx.GetAllRecentGroups()
# [
#     ('工作群', '500')
#     ('街坊群', '456')
#     ('八卦群', '123')
#     ...
# ]
```

返回值(List[Tuple[str, str]]):

返回值格式为list，列表元素为元组，格式为('群聊名', '人数')

6. 群管理、好友管理

6.1 群、好友通用管理

6.1.1 消息免打扰

wx.MuteNotifications

参数:

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

示例:

```python
who = '张三'
is_mute = True   # True为开启免打扰，False为关闭免打扰

wx.ChatWith(who)   # 先将微信主窗口切换到指定群聊天页面，详见 7.1`ChatWith`方法
wx.MuteNotifications(is_mute)
```

返回值：无

6.1.2 获取当前聊天信息

wx.CurrentChat

参数:

参数	类型	默认值	说明
details	bool	False	是否获取当前聊天对象详情信息，默认为False

示例:

```python
details = True

chat_details = wx.CurrentChat(details)
```

返回值(两种情况):

- details == False -> str

  当不获取详情信息的时候，返回当前聊天页面对象的名字

- details == False -> Dict

  当获取详情信息的时候，返回值为dict，例如：{'group_member_count': 264, 'chat_type': 'group', 'chat_name': '交流群'}

6.2 群管理相关

6.2.1 邀请入群

wx.AddGroupMembers

参数：

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

示例：

```
members = [
    '好友1',
    '好友2',
    '好友3'
]
group = '交流群'
wx.AddGroupMembers(group, members)
```

返回值：无

6.2.2 修改群聊名、备注、群公告、我在本群的昵称

wx.ManageGroup

参数：

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**

示例：

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

返回值(Dict[str, bool])：

返回值为一个dict，key为设置项目的名，str格式，value为该项是否设置成功，bool格式

6.2.3 获取群好友列表

wx.GetGroupMembers

参数：

参数	类型	默认值	说明
add_friend_mode	bool	False	是否开启添加好友模式，默认False

示例：

```
who = '工作群'
wx.ChatWith(who)  # 先将微信主窗口切换到指定群聊天页面，详见 7.1`ChatWith`方法
member_list = wx.GetGroupMembers()
```

返回值(两种情况)：

- add_friend_mode == False -> List[str]

  当未开启添加好友模式时，返回值的list中为str格式的群成员昵称或备注名

- add_friend_mode == True -> List[GroupMemberElement]

  当开启添加好友模式时，返回值的list中为GroupMemberElement对象，可用该对象的add_friend方法来进行添加群好友的操作，详见 8.4`GroupMemberElement`相关说明

6.3 好友管理相关

6.3.1 管理好友

wx.ManageFriend

参数：

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

示例：

```python
who = '张三'
wx.Chatwith(who)  # 先将微信主窗口切换到指定群聊天页面，详见 7.1`ChatWith`方法

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

返回值(bool):

返回值为bool格式，是否修改成功

7. 页面切换

7.1 切换微信主窗口聊天页面

wx.ChatWith

参数:

参数名	类型	默认值	说明
who	str	/	要打开的聊天框好友名或群名
exact	bool	False	who参数是否精确匹配，默认False

示例:

```python
who = '张三'
wx.ChatWith(who)
```

返回值(str):

返回值为切换后的聊天名，一般等于who变量，在不精确匹配下可能不一样

7.2 切换到通讯录页面

wx.SwitchToContact

参数：无

示例:

```python
wx.SwitchToContact()
```

返回值：无

7.3 切换到聊天页面

wx.SwitchToChat

参数：无

示例:

```python
wx.SwitchToChat()
```

返回值：无

8. 其他对象

8.1 消息对象`Message`

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，一共有五种`Message`对象，分别是系统消息`SysMessage`、时间消息`TimeMessage`、撤回消息`RecallMessage`、自己发出的消息`SelfMessage`、朋友发来的消息`FriendMessage`

