

Lab 5 – WinSock

Exercice 8 : Créer un chat client

Afin de réaliser ce chat client j'ai créé des fonctions à partir des exemples de code des exercices précédents :

Une pour recevoir des données et les afficher dans la console :

```
int receive_data(SOCKET * s, WSADATA * wsa)
```

Une pour envoyer un message :

```
int send_data(char * IP, int port, char * message, SOCKET * s, WSADATA * wsa)
```

Ces deux fonctions ont en paramètre le socket actuel et la variable WSADATA. J'utilise `receive_data` autant chez le client comme chez le serveur par contre seul le client envoie des données.

Afin de me connecter au serveur j'utilise l'adresse 127.0.0.1. Après avoir initialisé winsock, le serveur se prépare à écouter : il attend donc une connexion d'un éventuel client. Une fois qu'une connexion est établie il imprime dans la console le message reçu puis se ferme.

Pour réaliser le chat client, j'ai décidé que chaque personne impliquée dans le chat aurait une application client et une application serveur. L'application client sert uniquement à envoyer des données tandis que l'application serveur à en recevoir. Voici une capture d'écran de comment la connexion s'établit entre les deux personnes au début :

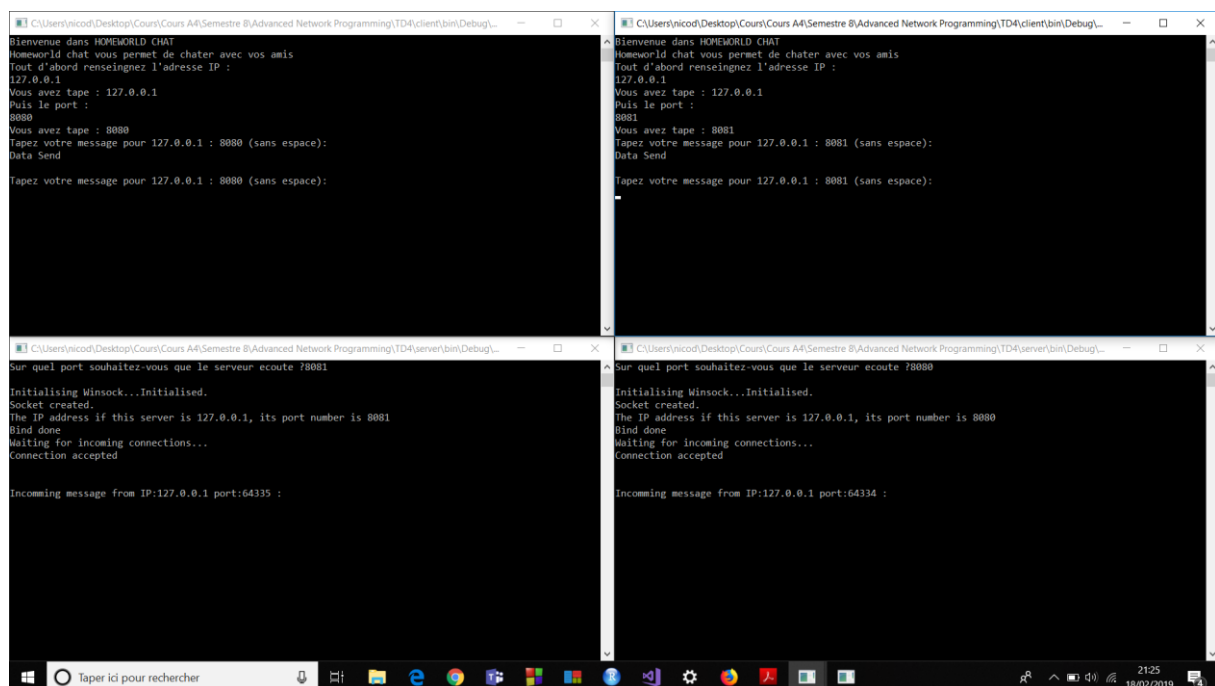
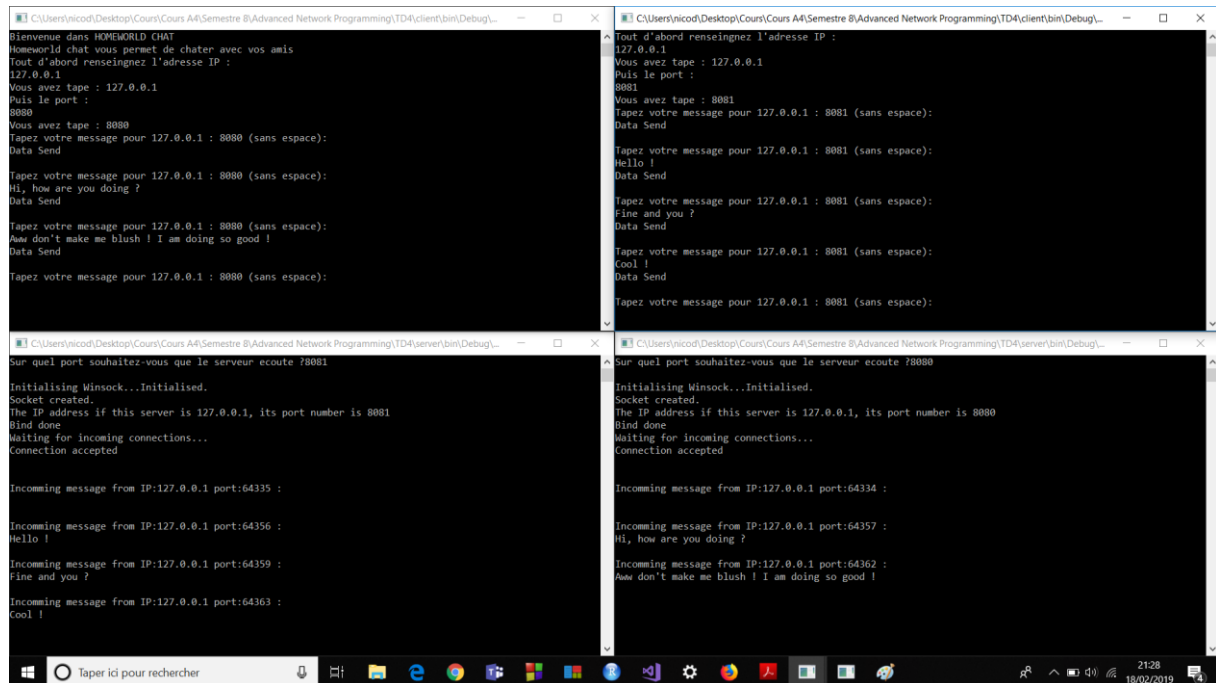


