python中基于PYQT5实现聊天

组长:郑晗

组员:方炫棋，程青红，张韶箫，张登玲

PYQT5是Python的一个GUI（Graphical User Interface，图形用户界面）框架。

主要模块

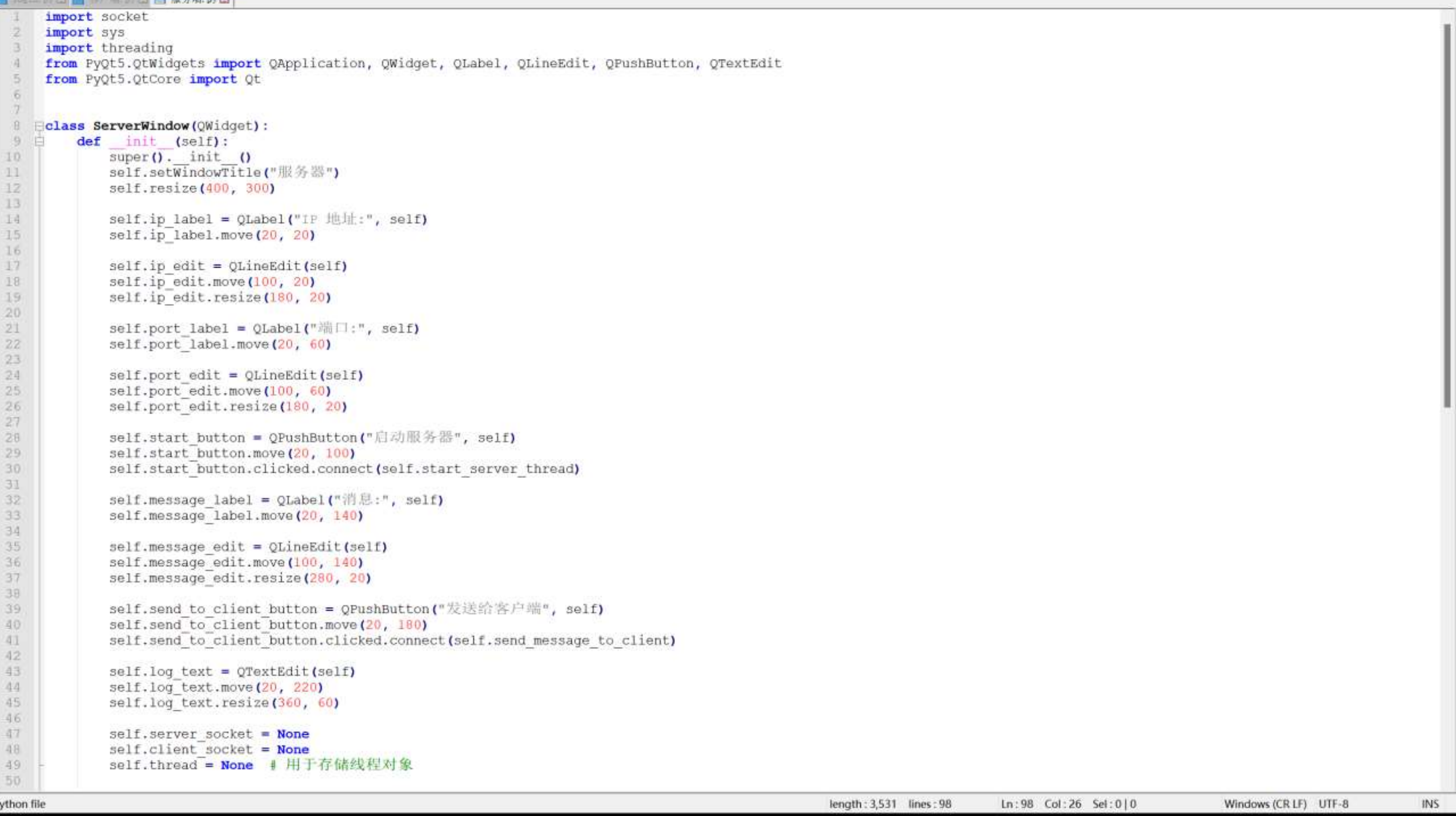
-QtCore：包含了核心的非GUI功能，如信号与槽机制、事件循环、定时器等，是整个框架的基础。
-QtGui：提供了基本的图形界面元素和绘图功能，如窗口、按钮、文本框、图形绘制等。
-QtWidgets：是QtGui的扩展，提供了更丰富的标准GUI组件，如各种布局管理器、菜单、工具栏等。
-QtNetwork：用于网络编程，支持TCP/IP、UDP等协议，方便实现网络通信功能。
-QtSql：提供了与数据库交互的功能，支持多种数据库系统，如SQLite、MySQL、Oracle等。

应用场景

- 桌面应用开发：可用于开发各种类型的桌面应用程序，如文本编辑器、图像查看器、音乐播放器等
- 系统工具开发：用于开发系统管理工具、监控工具等，方便系统管理员对系统进行管理和维护。

