Threads de Control

- Programación secuencial
  - Cada línea del programa se ejecuta en el mismo orden en el cual fue invocada.
- Programación paralela
  - Una parte del programa puede invocar a uno o más proceso que se ejecute en forma paralela.
  - Generalmente se utilizan en una arquitectura multi procesadores.
  - Los S.O. modernos implementan multi-procesos sin la necesidad de multi procesadores.

Threads de Control

- Java provee un mecanismo de Threads, que permite ejecutar varios procesos en forma paralela.
- La manera de usar procesos paralelos es extendiendo la clase Thread e implementar el método run();
- Para la invocación se utiliza el método start();

Threads de Control

- Ejemplo
  - `public class DosThreads {`
    - `public static void main(String[] args){`
      - `new ThreadsSimple("Santiago").start();`
      - `new ThreadsSimple("Arica").start();`
      - `}`
  - `}`
  - Donde la clase ThreadsSimple implementa un proceso en paralelo.

Threads de Control

```
· public class ThreadsSimple extends Thread {
    - public ThreadsSimple(String str) {
        • super(str);
        • }

        • public void run() {
        - for (int i = 0; i < 10; i++) {
            » System.out.println(i + " " + getName());
            » try {
            » sleep((int) (Math.random() * 1000));
            » } catch (InterruptedException e) {}
        - }
        - System.out.println("HECHO! " + getName());
        • }
    -  }
```

- ver: ThreadsImpPack.java y ThreadsSimple.java

Threads de Control

- Si varios Threads deben compartir la misma estructura de datos o espacio de memoria, esto podría llevar a inconsistencia.
- Para evitar las inconsistencias, existe la sincronización.

Threads de Control

```
public class CubbyHole {
    private int contents;
    private boolean available = false;

    public synchronized int get() {
        while (available == false) {
            try {
                wait();
            } catch (InterruptedException e) { }
        }
        available = false;
        notifyAll();
        return contents;
    }

    public synchronized void put(int value) {
        while (available == true) {
            try {
                wait();
            } catch (InterruptedException e) { }
        }
        contents = value;
        available = true;
        notifyAll();
    }
}
```

* la palabra clave synchronized se agrega a los metodos
* los metodos wait() y notifyAll() del objeto Thread se usan para comunicarse entre Threads

Red

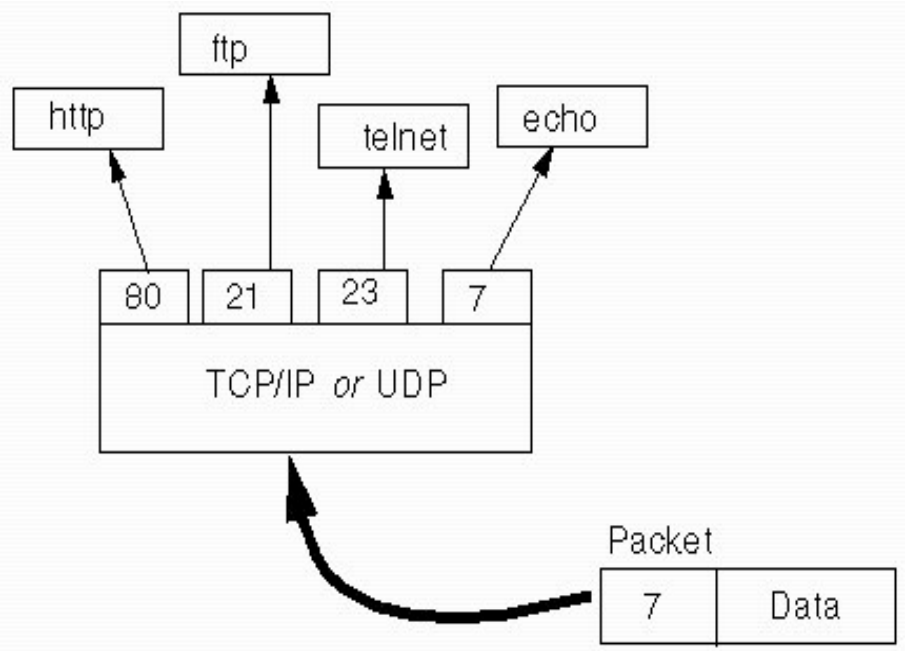
- Trabajo en Red
    - El entorno Java es altamente considerado en parte por su capacidad para escribir programas que utilizan e interactúan con los recursos de Internet y la World Wide Web.

