Documentazione del Progetto chat_room.py [Traccia 1]

Introduzione

`chat_room.py` è un'applicazione client-server che consente agli utenti (appartenenti ad una stessa rete) di comunicare in una chat comune. Il programma è stato sviluppato in Python facendo uso delle librerie `socket` (per le connessioni e lo scambio di messaggi), `threading` (per rendere simultanei invio e recezione dei messaggi, e per gestire utenti multipli) e `tkinter` (per la finestra grafica del lato client).

Funzionalità Principali

Server:

- Gestisce la connessione dei client.
- Trasmette i messaggi inviati dai client agli altri utenti connessi, in modalità broadcast (escluso l'eventuale mittente).
- Gestisce le attività degli utenti, come l'ingresso e l'uscita dalla chat.

Client:

- Connette il client al server.
- Visualizza la chat in una finestra grafica.
- Invia e riceve messaggi dalla chat.

Componenti del Codice

Server-Side

- **`server_mode()`**: Funzione principale per avviare la modalità server.
- **`server_user_manager(user)`**: Gestisce la connessione di un singolo utente al server. Multipli utenti vengono gestiti tramite threads diversi.
- **`server_broadcast(message, sender=None)`**: Trasmette un messaggio a tutti gli utenti connessi al server, escluso l'eventuale sender.

Client-Side

- **`client_mode()`**: Funzione principale per avviare la modalità client. Consente la connessione con un server e inizializza la recezione, l'invio di messaggi e la finestra dell'applicazione.
- **`client_generate_window(client_socket)`**: Genera e visualizza la finestra dell'applicazione client.
- **`client_receive()`**: Riceve i messaggi dal server e li visualizza nella finestra dell'applicazione client (gestito tramite thread).
- **`client_send()`**: Invia un messaggio al server e lo visualizza nella finestra dell'applicazione client.
- **`client_closing_window()`**: Chiude la finestra dell'applicazione client, e torna alle impostazioni di connessione su console.

Altro

- **`mode_selector()`**: Consente all'utente di selezionare la modalità (server o client) in cui eseguire l'applicazione.

Esecuzione del Codice

1. Avviare il programma con

   ```
   python chat_room.py
   ```

   (o, su Windows, avviando lo script `start.bat`).
2. Selezionare la modalità in cui si vuole eseguire l'applicativo: `0` per la modalità server, `1` per la modalità client.
3. **Modalità Server**
   1. Il programma creerà la socket per il server all'indirizzo IP della macchina sulla quale è eseguito, e sulla porta 53000, e stamperà queste informazioni su console.
   2. Il server da adesso in poi trascriverà in console tutti i messaggi arrivati, e gli avvisi di entrata e uscita dalla chat.
   3. Alla chiusura della console, il server interromperà tutte le connessioni con gli utenti, chiudendo le loro finestre di chat e facendoli tornare alle opzioni di connessione.
4. **Modalità Client**
   1. Il programma richiederà di inserire l'IP e la porta alle quali connettersi (in caso di campi vuoti, tenterà la connessione all'IP della macchina stessa e alla porta 53000).
   2. Se la connessione andrà a buon fine, il programma chiederà all'utente di inserire un username da usare nello scambio di messaggi. Altrimenti, tornerà alle impostazioni di connessione.
   3. Una volta inserito l'username, si aprirà la finestra di chat. Tutti gli utenti connessi riceveranno l'avviso da parte del server che un nuovo utente si è collegato, indicandone l'username.
   4. Da adesso, i client potranno scambiarsi messaggi. Per inviarne uno, basta digitarlo dove indicato e premere invio. Tutti i messaggi ricevuti saranno visibili nel box della finestra. I messaggi inviati da un certo client verranno visualizzati dallo stesso con l'intestazione "You:". Ad ogni messaggio viene allegato l'orario nel quale è stato inviato.
   5. Per uscire dalla chatroom, basta chiudere la finestra di chat. Il server avviserà tutti gli utenti rimasti che un certo utente ha lasciato la chat. A questo punto, il client uscito dal server riaccede su console alle impostazioni di connessione, per potersi riconnettere allo stesso server, o ad un altro.

Requisiti

- **Python 3.x**
- **Librerie Python:** socket, threading, tkinter
- **Connessione ad una rete** che non blocchi i pacchetti

Considerazioni aggiuntive

Ho scelto di sviluppare entrambi i lati del software in un unico file per permettere a diversi utenti, scaricando un solo .py, di hostare la propria chatroom sulla propria macchina, oltre a poter diventare client di molteplici server diversi.

Autore

Francesco Tonelli, matricola 0001071531