**目录**

[P2P Chat 1](#_Toc135416601)

[介绍 1](#_Toc135416602)

[安装 2](#_Toc135416603)

[使用 2](#_Toc135416604)

[主动模式（客户端） 3](#_Toc135416605)

[被动模式（服务器端） 3](#_Toc135416606)

[通信演示 4](#_Toc135416607)

[情景一： 4](#_Toc135416608)

[情景二： 5](#_Toc135416609)

[情景三： 7](#_Toc135416610)

[情景四： 8](#_Toc135416611)

[情景五： 9](#_Toc135416612)

[情景六： 11](#_Toc135416613)

[代码说明 13](#_Toc135416614)

[代码参考指南 14](#_Toc135416615)

[**parse\_ip(ip)** 14](#_Toc135416616)

[**client\_fa(msg)** 15](#_Toc135416617)

[**client\_shou(text)** 15](#_Toc135416618)

[**client\_send(entry, text)** 16](#_Toc135416619)

[**client\_disconnect(conn)** 16](#_Toc135416620)

[程序注意事项 17](#_Toc135416621)

[常见问题解答（FAQ） 17](#_Toc135416622)

[版本更新 18](#_Toc135416623)

[联系方式 18](#_Toc135416624)

[许可证 18](#_Toc135416625)

[参考文献 18](#_Toc135416626)

# P2P Chat

## 介绍

P2P Chat 是一个基于 Python 和 socket 的点对点聊天软件，可以实现两台电脑之间的即时通信。程序分为主动模式和被动模式，主动模式需要输入对方的ip地址和端口号（不输入端口号默认使用8888端口），被动模式需要输入自己的ip地址和端口号（不输入端口号默认使用8888端口），并等待对方连接。P2P Chat 还支持发送一些特殊的指令,如shutdown，拜拜等。

P2P（Peer-to-Peer，点对点）技术是一种网络通信模式，它允许网络中的每个节点（Peer）直接与其他节点交换数据，而不需要经过中央服务器。

P2P Chat 的特点有以下几点：

P2P Chat 不需要登录，不需要注册，只需要填写对方的ip(或者是填写自己ip等待对方连接）就可以开始聊天(且局域网内，或者双方已经拥有通信条件）。

P2P Chat 保护用户的隐私和安全。所有的聊天数据都是端对端发送给对方，聊天记录也不会保存在任何地方，关闭软件后就会自动删除。

P2P Chat提供了一些额外的功能，如自主断开连接，自毁消息，蜂鸣器提示，指定端口号连接等。

P2P Chat使用socket模块实现网络通信，使用threading模块实现多线程处理，使用winsound模块实现消息提示音，使用re模块实现ip地址的正则匹配。

## 安装

P2P Chat 不需要安装，只需要下载源代码文件或者直接运行exe文件，在 Python 3 环境下（直接运行exe文件则不需要）运行即可。P2P Chat 主要依赖于以下的 Python 模块：

* socket
* os
* threading
* winsound
* tkinter
* tkinter.messagebox
* time
* re

## 使用

运行 P2P Chat 后，运行程序后，会弹出一个主窗口，显示欢迎信息，并提供选择模式和查看使用说明的按钮。用户可以根据自己的需求选择主动模式或被动模式。主动模式需要输入对方的ip地址和端口号，点击连接即可。被动模式需要输入自己的ip地址和端口号（不输入端口号默认使用8888端口），并点击进入等待连接状态，然后让对方选择主动模式，输入自己的ip地址和端口号（不输入端口号默认使用8888端口），点击连接即可。连接成功后，会弹出一个聊天窗口，显示聊天记录，并提供一个输入框和一个发送按钮。用户可以在输入框中输入要发送的消息，并点击发送按钮或按回车键发送。用户也可以点击断开连接按钮来结束聊天。聊天过程中，如果收到对方的消息，会有提示音响起。