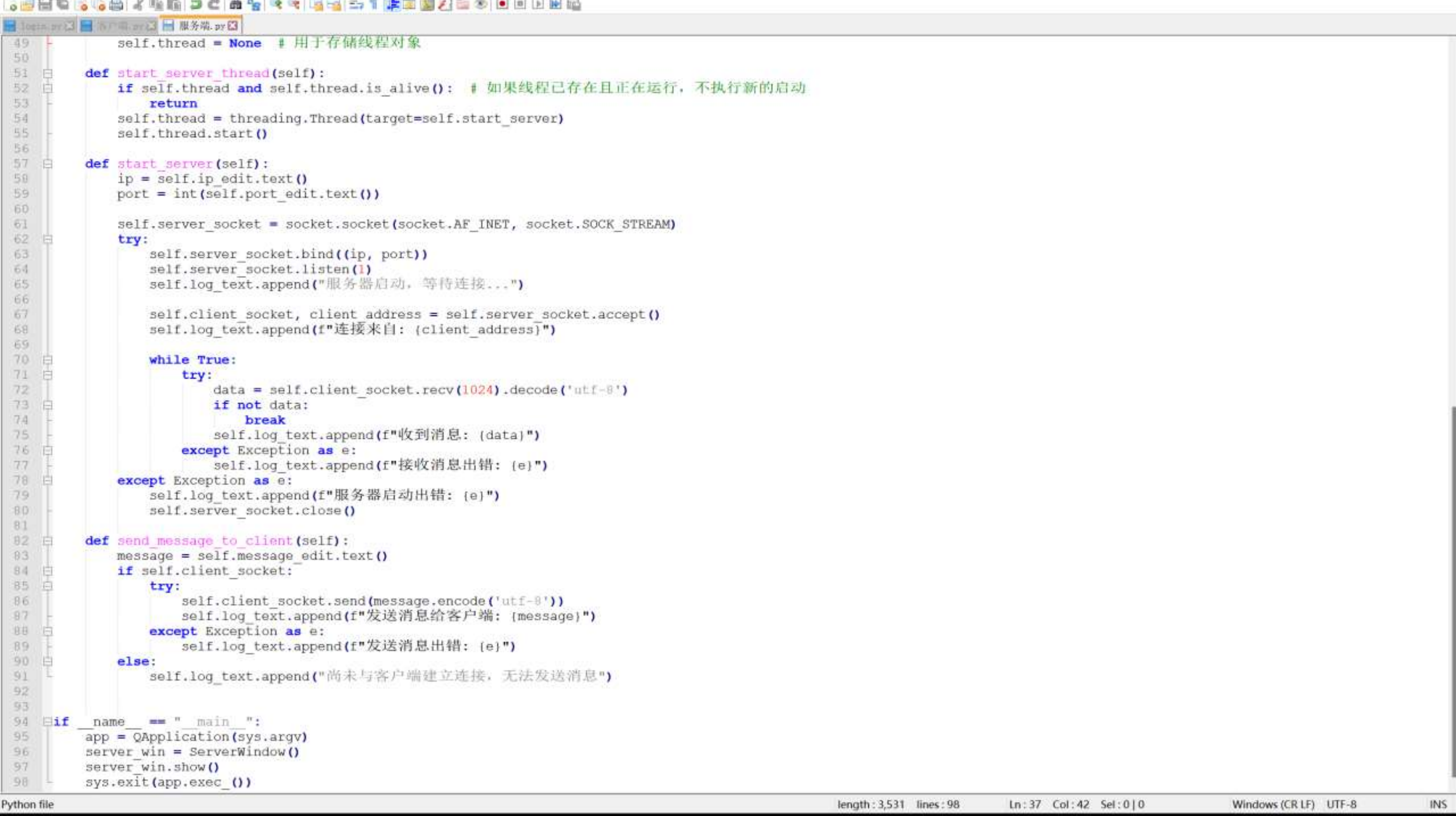
1

服务器代码展示



```python
import socket
import sys
import threading
from PyQt5.QtWidgets import QApplication, QWidget, QLabel, QLineEdit, QPushButton, QTextEdit
from PyQt5.QtCore import Qt


class ServerWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("服务器")
        self.resize(400, 300)

        self.ip_label = QLabel("IP 地址:", self)
        self.ip_label.move(20, 20)

        self.ip_edit = QLineEdit(self)
        self.ip_edit.move(100, 20)
        self.ip_edit.resize(180, 20)

        self.port_label = QLabel("端口:", self)
        self.port_label.move(20, 60)

        self.port_edit = QLineEdit(self)
        self.port_edit.move(100, 60)
        self.port_edit.resize(180, 20)

        self.start_button = QPushButton("启动服务器", self)
        self.start_button.move(20, 100)
        self.start_button.clicked.connect(self.start_server_thread)

        self.message_label = QLabel("消息:", self)
        self.message_label.move(20, 140)

        self.message_edit = QLineEdit(self)
        self.message_edit.move(100, 140)
        self.message_edit.resize(280, 20)

        self.send_to_client_button = QPushButton("发送给客户端", self)
        self.send_to_client_button.move(20, 180)
        self.send_to_client_button.clicked.connect(self.send_message_to_client)

        self.log_text = QTextEdit(self)
        self.log_text.move(20, 220)
        self.log_text.resize(360, 60)

        self.server_socket = None
        self.client_socket = None
        self.thread = None  # 用于存储线程对象

```

ython file   length : 3,531   lines : 98   Ln : 98   Col : 26   Sel : 0 | 0   Windows (CR LF)   UTF-8   INS

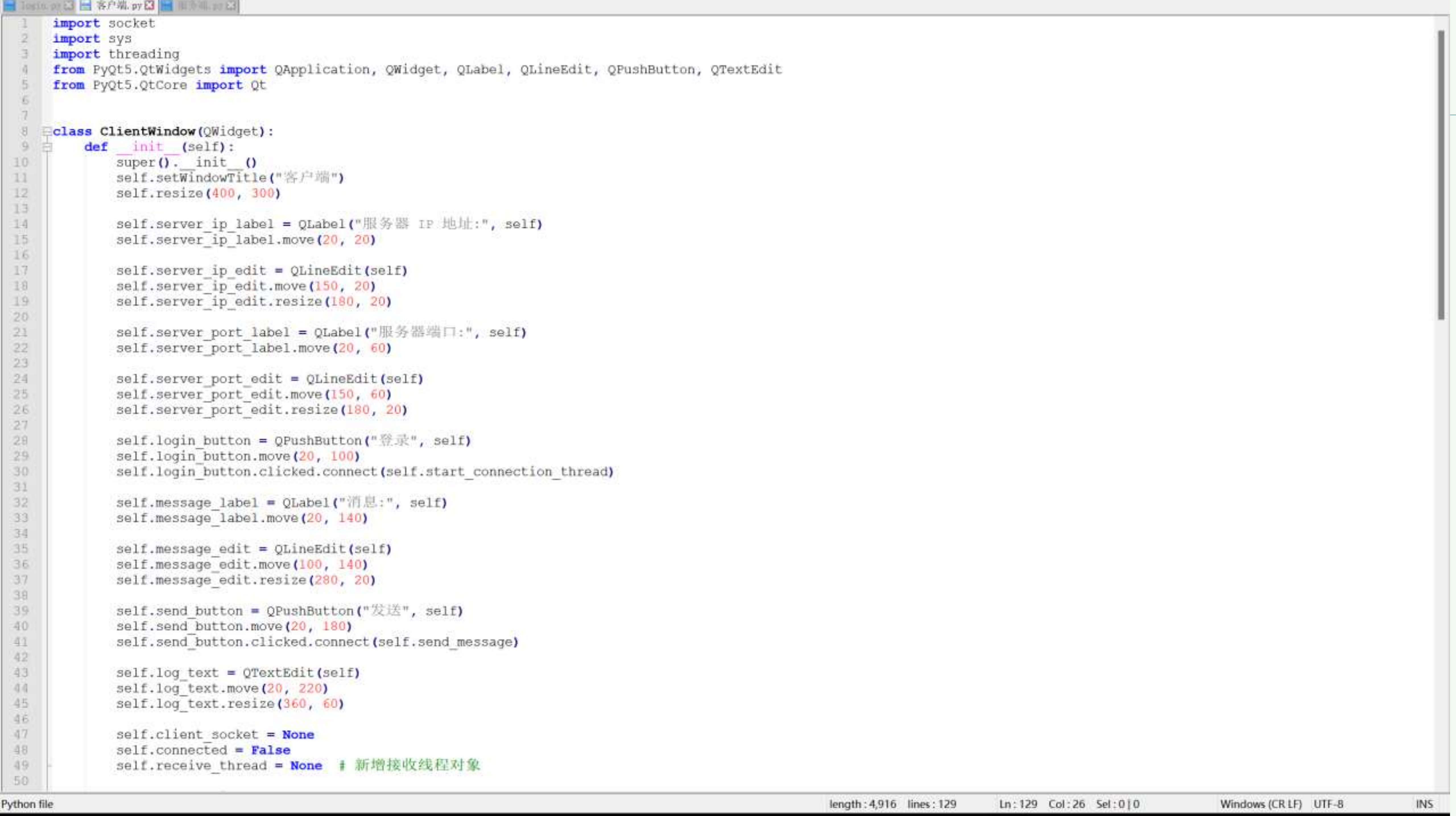

login.py 客户端.py 服务端.py

```python
        self.thread = None  # 用于存储线程对象

    def start_server_thread(self):
        if self.thread and self.thread.is_alive():  # 如果线程已存在且正在运行，不执行新的启动
            return
        self.thread = threading.Thread(target=self.start_server)
        self.thread.start()

    def start_server(self):
        ip = self.ip_edit.text()
        port = int(self.port_edit.text())

        self.server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        try:
            self.server_socket.bind((ip, port))
            self.server_socket.listen(1)
            self.log_text.append("服务器启动，等待连接...")

            self.client_socket, client_address = self.server_socket.accept()
            self.log_text.append(f"连接来自: {client_address}")

            while True:
                try:
                    data = self.client_socket.recv(1024).decode('utf-8')
                    if not data:
                        break
                    self.log_text.append(f"收到消息: {data}")
                except Exception as e:
                    self.log_text.append(f"接收消息出错: {e}")
        except Exception as e:
            self.log_text.append(f"服务器启动出错: {e}")
            self.server_socket.close()

    def send_message_to_client(self):
        message = self.message_edit.text()
        if self.client_socket:
            try:
                self.client_socket.send(message.encode('utf-8'))
                self.log_text.append(f"发送消息给客户端: {message}")
            except Exception as e:
                self.log_text.append(f"发送消息出错: {e}")
        else:
            self.log_text.append("尚未与客户端建立连接，无法发送消息")


if __name__ == "__main__":
    app = QApplication(sys.argv)
    server_win = ServerWindow()
    server_win.show()
    sys.exit(app.exec_())
```

Python file   length : 3,531   lines : 98   Ln : 37   Col : 42   Sel : 0 | 0   Windows (CR LF)   UTF-8   INS

