Rapport TP3 réseau

RIDA VERDU

Novembre 2022

Exercice 2

```java
// ScannerTCPv1.java
import java.util.*;
import java.lang.*;
import java.net.*;
import java.io.*;

public class ScannerTCPv1 {

    public static int []openPorts = new int[65535];

    public static boolean isPortAvailable(int port){
        try {
            (new Socket("localhost", port)).close();
            return true;
        }
        catch(Exception e){
            return false;
        }
    }

    public static void main(String []args){
        int [] portList = java.util.stream.IntStream.rangeClosed(1024,
            openPorts.length).toArray();

        System.out.printf("Voici les ports ouverts :\n");
        for(int i = 0; i < portList.length; i++ ){
            if(isPortAvailable(i)){
                openPorts[i] = portList[i];
                System.out.printf("%d Open\n", i);
            }
        }
    }
}
```

```java
// ScannerTCPv2.java
import java.util.*;
import java.lang.*;
import java.net.*;
import java.io.*;

public class ScannerTCPv2 {

    public static int []openPorts = new int[81];
    public static String remoteHost;

    public static boolean isRemotePortAvailable(int port){
        try {
            (new Socket(remoteHost, port)).close();
            return true;
        }
        catch(IOException e){
            return false;
        }
    }

    public static void main(String []args){
        int [] portList = java.util.stream.IntStream.rangeClosed(0,
            openPorts.length).toArray();
        remoteHost = "149.62.158.51";
        System.out.printf("Voici les ports ouverts :\n");

        for(int i = 0; i < portList.length; i++ ){
            if(isRemotePortAvailable(i)){
                openPorts[i] = portList[i];
                System.out.printf("%d Open\n", i);
            }
        }
    }
}
```

```java
// ScannerUDPv1.java
import java.util.*;
import java.lang.*;
import java.net.DatagramSocket;

public class ScannerUDPv1 {

    public static int []openPorts = new int[65535];

    public static boolean isRemotePortAvailable(int port){
        try {
            (new DatagramSocket(port)).close();
```

```
            return true;
        }
        catch(Exception e){
            return false;
        }
    }

    public static void main(String []args){
        int [] portList = java.util.stream.IntStream.rangeClosed(1024,
            openPorts.length).toArray();

        System.out.printf("Voici les ports ouverts :\n");
        for(int i = 0; i < portList.length; i++ ){
            if(isRemotePortAvailable(i)){
                openPorts[i] = portList[i];
                System.out.printf("%d Open\n", i);
            }
        }
    }
}
```

Exercice 3

```
// HTTPdServerV1.java
import java.io.*;
import java.net.*;

public class HTTPdServerV1 {

    public static void main(String[] args){

        int port = Integer.parseInt(args[0]);
        ServerSocket serverSocket = null;
        try {
            serverSocket = new ServerSocket(port);
        }
        catch(IOException e){
            System.err.println("Le port spcifi n'est pas disponible.");
            return;
        }

        System.err.println("Instance du server est lanc sur le port " +
            port);

        try{
            while (true) {
                Socket clientSocket = serverSocket.accept();
```

```
                System.err.println("Client  dtct ");

                BufferedWriter out = new BufferedWriter(new
                    OutputStreamWriter(clientSocket.getOutputStream()));


                out.write("HTTP/1.0 200 OK\r\n");
                out.write("Server: HandMade ESIEA\r\n");
                out.write("Content-Type: text/html\r\n");
                out.write("Content-Length: 59\r\n");
                out.write("\r\n");
                out.write("<!DOCTYPE html>");
                out.write("<html><head><title>tete de
                    page</title></head><body><p>Super page
                    Internet</p></body></html>");

                System.err.println("Fin de communication.");
                out.close();
                clientSocket.close();
            }
        }
        catch(Exception e) {
            System.err.println("Fermeture de la session du server");
            try {
                serverSocket.close();
            }
            catch(IOException y){
                System.err.println("Erreur lors de la fermeture du
                    server");
            }

        }

    }
}
```

