

I2180 : Exercices de Programmation Système (8)

8.1. Exercice - appel système : socket

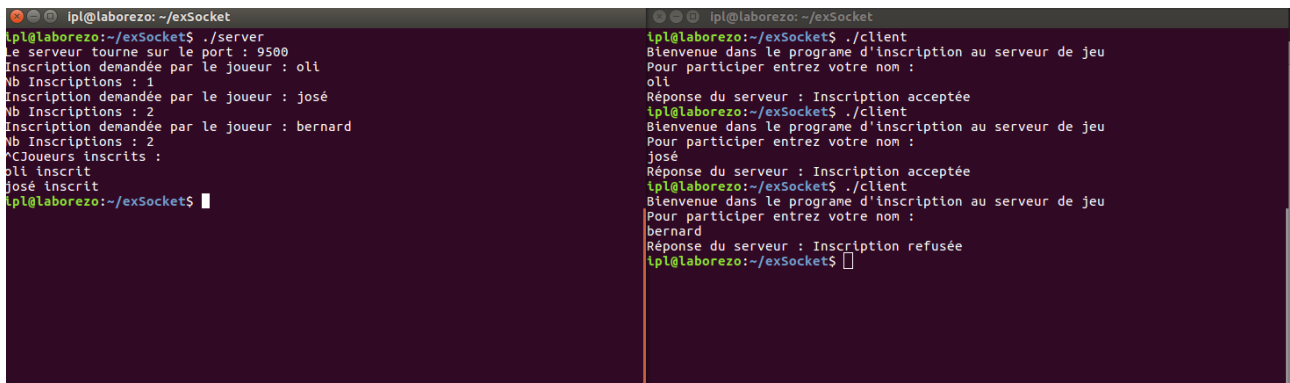
Ecrivez 2 programmes : un client et un serveur. Le serveur sera un serveur d'inscription permettant de rejoindre un salon de jeu. Le client sera un programme utilisé par les joueurs pour envoyer leur demande d'inscription au serveur. Le client enverra un message de type « INSCRIPTION_REQUEST » au serveur et le serveur répondra par « INSCRIPTION_OK » ou « INSCRIPTION_KO ». Le serveur autorisera un maximum de X joueurs (X étant une constante paramétrable).

Pour vous aider, nous avons déjà construit la structure des programmes (ressources sur Moodle):

1. Un fichier « messages.h » contient les messages qui seront échangés entre le serveur et le client ainsi que leur structure.
2. Le « server » utilisera une fonction « initSocketServer » qui renverra un socket d'écoute.
3. Le « client » utilisera une fonction « initSocketClient » qui renverra un socket de connexion au serveur.
4. Un Ctrl-C sur le serveur affichera le nom des inscrits avant de terminer le processus.

Nous vous demandons d'écrire les fonctions « initSocketServer » et « initSocketClient » afin de faire fonctionner ces 2 programmes.

Résultat attendu :



```
iplt@laborezo:~/exSocket
iplt@laborezo:~/exSocket$ ./server
Le serveur tourne sur le port : 9500
Inscription demandée par le joueur : oli
Nb Inscriptions : 1
Inscription demandée par le joueur : jose
Nb Inscriptions : 2
Inscription demandée par le joueur : bernard
Nb Inscriptions : 2
^CJoueurs inscrits :
oli inscrit
jose inscrit
iplt@laborezo:~/exSocket$

iplt@laborezo:~/exSocket$ ./client
Bienvenue dans le programme d'inscription au serveur de jeu
Pour participer entrez votre nom :
oli
Réponse du serveur : Inscription acceptée
iplt@laborezo:~/exSocket$ ./client
Bienvenue dans le programme d'inscription au serveur de jeu
Pour participer entrez votre nom :
jose
Réponse du serveur : Inscription acceptée
iplt@laborezo:~/exSocket$ ./client
Bienvenue dans le programme d'inscription au serveur de jeu
Pour participer entrez votre nom :
bernard
Réponse du serveur : Inscription refusée
iplt@laborezo:~/exSocket$
```

8.2. Aspirateur site Web - appel système : socket

Ecrivez un programme utilisant un socket de connexion dans le but de récupérer les différentes pages HTML d'un site web. Pour cet exercice, nous nous limiterons à aspirer quelques pages HTML du site « <http://ochoquet.be/syllabusHTML/> ». Plus précisément, pour chaque page à récupérer (voir sitemap.txt), vous vous connecterez au site « ochoquet.be » via un socket, vous enverrez une requête http GET, vous récupérerez la réponse et vous l'écrirez dans un fichier.

Pour vous aider, voici quelques indications :

1. Voici un exemple de requête HTTP qui peut être envoyée via un socket de connexion :
« GET http://ochoquet.be/syllabusHTML/index.html HTTP/1.0\r\nHost: ochoquet.be\r\n\r\n »
2. Vous disposez d'un fichier « sitemap.txt » listant les pages HTML à récupérer.
3. Vous disposez d'une fonction « hostname_to_ip » permettant de convertir un nom de domaine en une adresse IP (voir utils.c)

4. Vous disposez d'une fonction «readline» permettant de lire un fichier ligne par ligne (voir utils.c)
5. Vous devez effectuer une connexion pour chaque page à récupérer. Réutilisez votre fonction « initSocketClient » faite à l'exercice précédent.
6. Pour obtenir le nom d'une page à partir des URL présentes dans le fichier « sitemap.txt », vous pouvez utiliser la fonction C « strchr » de cette façon :
`sprintf(pageName, "%s", strchr(ligne, '/') + 1);`

Résultat attendu :

```
ipl@laborezo:~/ExSocket8.2$ ./client
IPv4 ochoquet.be : 213.186.33.4
Ligne : http://ochoquet.be/syllabusHTML/index.html ,strlen :42
Page Name : index.html
Ligne : http://ochoquet.be/syllabusHTML/internet.html ,strlen :45
Page Name : internet.html
Ligne : http://ochoquet.be/syllabusHTML/html5.html ,strlen :42
Page Name : html5.html
Ligne : http://ochoquet.be/syllabusHTML/css3.html ,strlen :41
Page Name : css3.html
ipl@laborezo:~/ExSocket8.2$
```

Et le répertoire courant contiendra les fichiers « index.html », « internet.html », « html5.html », « css3.html »

Amélioration possible : retirer les lignes headers de la réponse http c'est-à-dire ceci (Ex.) :

```
HTTP/1.1 200 OK
Set-Cookie: 240planBAK=R2339302148; path=/; expires=Wed, 27-Mar-2019 15:54:08 GMT
Date: Wed, 27 Mar 2019 14:36:37 GMT
Content-Type: text/html
Content-Length: 2390
Connection: close
Set-Cookie: 240plan=R915039546; path=/; expires=Wed, 27-Mar-2019 15:54:08 GMT
Server: Apache
Accept-Ranges: bytes
Vary: Accept-Encoding
X-IPLB-Instance: 178
```