Le but de ce TP est de développer un *framework* permettant d'échanger des données entre deux ordinateurs distants. En réalité, il s'agit de deux applications différentes (Une pour le serveur et l'autre cliente).

Application Serveur

Il s'agit de développer une application qui jouera le rôle d'un serveur log permet d'enregistrer les messages provenant des utilisateurs distants. Les messages reçus seront affichés d'abord dans un JTextArea et ensuite stockés dans un fichier log.txt sur le serveur.

Avant de commencer, créez un nouveau projet.

Pour pourvoir accepter des connexions on utilisera l'objet *ServerSocket* comme suit

```
ServerSocket server = new ServerSocket(port);
```

La classe *ServerSocket* permet d'obtenir un écouteur de connexion TCP. Les demandes de connexion peuvent provenir de la classe *Socket* (que vous devez ajouter par la suite à votre application cliente).

Pour accepter les connexions, il s'agit d'appeler la méthode *accept()*de la classe *ServerSocket.* La méthode *accept()* est une méthode bloquante et retourne un objet de type Socket, une fois une connexion avec un Client est établie.

```
Socket client = server.accept();
```

Utilisez les objets *InputStreamReader* et *BufferedReader* pour récupérer les messages envoyés par l'utilisateur.

```
InputStreamReader is= new InputStreamReader(client.getInputStream());

BufferedReader reader = new BufferedReader(is);
```

Faites le nécessaire pour stocker les informations dans un fichier texte et les afficher dans un *JTextArea*. (Attention au blocage de la fenêtre, Proposez une solution).

Client Java

Créez un nouveau projet qui vous permet de vous connecter au serveur et lui envoyer des messages.

Il s'agit de créer un objet de type Socket.

```
Socket socket = new Socket(adresse du serveur, numéro du port);
```

Votre application doit permettre à un utilisateur d'envoyer des messages saisis à travers l'interface graphique.

Pour envoyer des messages, vous allez utiliser une instance de la classe PrintWriter et lui passer en paramètre l'OutputStream du socket (socket.getOutputStream())

Ainsi, pour envoyer du texte il suffit d'appeler la méthode println du printWriter .

Exercice

Complétez votre application serveur pour la rendre capable d'accepter plusieurs demandes de connexion simultanément.

Bon travail