

TD sockets : Un serveur qui met des chaînes à l'envers

Objectifs :

Parallélisme via les threads, sans ressource partagée.
Client/serveur.
Communication par sockets.

Exercice 1 : connexion et service

Dans ce TD, vous devrez réaliser votre première application client/serveur. L'idée est de fournir sur le port 1000 du serveur le service d'inversion de chaîne de caractères. Le client qui se connecte envoie une chaîne de caractère, le serveur l'inverse et renvoie la chaîne inversée. La classe `java.lang.StringBuffer` permet avec la méthode `reverse()` de faire ce travail (cf javadoc).

La chaîne peut être le résultat d'une saisie clavier ou, dans un premier temps, codée en dur dans le main du client.

Exemple de communication (en italique les saisies clavier) :

Coté serveur :	Coté client 1:	Coté client 2:
Serveur lancé sur le port 1234	Connecté au serveur appserv.brette.fr port 1234	Connecté au serveur appserv.brette.fr port 1234
Connexion 1 démarrée	Tapez une chaîne de caractères	Tapez une chaîne de caractères
Connexion 2 démarrée	➤ <i>bonjour</i>	➤ _____
Connexion 1 a reçu <i>bonjour</i>	Votre chaîne inversée	
Connexion 1 terminée	<i>ruojnob</i>	

N'oubliez pas que le serveur doit pouvoir gérer plusieurs clients en parallèle.

Remarques :

Vous pouvez vous contenter du coté client d'une classe (celle de l'application) avec son main.

Du coté serveur, il vous faut une classe pour le serveur (Serveur) et une pour le service (ServiceInversion) en plus de la classe de l'application. Le main se contentera d'instancier Serveur au port 1234.

Les applications serveur et client devront être développées dans 2 projets différents, l'idée étant de pouvoir les déployer séparément.

N'oubliez pas de fermer la socket des deux cotés, la connexion pouvant être interrompue d'un coté ou de l'autre.

Exercice 2 : le protocole de communication client/serveur « BTTP »

Afin de faciliter la maintenance évolutive (d'autres services ou des services nécessitant plus d'un échange), on établit un protocole simple à base de questions-réponses entre le service et le client (protocole de transfert de texte Brette : le protocole BTTP)

- 1 ; le service envoie un texte au client ; ce texte implique une réponse du client
2. le client affiche ce texte dans la console et saisie la réponse qui est envoyée au service

Cet échange se poursuit jusqu'à ce que, soit le service ferme la connexion (client jugé indésirable ou autre critère à déterminer), soit le client, via sa réponse, indique qu'il souhaite arrêter les échanges (réponse « N » à la question « Voulez-vous recommencer ? » ou autre critère à déterminer).

Le respect de cet échange permet d'envisager d'écrire un client BTTP totalement général (indépendant du service, des questions posées, etc) à base d'une répétition de cycles

« je reçois une string – je l'affiche »

« je saisie une string, je l'envoie »

Ce client se connectera en tapant une url « BTTP:host:port » - par défaut « BTTP:localhost:1234 »

Exemple pour votre service d'inversion :

1. le premier échange est l'envoi par le service de « Tapez une chaîne de caractères » au client
2. le client affiche cette invite à l'écran, saisie au clavier un texte et l'envoie au service
3. le service envoie le texte inversé et ferme la connexion
4. le client affiche le texte inversé et ferme la connexion

Dans tous les cas, le premier échange se fait dans le sens service → client

Dans le cas présent, la fin des échanges s'est faite à l'initiative du service

Si on envisageait différents services au choix, le premier échange serait l'envoi du menu et la fin des échanges pourrait être une option « Fin » du menu