Chat client-server - documentazione

Chat client-server 1.0

Passaggi per la connessione:
Creazione del server
Creazione del client che si dovrà collegare al server
Dopo aver instaurato la connessione il client invia il nome al server
Il client invia il messaggio al server, questo lo invia all'altro client
Viceversa il secondo client deve inviarlo al primo
per uscire dalla chat scrivere "Logout"

Classe Server
classe Client
classe per la gestione dei client

La prima classe viene usata per creare il server, ricevere e accettare le varie richieste di comunicazione da parte dei client

La seconda classe verrà usata per inserire i nomi dei client all'interno del vettore per usarli in seguito nella ricerca del destinatario.
Qui utilizzo Stringtokenizer per separare il messaggio dal destinatario tramite il formato:
messaggio#destinatario

Classe Client
La classe contiene 2 thread, uno per l'invio del messaggio, l'altro per la lettura del messaggio.
WriteUTF e ReadUTF utilizzano la codifica UTF-8 ovvero una codifica usata per rappresentare i caratteri Unicode come **#** o **:**
Il **#** viene usato come separatore tra il messaggio e il destinatario.

Chat client-server 2.0

Primo obiettivo: gestione della disconnessione del client

```
if(ricevuta.equals("Logout")){
                this.login=false;
                this.s.close();
                break;
            }
```

Secondo obiettivo: i client devono conoscere i nomi dei client connessi

Per fare ciò invio al client connesso ogni client mediante l'uso della variabile i:

IDEA INIZIALE

```
if(i == 0){
            System.out.println("Sei il Client 0 e sei il primo nella chat");
        }
        else if(i > 0){
            System.out.println("Sei il Client " + i);
            while(i == 0){
              System.out.println("Nella chat è presente: Client " + i - -);
            }
        }
```

```
{
… //varie righe di codice
}

i++;
```

CONCLUSIONI: facendo così c'è il problema che, una volta che i diminuisce, resti a 0 e quando aumenta torna a 1 rimanendo in un ciclo infinito

_____

```
if(i == 0){
            System.out.println("Sei il Client 0 e sei il primo nella chat");
          }
          else if(x > 0){
            System.out.println("Sei il Client " + i);
int x = i;
            while(x == 0){
                System.out.println("Nella chat è presente: Client " + x - -);
            }
          }
{
… //varie righe di codice
}

i++;
```

Aggiungo una variabile x = i in modo tale da usare x solo per diminuirla e mostrare ai client tutti quelli già connessi nella chat, mentre così i rimane con lo stesso valore senza modificarla. Quando x viene diminuita, prima del while, viene posta nuovamente uguale a i.

Per far mostrare al client i vari messaggi del server riguardante chi è connesso nella chat, sostituisco i vari System.out.println precedenti con dis.writeBytes() che mi consente di inviare la stringa del messaggio al client. Nella classe client ho creato un metodo per poter mostrare all'utente il messaggio con un System.out.println().

Conclusioni sull'obiettivo
Il programma mostrava un errore nel STR() che veniva risolto togliendo static al main. Facendo ciò, il programma non cominciava a causa della mancanza del main statico, per risolvere ciò ho creato un'ulteriore classe, sia per il server che per il client, Main dove poter inizializzare le altre classi contenenti i metodi.