Übungsblatt 6 (Musterlösung)

In den folgenden Aufgaben geht es darum das hier angegebene (sehr einfache) Protokoll zu implementieren:
Protokoll:

1) Nach dem Verbindungsaufbau sendet der Client:
   „HALLO: Ich bin der Client von <Ihr Vorname>.“
2) Nachdem der Server ein „HALLO“ empfangen hat, sendet er:
   „HALLO: Ich bin der Server von <Ihr Vorname>.“
3) Der Client sendet danach:
   „WIEDERSEHEN: Bis zum naechsten Mal.“.
4) Der Server sendet danach auch:
   „WIEDERSEHEN: Bis zum naechsten Mal.“.
5) Der Server schließt die Verbindung und wartet auf den nächsten Client.

1. TCP-Server

a) Neues Projekt:
   - Legen Sie in Eclipse ein neues Java-Projekt mit Namen „Netzwerke“ an.
   - Erzeugen sie in diesem Projekt ein Package „sockets“.
   - Legen Sie in diesem Package eine Klasse „SimpleServerTCP“ an.
   - Kopieren Sie das weiter unten vorgegebene Code-Fragment in diese Klasse.
   - Der Server soll auf Port 4444 lauschen.
   - Der Server soll mit dem Befehl System.out.printf() eine Meldung auf der Konsole ausgeben
     - wenn er gestartet ist,
     - wenn er eine neue Verbindung hat,
     - und wenn er etwas empfangen hat.
   - Der Server soll Clients hintereinander abarbeiten können, ohne dass er sich zwischendurch beendet.

b) Implementieren Sie oben dargestellte Protokoll mit einem TCP-Socket.
   - Benutzen Sie beim Senden immer „/r/n“ um beim Senden einen Zeilenumbruch hinzuzufügen.
   - Achten Sie darauf, dass am Ende alle Streams und Sockets mit dem Befehl close() geschlossen werden.

c) Testen Sie Ihren Server mit Telnet. Benutzen Sie dazu die Software Putty.
   - Falls Putty nicht installiert ist, können Sie eine „putty.exe“ von http://www.putty.org/ herunterladen.
   - Einstellungen:
     - “RAW” (nicht Telnet)
     - Ihre eigene IP ist 127.0.0.1 oder “localhost”
     - Port = 4444

d) Testen Sie den Server eines anderen Studenten mit Telnet
   - Die IP eines Rechners kann man mit dem Befehl „ipconfig“ ermitteln (unter Linux/Mac: „ifconfig“)

Software-Fragment (siehe iLearn)

```
package sockets;

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

public class SimpleServerTCP {

	public static void main(String args[]) throws Exception {
		System.out.printf("Server gestartet\n");

		// Erzeugen Sie hier die notwendigen Sockets
		// Rufen Sie dann die Methode "protocol()" auf,
		// um einen Client abzuarbeiten
	}

	public static void protocol(Socket socket) throws Exception {

		// Hier wird ein einzelner Client bedient
	}
}
```

Lösung

```
package sockets;

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

public class SimpleServerTCP {

	public static void main(String args[]) throws Exception {
		System.out.printf("Server gestartet\n");
		ServerSocket serverSocket = new ServerSocket(4444);

		while (true) {
			Socket socket = serverSocket.accept();
			System.out.printf("Verbunden mit Client!\n");
			protocol(socket);
		}
	}

	public static void protocol(Socket socket) throws Exception {
		Scanner inFromClient = new Scanner(socket.getInputStream());
		PrintStream outToClient = new PrintStream(socket.getOutputStream());

		String line = inFromClient.nextLine();
		System.out.printf("Empfangen: %s\n", line);

		outToClient.printf("HALLO: Ich bin der Server von Andreas.\r\n");

		line = inFromClient.nextLine();
```

```
            System.out.printf("Empfangen: %s\n", line);

            outToClient.printf("WIEDERSEHEN: Bis zum naechsten Mal.\r\n");

            outToClient.close();
            inFromClient.close();
            socket.close();
      }
}
```

2. TCP-Client

a) Anlegen des Client:
   - Legen Sie im Package „sockets“ eine Klasse „SimpleClientTCP“ an.
   - Kopieren Sie das weiter unten vorgegebene Code-Fragment in diese Klasse.
   - Der Client soll mit dem Befehl System.out.printf() eine Meldung auf der Konsole ausgeben
     - wenn er gestartet ist
     - und wenn er etwas empfangen hat.

b) Implementieren Sie oben dargestellte Protokoll mit einem TCP-Socket.
c) Testen Sie Ihren Client mit Ihrem eigenen Server.
d) Testen Sie Ihren Client mit den Servern anderer Studenten.