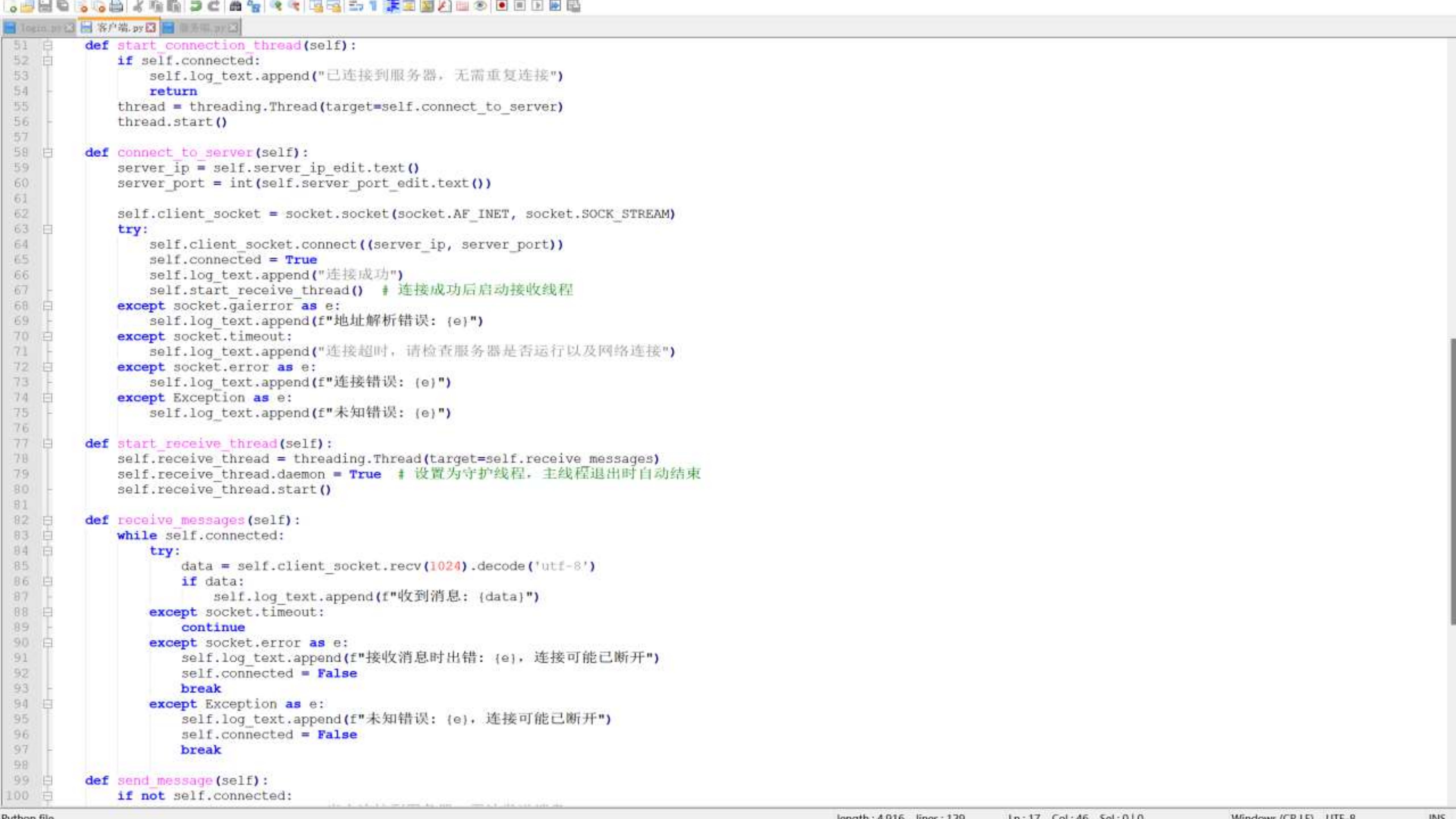
2

客服端代码展示

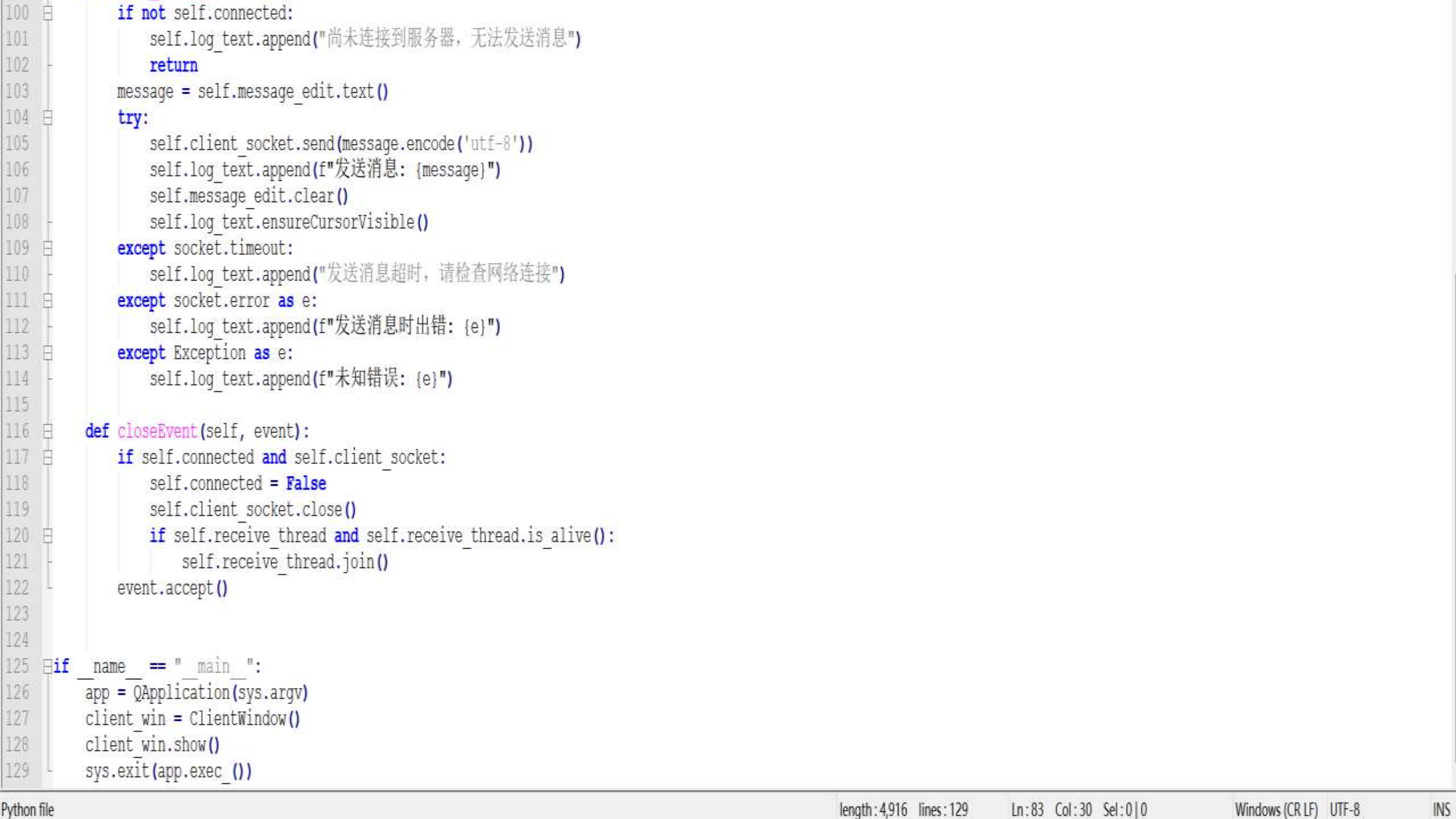

login.py 客户端.py 服务端.py

```python
import socket
import sys
import threading
from PyQt5.QtWidgets import QApplication, QWidget, QLabel, QLineEdit, QPushButton, QTextEdit
from PyQt5.QtCore import Qt


class ClientWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("客户端")
        self.resize(400, 300)

        self.server_ip_label = QLabel("服务器 IP 地址:", self)
        self.server_ip_label.move(20, 20)

        self.server_ip_edit = QLineEdit(self)
        self.server_ip_edit.move(150, 20)
        self.server_ip_edit.resize(180, 20)

        self.server_port_label = QLabel("服务器端口:", self)
        self.server_port_label.move(20, 60)

        self.server_port_edit = QLineEdit(self)
        self.server_port_edit.move(150, 60)
        self.server_port_edit.resize(180, 20)

        self.login_button = QPushButton("登录", self)
        self.login_button.move(20, 100)
        self.login_button.clicked.connect(self.start_connection_thread)

        self.message_label = QLabel("消息:", self)
        self.message_label.move(20, 140)

        self.message_edit = QLineEdit(self)
        self.message_edit.move(100, 140)
        self.message_edit.resize(280, 20)

        self.send_button = QPushButton("发送", self)
        self.send_button.move(20, 180)
        self.send_button.clicked.connect(self.send_message)

        self.log_text = QTextEdit(self)
        self.log_text.move(20, 220)
        self.log_text.resize(360, 60)

        self.client_socket = None
        self.connected = False
        self.receive_thread = None  # 新增接收线程对象

```

Python file length : 4,916 lines : 129 Ln : 129 Col : 26 Sel : 0 | 0 Windows (CR LF) UTF-8 INS



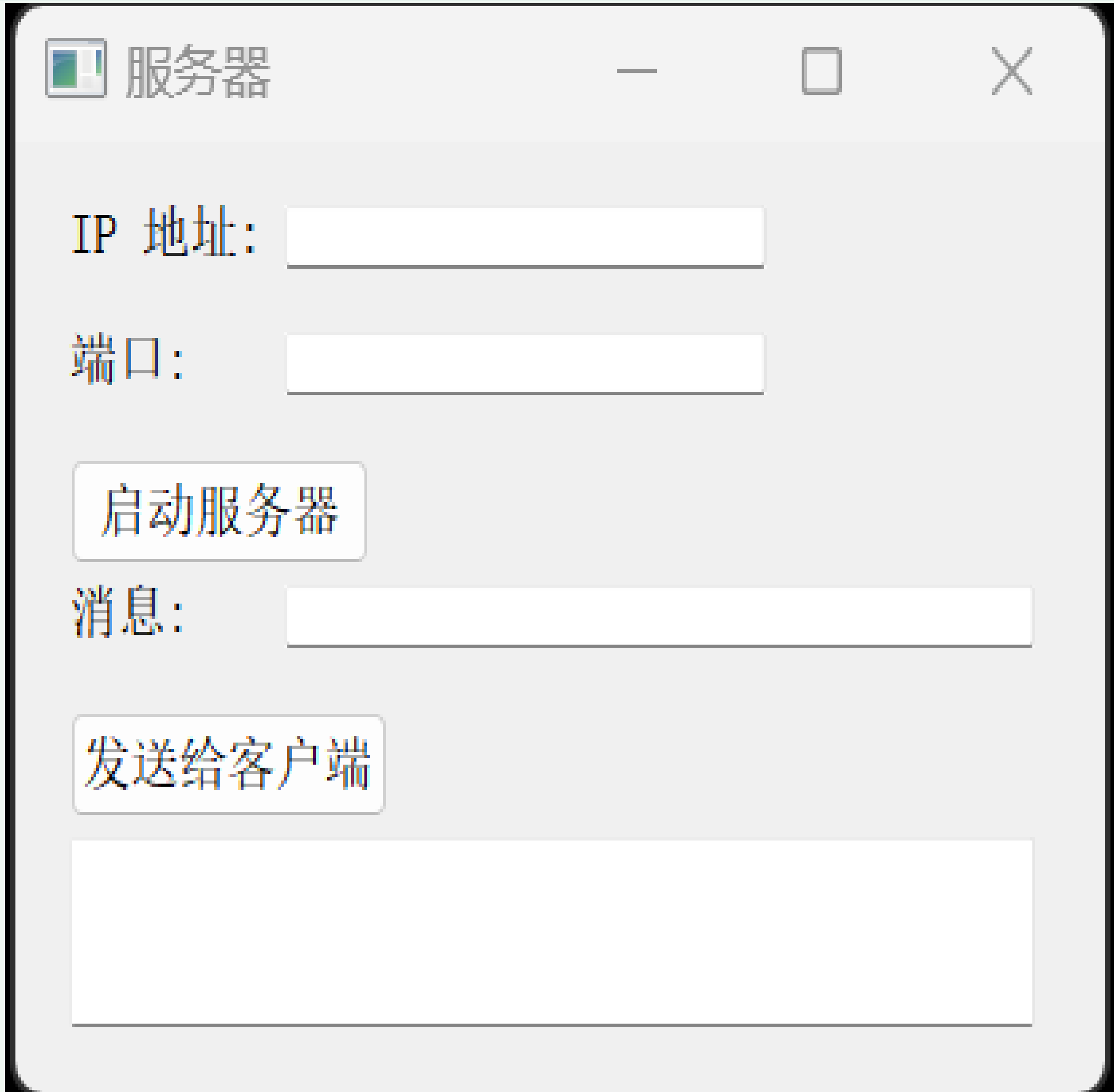
```
    def start_connection_thread(self):