我们将消息对象命名为msg，后续文档内不再重复定义：

```
msgs = wx.GetAllMessage()  # msgs为消息对象组成的list
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
```

所有消息对象都支持以下属性：

属性名	类型	说明
type	str	消息来源类型
mtype	str	消息内容类型，仅设置了保存某种类型的消息才会有，否则为空
content	str	消息内容
sender	str	发送者
info	list	原始消息信息，包含了消息的所有信息
control	uiautomation.Control	该消息的uiautomation控件
id	str	消息id

所有消息对象都支持以下方法：

- roll_into_view

  将该消息滚动至聊天窗口可见位置，便于操作

  ```
  msg.roll_into_view()
  ```

8.1.1 系统消息`SysMessage`

支持属性：

属性名	类型	说明
type	str	消息类型，固定为`sys`
content	str	消息内容
sender	str	发送者，固定为`SYS`
info	list	原始消息信息，包含了消息的所有信息
control	uiautomation.Control	该消息的uiautomation控件
id	str	消息id

```
...  # 此处省略wx对象的初始化
msgs = wx.GetAllMessage()
for msg in msgs:
    if msg.type == 'sys':
        print(f'【系统消息】{msg.content}')
```

8.1.2 时间消息`TimeMessage`

支持属性:

属性名	类型	说明
type	str	消息类型，固定为`time`
content	str	消息内容
sender	str	发送者，固定为`Time`
time	str	时间消息内容，格式为`%Y-%m-%d %H:%M`
info	list	原始消息信息，包含了消息的所有信息
control	uiautomation.Control	该消息的uiautomation控件
id	str	消息id

```
...  # 此处省略wx对象的初始化
msgs = wx.GetAllMessage()
for msg in msgs:
    if msg.type == 'time':
        print(f'【时间消息】{msg.time}')
```

8.1.3 撤回消息`RecallMessage`

支持属性:

属性名	类型	说明
type	str	消息类型，固定为`recall`
content	str	消息内容
sender	str	发送者，固定为`Recall`
info	list	原始消息信息，包含了消息的所有信息
control	uiautomation.Control	该消息的uiautomation控件
id	str	消息id

```
...  # 此处省略wx对象的初始化
msgs = wx.GetAllMessage()
for msg in msgs:
    if msg.type == 'recall':
        print(f'【撤回消息】{msg.content}')
```

8.1.4 自己发出的消息`SelfMessage`

支持属性：

属性名	类型	说明
type	str	消息类型，固定为`self`
mtype	str	消息内容类型，text、image、video、file、voice、card
content	str	消息内容
sender	str	发送者
info	list	原始消息信息，包含了消息的所有信息
control	uiautomation.Control	该消息的uiautomation控件
id	str	消息id

```
...  # 此处省略wx对象的初始化
msgs = wx.GetAllMessage()
for msg in msgs:
    if msg.type == 'self':
        print(f'{msg.sender}：{msg.content}')
```

支持方法

- 引用消息`quote`方法

  参数：

参数	类型	默认值	说明
msg	str	/	要回复的消息内容

  示例：

```
text = '补充说明一下：xxxx'
msg.quote(text)
```

返回值：无

- 转发消息`forward`方法

  参数：

参数	类型	默认值	说明
friend	str|list	/	要转发给谁，昵称、备注、微信号都可以，手机号不可以，**`list`类型可以多选转发对象**

  示例：

```
# 单个转发对象
friend = '好友昵称'
msg.forward(friend)

# 多个转发对象
friend = [
    '好友A',
    '好友B',
    '好友C',
    ...
]
msg.forward(friend)
```

  返回值(bool)：

  返回值为bool格式，是否转发成功

  如果为多个转发对象，转发成功返回True，失败则返回失败列表

- 解析合并消息`parse`方法

  参数：无

  示例：

```
# 解析合并消息内容，可以用msg.content == '[聊天记录]' 来判断是否为合并消息
if msg.content == '[聊天记录]':
    parse_result = msg.parse()
```