Software-Fragment (siehe iLearn)

```
package sockets;

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

public class SimpleClientTCP {
      public static void main(String args[]) throws Exception {
            System.out.printf("Client gestartet\n");

            Scanner inFromUser = new Scanner(System.in);
            System.out.printf("Bitte IP-Adresse des Servers eingeben: ");
            String serverIP = inFromUser.nextLine();
            inFromUser.close();

            // Erzeugen Sie hier die notwendigen Sockets
            // Rufen Sie dann die Methode "protocol()" auf,
            // um mit dem Server zu kommunizieren.
      }

      public static void protocol(Socket socket) throws Exception {

            // Kommunikation mit dem Server
      }
}
```

Lösung

```
public class SimpleClientTCP {
	public static void main(String args[]) throws Exception {
		System.out.printf("Client gestartet\n");

		Scanner inFromUser = new Scanner(System.in);
		System.out.printf("Bitte IP-Adresse des Servers eingeben: ");
		String serverIP = inFromUser.nextLine();
		inFromUser.close();

		Socket socket = new Socket(serverIP, 4444);
		protocol(socket);
	}

	public static void protocol(Socket socket) throws Exception {
		PrintStream outToServer = new PrintStream(socket.getOutputStream());
		Scanner inFromServer = new Scanner(socket.getInputStream());

		outToServer.printf("HALLO: Ich bin der Client von Andreas.\r\n");

		String line = inFromServer.nextLine();
		System.out.printf("Empfangen: %s\n", line);

		outToServer.printf("WIEDERSEHEN: Bis zum naechsten Mal.\r\n");

		line = inFromServer.nextLine();
		System.out.printf("Empfangen: %s\n", line);

		outToServer.close();
		inFromServer.close();
		socket.close();
	}
}
```

3. UDP-Client und Server

Realisieren Sie den Client und den Server aus den Aufgaben 1 und 2 als UDP-Socket, mit Server-Port 5555.
Tipp: Um die Empfangenen Daten auszugeben benutzen Sie bitte den Befehl:

- System.out.printf("Empfangen: %s\n", new String(receivePacket.getData()));

Lösung

```
package sockets;

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

public class SimpleServerUDP {

	public static void main(String args[]) throws Exception {
		System.out.printf("Server gestartet\n");

		while (true) {
			protocol();

		}
	}

	public static void protocol() throws Exception {
		byte[] receiveData = new byte[1024];
		byte[] sendData = new byte[1024];

		DatagramSocket socket = new DatagramSocket(5555);

		DatagramPacket receivePacket = new DatagramPacket(receiveData,
		receiveData.length);
		socket.receive(receivePacket);
		System.out.printf("Empfangen: %s\n", new String(receivePacket.getData()));

		InetAddress IPAddress = receivePacket.getAddress();
		int port = receivePacket.getPort();

		sendData = "HALLO: Ich bin der Server von Andreas.\r\n".getBytes();
		DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,
		IPAddress, port);
		socket.send(sendPacket);

		receivePacket = new DatagramPacket(receiveData, receiveData.length);
		socket.receive(receivePacket);
		System.out.printf("Empfangen: %s\n", new String(receivePacket.getData()));

		sendData = "WIEDERSEHEN: Bis zum naechsten Mal.\r\n".getBytes();
		sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress,
		port);
		socket.send(sendPacket);

		socket.close();
	}
}
```

```
package sockets;

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

public class SimpleClientUDP {
	public static void main(String args[]) throws Exception {
		System.out.printf("Client gestartet\n");

		Scanner inFromUser = new Scanner(System.in);
		System.out.printf("Bitte IP-Adresse des Servers eingeben: ");
		String serverIP = inFromUser.nextLine();
		inFromUser.close();

		InetAddress IPAddress = InetAddress.getByName(serverIP);
		protocol(IPAddress);
	}

	public static void protocol(InetAddress IPAddress) throws Exception {
		DatagramSocket socket = new DatagramSocket();
		int port = 5555;

		byte[] sendData = new byte[1024];
		byte[] receiveData = new byte[1024];

		sendData = "HALLO: Ich bin der Client von Andreas.\r\n".getBytes();
		DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,
		IPAddress, port);
		socket.send(sendPacket);

		DatagramPacket receivePacket = new DatagramPacket(receiveData,
		receiveData.length);
		socket.receive(receivePacket);
		System.out.printf("Empfangen: %s\n", new String(receivePacket.getData()));

		sendData = "WIEDERSEHEN: Bis zum naechsten Mal.\r\n".getBytes();
		sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress,
		port);
		socket.send(sendPacket);

		receivePacket = new DatagramPacket(receiveData, receiveData.length);
		socket.receive(receivePacket);
		System.out.printf("Empfangen: %s\n", new String(receivePacket.getData()));

		socket.close();
	}
}
```

Viel Erfolg !!!