EPSI – JAVA – 3ème année –Socket, Thread et JDBC

Pour ce TP, vous devez repartir du travail effectué lors de la seconde séance ou de son corrigé.

Exercie1 : Application client/serveur

L'objectif de cet exercice est de transformer votre application Console en une application de type Client/Serveur.

1. Mettez à jour votre application pour permettre à l'application de se lancer en mode serveur quand l'option -s est passée sur la ligne de commande. En l'absence d'option l'application doit se lancer en mode client.
2. Ecrivez une classe `Server` qui écoute les connexions entrantes sur le port 5123
    a. Utilisez une boucle infinie afin de pouvoir recevoir plusieurs connexions
3. Ecrivez une classe `Client` qui se connecte à votre serveur. Testez que la connexion s'établie bien entre l'instance cliente et l'instance serveur
4. Encapsulez dans une classe `AuthentifcationRequest` les informations nécessaires pour authentifier un utilisateur.
    a. Envoyez cette requête depuis le client en sérialisant l'objet `AuthentifcationRequest`
    b. Gérez cette requête côté serveur en désérialisant ce même objet
    c. De la même manière, gérez la réponse à l'aide d'une classe `AuthenticationResponse`
5. Côté serveur, simulez un temps de réponse élevé (quelques secondes) à l'aide de `Thread.sleep(int millisecondes)`
6. Essayez de lancer plusieurs clients simultanément, quel comportement constatez-vous ?
7. Si vous avez du temps, implémentez les requêtes et les réponses permettant de rechercher un taxi, afficher les statistiques etc… Implémentez une classe mère `Request` afin de gérer plus facilement les différentes requêtes

Exercice 2 : Threading

Afin de gérer plus d'un client à la fois, vous devez améliorer le code de votre serveur pour gérer plusieurs connexions simultanées.

1. Ecrivez une classe `RequestHandler` et qui implémente l'interface `Runnable`
    a. Cette classe doit notamment être construite avec un objet xxxRequest et la socket venant d'être créée à la suite de la connexion d'un nouveau client
    b. Déplacez le code de gestion des requêtes de votre classe Serveur vers la classe `RequestHandler`
2. Instanciez un nouveau thread RequestHandler à chaque nouvelle connexion entrante
3. Reprenez le test de l'exercice 1 et vérifiez que votre programme est à présent capable de gérer plusieurs connexions simultanées
4. Si vous avez du temps, essayez de gérer un pool de thread afin de ne pas créer un nouveau thread à chaque connexion entrante, ce qui est une opération couteuse

Exercice 3 : JDBC

Le stockage actuel de vos données est effectué de manière peu industrielle à l'aide d'objets sérialisés sur votre disque local. L'utilisation d'une base de données permettra de simplifier la gestion de ces données.

1. Si ce n'est pas déjà le cas, installer MySQL sur votre poste
2. A l'aide d'un outil graphique (phpMyAdmin, MySQL Workbench) ou en ligne de commande, executez le script de création de la base se trouvant sur le dépôt du cours
3. Dans votre projet, importez le Driver JDBC se trouvant également dans votre dépôt
    a. Build Path > Add external archive
4. Testez le bon fonctionnement du Driver, puis remplacez votre code de sauvegarde sur disque par une sauvegarde en base