École d'ingénieurs
Télécom Physique Strasbourg

Chatroom : Server Clients

AYMANE ELMAHI MOHAMED FAID

Introduction

Ce projet est le fruit du TP commencé en classe avec le professeur Pascal Mérindol. Ce TP est une initiation à la programmation socket et aux fonctions associées (select(), accept(), socket() ...).

Ce projet consiste d'un "chatroom " centralisé de maximum 10 clients avec la possibilité d'envoyer un message privé à un utilisateur spécifique sur le terminal linux.

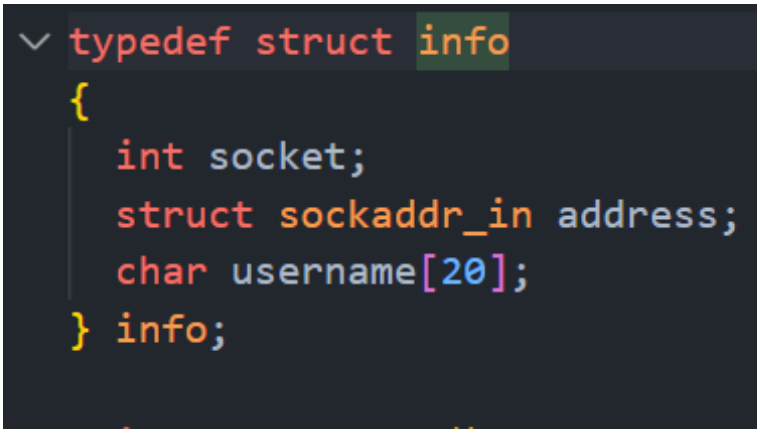
Le "chatroom" utilise le protocole TCP pour l'échange des données et l'utilisation de select qui gère plusieurs clients à la fois.

Notre programme est écrit en Langage C.

Implementation :

On a utilisé deux structures :

Une structure “message” pour gérer les messages des différents clients (leur types, id de l'émetteur et le type), ainsi qu'une structure “info” plus compacte pour initialiser les sockets , le port , l'adresse IP ..



Composition des fichiers:

message.h : Nous avons énuméré les différents types possibles de messages (MESSAGE_TO_ALL, PRIVATE,CONNECTING,...) ainsi que la structure qui accepte les données d'un message.

colors.h : Les couleurs utilisées pour les messages à afficher.

tools.h : Toutes les inclusions nécessaires ainsi que les fonctions "delete_line()" et "*cleanbuffer()*" pour gérer des problèmes de buffer et d'affichage. Il comporte aussi la structure nécessaire pour accueillir les informations de chaque client et sa socket.

server-tcp.c : Programme pour gérer le serveur qui accueille les clients. On initialise les descripteurs, on et on attend l'arrivée d'un message qu'on traite ensuite dépendamment du type du message. On gère aussi les cas de nouvelles connexions et de déconnexions.

client-tcp.c: programme qui se connecte au serveur et lui envoie le message entré par l'utilisateur.

Réalisation :

Après s'être mis d'accord sur les structures à utiliser et les fonctionnalités à implémenter, le travail a été réparti de la manière suivante:

Aymane Elmahi : écriture du code du serveur afin de gérer les différents connexions et déconnexions des clients sur le serveur.

Mohamed Faid : écriture du code du client qui gère les différents types de message (de groupe ou privé) et la connexion au serveur.

Utilisation du programme

Pour lancer le projet, il faut utiliser un terminal.
Le projet peut être compilé grâce à la commande ***make***.
On peut démarrer le serveur grâce à la commande :
./server <port>
après avoir choisi un numéro de port convenable.
Ensuit il est possible d'ouvrir un nouveau terminal par exemple et rejoindre le serveur grâce à la commande :
./client <ip> <port>
(Il est possible d'utiliser l'adresse IP locale 127.0.0.1 et le même port choisi par le serveur).
En tapant ***/help*** dans le terminal d'un client, il est possible d'afficher les commandes disponibles :
/help or /h
/quit or /q
/list or /l
/m <username> <message>
Chaque commande vient avec une explication.

Difficultés rencontrées :

L'exemple vu en TP avec un seul client était simpliste mais fondamental pour la réalisation de ce programme , la difficulté était particulièrement algorithmique, surtout pour permettre à un client d'envoyer un message privé à quelques clients seulement et non à un client spécifique ou à tous les clients.
La durée du projet était correcte grâce au prérequis acquis en TP et en TCP/IP.

Conclusion :

La réalisation d'un "chatroom" était très intéressante, surtout la découverte de nouvelles fonctions comme select() qui a permet de gérer plusieurs clients.
Prochainement on voudrait améliorer notre projet vers un tchat pair-to-pair décentralisé.