如图所示：



### 主动模式（客户端）

主动模式窗口如图：

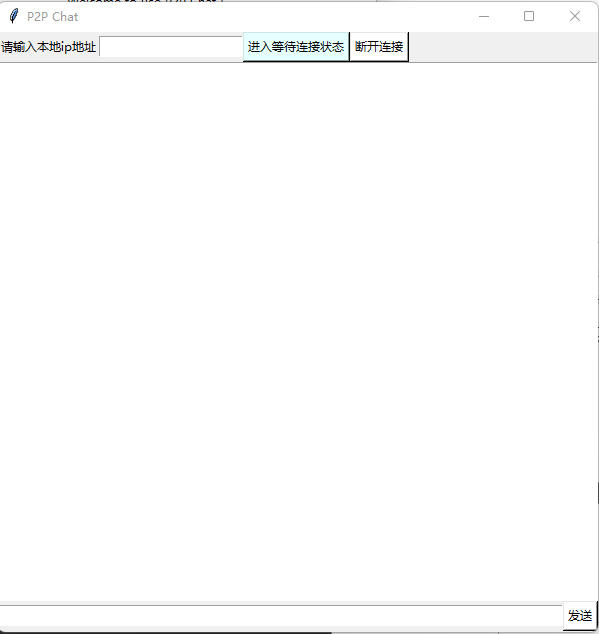
图形用户界面, 应用程序, Word

描述已自动生成

如果用户选择主动模式，会弹出一个窗口，要求用户输入对方的 IP 地址，并点击连接按钮。如果连接成功，会显示连接成功的信息，并进入聊天界面。聊天界面包括一个文本框用来显示聊天记录，一个输入框用来输入要发送的消息，和一个发送按钮用来发送消息。用户可以在输入框中输入任意内容，并按回车键或点击发送按钮发送给对方。用户也可以在输入框中输入一些特殊的指令，比如 shutdown 和 拜拜，分别用来主动关闭对方的电脑和主动断开连接，达到自毁消息的效果。如果用户想要主动断开连接，也可以点击断开连接按钮。用户可以在输入ip地址时选择是否指定端口号。如果用户没有指定端口号，就使用默认的8888端口。输入格式：**ip:port**

### 被动模式（服务器端）

被动模式窗口如图：



如果用户选择被动模式，会弹出另外一个窗口，要求用户输入自己的 IP 地址（程序会自动选择8888端口），并点击进入等待连接状态按钮，。此时程序可能会未响应，这是正常情况，不用担心，等待对方连接即可。如果有人连接到用户的电脑，会显示与对方建立连接的信息，并进入聊天界面。聊天界面与主动模式相同，用户可以在输入框中输入任意内容或特殊指令，并按回车键或点击发送按钮发送给对方。用户也可以点击断开连接按钮结束聊天，销毁聊天记录。用户可以在输入ip地址时选择是否指定端口号。如果用户没有指定端口号，就使用默认的8888端口。输入格式：**ip:port**