Red

- Para transmisión de datos por Red, se utiliza uno de dos protocolos:
  - TCP:
    - TCP es un protocolo basado en conexión que proporciona un flujo fiable de datos entre dos computadores.
  - UDP:
    - UDP es un protocolo que envía paquetes de datos independientes, desde un computador a otro sin garantías sobre su llegada.

Red

- Para establecer una conexión de se debe especificar el protocolo y el puerto.



Red

- URL
  - URL es un acrónimo que viene de Uniform Resource Locator y es una referencia (una dirección) a un recurso de Internet.
  - http://java.sun.com/
- Clase URL
  - La clase URL proporciona varios métodos que permiten preguntar a los objetos URL.

Red

- Crear un Objeto URL
  - URL linuxCenter = new URL("http://ww.linuxcenter.cl");
  - URL(URL URLbase, String URLrelativa)
    - URL linuxProductos = new URL (”linuxCenter,”productos.html”)

➡Existen más constructores que pueden ser vistos en la API de JAVA.

Red

- Excepciones en los constructores de URLs

```
- try {
-     URL myURL = new URL(. . .)
- } catch (MalformedURLException e) {
-     . . .
-     // Aquí va el código del
  manejador de excepciones
-     . . .
- }
```

Red

- getProtocol()
  - Devuelve el componente identificador de protocolo de la URL.
- getHost()
  - Devuelve el componente nombre del host de la URL.
- getPort()
  - Devuelve el componente número del puerto de la URL. Este método devuelve un entero que es el número de puerto. Si el puerto no está selccionado, devuelve -1.
- getFile()
  - Devuelve el componente nombre de fichero de la URL.

Red

- getRef()
  - Obtiene el componente referencia de la URL.
  - Ejemplo:

```
- import java.net.*;
- class ParseURL {
-     public static void main(String[] args) {
-         URL aURL = null;
-         try {
-             aURL = new
  URL("http://java.sun.com:80/tutorial/intro.html#DOWNLOADING");
-             System.out.println("protocol = " + aURL.getProtocol());
-             System.out.println("host = " + aURL.getHost());
-             System.out.println("filename = " + aURL.getFile());
-             System.out.println("port = " + aURL.getPort());
-             System.out.println("ref = " + aURL.getRef());
-         } catch (MalformedURLException e) {
-             System.out.println("MalformedURLException: " + e);
-         }
-     }
- }
```

Red

- Resultado del ejemplo:
  - protocol = http
  - host = java.sun.com
  - filename = /tutorial/intro.html
  - port = 80
  - ref = DOWNLOADING

Red

- Leer desde una URL

```
import java.net.*;
import java.io.*;

class OpenStreamTest {
    public static void main(String[] args) {
        try {
            URL yahoo = new URL("http://www.yahoo.com/");
            DataInputStream dis = new DataInputStream(yahoo.openStream());
            String inputLine;

            while ((inputLine = dis.readLine()) != null) {
                System.out.println(inputLine);
            }
            dis.close();
        } catch (MalformedURLException me) {
            System.out.println("MalformedURLException: " + me);
        } catch (IOException ioe) {
            System.out.println("IOException: " + ioe);
        }
    }
}
```

Red

- Resultado

- <html><head>
- <script language=javascript>
- var now=new Date,t1=0,t2=0,t3=0,t4=0,t5=0,t6=0,hp=0,cc='',ylp='';t1=now.getTime();
- </script>