        if self.connected:
            self.log_text.append("已连接到服务器，无需重复连接")
            return
        thread = threading.Thread(target=self.connect_to_server)
        thread.start()

    def connect_to_server(self):
        server_ip = self.server_ip_edit.text()
        server_port = int(self.server_port_edit.text())

        self.client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        try:
            self.client_socket.connect((server_ip, server_port))
            self.connected = True
            self.log_text.append("连接成功")
            self.start_receive_thread()  # 连接成功后启动接收线程
        except socket.gaierror as e:
            self.log_text.append(f"地址解析错误: {e}")
        except socket.timeout:
            self.log_text.append("连接超时，请检查服务器是否运行以及网络连接")
        except socket.error as e:
            self.log_text.append(f"连接错误: {e}")
        except Exception as e:
            self.log_text.append(f"未知错误: {e}")

    def start_receive_thread(self):
        self.receive_thread = threading.Thread(target=self.receive_messages)
        self.receive_thread.daemon = True  # 设置为守护线程，主线程退出时自动结束
        self.receive_thread.start()

    def receive_messages(self):
        while self.connected:
            try:
                data = self.client_socket.recv(1024).decode('utf-8')
                if data:
                    self.log_text.append(f"收到消息: {data}")
            except socket.timeout:
                continue
            except socket.error as e:
                self.log_text.append(f"接收消息时出错: {e}，连接可能已断开")
                self.connected = False
                break
            except Exception as e:
                self.log_text.append(f"未知错误: {e}，连接可能已断开")
                self.connected = False
                break

    def send_message(self):
        if not self.connected:
```