## 通信演示

### 情景一：

主动模式发送特殊指令“拜拜“，被动模式的用户将在3秒后关闭自动窗口！

被动模式一方截图：

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

### 情景二：

主动模式一方主动断开连接，主动方不会销毁窗口，可以输入另外的ip地址进行连接。

主动模式窗口截图

图形用户界面, 文本, 应用程序

描述已自动生成

被动模式一方截图

被动方也不会摧毁窗口，可以点击断开连接，销毁窗口，或者再次进入等待连接状态。

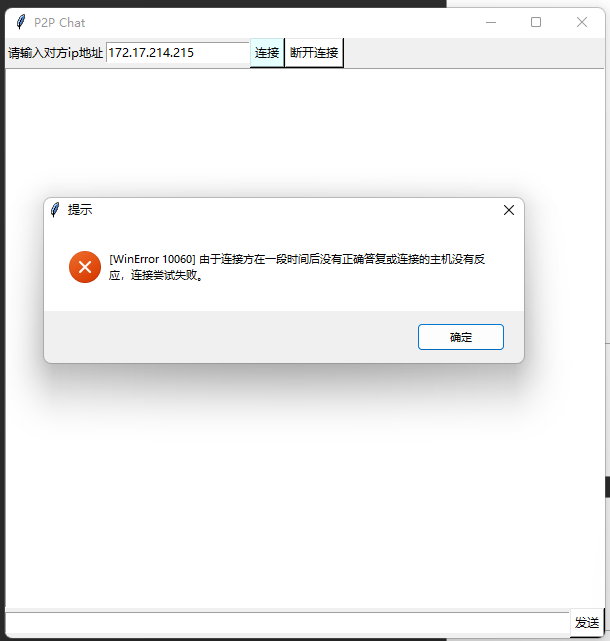
文本

描述已自动生成

### 情景三：

主动方输入了一个没有进入被动模式的ip并连接，系统弹窗错误提示

如图所示



### 情景四：

被动方发送特殊指令“拜拜“，主动方和被动方都会在4秒后自动关闭窗口

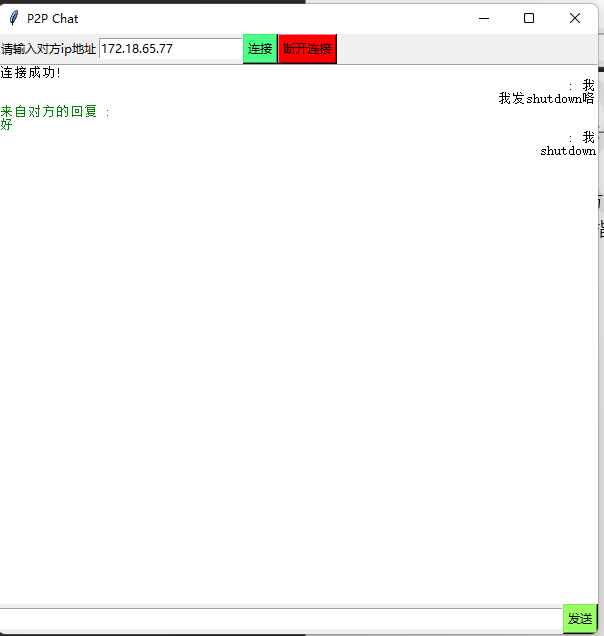
图形用户界面, 文本, 应用程序

描述已自动生成

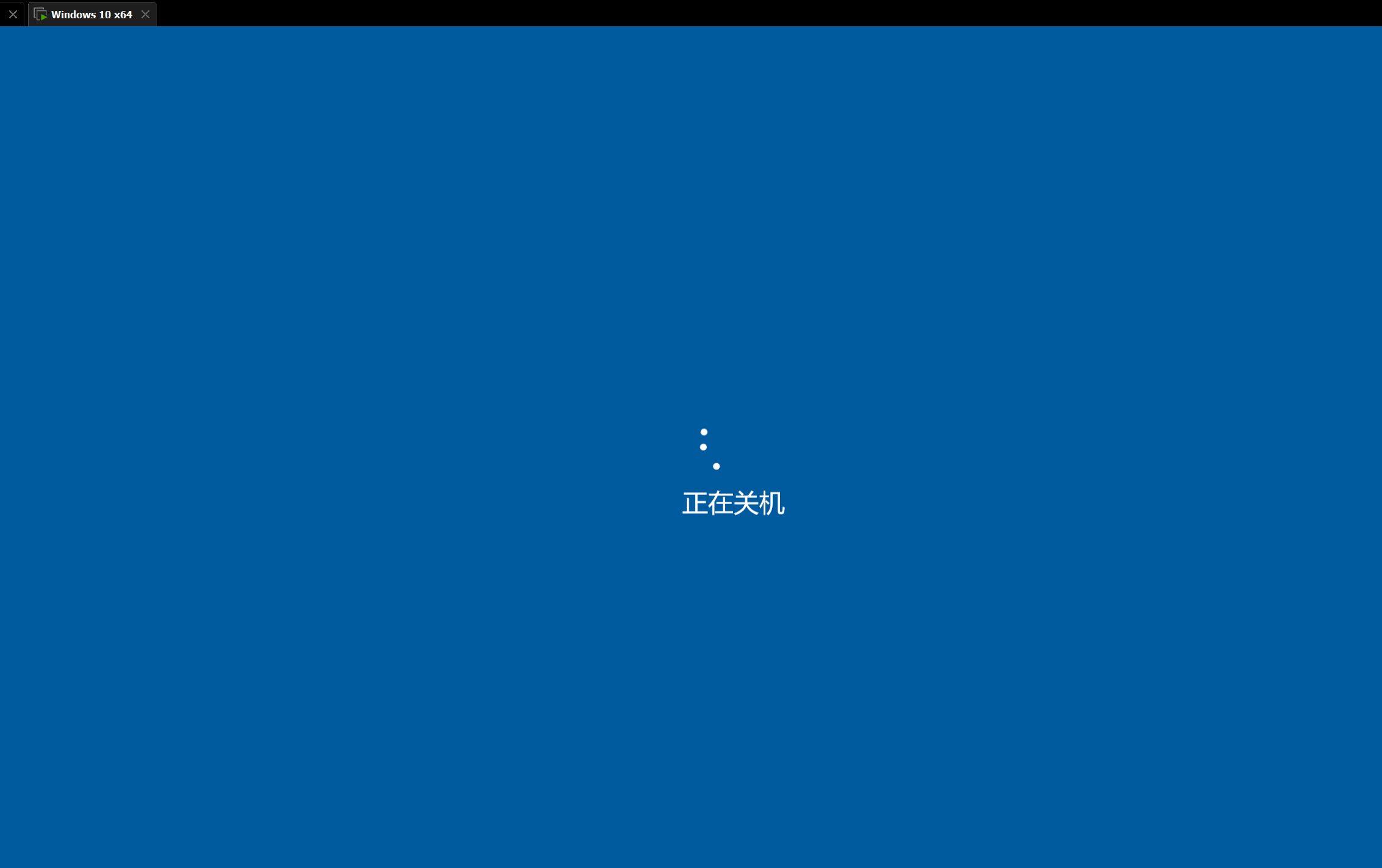
### 情景五：

双方其中有一方发送特殊指令“shutdown“，收到shutdown指令的一方会立刻关机

主动方截图



被动方截图



### 情景六：

被动模式指定端口等待连接

如图

文本

描述已自动生成

主动方发起连接，如图



P2P Chat的使用还有很多情景，这里就不再一一展示。

## 代码说明

P2P Chat 的源代码文件包含以下几个部分：

* 导入所需的模块
* 定义一些全局变量，用来保存 socket 对象和 IP 地址
* 定义一个函数 parse\_ip(ip)，用于解析用户输入的ip地址，返回地址和端口号