  返回值(List[List[str]])：

  返回值为list，list中元素为包含消息详情的list，消息详情有三个：消息发送者昵称、消息内容、时间

- 拍一拍`tickle`方法

  参数：无

  示例：

```python
# 拍一拍自己
msg.tickle()
```

返回值：无

- 点击`click`方法

  参数：无

  示例：

```python
# 点击这个消息（图片、链接等）
msg.click()
```

8.1.5 朋友发来的消息`FriendMessage`

支持属性：

属性名	类型	说明
type	str	消息类型，固定为`friend`
mtype	str	消息内容类型，text、image、video、file、voice、card
content	str	消息内容
sender	str	发送者
sender_remark	str	发送者备注名
info	list	原始消息信息，包含了消息的所有信息
control	uiautomation.Control	该消息的uiautomation控件
id	str	消息id

```python
...  # 此处省略wx对象的初始化
msgs = wx.GetAllMessage()
for msg in msgs:
    if msg.type == 'friend':
        sender = msg.sender # 这里可以将msg.sender改为msg.sender_remark，获取备注名
        print(f'{sender}：{msg.content}')
```

支持方法

- 引用消息`quote`方法

  参数：

参数	类型	默认值	说明
msg	str	/	要回复的消息内容

  示例：

```
text = '补充说明一下：xxxx'
msg.quote(text)
```

返回值：无

- 转发消息`forward`方法

  参数：

参数	类型	默认值	说明
friend	str	/	要转发给谁，昵称、备注、微信号都可以，手机号不可以

  示例：

```
friend = '好友昵称'
msg.forward(friend)
```

  返回值(bool)：

  返回值为bool格式，是否转发成功

- 解析合并消息`parse`方法

  参数：无

  示例：

```
# 解析合并消息内容，可以用msg.content == '[聊天记录]' 来判断是否为合并消息
if msg.content == '[聊天记录]':
    parse_result = msg.parse()
```

  返回值(List[List[str]])：

  返回值为list，list中元素为包含消息详情的list，消息详情有三个：消息发送者昵称、消息内容、时间

- 获取发送者信息`sender_info`方法

  参数：无

  示例：

```
if msg.type == 'friend':
    info = msg.sender_info()
```

  返回值(Dict[str, str])：

  返回值为一个dict，包含了发送人的基本信息，例如：

```
{'nickname': '张三',
'id': 'abc1234',
'remark': None,
'tags': 'wxauto',
'source': '通过扫一扫添加',
'signature': '个性签名',
'area': '上海',
'same_group': '1个'}
```

- 添加发送人为好友`add_friend`方法

  参数：

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

  示例：

  ```
  addmsg = '你好，我是xxxx'       # 添加好友的消息
  remark = '备注名字'             # 备注名
  tags = ['朋友', '同事']         # 标签列表
  msg.add_friend(addmsg=addmsg, remark=remark, tags=tags)
  ```

  返回值(int)：

  0 - 添加失败

  1 - 发送请求成功

  2 - 已经是好友

  3 - 对方不允许通过群聊添加好友

- 拍一拍`tickle`方法

  参数：无

  示例：

  ```
  # 拍一拍发这条消息的好友（群友）
  msg.tickle()
  ```

  返回值：无

- 点击`click`方法

  参数：无

  示例：

  ```
  # 点击这个消息（图片、链接等）
  msg.click()
  ```

8.2 微信子窗口对象`ChatWnd`

微信子窗口对象指的是从微信主窗口独立出来的聊天窗口，主要用于监听模式下对聊天窗口的各种操作。

支持属性：

参数	类型	说明
who	str	该子窗口对应的好友/群名
UiaAPI	str	该子窗口的uiautomation控件
editbox	str	该子窗口中消息编辑框的uiautomation控件
savepic	str	用于设置监听模式时，是否自动下载图片
savevideo	bool	False
savefile	str	用于设置监听模式时，是否自动下载文件
savevoice	str	用于设置监听模式时，是否自动获取语音转文字
parseurl	str	用于设置监听模式时，是否自动解析卡片链接

