INF3405 – Réseaux Informatiques

Hiver 2023

TP No. 1 :

Projet en réseaux informatiques

Gestionnaire de fichier

Groupe 02, 04, 05

2153068 – Selamet Zamanoglu

2148614 – Ilias Bakhbukh

2145194 – Mohamed Chaoui

Soumis à : Bilal Itani, Mehdi Kadi

17 février 2023

Introduction :

Il y a deux objectifs principaux pour ce laboratoire. Il permet à l'étudiant de se familiariser avec l'utilisation des sockets pour la communication client/serveur et d'utiliser les threads pour le développement des applications réseau. Ces objectifs permettent aussi l'évaluation des qualités 4 (conception) et 7 (communication) d'un étudiant en ingénierie. L'étudiant est placé dans le contexte qu'il est fatigué de ne pas avoir suffisamment d'espace de stockage sur les plateformes de stockage cloud courantes, décide de créer sa propre application client-serveur. Il utilise un vieil ordinateur donné par sa grand-mère comme serveur de stockage et se concentre sur le développement de l'application plutôt que sur l'interface graphique.

Présentation :

Pour les commandes, le client lance une commande tant que ce n'est pas la commande « exit » dans une boucle « do while ». On appelle la fonction « readCommand » qui prend comme argument l'input du client. Son but est de séparer en deux parties : nom de la commande et options de la commande (nom de dossier, nom de fichier, etc.). Le client envoie cette commande avec des appels à « out.writeUTF » pour que le « ClientHandler » traite la commande en question dans un switch case.

1. Pour la saisie des paramètres du serveur dans l'interface console du client et celle du serveur, on a utilisé un « Scanner input » pour la saisie des paramètres dans l'interface console. Ensuite, on appelle nos fonctions « readAdress et readPort », qui sont implémentées dans notre classe « Tools », qui prennent l'input en question comme argument.

2. Pour vérifier la validité de l'adresse IP saisie et le numéro de port, on a utilisé la librairie native de Java intitulé « regex » pour la validation d'une adresse IP et le numéro de port dans nos fonctions « ipValidation et portValidation ». On définit une expression régulière selon les formats voulus (IP et port) et on les utilise dans l'appel des fonctions « Pattern.compile » et « matcher » pour la validation de l'input.

3. Pour le téléversement, on appelle la fonction « upload » qui prend en entrée une chaîne de caractères "nameAndFormat" représentant le nom et le format du fichier à écrire, ainsi qu'un objet "DataInputStream" qui est utilisé pour lire les données d'entrée. La méthode crée un nouvel objet "File" à partir du nom de fichier et du répertoire courant, puis ouvre un flux de sortie pour écrire les données dans le fichier. La méthode utilise une boucle "while" pour lire les données d'entrée à partir du flux de données, puis écrit les données lues dans le fichier. La boucle s'arrête lorsque la fin du flux est atteinte ou si une erreur de délai d'attente se produit. Enfin, la méthode ferme le flux de sortie et renvoie "true" si l'opération s'est déroulée avec succès ou "false" s'il y a eu une erreur. De la même manière, pour le téléchargement, on appelle la fonction « download » qui prend en paramètre le nom du fichier et un objet DataOutputStream pour écrire les données du fichier à transmettre. Elle ouvre le fichier, lit son contenu en octets dans un tableau de bytes, puis écrit les données sur le flux de sortie jusqu'à la fin du fichier. Si le nom de fichier est vide, la fonction affiche un message d'erreur. La fonction gère les exceptions liées à l'entrée/sortie.

4. Pour la commande « cd », on appelle la fonction « cd » avec l'argument « commandOption qui correspond à la 2e partie d'un input du client. Elle réassigne une variable File qui s'appelle « currentFile » à un nouveau File. On ajoute au chemin de l'ancien File la partie « commandOption ». Pour finir, on affiche au client le message « "Vous êtes dans le dossier "+currentFile » qui montre le chemin absolu du répertoire courant.

5. Pour la commande « ls », on appelle la fonction « ls » utilise la méthode "listFiles()" sur la variable « currentFile » pour obtenir un tableau des fichiers et dossiers, et utilise une boucle "for" pour parcourir le tableau. Pour chaque élément du tableau, la méthode utilise "isDirectory()" et "isFile()" pour déterminer s'il s'agit d'un dossier ou d'un fichier, puis ajoute la chaîne appropriée à la chaîne de caractères "returnString". La méthode renvoie finalement "returnString" contenant la liste des fichiers et dossiers.

6. Pour la commande « mkdir », on appelle la fonction « mkdir » (la nôtre) qui prend comme argument "commandOption" représentant le nom du dossier à créer.

On crée un nouvel objet « File » en utilisant le répertoire courant et commandOption. La fonction appelle la méthode native de java « mkdir » pour créer le dossier voulu et retourne un booléen selon le succès de l'opération.

7. Après que le client fait une commande « exit », le code sort de la boucle « do while » et on appelle « input.close() » pour fermer le scanner des saisies du client dans l'interface de la console et « socket.close() » pour fermer le socket qui permet la communication entre un client et le serveur.

8. Pour l'affichage en temps réel des demandes à traiter, on appelle la fonction «System.out.println » pour les afficher, mais le client envoie les commandes au serveur avec la fonction « out.writeUTF ».

Difficultés rencontrées :

Envoyer une image 4K était compliqué, car on ne pouvait pas envoyer toute l'image en 1 coup. On a dû envoyer l'image en plusieurs paquets de 8kb. On a aussi eu de la difficulté par rapport à la duplication de code et de lisibilité. On a dû créer une classe « Tools » pour séparer en fonctions les solutions pour le tp.

Critiques et améliorations :

On n'a pas vraiment de critiques et améliorations par rapport au laboratoire. Nous avons trouvé ce laboratoire assez utile et intéressant à faire.

Conclusion :

Ce laboratoire sort des tp ordinaires auxquels ont été habitué. On apprit à faire communiquer un client et un serveur grâce à la réseautique avec les sockets. Nos attentes ont été comblées.