Exercice 4

```
// HTTPdServerV2.java
import java.io.*;
import java.net.*;
import java.nio.charset.StandardCharsets;

public class HTTPdServerV2 {

    public static void main(String[] args){
```

```
int port = Integer.parseInt(args[0]);
ServerSocket serverSocket = null;
try {
    serverSocket = new ServerSocket(port);
}
catch(IOException e){
    System.err.println("Le port spcifi n'est pas disponible.");
    return;
}

System.err.println("Instance du server est lanc sur le port " +
     port);

try{
    while (true) {
        Socket clientSocket = serverSocket.accept();
        System.err.println("Client  dtct ");

        BufferedWriter out = new BufferedWriter(new
            OutputStreamWriter(clientSocket.getOutputStream()));

        InputStream indexFile = new FileInputStream("index.html");


        byte[] buffer=new byte[4096];
        int size;
        String s;

        out.write("HTTP/1.0 200 OK\r\n");
        out.write("Server: HandMade ESIEA\r\n");
        out.write("Content-Type: text/html\r\n");
        out.write("Content-Length: 59\r\n");
        out.write("\r\n");

        while ((size = indexFile.read(buffer)) > 0){
            s = new String(buffer, StandardCharsets.UTF_8);
            out.write(s, 0, size);
        }
        indexFile.close();

        System.err.println("Fin de communication.");
        out.close();

        clientSocket.close();
    }
}
catch(Exception e) {
    System.err.println("Fermeture de la session du server");
    try {
        serverSocket.close();
```

```
            }
            catch(IOException y){
                System.err.println("Erreur lors de la fermeture du
                    server");
            }

        }

    }
}
```

```
// Index.html
<!DOCTYPE html>
<html>
    <head>
        <title>tete de page</title>
    </head>
    <body>
        <p>Super page Internet</p>
    </body>
</html>
```

Exercice 5

```
// HTTPdServerV3.java
import java.io.*;
import java.net.*;

public class HTTPdServerV3 {

    public static void main(String[] args){

        int port = Integer.parseInt(args[0]);
        ServerSocket serverSocket = null;
        try {
            serverSocket = new ServerSocket(port);
        }
        catch(IOException e){
            System.err.println("Le port spcifi n'est pas disponible.");
            return;
        }

        System.err.println("Instance du server est lanc sur le port " +
             port);

        try{
```

```
            while (true) {

                Socket clientSocket = serverSocket.accept();
                System.err.println("Client  dtct ");


                BufferedReader in = new BufferedReader(new
                    InputStreamReader(clientSocket.getInputStream()));
                BufferedWriter out = new BufferedWriter(new
                    OutputStreamWriter(clientSocket.getOutputStream()));


                String s;
                while ((s = in.readLine()) != null) {
                    System.out.println(s);
                }
                out.write("HTTP/1.0 200 OK\r\n");
                out.write("Server: HandMade ESIEA\r\n");
                out.write("Content-Type: text/html\r\n");
                out.write("Content-Length: 59\r\n");
                out.write("\r\n");
                out.write("<!DOCTYPE html>");
                out.write("<html><head><title>tete de
                    page</title></head><body><p>Super page
                    Internet</p></body></html>");

                System.err.println("Fin de communication.");
                out.close();
                in.close();
                clientSocket.close();
            }
        }
        catch(Exception e) {
            System.err.println("Fermeture de la session du server");
            try {
                serverSocket.close();
            }
            catch(IOException y){
                System.err.println("Erreur lors de la fermeture du
                    server");
            }

        }

    }
}
```