我们将微信子窗口对象命名为chat，后续文档内不再重复定义：

```
who = '张三'
savepic = True
savefile = False
savevoice = True
parseurl = False
wx.AddListenChat(
    who
    savepic=savepic,
    savefile=savefile,
    savevoice=savevoice,
    parseurl=parseurl
)   # 该方法会将“张三”的聊天窗口加入监听，并把他的聊天窗口独立出来，在wx.listen这个dict中记录

wx.listen
# {'张三': <wxauto Chat Window at 0xbac39c32 for 张三>}

chat = wx.listen[who]  # 后续文档内不再重复定义chat
```

8.2.1 加载该子窗口当前UI框架更多消息

chat.LoadMoreMessage

参数：

参数名	类型	默认值	说明
interval	float	0.3	滚动间隔时间，看自己电脑卡顿程度调整，默认0.3秒

```
>>> chat.LoadMoreMessage()
```

返回值(bool):

成功加载返回True，否则返回False

8.2.2 获该子窗口当前UI框架已加载所有聊天记录

chat.GetAllMessage

参数：

参数名	类型	默认值	说明
savepic	bool	False	是否自动保存聊天图片
savevideo	bool	False	是否自动保存聊天视频
savefile	bool	False	是否自动保存聊天文件
savevoice	bool	False	是否自动保存聊天语音转文字内容
parseurl	bool	False	是否自动获取卡片链接

示例：

```
savepic = False
savefile = False
savevoice = False
parseurl = False
msgs = chat.GetAllMessage(savepic, savefile, savevoice, parseurl)
```

返回值(List[Message])：

返回一个消息列表，列表元素为`Message`对象，详见8.1 `Message`对象的相关说明

8.2.3 发送文字消息（普通模式）

chat.SendMsg

方法说明：

给指定人员（群组）发送消息

参数：

参数名	类型	默认值	说明
msg	str	/	要发送的文字内容
clear	bool	True	是否清除原本聊天编辑框的内容
at	list,str	None	要@的人，可以是一个人或多个人，格式为str或list，例如："张三"或["张三", "李四"]

返回值：

如果发送成功则返回True，否则为None

示例：

```
msg = '''这是一条消息
这是第二行
这是第三行
'''
chat.SendMsg(msg)
```

8.2.4 发送文字消息（打字机模式）

chat.SendTypingText

参数：

参数名	类型	默认值	说明
msg	str	/	要发送的文字内容
clear	bool	True	是否清除原本聊天编辑框的内容
at	list,str	None	要@的人，可以是一个人或多个人，格式为str或list，例如："张三"或["张三", "李四"]

返回值：无

示例：

```
# 发送消息
msg = '''这是一条消息
这是第二行
这是第三行'''
chat.SendTypingText(msg)

# 换行及@功能
msg = '''各位下午好
{@张三}负责xxx
{@李四}负责xxxx'''
chat.SendTypingText(msg)
```

8.2.5 发送文件、图片、视频消息

chat.SendFiles

参数：

参数名	类型	默认值	说明
filepath	str | list	/	指定文件路径，单个文件str，多个文件list

返回值(bool)：

如果发送成功则返回True，否则为False

示例：

```
# 指定文件路径（绝对路径）
files = ['D:/test/test1.txt', 'D:/test/test2.txt', 'D:/test/test3.txt']
chat.SendFiles(files)
```

8.2.6 发送@所有人消息

chat.AtAll

参数：

参数名	类型	默认值	说明
msg	str	None	要发送的文字内容，可为空

返回值：无

示例：

```
msg = '通知：XXXXXXXXXX'
chat.AtAll(msg=msg)
```

8.2.7 发送自定义表情包

chat.SendEmotion

参数：

参数名	类型	默认值	说明
emotion_index	int	/	表情索引，从0开始计数

示例：

```
emotion_index = 25   # 要发送第25个自定义表情
chat.SendEmotion(emotion_index)
```

返回值(bool)：

如果发送成功则返回True，否则为False

8.2.8 获取该子窗口信息

chat.ChatInfo

参数：无

示例：

```
info = chat.ChatInfo()
```

返回值(Dict)：

返回值为dict，该子窗口的信息，例如：{'chat_type': 'group', 'chat_name': '交流群', 'group_member_count': 477}

