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

Objectifs :

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

Enoncé :

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 caractère. Le client qui se connecte envoie une chaîne de caractère, le serveur l'inverse et renvoie la chaîne inversée.
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 acsi.iut.univ-paris5.fr 1234	Connecté au serveur acsi.iut.univ-paris5.fr 1234
	>*bonjour*	>*salut*
Connexion 1 reçoit bonjour	ruojnob	tulas
Connexion 2 reçoit salut	> *ca va bien ?*	>
Connexion 1 reçoit ca va bien ?	? neib av ac	… coupure par le client 2 ……..
Connexion 2 interrompue	>	
	ETC …….	

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 l'application. Le main se contentera d'instancier Serveur.

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

Dans un premier temps, testez l'échange d'une string avec un client, puis faites plusieurs clients, enfin des échanges comme sur l'exemple ci-dessus.

N'oubliez pas de fermer la socket des deux cotés, la connexion pouvant être interrompue d'un coté ou de l'autre. Si on suppose qu'à priori, les connexions sont infinies, on pourra tester une IOException pour savoir qu'un des deux cotés à fermé la communication.