Java tutorial

Hálózati alkalmazások

- A Java egyik fő erőssége
- Célok
  - Kliens-szerver architektúra
  - Elosztott rendszer
- Lehetőségek
  - Appletek
  - Szervletek
  - TCP/IP vagy UDP
  - RMI, CORBA

Internet hálózatok

- Számítógépek azonosítása: IP cím
  - 32 bites szám, pl. 157.181.42.42
  - Neveket rendelünk a gépekhez: DNS
- Szerver programok azonosítása
  - IP cím mellett egy portszám is, 0-65535
  - 0-1023: rendszergazda jog kell hozzájuk
  - 1-255: általánosan elterjedt szolgáltatások
    - HTTP - 80, TELNET - 23, SSH - 22, stb.

IP, TCP, UDP

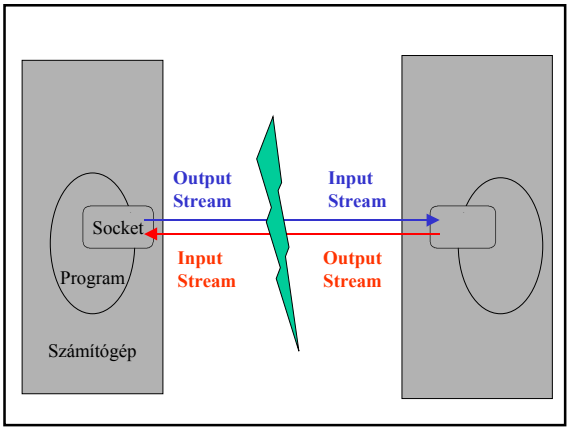
- IP: csomagkapcsolt - az alap
- UDP: szintén csomagkapcsolt (mint egy levél)
  - Az IP jellemzőit örökli (pl. csomagméret korlátozás)
  - A csomagok érkezési sorrendje nem definiált
  - Csomagok el is veszhetnek
  - Nincs visszaigazolás a fogadótól
  - Hatékony
- **TCP**: kapcsolat-elvű (mint egy telefonbeszélgetés)
  - Garantálja, hogy minden elküldött információ a küldés sorrendjében megérkezik
  - Sok erőforrás (IP csomagok visszaigazolása, stb.)

Kliens-szerver architektúra

- Asszimetrikus kapcsolatfelvétel
  - Szerver: passzív
    egy gép egy portján várja a kliensek kapcsolatfelvételi igényét
  - Kliens: aktív
    ismeri a szerver helyét, kezdeményezi a kapcsolatot
- Kommunikáció: már szimmetrikus
  - Mindketten küldhetnek információt a másiknak
- Egy szerverhez több kliens egy időben

Socket

- A BSD UNIX világból származik
- A kommunikációs csatorna végpontjai
- java.net.Socket
  - írni lehet rá, ami elmegy a másik gépre a másik folyamatnak
  - olvasni lehet róla, amit a másik gépen a másik folyamat ír



TCP kapcsolat kiépítése

- Szerver:
  - java.net.ServerSocket létrehozása adott porton
  - kliens várása
  - a létrejött kapcsolatot egy Socket reprezentálja
- Kliens
  - egy Socket létrehozása: adott számítógép adott portján figyelő szerverprogramra való rácsatlakozás

Java tutorial

Java-ul

- Szerver

```
ServerSocket ss = new ServerSocket(1234);
Socket s = ss.accept();
```

- Kliens

```
Socket s = new Socket("www.elte.hu",80);
```

Kommunikáció a kapcsolat felépítése után

- Mind a kliensen, mind a szerveren lekérjük a bemeneti és kimeneti csatornákat
  - A kliens Socket-jának kimenetije rá van csatlakoztatva a szerver Socket-jának bemenetijére
  - és fordítva...
  - Ezeken keresztül lehet kommunikálni

```
InputStream is  = s.getInputStream();
OutputStream os = s.getOuputStream();
```

```
import java.net.*; import java.io.*;
public class Böngésző {
  public static void main(String[] args)
  throws IOException {
     Socket s = new Socket("www.elte.hu", 80);
     InputStream is = s.getInputStream();
     OutputStream os = s.getOutputStream();
     PrintWriter out = new PrintWriter(
                    new OutputStreamWriter(os) );
     BufferedReader in = new BufferedReader(
                       new InputStreamReader(is) );
     out.println("GET /index.html HTTP/1.0");
     out.println();
     out.flush();
     String line;
     while ((line = in.readLine()) != null)
       System.out.println(line);
     in.close(); out.close(); s.close();
  }
}
```

```
import java.net.*; import java.io.*;
public class Ismétlő {
  public static void main(String[] args)
  throws IOException {
     ServerSocket ss = new ServerSocket(1234);
     Socket s = ss.accept();
     InputStream is = s.getInputStream();
     OutputStream os = s.getOutputStream();
     PrintWriter out = new PrintWriter(
                    new OutputStreamWriter(os) );
     BufferedReader in = new BufferedReader(
                       new InputStreamReader(is) );
     String line;
     while ((line = in.readLine()) != null){
        out.println(line);
        out.flush();
     }
     in.close(); out.close(); s.close();
  }
}
```

Feladat

- Írj chat programot. Paraméter nélkül elindítva szerverként viselkedjen. A kliens elindításához parancssori argumentumban kell megadni a szerver IP címét.
- Ha nem AWT-s programot írsz, hanem alfanumerikusat, akkor két végrehajtási szálat kell írnod!
- Írd meg úgy, hogy egyszerre több kliens is lehessen összekapcsolódva! Ehhez többszálú programot kell írnod!

Java tutorial

Datagram

- Az UDP protokoll megvalósítása: DatagramSocket és DatagramPacket
- Példa: kliens küld szervernek, szerver visszaküldi dátummal
- Multicast lehetőség

```
DatagramSocket socket = new DatagramSocket();
byte[] buf;

buf = args[1].getBytes();
InetAddress address = InetAddress.getByName(args[0]);
DatagramPacket packet = new DatagramPacket(buf, buf.length, address, 4445);
socket.send(packet);

buf = new byte[256];
packet = new DatagramPacket(buf, buf.length);
socket.receive(packet);

System.out.println(new String(packet.getData()));
socket.close();
```

```
DatagramSocket socket = new DatagramSocket();
byte[] buf;

buf = new byte[100];
DatagramPacket packet = new DatagramPacket(buf, buf.length);
socket.receive(packet);

buf = (new String(packet.getData()) + new Date().toString()).getBytes();
InetAddress address = packet.getAddress();
int port = packet.getPort();
packet = new DatagramPacket(buf, buf.length, address, port);
socket.send(packet);

socket.close();
```

Elosztott rendszerek

- A program különböző komponensei különböző gépeken futnak.
  - Ebbe belefér a kliens/szerver architektúra is
- A komponensek kommunikációja: objektumok egymás metódusait hívják
  - A kommunikációs részleteket elrejtik a programozó elől
- Megvalósítás
  - Remote Method Invocation (Java-Java)
  - CORBA (Java-*)

Java tutorial