8.3 接收到的新好友请求对象`NewFriendsElement`

支持属性：

参数	类型	说明
ele	uia.Control	该对象对应的uiautomation控件
name	str	对方的昵称
msg	str	对方发来的申请信息

```
new_friend_list = wx.GetNewFriends()
# 获取一个新好友请求对象
new: NewFriendsElement = new_friend_list[0]  # 假设有新好友请求的情况
```

8.3.1 接受好友请求

new.Accept

参数：

参数名	类型	默认值	说明
remark	str	None	备注名
tags	list	None	标签列表
permission	str	'朋友圈'	朋友圈权限, 可选值：'朋友圈', '仅聊天'

示例：

```
remark = '同事张三'
tags = ['同事']
new.Accept(remark=remark, tags=tags)
```

返回值：无

> 注：接受好友请求后并不会自动切换到聊天页面，如果要进行聊天页面的操作请主动调用7.3`SwitchToChat`方法切换至聊天页面

8.3.2 获取请求人微信号（仅接受好友后可用，否则返回None）

new.GetAccount

参数：

参数名	类型	默认值	说明
wait	float	5.0	等待时间

示例：

```
account = new.GetAccount(10)  # 10秒内获取不到则结束
```

返回值(str)：

如果获取到，则为str类型的微信号信息，如果未获取到，则为None

8.4 群成员对象`GroupMemberElement`

支持属性：

参数	类型	说明
UiaAPI	uia.Control	该对象对应的uiautomation控件
nickname	str	对方的昵称

```
who = '工作群'
add_friend_mode = True
wx.ChatWith(who)  # 先将微信主窗口切换到指定群聊天页面，详见 7.1`ChatWith`方法
member_list = wx.GetGroupMembers(add_friend_mode)
group_member = member_list[0]  # 取第一个为群成员对象案例
```

8.4.1 向该成员发起好友申请

group_member.add_friend

参数：

参数名	类型	默认值	说明
addmsg	str	None	发起好友申请的消息
remark	str	None	备注名
tags	list	None	标签列表
permission	str	'朋友圈'	朋友圈权限, 可选值：'朋友圈', '仅聊天'

示例：

```
addmsg = '你好，我是xxxx'       # 添加好友的消息
remark = '备注名字'             # 备注名
tags = ['朋友', '同事']         # 标签列表
group_member.add_friend(addmsg=addmsg, remark=remark, tags=tags)
```

返回值(int)：

```
0 - 添加失败
1 - 发送请求成功
2 - 已经是好友
3 - 对方不允许通过群聊添加好友
```

8.6 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

参数：

参数名	默认值	说明
app_path	None	微信客户端路径，如果无法启动，则需要手动传入该参数

```python
from wxautox.elements import LoginWnd

app_path = "D:/App/wechat/wechat.exe"
wxlogin = LoginWnd(app_path)
```

8.6.1 获取登录二维码

```python
from wxautox.elements import LoginWnd

# 当客户端掉线时

## 1. 获取二维码
wxlogin = LoginWnd()
if wxlogin.UiaAPI.Exists(3):
    wxlogin.reopen()    # 重新打开微信登录窗口，这里是为了确保没有其他弹窗影响操作
    qrcode = wxlogin.get_qrcode()    # 保存并返回二维码图片地址
    ...  # 拿到的二维码自行处理
```