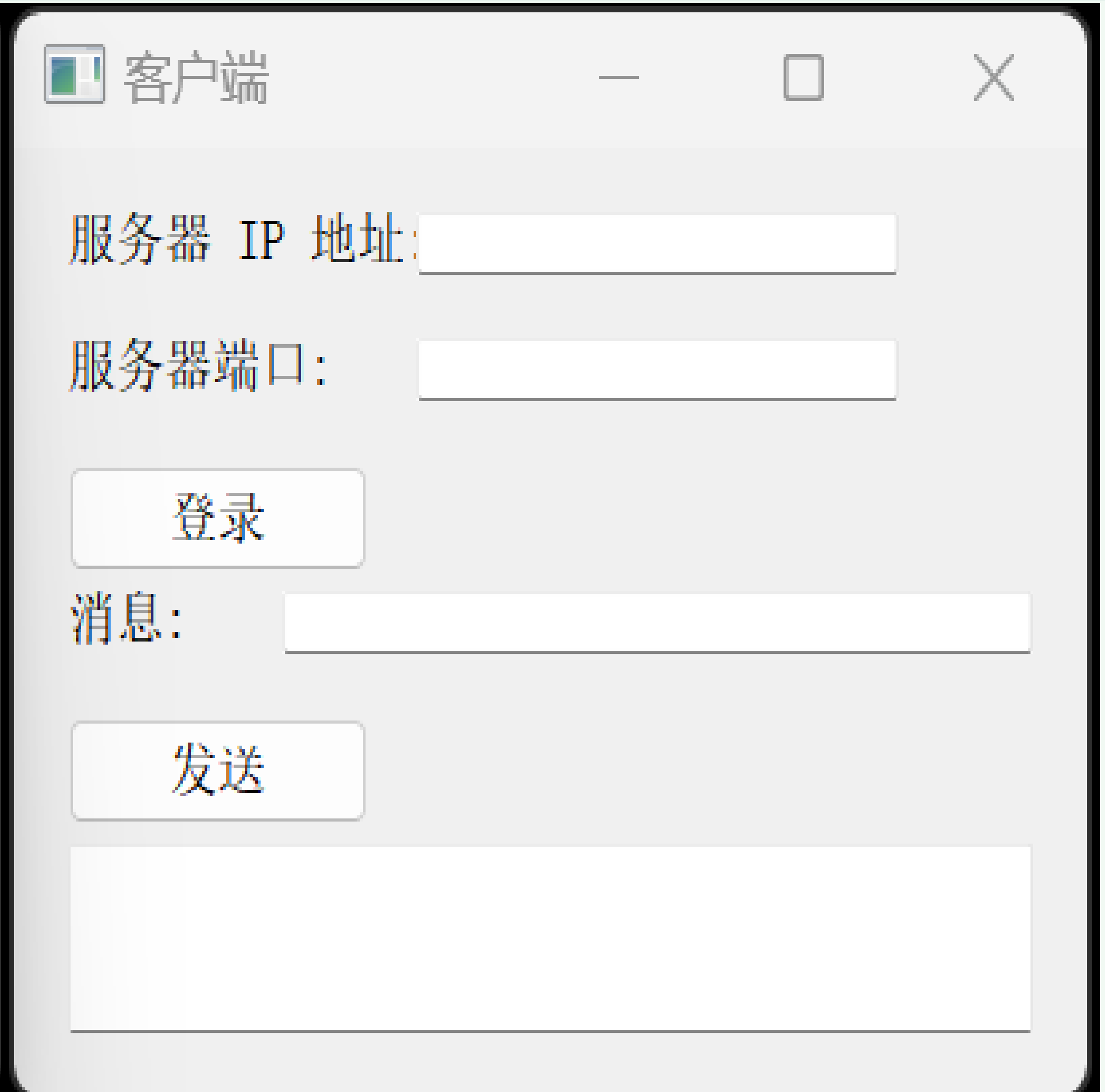


```python
        if not self.connected:
            self.log_text.append("尚未连接到服务器，无法发送消息")
            return
        message = self.message_edit.text()
        try:
            self.client_socket.send(message.encode('utf-8'))
            self.log_text.append(f"发送消息: {message}")
            self.message_edit.clear()
            self.log_text.ensureCursorVisible()
        except socket.timeout:
            self.log_text.append("发送消息超时，请检查网络连接")
        except socket.error as e:
            self.log_text.append(f"发送消息时出错: {e}")
        except Exception as e:
            self.log_text.append(f"未知错误: {e}")

    def closeEvent(self, event):
        if self.connected and self.client_socket:
            self.connected = False
            self.client_socket.close()
            if self.receive_thread and self.receive_thread.is_alive():
                self.receive_thread.join()
        event.accept()


if __name__ == "__main__":
    app = QApplication(sys.argv)
    client_win = ClientWindow()
    client_win.show()
    sys.exit(app.exec_())
```

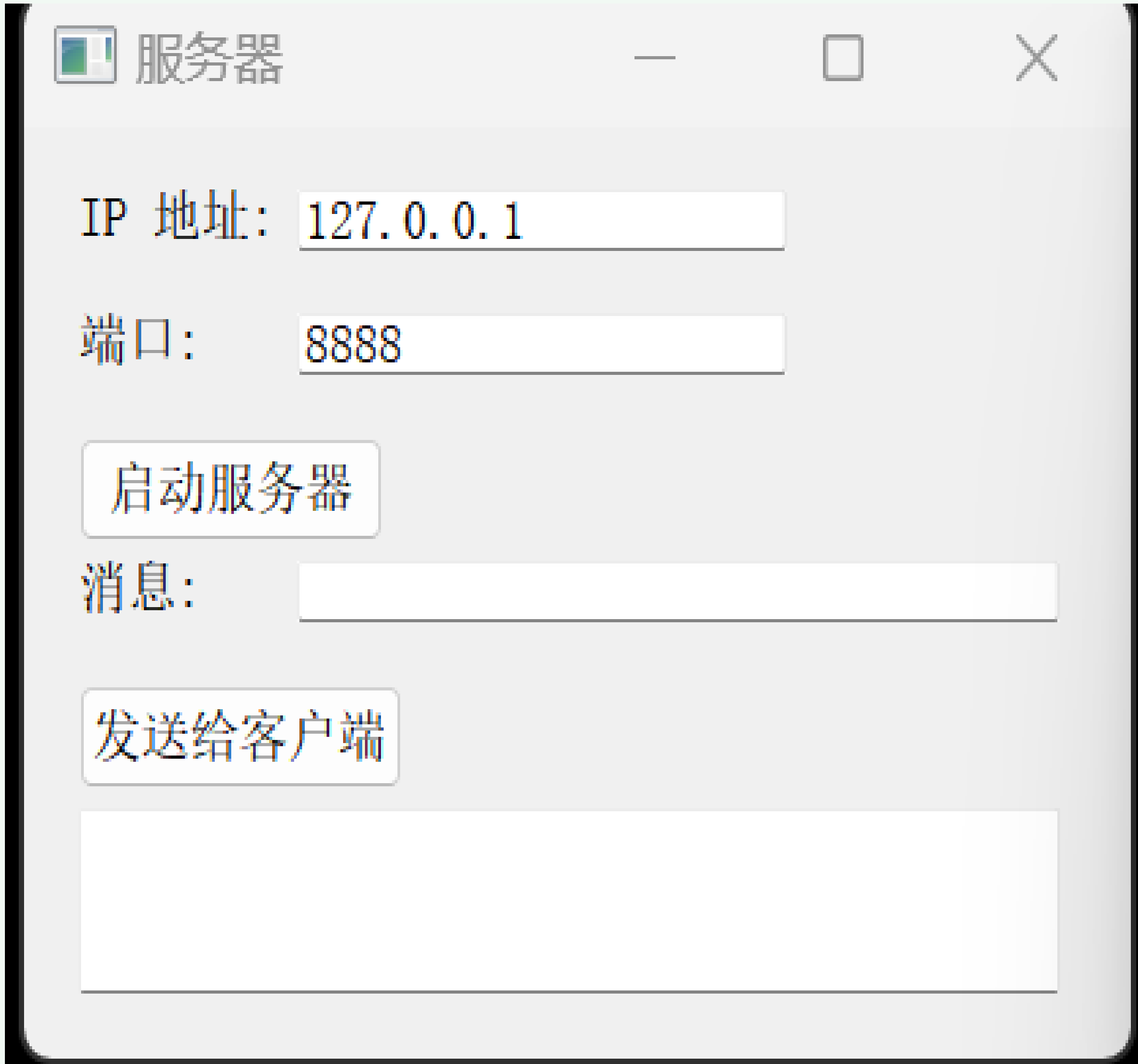
Python file   length : 4,916   lines : 129   Ln : 83   Col : 30   Sel : 0 | 0   Windows (CR LF)   UTF-8   INS