Figure 1 Etablir la connexion avant de discuter

Chaque serveur demande à son lancement sur quel port il va écouter et chaque client à son lancement demande l'adresse IP du serveur (qui est ici 127.0.0.1 car tout se passe en local). Pour que l'on puisse en fait saisir des messages avec espace j'ai utilisé la fonction `fgets` qui semble envoyer toujours une partie de ce qui est dans le buffer dès que la connexion s'établit mais ensuite fonctionne comme elle le devrait. Voici un extrait d'une conversation (passionnante) :



```

C:\Users\nicod\Desktop\Cours A4\Semestre 8\Advanced Network Programming\TD4\client\bin\Debug>
Bienvenue dans HOMERLORD CHAT
Homeworld chat vous permet de chatter avec vos amis
Tout d'abord renseignez l'adresse IP :
127.0.0.1
Vous avez tape : 127.0.0.1
Puis le port :
8080
Vous avez tape : 8080
Tapez votre message pour 127.0.0.1 : 8080 (sans espace):
Data Send

Tapez votre message pour 127.0.0.1 : 8080 (sans espace):
Hi, how are you doing ?
Data Send

Tapez votre message pour 127.0.0.1 : 8080 (sans espace):
Aw! don't make me blush ! I am doing so good !
Data Send

Tapez votre message pour 127.0.0.1 : 8080 (sans espace):

C:\Users\nicod\Desktop\Cours A4\Semestre 8\Advanced Network Programming\TD4\server\bin\Debug>
Sur quel port souhaitez-vous que le serveur écoute ?8080

Initialising Winsock...Initialised.
Socket created.
The IP address if this server is 127.0.0.1, its port number is 8080
Bind done
Waiting for incoming connections...
Connection accepted

Incoming message from IP:127.0.0.1 port:64335 :

Incoming message from IP:127.0.0.1 port:64356 :
Hello !

Incoming message from IP:127.0.0.1 port:64359 :
Fine and you ?

Incoming message from IP:127.0.0.1 port:64363 :
Cool !

C:\Users\nicod\Desktop\Cours A4\Semestre 8\Advanced Network Programming\TD4\client\bin\Debug>
Tout d'abord renseignez l'adresse IP :
127.0.0.1
Vous avez tape : 127.0.0.1
Puis le port :
8081
Vous avez tape : 8081
Tapez votre message pour 127.0.0.1 : 8081 (sans espace):
Data Send

Tapez votre message pour 127.0.0.1 : 8081 (sans espace):
Hello !
Data Send

Tapez votre message pour 127.0.0.1 : 8081 (sans espace):
Fine and you ?
Data Send

Tapez votre message pour 127.0.0.1 : 8081 (sans espace):
Cool !
Data Send

Tapez votre message pour 127.0.0.1 : 8081 (sans espace):

```

Figure 2 Conversation entre les deux chacun à son serveur et son client à gauche ou à droite de l'écran

Pour lancer ces programmes je me sers directement des exécutables créés par Codeblocks en allant dans le dossier `bin > Debug`.