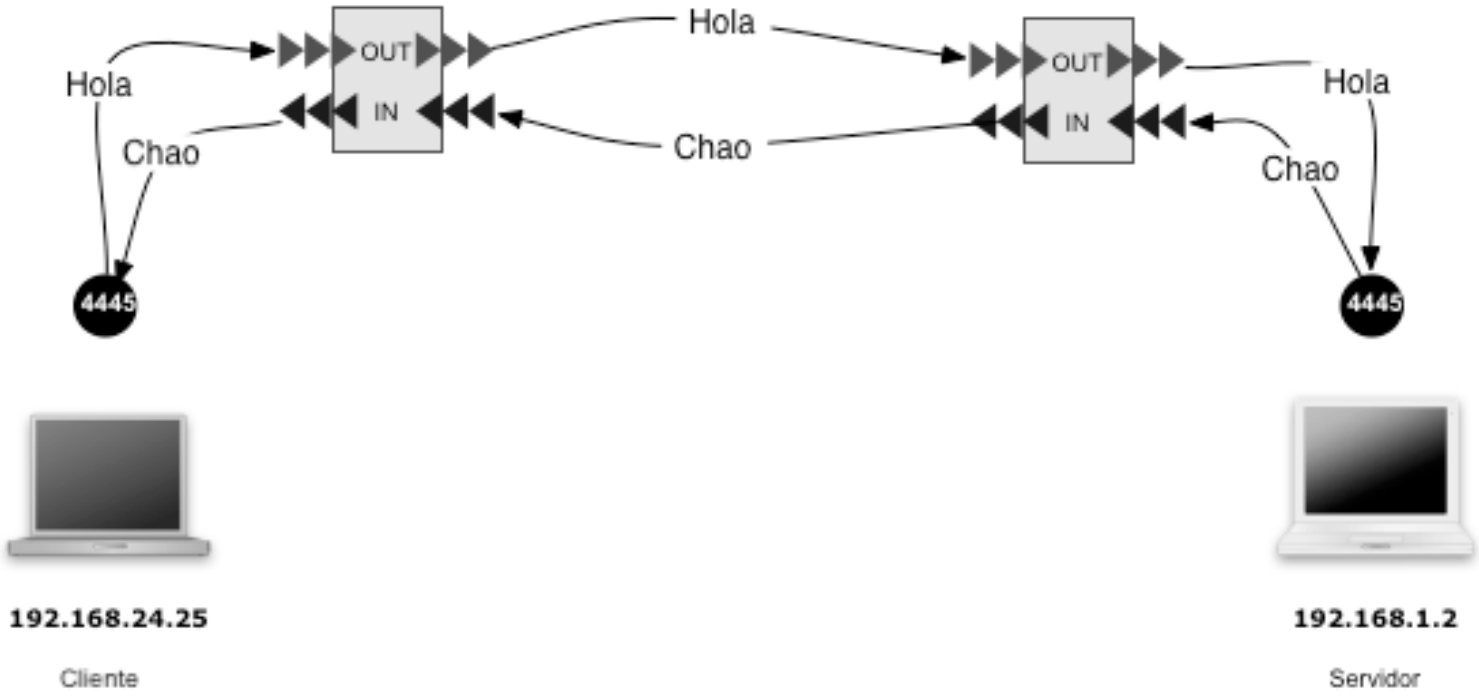
- <title>Yahoo!</title>
- <meta http-equiv="PICS-Label" content='(PICS-1.1 "http://www.icra.org/ratingsv02.html" l r (cz 1 lz 1 nz 1 oz 1 vz 1) gen true for "http://www.yahoo.com" r (cz 1 lz 1 nz 1 oz 1 vz 1) "http://www.rsac.org/ratingsv01.html" l r (n 0 s 0 v 0 l 0) gen true for "http://www.yahoo.com" r (n 0 s 0 v 0 l 0))'>
- <base href="http://www.yahoo.com/_ylh=X3oDMTEwZGh2NmNjBF9TAzI3MTYxNDkEdGVzdAMwBHRtcGwDaW5kZXgtdGJs/" target=_top>
- <script language=javascript><!--
- lck='',sss=1101710360,ylp='http://geo.yahoo.com/serv2?t=1101710360&_ylp=A9htdkYYxKpBr7MAdk71cSkA';//--></script><script language=javascript>
- var b,d,l='',n='0',r,s,y;
- .
- .
- <script language=javascript>d.sf1.p.focus();</script>
- <script language=javascript>
- now=new Date;
- t4=now.getTime();
- </script>
- </body>

Red

- La Clase URL me permite interactuar a nivel de la WEB.
- La Clase Socket me permite interactuar a nivel de cualquier aplicación.
- Java nos provee de herramientas para trabajar con la internet en una libreria:
  - java.net.*

Red

- Sockets


Hola
Chao
OUT
IN
Hola
Chao
OUT
IN
Hola
Chao
4445
4445
192.168.24.25
Cliente
192.168.1.2
Servidor

Red

- Ejemplo: Sockets
  - Cree un programa que se conecte al servidor que tiene corriendo el profesor usando la IP de él.
  - ver: EchoServer.java y EchoClient.java