结果展示





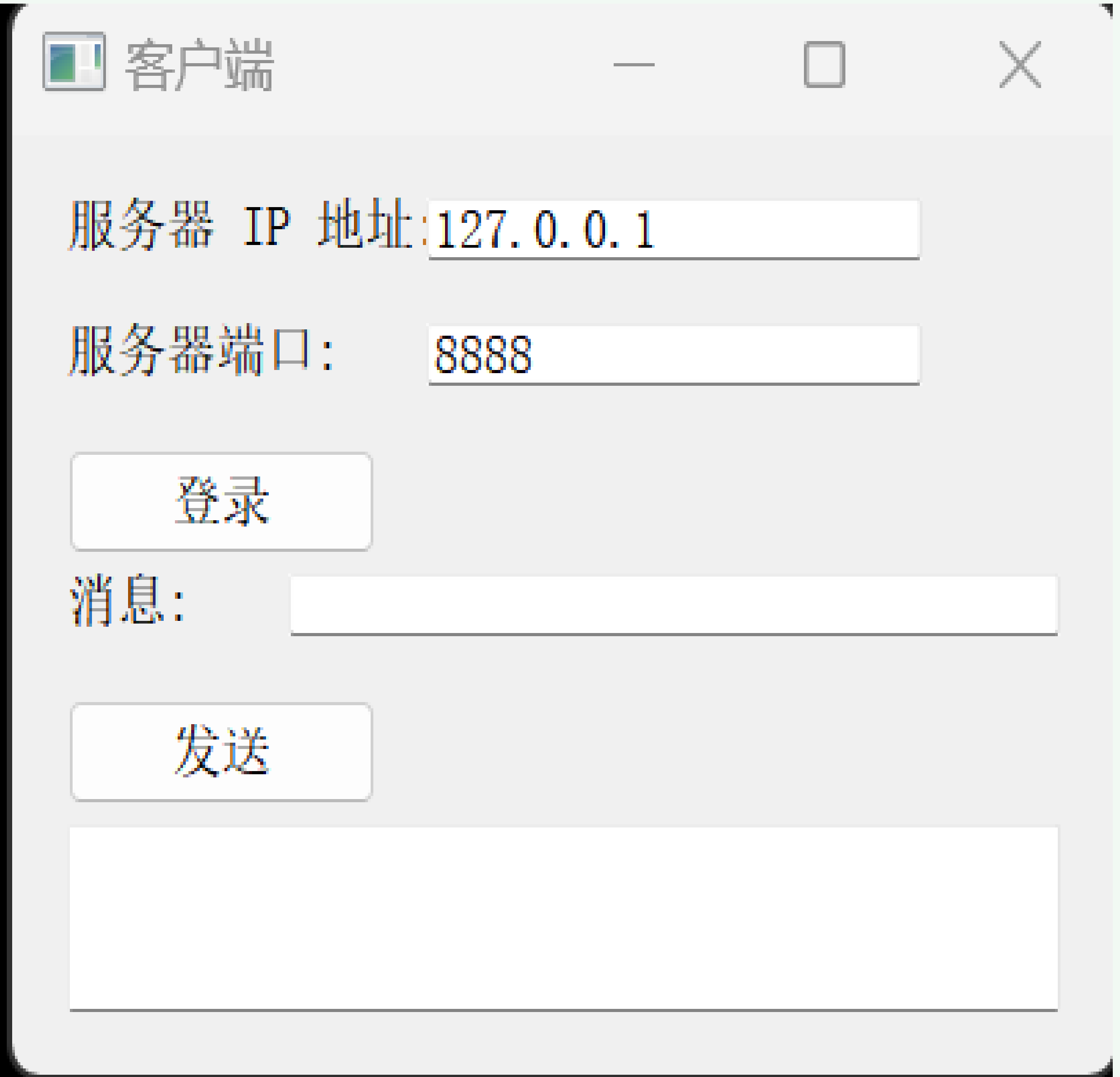
结果展示


服务器

IP 地址: 127.0.0.1

端口: 8888

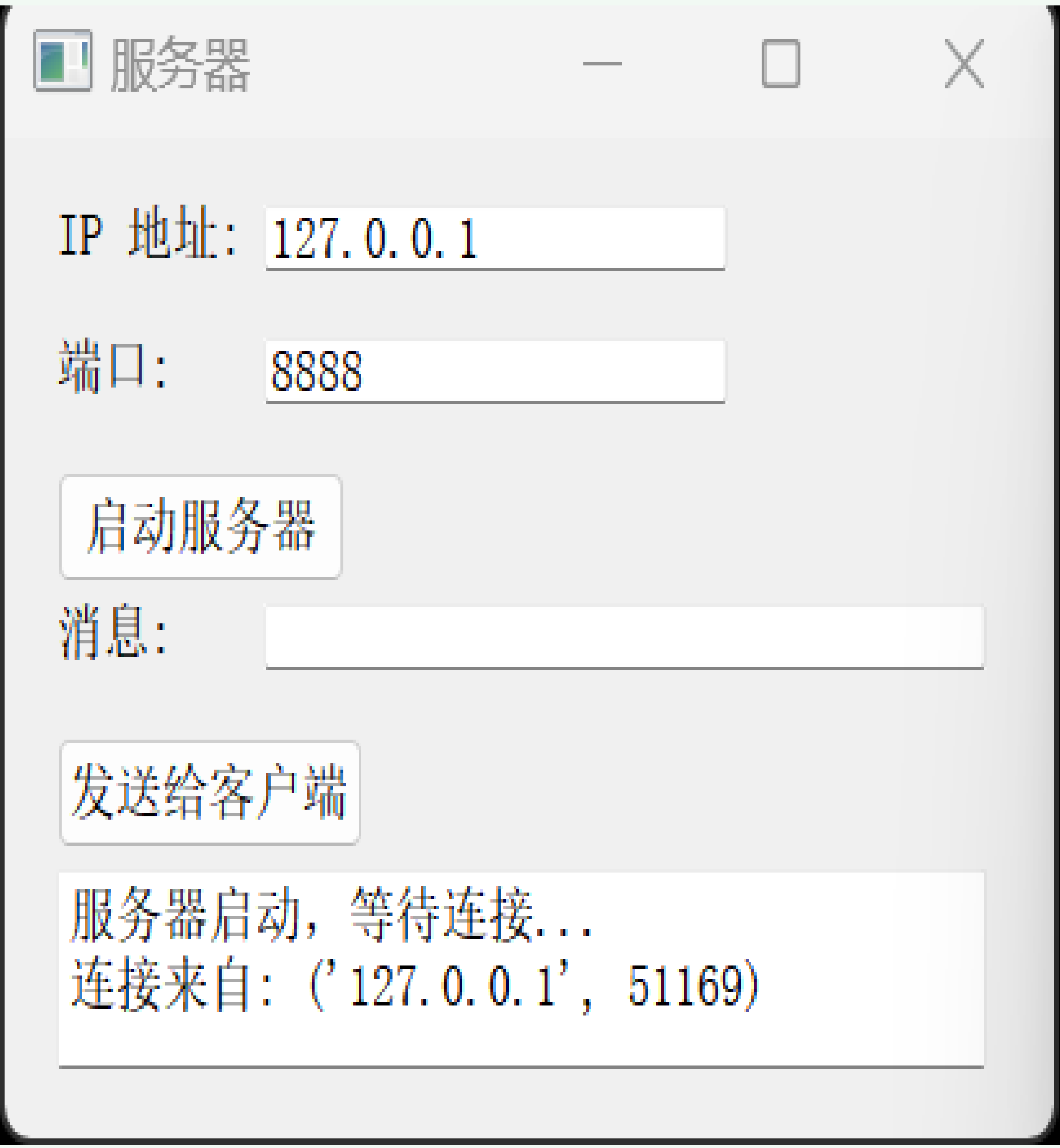
消息:


客户端

服务器 IP 地址: 127.0.0.1

服务器端口: 8888

消息:

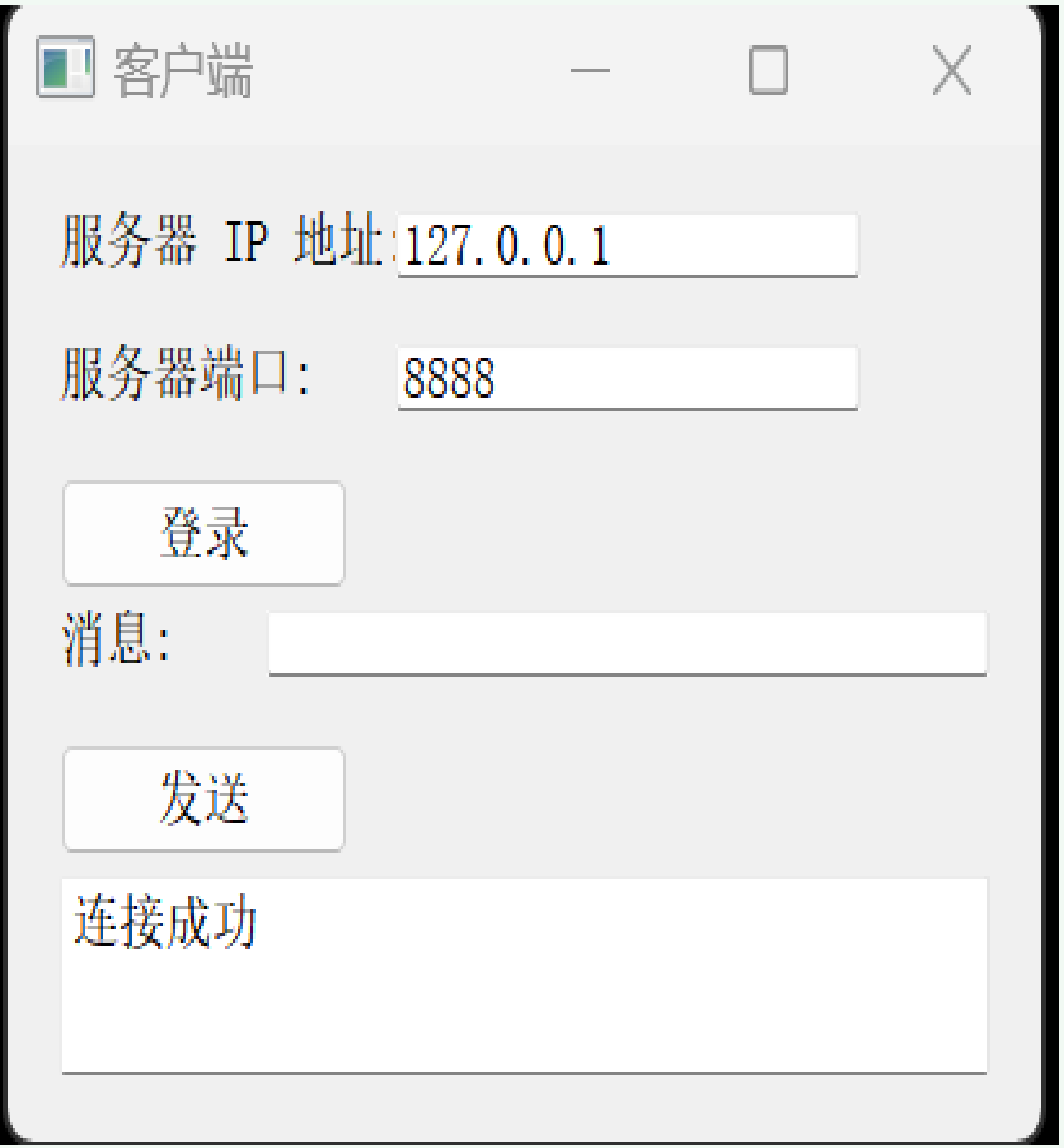


IP 地址：127.0.0.1

端口：8888

消息：

```
服务器启动，等待连接...
连接来自：('127.0.0.1', 51169)
```



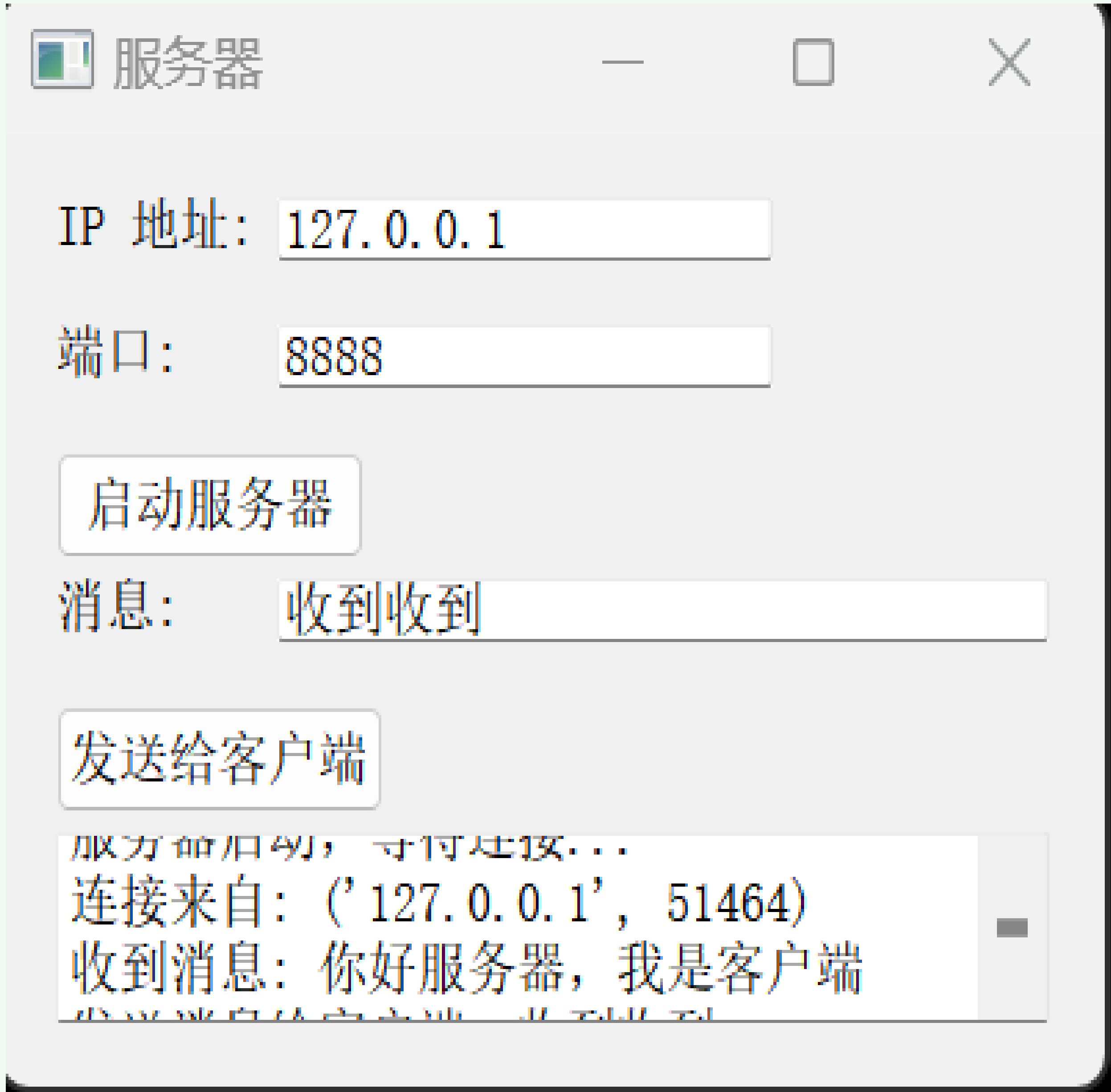
服务器 IP 地址：127.0.0.1

服务器端口：8888

消息：

```
连接成功
```

结果展示

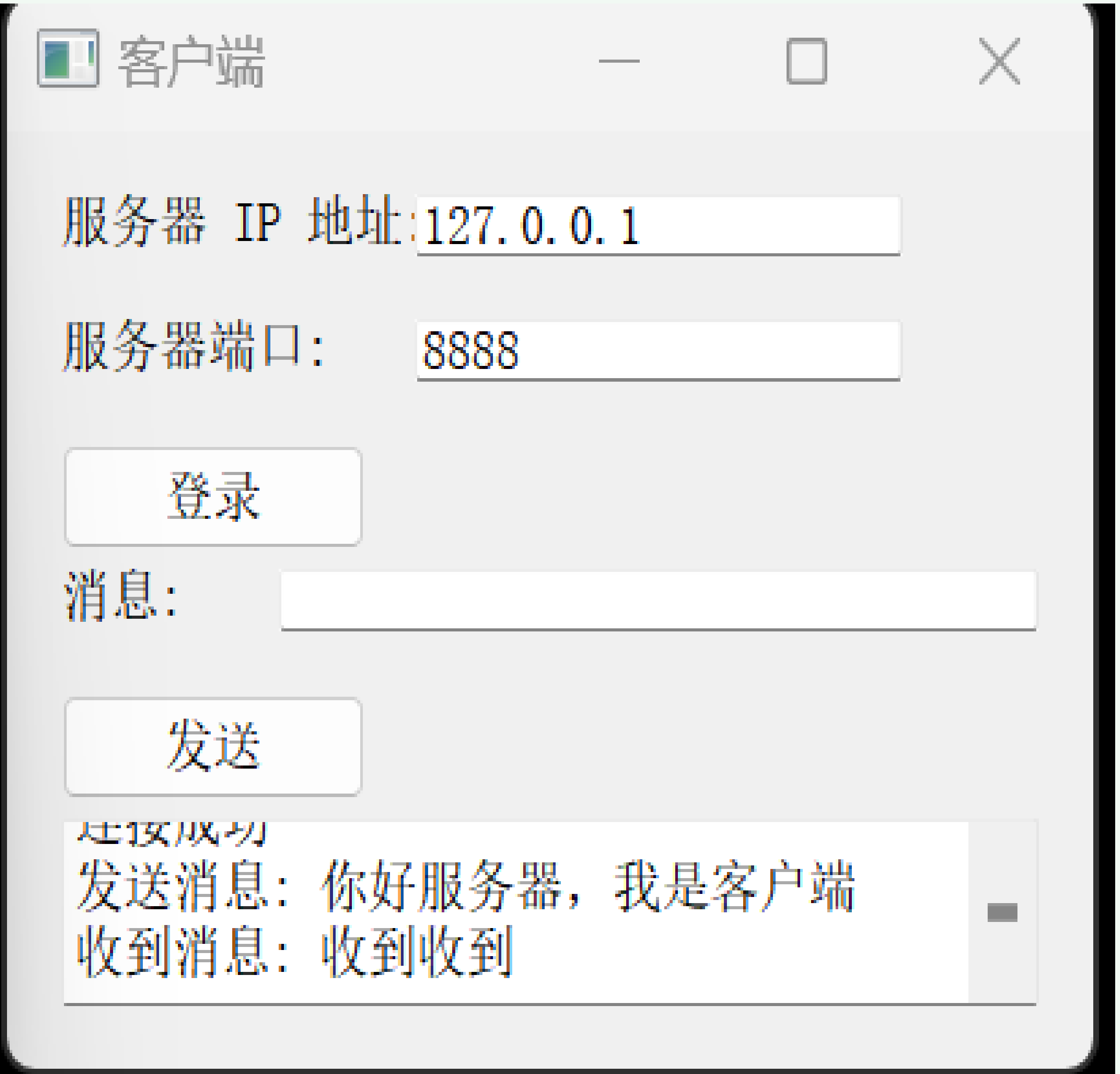


IP 地址: 127.0.0.1

