TCP Sockets

FHNW
FS 2015
Prof. Dr. Richard Gut

Basierend auf:
http://www.javaworld.com/article/2077322/core-java/sockets-programming-in-java-a-tutorial.html

TCP/IP und UDP/IP Kommunikation

- Es gibt zwei Arten der Socket - Kommunikation: Datagram - Kommunikation und Stream - Kommunikation.
- Die Datagram-Kommunikation (UDP) ist verbindungslos und somit muss jedes Mal die Information zu den Sockets mitgegeben werden.
- Die Stream-Kommunikation verwendet das TCP (Transfer Control Protocol) und ist verbindungsorientiert, d.h. die Verbindung wird nur einmal aufgebaut.
- Java verfügt über die notwendigen Klassen zur Socket - Kommunikation. Wir beschränken uns auf die wichtigere TCP – Kommunikation.

Öffnen einer Socket - Verbindung

Server:

```
ServerSocket serviceSocket;

try {
          serviceSocket = new ServerSocket(11112);
} catch (IOException e) {
         System.out.println(e);
}
```

Verbindung akzeptieren:

```
try {
       client = serviceSocket.accept();
} catch (IOException e) {
       System.out.println(e);
}
```

- Die Zahl bezeichnet den Port, der zur Kommunikation verwendet wird. Die Werte von 0 bis 1023 sind für privilegierte Benutzer (Super oder Root) reserviert!
- Damit steht aber noch keine Verbindung. Der Server-Socket hat nur die Fähigkeit Anfragen an den Port zu akzeptieren.
- Mittels serviceSocket.accept() wird eine allfällige Anfrage akzeptiert.

Öffnen einer Socket - Verbindung

Client:

```
Socket server;
server = new Socket("127.0.0.1", 11112);
```

- Damit besteht nun eine Punkt zu Punkt Verbindung serverSocket <--> clientSocket.
- Auf die beiden Sockets wird nun ein InputStream und OutputStream geöffnet.
- Über die Streams erfolgt dann die eigentliche Kommunikation.

Öffnen einer Socket - Verbindung

Server:
Input - Streams erstellen:

```
try {
        BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(
        client.getInputStream()));
}
catch (IOException e) {
   System.out.println(e);
}
```

Output-Stream erstelle:

```
try {
        PrintWriter printWriter = new PrintWriter(new OutputStreamWriter(
        client.getOutputStream()));
}
catch (IOException e) {
   System.out.println(e);
}
```

Der Socket **client** ist die Verbindung zum Client!

Öffnen einer Socket - Verbindung

Client:
Input - Streams erstellen:

```
    try {
           BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(
           server.getInputStream()));
    }
    catch (IOException e) {
       System.out.println(e);
    }
```

Output-Stream erstelle:

```
    try {
           PrintWriter printWriter = new PrintWriter(new OutputStreamWriter(
           server.getOutputStream()));
    }
    catch (IOException e) {
       System.out.println(e);
    }
```

Der Socket **server** ist die Verbindung zum Server!

Schliessen einer Socket - Verbindung

- Client Seite:

```
try {
   printWriter.close();
   bufferedReader.close();
   server.close();
} catch (IOException e) {
   System.out.println(e);
}
```

- Server Seite:

```
try {
   printWriter.close();
   bufferedReader.close();
   server.close();
   serverSocket.close();
  }catch (IOException e) {
    System.out.println(e);
 }
```