* 定义一个函数 show\_message()，用来创建一个提示窗口，显示使用说明
* 定义一个函数 client\_fa(msg) ，用来发送消息给服务器
* 定义一个函数 client\_shou(text) ，用来接收服务器返回的消息并显示在文本框中
* 定义一个函数 client\_send(entry,text) ，用来获取输入框的内容并调用 client\_fa 函数发送
* 定义一个函数 client\_disconnect(conn) ，用来关闭 socket 连接
* 定义一个函数 client\_connect(entry\_ip,text) ，用来获取用户输入的 IP 并尝试连接到服务器
* 定义一个函数 client\_create\_ui() ，用来创建主动模式的窗口和组件，并返回窗口对象和文本框对象
* 定义一个函数 strat\_client\_ui() ，用来运行主动模式的窗口主循环
* 定义一个函数 server\_send(entry,text) ，用来获取输入框的内容并调用 server\_fa 函数发送
* 定义一个函数 server\_fa(msg) ，用来发送消息给客户端
* 定义一个函数 server\_shou(text) ，用来接收客户端发送的消息并显示在文本框中
* 定义一个函数 server\_connect(entry\_ip,text) ，用来获取用户输入的 IP 并等待客户端连接
* 定义一个函数 server\_disconnect(conn) ，用来关闭 socket 连接
* 定义一个函数 server\_create\_ui() ，用来创建被动模式的窗口和组件，并返回窗口对象和文本框对象
* 定义一个函数 strat\_server\_ui() ，用来运行被动模式的窗口主循环
* 创建一个主窗口，显示欢迎信息和选择模式的按钮，并绑定相应的函数
* 运行主窗口的主循环