Exercice 6

```java
// HTTPdServerV4.java
import java.io.*;
import java.net.*;
import java.nio.charset.StandardCharsets;

public class HTTPdServerV4 {

    public static void main(String[] args){

        int port = Integer.parseInt(args[0]);
        ServerSocket serverSocket = null;
        try {
            serverSocket = new ServerSocket(port);
            serverSocket.setSoTimeout(10000);
        }
        catch(IOException e){
            System.err.println("Le port spcifi n'est pas disponible.");
            return;
        }

        System.err.println("Instance du server est lanc sur le port " +
             port);

        try(
            Socket clientSocket = serverSocket.accept();
            BufferedWriter out = new BufferedWriter(new
                 OutputStreamWriter(clientSocket.getOutputStream()));
            BufferedReader in = new BufferedReader(new
                 InputStreamReader(clientSocket.getInputStream()));
        )
        {
            while (true) {
                System.err.println("coute...");

                String filename;
                String inReq = in.readLine();
                if(inReq.split("/")[1].startsWith(" ")){
                    filename = "index.html";
                }else {
                    filename = inReq.split("/")[1].split(" ")[0];
                }
                System.out.printf("Nom du fichier charg :%s \n",
                     filename);
                InputStream indexFile = new FileInputStream(filename);

                byte[] buffer=new byte[4096];
                int size;
```

```
                String fileString;

                out.write("HTTP/1.0 200 OK\r\n");
                out.write("Server: HandMade ESIEA\r\n");
                out.write("Content-Type: text/html\r\n");
                out.write("Content-Length: 59\r\n");
                out.write("\r\n");

                while ((size = indexFile.read(buffer)) > 0){
                    fileString = new String(buffer,
                        StandardCharsets.UTF_8);
                    out.write(fileString, 0, size);
                }
                indexFile.close();

                System.err.println("Fin de communication.");
                in.close();
                out.close();
                clientSocket.close();
            }
        }
        catch(Exception e) {
            System.err.printf("Fermeture de la session du server : \n%s",
                e);
            try {
                serverSocket.close();
            }
            catch(IOException y){
                System.err.println("Erreur lors de la fermeture du
                    server");
            }

        }

    }
}
```

Exercice 7

```
// HTTPdServerV5.java
import java.io.*;
import java.net.*;
import java.nio.charset.StandardCharsets;
import java.util.concurrent.ThreadLocalRandom;

public class HTTPdServerV5 {
```

```
public static void main(String[] args){

    int port = Integer.parseInt(args[0]);
    ServerSocket serverSocket = null;
    try {
        serverSocket = new ServerSocket(port);
        serverSocket.setSoTimeout(10000);
    }
    catch(IOException e){
        System.err.println("Le port spcifi n'est pas disponible.");
        return;
    }

    System.err.println("Instance du server est lanc sur le port " +
         port);

    try(
        Socket clientSocket = serverSocket.accept();
        BufferedWriter out = new BufferedWriter(new
             OutputStreamWriter(clientSocket.getOutputStream()));
        BufferedReader in = new BufferedReader(new
             InputStreamReader(clientSocket.getInputStream()));
    )
    {
        while (true) {
            System.err.println("coute...");

            String filename;
            String inReq = in.readLine();
            if(inReq.split("/")[1].startsWith(" ")){
                int randomNum = ThreadLocalRandom.current().nextInt(0,
                     1 + 1);
                if(randomNum == 1){
                    filename = "chromium.html";
                }else{
                    filename = "iceweasel.html";
                }
            }else {
                filename = inReq.split("/")[1].split(" ")[0];
            }
            System.out.printf("Nom du fichier charg :%s \n",
                 filename);
            InputStream indexFile = new FileInputStream(filename);

            byte[] buffer=new byte[4096];
            int size;
            String fileString;

            out.write("HTTP/1.0 200 OK\r\n");
            out.write("Server: HandMade ESIEA\r\n");
```

```
                out.write("Content-Type: text/html\r\n");
                out.write("Content-Length: 59\r\n");
                out.write("\r\n");

                while ((size = indexFile.read(buffer)) > 0){
                    fileString = new String(buffer,
                        StandardCharsets.UTF_8);
                    out.write(fileString, 0, size);
                }
                indexFile.close();

                System.err.println("Fin de communication.");
                in.close();
                out.close();
                clientSocket.close();
            }
        }
        catch(Exception e) {
            System.err.printf("Fermeture de la session du server : \n%s",
                e);
            try {
                serverSocket.close();
            }
            catch(IOException y){
                System.err.println("Erreur lors de la fermeture du
                    server");
            }

        }

    }
}
```

```
// chromium.html
<!DOCTYPE html>
<html>
    <head>
        <title>tete de page</title>
    </head>
    <body>
        <p>CHROMIUM</p>
    </body>
</html>
```

```
// iceweasel.html
<!DOCTYPE html>
<html>
    <head>
```

```
        <title>tete de page</title>
    </head>
    <body>
        <p>ICEWEASEL</p>
    </body>
</html>
```