INF3405 –Réseaux informatiques

Automne 2023

TP No. 1

Groupe 02

2129466 – Pablo Cabale

2152451 – Raphael Ramanitranja

2149345 - Hamza Drif

Soumis à : Mehdi Kadi

Date de remise: 20 octobre 2023

Introduction

Le travail pratique consistait à développer une application permettant d'appliquer un filtre de Sobel à n'importe quelle image importée. Pour cela, nous avons dû mettre en place un serveur et un client qui communiquent entre eux, ainsi qu'une interface console pour les utilisateurs. L'application devait être capable de recueillir et valider les informations entrées au moment du lancement, en plus de gérer la connexion de plusieurs utilisateurs simultanément. Le serveur devait également stocker les identifiants des utilisateurs dans une base de données locale. L'objectif principal de ce travail était de nous familiariser avec les échanges entre le client et le serveur à l'aide des sockets, ainsi que le développement d'applications réseau, en mettant en œuvre la gestion des threads.

Présentation

Nous avons initialement regroupé les différentes fonctionnalités de notre application afin de créer des fichiers appropriés remplissant tous une fonction spécifique.

Client.java :

Cette classe, c'est celle que les clients auront accès. Avec cette classe, les utilisateurs pourront avoir une connexion avec le serveur, pour ensuite appliquer le filtre à leur image.

User.java :

Le fichier User.java contient la classe User qui à été instanciée afin de manipuler notre base de donnée locale (JSON) dans le fichier ClientHandler.java.

ClientHandler.java:

Cette classe s'occupe du service du serveur envers le client C'est-à-dire, la vérification de l'utilisateur et le traitement de l'image.

Serveur.java :

Cette classe s'occupe de la responsabilité de créer une connexion dans laquelle les clients peuvent se connecter pour ensuite leur rendre service.

Validation.java :

Le fichier Validation.java contient toute la logique permettant de vérifier et valider les informations entrées par les utilisateurs comprenant l'adresse ip et le port de l'ordinateur.

L'adresse ip doit être composée de 4 octets et elle doit également être écrite avec la bonne syntaxe tandis que le port doit être compris dans un intervalle de 5000 à 5050 inclusivement.

ImageTreatment.java :

La classe ImageTreatment.java s'occupe de tout le transfert d'image entre le client et le serveur ainsi que du traitement et de la sauvegarde de l'image. C'est ce fichier qui contient aussi l'algorithme de Sobel.

Nous avons finalement créé deux exécutables (Serveur.jar et Client.jar) à partir de nos fichiers .java. Il suffit de run le serveur avec la commande java -jar Serveur.jar et ensuite run le client avec la commande java -jar Client.jar.

Bibliothèques ajoutées :

1. Nous avons ajouté la bibliothèque ImageIO pour pouvoir écrire une BufferedImage en ByteArray afin de pouvoir renvoyer l'image traitée au client à partir du serveur. Ça facilitait la manipulation des BufferedImage.

2. Nous avons ajouté la bibliothèque GSON. Cette bibliothèque nous permet la gestion de fichiers JSON, avec lequel on utilise pour gérer la base de données des utilisateurs. GSON permet entre autres de créer, modifier et écrire un fichier JSON.

Difficultés rencontrées

La principale difficulté à laquelle nous avons fait face était au niveau de la division des tâches. On ne savait pas vraiment par où commencer car nous étions incertains par rapport à ou coder en premier et Les objectifs du travail pratique étaient clairs mais il fallait ensuite déterminer le nombre de fichiers que nous allions créer ainsi que leur contenu et la fonction de chacun. Nous avons donc décortiqué l'énoncé pour identifier les fonctionnalités principales et les regrouper dans de différentes classes côté serveur et côté client. Une autre embûche était plus liée au code dans le sens ou bien qu'il n'y avait relativement pas beaucoup de code à écrire, il fallait passer un peu de temps pour comprendre le code initial donné dans le projet de base et aussi apprendre les nouvelles libraires comme Socket, Thread et Gson.

Critiques et améliorations

Nous avons trouvé que le contenu et les objectifs de travail pratique n'avait pas beaucoup de lien avec la matière vue en cours. Les seules diapositives pertinentes étaient les dernières vers la fin du premier chapitre qui présentaient les fichiers initiaux du projet mais à part ça il fallait chercher nous-mêmes afin de comprendre le code. De plus, nous avons eu un peu de mal à comprendre pour quels formats d'image il fallait écrire le code. Celui que nous remettons marche seulement pour les images .jpg, mais il aurait été facile de modifier cela pour inclure d'autres formats d'images.

Conclusion

En résumé, ce laboratoire nous a permis d'en apprendre plus sur les différents aspects de la programmation et de la communication réseaux ainsi que la communication client-serveur grâce aux sockets et aux threads.