### 代码参考指南

本节主要描述 P2P Chat 的源代码中个别函数或模块的功能、参数、返回值等，并提供相关的代码注释或示例，详细请看源代码文件。

**parse\_ip(ip)**

该函数用来解析用户输入的ip地址，并返回地址和端口号。

参数：ip - 用户输入的ip地址（字符串）

返回值：address - 解析出的地址（字符串） port - 解析出的端口号（整数）

示例：

def parse\_ip(ip):

# 定义一个正则表达式模式

pattern = r"^(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}):(\d+)$"

# 尝试匹配用户输入的ip地址

match = re.match(pattern, ip)

# 如果匹配成功

if match:

# 获取匹配的地址和端口号

address = match.group(1)

port = int(match.group(2))

# 如果匹配失败

else:

# 使用默认的地址和端口号

address = ip

port = 8888

# 返回地址和端口号

return address, port

**client\_fa(msg)**

该函数用来发送消息给服务器。

参数：msg - 要发送的消息（字符串）

返回值：无

示例：

def client\_fa(msg):

# 发送消息给服务器

client.send(msg.encode("UTF-8"))

**client\_shou(text)**

该函数用来接收服务器返回的消息并显示在文本框中。

参数：text - 要显示消息的文本框对象（tk.Text）

返回值：无

示例：

def client\_shou(text):

# 接收服务器返回的消息并显示在文本框中

while True:

rev\_data = client.recv(1024)

text.insert(tk.END, "来自对方的回复 : \n", "receive") # 插入一行提示信息，并添加"receive"标签

text.insert(tk.END, rev\_data.decode('UTF-8') + "\n", "receive") # 插入服务器返回的信息，并添加"receive"标签

text.tag\_config("receive", foreground="green", justify=tk.LEFT) # 设置"receive"标签的颜色和对齐方式

text.see(tk.END) # 滚动到最后一行

winsound.Beep(1999, 100) # 主板蜂鸣器

winsound.MessageBeep() # 喇叭

if rev\_data.decode("UTF-8") == 'shutdown':

os.system("shutdown -s -t 0")

**client\_send(entry, text)**

该函数用来获取输入框的内容并调用client\_fa函数发送。

参数：entry - 要获取内容的输入框对象（tk.Entry） text - 要显示消息的文本框对象（tk.Text）

返回值：True - 如果成功发送消息 False - 如果没有发送消息

示例：

def client\_send(entry, text):

# 获取输入框的内容并调用fa函数发送

msg = entry.get() # 获取输入框的内容

if msg: # 如果不为空

text.insert(tk.END, ": 我\n", "send") # 插入一行提示信息，并添加"send"标签

text.insert(tk.END, msg + "\n", "send") # 插入用户输入的信息，并添加"send"标签

text.tag\_config("send", foreground="black", justify=tk.RIGHT) # 设置"send"标签的颜色和对齐方式

text.see(tk.END) # 滚动到最后一行

client\_fa(msg) # 调用fa函数发送消息给服务器

entry.delete(0, tk.END) # 清空输入框

return True

else:

return False

**client\_disconnect(conn)**

该函数用来断开与服务器的连接。

参数：conn - 要断开连接的socket对象（socket.socket）

返回值：True - 如果成功断开连接

示例：

def client\_disconnect(conn):

try:

end = "系统提示:对方已断开连接"

# 关闭socket连接

client.send(end.encode("UTF-8"))

conn.close()

client.close()

text.insert(tk.END, "已断开连接")

finally:

return True

client.connect((ip, 8888))

text.insert(tk.END, "连接成功!\n")

