

---

# Réseaux - Travaux Pratiques

## Programmation Client-Serveur

---

Le but du TP est de programmer deux applications client/serveur : une utilisant le protocole UDP et l'autre le protocole TCP. Vous devez programmer chacune de ces applications en Java.

- Le protocole UDP utilise un mode sans connexion ; chaque échange se fait de manière indépendante.
- Le protocole TCP utilise un mode d'échange fiable avec connexion ; avant d'échanger les messages, les deux correspondants établissent une liaison qui est utilisée pour l'échange des messages.

### 1 Avec le protocole UDP

On souhaite réaliser un service (en utilisant le protocole UDP) qui permet de renvoyer en majuscule à un client toute chaîne de caractères envoyée au serveur.

1. Ecrire le serveur à partir du code fourni ci-dessous.
2. Ecrire le client à partir du code fourni ci-dessous.
3. Effectuer les tests.
4. Gérer les exceptions (au lieu de les laisser simplement passer).

```
import java.net.*;

public class UDPServer {
    public static void main(String[] args) throws Exception {
        if (args.length != 1) {
            System.out.println("Usage: java UDPServer <port>");
            System.exit(1);
        }
        int serverPort = Integer.parseInt(args[0]);

        // On cree une socket en ecoute sur le port specifie
        DatagramSocket datagramSocket = ...

        byte[] data = new byte[1024];

        while (true) {
            // On attend un datagramme
            DatagramPacket receivePacket = ...
            ...

            // Recuperation du contenu
            InetAddress clientAddress = ...
            int clientPort = ...
            byte[] outData = new String(data).toUpperCase().getBytes();
            // NB: data est equivalent a receivePacket.getData()

            // on envoie le packet
            DatagramPacket sendPacket = ...
            ...
        }
    }
}
```

```

import java.net.*;
import java.util.Scanner;

public class UDPClient {
    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.out.println("Usage : java UDPClient <machine> <port>");
            System.exit(1);
        }
        InetAddress serverAddress = ...
        int serverPort = ...

        // On cree une socket en laissant au systeme choisir un numero de port
        DatagramSocket datagramSocket = ...

        Scanner scanner = new Scanner(System.in);
        System.out.println("Entrez une phrase");
        byte[] data = scanner.nextLine().getBytes();

        // On envoie le datagramme
        DatagramPacket outDatagram = ...
        ...

        // NB : je peux reutiliser le meme tableau de byte car je sais que le serveur me renvoie un tableau de meme taille
        DatagramPacket inDatagram = ...
        ...
        System.out.println("Receive :" + new String(inDatagram.getData()));

        // on ferme
        ...
    }
}

```

## 2 Avec le protocole TCP

On veut réaliser un petit “chat” entre simplement deux individus. Ces deux individus “parlant” à tour de rôle, on a décidé d’utiliser TCP (sinon, de manière plus classique, c’est UDP qui est utilisé pour ce genre de services).

1. Ecrire le serveur à partir du code fourni ci-dessous.
2. Tester le serveur en utilisant simplement telnet. Comment fait-on ?
3. Ecrire le client à partir du code fourni ci-dessous.
4. Effectuer les tests.
5. Gérer les exceptions (au lieu de les laisser simplement passer).

```

import java.io.*;
import java.net.*;
import java.util.Scanner;

public class TCPServer {
    public static void main(String[] args) throws Exception {
        if (args.length != 1) {
            System.out.println("Usage : java TCPServer <port>");
            System.exit(1);
        }
        int serverPort = ...

        Scanner scanner = new Scanner(System.in);

        // On cree un serveur socket en precisant le numero de port d'ecoute du serveur
        ServerSocket serverSocket = new ServerSocket(serverPort);

        // On attend une demande de connexion de la part d'un client
        Socket socket = ...

        // On recupere les flux d'entree/sortie

```

```

BufferedReader in = ...
PrintWriter out = ...

out.println("Welcome, start the discussion");
while (true) {
    // dialogue à tour de rôle, exit pour finir
}
// on ferme
}
}

import java.io.*;
import java.net.*;
import java.util.Scanner;

public class TCPClient {
    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.out.println("Usage : java UDPCClient <machine> <port>");
            System.exit(1);
        }
        InetAddress serverAddress = ...
        int serverPort = ...

        Scanner scanner = ...

        // On cree une socket en precisant l'adresse du serveur (machine) et le numero de port d'ecoute du serveur
        Socket socket = ...

        // On recupere les flux d'entree/sortie
        BufferedReader in = ...
        PrintWriter out = ...

        while (true) {
            // dialogue à tour de rôle, exit pour finir
        }
        // on ferme
    }
}

```