8.6.2 自动登录

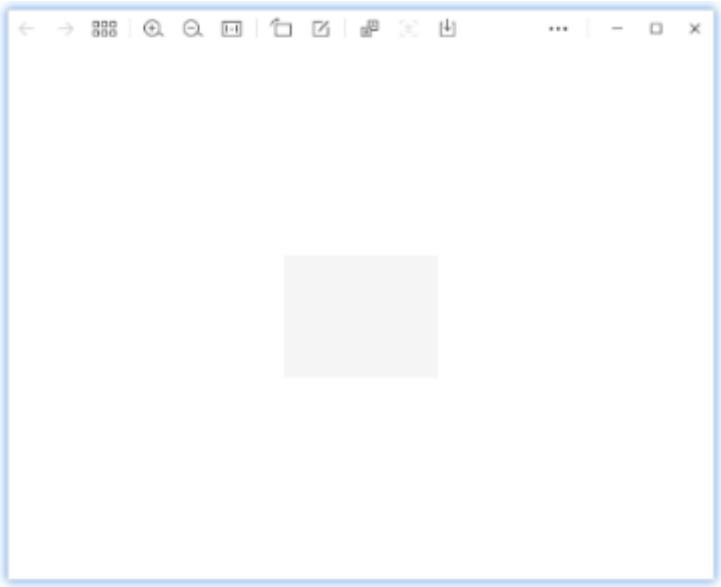
```python
## 2. 自动登录（如果是被踢下线则需要重新扫码登录）
wxlogin = LoginWnd()
if wxlogin.UiaAPI.Exists(3):
    wxlogin.reopen()    # 重新打开微信登录窗口，这里是为了确保没有其他弹窗影响操作
    login_result = wxlogin.login()    # 当login_result为True时，登录成功，None则登录失败需要扫码
```

8.7 微信图片/视频窗口对象

微信图片/视频窗口对象指的是点击图片或视频后，打开的窗口。

参数:

参数名	默认值	说明
video	False	默认为图片窗口，如果是视频窗口则需指定为True



```python
from wxautox.elements import WeChatImage
# 获取该实例前需要打开这个窗口
... # 假设你已经拿到一个msg对象
msg.click()  # 点击图片/视频打开这个窗口

if msg.mtype == 'image' or msg.content == '[图片]':
    image = WeChatImage()
elif msg.mtype == 'video' or msg.content == '[视频]':
    image = WeChatImage(video=True)

```

8.7.1 保存

image.Save

下载图片/视频

参数:

参数名	类型	默认值	说明
savepath	str	''	绝对路径，包括文件名和后缀，不填则自动生成
timeout	int	None	保存超时时间，默认10秒

返回值（str）：

保存的路径

8.7.2 文字识别

image.OCR

调用微信的图片文字识别功能

```python
text = image.OCR()
print(text)
```

返回值（str）：

文字识别结果

8.7.3 上一张

image.Previous

切换到上一张图片

```python
image.Previous()
```

返回值（bool）：

成功True，失败False

8.7.4 下一张

image.Next

切换到下一张图片

```
image.Next()
```

返回值（bool）：

成功True，失败False

8.7.5 关闭窗口

image.Close

```
image.Close()
```

返回值：

无

三、朋友圈相关

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

```
from wxautox import WeChat

wx = WeChat()
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

```python
# 获取当前页面的朋友圈内容
moments = pyq.GetMoments()

# 通过`next_page`参数获取下一页的朋友圈内容
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

```python
# 刷新朋友圈
pyq.Refresh()
```

1.3 关闭朋友圈

```python
# 关闭朋友圈
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。

```python
# 获取朋友圈对象
moments = pyq.GetMoments()

# 获取第一条朋友圈
moment = moments[0]
```

2.1 获取朋友圈内容

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

```
#     '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
#     'time': '4分钟前',              # 发送时间
#     'img_count': 3,                 # 图片数量
#     'comments': [],                 # 评论
#     'addr': '',                     # 发送位置
#     'likes': []                     # 点赞
# }

moment.sender
# '天天鲜花2号客服'

moment.content
# '客订花束'

moment.time
# '4分钟前'

