Client/Serveur multi-threader, multi-client

# Architecture :

Utilisation des sockets pour la transmission des messages.

Utilisation d’1 thread par client

Il faut 3 classes :

* Serveur :
  + Un ServerSocket
  + Un Socket : va recevoir le socket correspondant à chaque connection au serveur par la méthode accept().
  + Un tableau contenant les thread correspondants aux clients connectés (limité à 10 dans le code)
  + A chaque connexion (accept()), si il reste de la place dans le tableau, on créer un thread qu’on stock dans ce même tableau.
* Client :
  + Un socket
  + Un stream de sortie (PrintStream)
  + Un stream d’entrée (DataInputStream)
  + Un lecteur (BufferedReader)
  + Un Thread qui va lire tout ce qui vient du serveur
* Thread :
  + Un stream de sortie (PrintStream)
  + Un stream d’entrée (DataInputStream)
  + Le socket correspondant au client
  + Le tableau contenant tous les threads des clients connectés
  + Un ID, qui permet de faire des messages personnels (envoyer à 1 seule personne)
  + Permet l’identification (avec un identifiant unique) et l’envoie de message.

# Lancement du programme :

1. Lancer le serveur
2. Lancer 1 client, entrer son identifiant et recommencer jusqu’à atteindre le nombre de client voulu dans les limites du serveur (10)