端口: 8888

消息: 收到收到

```
服务器启动，等待连接...
连接来自：('127.0.0.1', 51464)
收到消息：你好服务器，我是客户端
发送消息给客户端：收到收到
```



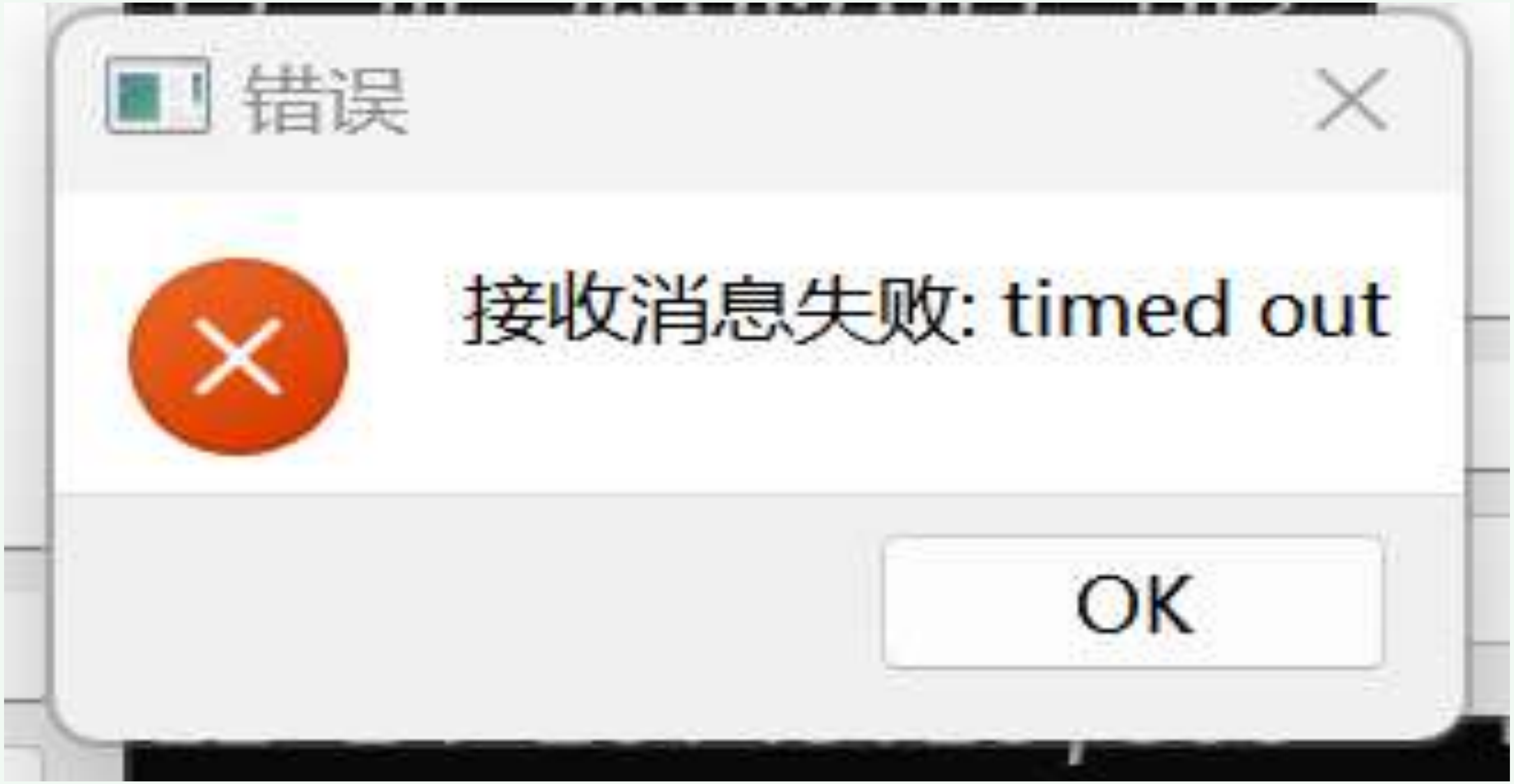
服务器 IP 地址: 127.0.0.1

服务器端口: 8888

消息:

```
连接成功
发送消息：你好服务器，我是客户端
收到消息：收到收到
```

存在问题：可以对话但是会出现连接超时



接收消息失败: timed out

谢谢

汇报人：方炫琪