Compte-rendu du TP3 de MRI : Les Sockets

Pierre VANNIER & Salomé COAVOUX

17 février 2014

Le code se suffit à lui-même pour répondre à la plupart des questions. Vous trouverez ici quelques points qu'il nous a paru bon d'éclairer.

1 EXERCICE 1 : SERVEUR ET CLIENT ECHO

Q4 "Faites en sorte que votre serveur supporte l'arrêt brutal du client sans planter en gérant correctement les exceptions dans traiterSocketCliente."

Pour que le serveur supporte l'arrêt brutal du client, l'exception levée déclenche la fermeture de la socket cliente. Ainsi, le serveur peut retourner en attente de connexion.

2 EXERCICE 2 : AJOUT DU NOM

Q3 "Modifiez le code de votre client pour envoyer un message au client en cas d'erreur sur le nom."

Si aucun nom n'est renseigné, un message est affiché sur la sortie d'erreur en indiquant le nom par défaut ("Anonymous"). Le client et le serveur continuent normalement leur échange.

3 EXERCICE 4 : GESTIONS DE PLUSIEURS CLIENTS

Q3 "Testez avec telnet que votre serveur fonctionne."

terminal :

```
$ telnet 127.0.0.1 9999
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
```

Eclipse :

```
Serveur en attente.
Le client /127.0.0.1 est connecté.
Serveur en attente.
```

4 EXERCICE 5 : UN VRAI CHAT

Q5 "Testez votre programme avec telnet et différents clients utilisant des charset différents. Le code des clients n'est pas adapté, pourquoi ?"

Les différents clients ne sont pas capable de lire des messages encodés avec un charset différent du leur.

5 CONCLUSION

La plupart des problèmes ont été réglés dans ce TP, grâce notamment au mot-clé *finally* à la fin d'un *try/catch*. Le plus compliqué a été de bien comprendre le fonctionnement des sockets, et de les utiliser en les combinant avec des *Threads* pour être capable d'écouter et d'envoyer en même temps.