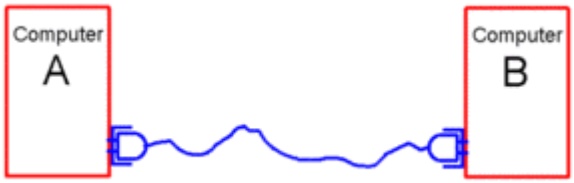
Übungsblatt 15 Java

In den RFCs (Request for Comments) ist ein Socket als ein Tupel aus Ziel- und Quell-IP-Adresse, Ziel- und Quell-Port und Netzwerkprotokoll beschrieben.



Der Ziel-Port wählt einen bestimmten Dienst aus, der Quell-Port wird zufällig generiert. Auf diese Weise sind mehrere Verbindungen zu einem Dienst zwischen zwei Rechnern (hosts) möglich.
Wir benutzen ein Socket (engl. Steckdose, Buchse) als eine Software-Schnittstelle zum Zweck der Nutzung einer bidirektionalen Verbindung zwischen Computern. Diese Schnittstelle wird in objektorientierten Sprachen als ein spezielles Objekt auftreten, dass die Handhabung erleichtert.

Vergleich mit einer Telefonverbindung
Man kann das Socket-Konzept mit einer Telefonverbindung vergleichen. Das Telefongerät entspricht dem Socket. Die IP-Adressen wären die Telefonnummern, die Ports wären die Nummern der Durchwahlverbindungen. Um einen Telefon-Dienst in Anspruch nehmen zu können, muss man also die Nummer samt Durchwahl des Dienstes kennen. Gewöhnlich meldet sich 'der Dienst' nach einer Anwahl und nach Austausch von 'Anmeldeinformationen' steht die Verbindung solange bis einer der beiden Teilnehmer auflegt. Nach dem Zustandekommen ist eine Verbindung symmetrisch und fullduplex, dh. beide Teilnehmer können gleichzeitig 'senden'.

1. Installieren Sie unter Windows die einfachen TCP/IP Dienste und schreiben Sie einen Client für den qotd (Port 17) Dienst.

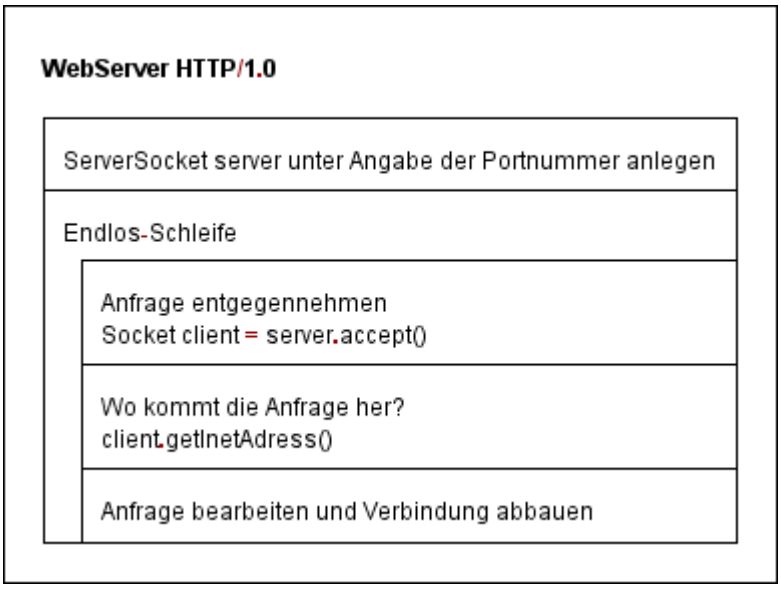
```
class QotdClient {
	public static void main(String[] args) {
		String server = "localhost";// server name
		int port = 17; // qotd port
		try {
			Socket socket = new Socket(server, port);
			InputStream is = socket.getInputStream();
			int data;
			while ((data = is.read()) != -1) {
				System.out.print((char) data);
			}
		} catch (UnknownHostException e) {
			e.printStackTrace();
		} catch (IOException e) {
			e.printStackTrace();
		}
	}
}
```

2. Verändern Sie obigen Client zur Abfrage der Uhrzeit und des Datums (daytime-Dienst Port 13). Mögliche Server: time.fu-berlin.de in Berlin, panix1.panix.com in New York oder localhost.

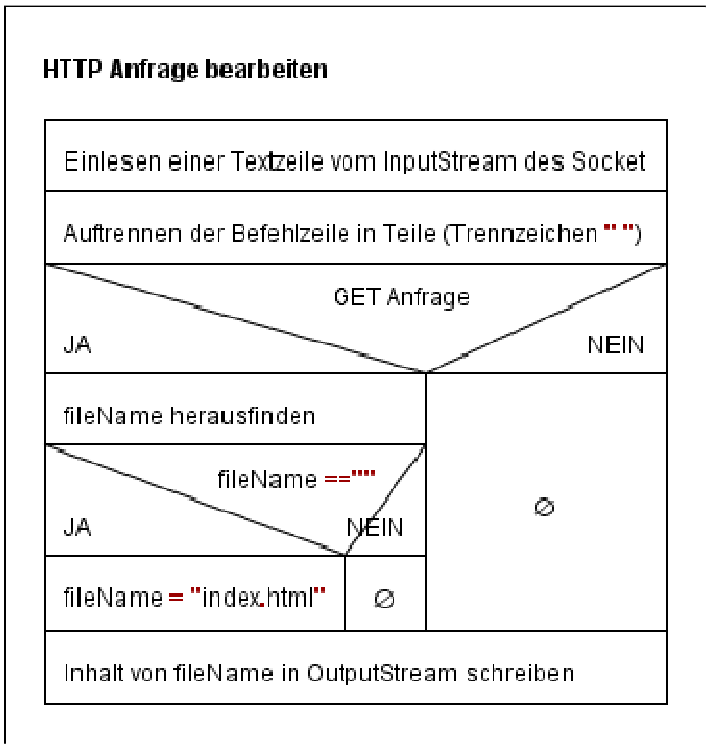
3. Senden Sie einige Grußworte an den Server (echo Port 7) und geben Sie die Antwort aus.

4. Schreiben sie ein Tool zur Abfrage von Web-Seiten (http Port 80). Experimentieren Sie mit dem http-Protokoll.

5. Erstellen Sie nach folgendem Struktogramm einen eigenen einfachen WebServer:



Die Verarbeitung einer Anfrage ist dem nächsten Struktogramm zu entnehmen:



Zum Lesen einer Datei bietet sich die Klasse FileInputStream an.

Da das Senden einer Datei über eine Netzwerkverbindung sehr langsam sein kann, bietet es sich an, die Bearbeitung einer Anfrage in einen seperaten Thread auszulagern.

6. Schreiben Sie eine einfache Client-Server-Software für einen Chat. Im linken Fenster sieht man den Server. Es kann eine Verbindung aus der Liste ausgewählt werden und bei Bedarf aus dem Chat entfernt werden. Die Clients haben eine Textanzeige (TextArea) mit dem Inhalt der gesamten Kommunikation. Der Benutzer kann Text links unten eingeben und an alle Partner im Chat versenden.