Laborator 5 – Aplicatii Java

Programarea in retea

Scrieti, compilati si rulati toate exemplele din acest laborator:

Programarea in retea implica trimiterea de mesaje si date intre aplicatii ce ruleaza pe calculatoare aflate intr-o retea locala sau conectate la Internet. Pachetul care ofera suport pentru scrierea aplicatiilor de retea este **java.net**. Clasele din acest pachet ofera o modalitate facila de programare in retea, fara a fi nevoie de cunostinte prealabile referitoare la comunicarea efectiva ıntre calculatoare.

Ce este un protocol ?

Un protocol reprezinta o conventie de reprezentare a datelor folosita in comunicarea intre doua calculatoare. Avand in vedere faptul ca orice informatie care trebuie trimisa prin retea trebuie serializata astfel incat sa poata fi transmisa secvential, octet cu octet, catre destinatie, era nevoie de stabilirea unor conventii (protocoale) care sa fie folosite atat de calculatorul care trimite datele cat si de cel care le primeste, pentru a se ”intelege” intre ele.
Doua dintre cele mai utilizate protocoale sunt TCP si UDP.

1. **TCP (Transport Control Protocol)** este un protocol ce furnizeaza un flux sigur de date intre doua calculatoare aflate in retea. Acest protocol asigura stabilirea unei conexiuni permanente intre cele doua calculatoare pe parcursul comunicatiei.
2. **UDP (User Datagram Protocol)** este un protocol bazat pe pachete independente de date, numite datagrame, trimise de la un calculator catre altul fara a se garanta in vreun fel ajungerea acestora la destinatie sau ordinea ın care acestea ajung. Acest protocol nu stabileste o conexiune permanta ıntre cele doua calculatoare.

Cum este identificat un calculator in retiea ?

Orice calculator conectat la Internet este identificat in mod unic de adresa sa IP (IP este acronimul de la Internet Protocol). Aceasta reprezinta un numar reprezentat pe 32 de biti, uzual sub forma a 4 octeti, cum ar fi de exemplu:
31.14.19.2 si este numit adresa IP numerică. Corespunzătoare unei adrese numerice exista si o adresa IP simbolică, cum ar fi www.utgjiu.ro pentru adresa numerică anterioară.
De asemenea, fiecare calculator aflat într-o rețea locală are un nume unic ce poat fi folosit la identificarea locală a acestuia.
Clasa Java care reprezintă noțiunea de adresă IP este **InetAddress**.

Ce este un port ?

Un calculator are în general o singură legătură fizică la rețea. Orice informație destinată unei anumite masini trebuie deci să specifice obligatoriu adresa IP a acelei masini. Insă pe un calculator pot exista concurent mai multe procese care au stabilite conexiuni în rețea, asteptând diverse informații.
Prin urmare, datele trimise către o destinație trebuie să specifice pe lânga adresa IP a calculatorului si procesul către care se îndreaptă informațiile respective. Identificarea proceselor se realizează prin intermdiul porturilor.
Un port este un număr pe 16 biți care identifică în mod unic procesele care rulează pe o anumită masină. Orice aplicație care realizează o conexiune în rețea va trebui să ataseze un număr de port acelei conexiuni. Valorile pe care le poate lua un număr de port sunt cuprinse între 0 si 65535 (deoarece sunt numere reprezentate pe 16 biți), numerele cuprinse între 0 si 1023 fiind însă rezervate unor servicii sistem si, din acest motiv, nu trebuie folosite în aplicații.

Clase de bază din java.net

Clasele din java.net permit comunicarea între procese folosind protocoalele TCP si UDP si sunt prezentate în tabel:

TCP	UDP
URL	DatagramPacket
URLConnection	DatagramSocket
Socket	MulticastSocket
ServerSocket	

Un obiect de tip URL poate fi folosit pentru:

- Aflarea informațiilor despre resursa referită (numele calculatorului gazdă, numele fisierului, protocolul folosit. etc).
- Citirea printr-un flux a continutului fisierului respectiv.
- Conectarea la acel URL pentru citirea si scrierea de informații.

Citirea conținutului unui URL
Orice obiect de tip URL poate returna un flux de intrare de tip InputStream pentru citirea continutului său. Secvența standard pentru această operatiune este prezentată în exemplul de mai jos, în care afisam conținutul resursei specificată la linia de comandă. Dacă nu se specifică nici un argument, va fi afisat fisierul index.html de la adresa:
http://www.infoiasi.ro.

Citirea conținutului unui URL
import java . net .*;

```
import java .io .*;
public class CitireURL {
        public static void main ( String [] args )
        throws IOException {
                String adresa = " http :// www. utgjiu .ro";
                if ( args . length > 0)
                adresa = args [0];
                BufferedReader br = null ;
                try {
                URL url = new URL ( adresa );
                InputStream in = url. openStream ();
                br = new BufferedReader ( new InputStreamReader (in));
                String linie ;
                while (( linie = br. readLine ()) != null ) {
                // Afisam linia citita
                System . out. println ( linie );
                }
        } catch ( MalformedURLException e) {
                System . err. println ("URL invalid !\n" + e);
        } finally {
                br. close ();
        }
        }
}
```

Conectarea la un URL

Se realizează prin metoda openConnection ce stabileste o conexiune bidirecțională cu resursa specificată. Această conexiune este reprezentată de un obiect de tip URLConnection, ce permite crearea atât a unui flux de intrare pentru citirea informațiilor de la URL-ul specificat, cât si a unui flux de iesire pentru scrierea de date către acel URL. Operațiunea de trimitere de date dintr-un program către un URL este similară cu trimiterea de date dintr-un formular de tip FORM aflat într-o pagină HTML. Metoda folosită pentru trimitere este POST.
In cazul trimiterii de date, obiectul URL este uzual un proces ce rulează pe serverul Web referit prin URL-ul respectiv (jsp, servlet, cgi-bin, php, etc).

Alte exemple:

http://www.tutorialspoint.com/java/java_networking.htm

http://cs.lmu.edu/~ray/notes/javanetexamples/ :

1. A Multi-User Chat Application
2. A Two-Player Networked Tic-Tac-Toe Game

http://www.java2novice.com/java_networking/ :
Java Networking Examples
This package provides the classes for implementing networking applications. The java.net package can be broadly classified into two sections.
A low level API, which deals with IP Addresses, Sockets and Network Interfaces
A high level API, which deals with URLs, URIs and Connections.
- See more at: http://www.java2novice.com/java_networking/#sthash.8BhHXuNA.dpuf

Referinte:

http://www.tutorialspoint.com/javaexamples/java_networking.htm
Curs practic de Java, Cristian Frasinaru – capitolul Programarea in retea
http://discipline.elcom.pub.ro/lpai/2006_LPAI_Curs_Cap5.pdf
http://staff.cs.upt.ro/~ionel/PRC/PRC5.pdf