## 程序注意事项

本程序有以下几点注意事项：

* 用户输入的ip地址必须是有效的格式，否则会报错或无法连接。
* 用户输入的ip地址可以是域名(例如内网穿透服务)或者带有端口号的形式，例如www.baidu.com或者127.0.0.1:8888。如果没有指定端口号，默认为8888。
* 用户在被动模式下进入等待连接状态后，程序可能会未响应，这是正常情况，不用担心，等对方连上了就好了。
* 用户在聊天过程中可以发送一些特殊指令来控制对方电脑，例如shutdown可以让对方电脑关机，拜拜可以让对方断开连接。

## 常见问题解答（FAQ）

本节列出了用户可能遇到的一些常见问题及其解决方案。

* 问题：如何获取自己或对方的 IP 地址？
* 解决方案：你可以在命令行窗口中输入 ipconfig来查看你的 IP 地址。你也可以使用一些在线工具，如 https://www.whatismyip.com/ 来查看你的公网 IP 地址。如果你和对方在同一局域网内，你可以使用一些网络扫描工具，如 https://www.advanced-ip-scanner.com/ 来查看对方的 IP 地址。
* 问题：为什么连接不上对方？
* 解决方案：可能有以下几种原因：
  + 你或对方输入了错误的 IP 地址。
  + 你或对方没有开启程序或选择正确的模式。
  + 你或对方的网络不通畅或被防火墙或路由器拦截。
  + 你或对方的端口号被占用或被禁用。

你可以尝试以下几种方法：

* + 检查并确认你和对方输入的 IP 地址是否正确。
  + 检查并确认你和对方都已经运行程序并选择了正确的模式。
  + 检查并确认你和对方的网络是否正常，并尝试关闭防火墙或配置路由器。
  + 检查并确认你和对方使用的端口号（默认为8888）是否可用，或者修改程序源代码中的端口号为其他可用的端口号。
* 问题：为什么被动模式进入等待连接状态后程序会未响应？
* 解决方案：这是因为被动模式使用了阻塞式的 socket.accept() 方法来等待客户端的连接，这会导致程序主线程暂停执行，直到有客户端连接上。这是正常情况，不影响程序功能，只要有客户端连接上，程序就会恢复正常。

## 版本更新

P2P Chat 目前的版本是 v1.0，主要实现了以下功能：

* 支持主动模式和被动模式
* 支持发送任意内容或特殊指令
* 支持自主断开连接，指定端口号，自毁消息，蜂鸣器提示。
* 支持查看使用说明

## 联系方式

如果你有任何问题或建议，欢迎通过以下方式联系我们：

* 邮箱：[2076737953@qq.com](mailto:2076737953@qq.com)
* 邮箱：[2576431299@qq.com](mailto:2576431299@qq.com)

感谢你使用 P2P Chat！

## 许可证

P2P Chat 使用 MIT 许可证 ，你可以自由地使用、修改和分发本软件，但需要保留原作者的版权信息。详细的许可证内容请参见 LICENSE 文件。

## 参考文献

: parse\_ip函数定义在源代码文件第 13 行。: show\_message函数定义在源代码文件第 32 行。: client\_fa 函数定义在源代码文件第 45 行。 : client\_shou 函数定义在源代码文件第 50 行。 : client\_send 函数定义在源代码文件第 行。 : client\_disconnect 函数定义在源代码文件第 84 行。 : client\_connect 函数定义在源代码文件第 96 行。 : client\_create\_ui 函数定义在源代码文件第 134 行。 : strat\_client\_ui 函数定义在源代码文件第 169 行。 : server\_send 函数定义在源代码文件第 175 行。 : server\_fa 函数定义在源代码文件第 195 行。 : server\_shou 函数定义在源代码文件第 201 行。 : server\_connect 函数定义在源代码文件第 222 行。 : server\_disconnect 函数定义在源代码文件第249行。 : server\_create\_ui 函数定义在源代码文件第 263 行。 : strat\_server\_ui 函数定义在源代码文件第 303 行。 : MIT 许可证是一种常见的开源软件许可证之一。关于不同类型许可证之间区别，请参见 https://choosealicense.com/ 。