# 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
...
```

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

```
# 获取朋友圈图片
images = moment.SaveImages()
# [
#     'D:/Images/微信图片_xxxxxx1.jpg',
#     'D:/Images/微信图片_xxxxxx2.jpg',
#     'D:/Images/微信图片_xxxxxx3.jpg',
#     ...
# ]
```

2.3 获取好友信息

```python
# 获取好友信息
moment.sender_info()
# {
#     'nickname': None,
#     'id': 'xxxxxxx',
#     'remark': None,
#     'tags': None,
#     'source': '通过搜索手机号添加',
#     'signature': None
# }
```

四、常见问题

1. 不同获取消息的方法有什么区别

wxauto中，有以下获取消息的方法，除GetAllMessage之外，其余方法均用于获取新消息

方法	说明
GetAllMessage	获取当前聊天页面中**已加载**的消息
GetAllNewMessage	获取微信主窗口中，所有**未设置消息免打扰**的新消息
GetNextNewMessage	获取微信主窗口中，其中一个**未设置消息免打扰**窗口的新消息
GetListenMessage	获取**监听模式**下聊天窗口的新消息

监听模式

GetListenMessage

调用AddListenMessage方法将目标聊天窗口独立出去加入监听列表，获取新消息

优点

- 不遗漏消息
- 读取速度快

缺点

- 数量限制，最多设置30个监听对象

全局模式

GetAllNewMessage、GetNextNewMessage

获取所有微信主窗口中，未被设置为消息免打扰的窗口中的新消息

优点

- 没有数量限制，无差别获取所有窗口新消息

缺点

- 有概率漏消息

该方法原理是获取会话列表中，聊天对象头像上的未读消息角标数字来判断新消息数，正常情况下按黑色流程读取未读消息，如果在程序读取完角标之后，读取完新消息之前，对方又发过来几条消息，就会漏掉相应数量的消息

- 必须进行UI操作，速度可能相较监听模式慢些

2. 为什么会掉线

掉线是微信3.9.9及以后的版本中加入的机制，客户端频繁操作导致的

plus版本会掉线吗

会，手动操作频繁也会掉线，是微信客户端的机制

如何规避

- 加延迟时间
- 用3.9.8版本客户端
- plus版本提供自动登录、获取二维码操作

3. 关闭远程连接后就报错了怎么办

因为windows默认的远程工具在断开连接时会锁定桌面，而该项目依赖UIAutomation接口，在锁屏时无法正常运行所以报错

如何规避

- 利用bat脚本使断开时不再锁屏

```
for /f "skip=1 tokens=3" %%s in ('query user %USERNAME%') do
(%windir%\System32\tscon.exe %%s /dest:console)
```

- 其他远程方式

  有很多远程方案，这里推荐使用VNC：

  1. 登录服务器，安装RealVNC server，设置登录密码和服务端口，默认为5900端口，但是最好改一下
  2. 登录云服务器服务商控制台，在防火墙打开上一步设置的vnc端口的访问权限
  3. 自己电脑安装RealVNC viewer，输入云服务器ip:端口进行连接，断开连接不会锁屏
  4. 服务器安装微信，部署代码即可运行

> 注：
>
> 1. 远程方案有很多，没有必须是VNC，只要不锁屏用什么都可以，向日葵也可以。
> 2. 云服务器最好买你所在城市或临近城市的，因为有几率触发微信异地登录风控。

4. 打包exe

```
Pyinstaller -F .\app.py --hidden-import comtypes.stream
```

5.支持Linux吗

不支持，基于windows官方API开发，只支持windows系统

6. 为什么安装成功但是无法导入

检查下安装wxautox的环境与你运行环境是否同一个python环境。

PyCharm默认会给你的项目创建一个虚拟环境，需要在虚拟环境中安装才可以调用

> 如果不清楚如何使用虚拟环境安装，可问AI ”怎么用pycharm的虚拟环境安装本地离线whl包“