

LPR 2005/ 2006

Lezione n.10

Callbacks: esempio di utilizzo

GESTIONE ELEZIONI

Si vuole gestire una elezione a cui partecipano un numero prefissato di candidati.

Si chiede di realizzare un server RMI che consentisse al client di votare un candidato e di richiedere il numero di voti ottenuti dai candidati fino ad un certo punto.

Il server deve notificare ogni nuovo voto ricevuto a tutti i clients che hanno votato fino a quel momento. La registrazione dei clients sul server avviene al momento del voto.

L'INTERFACCIA DEL CLIENT

```
import java.rmi.*;
```

```
public interface CallbackClientInterface extends Remote  
{ void notifyVote (String name) throws RemoteException;  
}
```

L'OGGETTO REMOTO CLIENT

```
import java.rmi.*;
```

```
import java.rmi.server.*;
```

```
public class CallbackClientImpl extends
```

```
    UnicastRemoteObject implements CallbackClientInterface
```

```
{
```

```
    public CallbackClientImpl( ) throws RemoteException {super();};
```

```
    public void notifyVote(String s)
```

```
    {
```

```
        System.out.println(s);
```

```
    }
```

```
}
```

IL CLIENT

```
import java.rmi.*;
```

```
import java.util.*;
```

```
public class CallbackClient
```

```
    {public static void main (String args[])
```

```
        {Scanner in = new Scanner(System.in);
```

```
        try{String nome = in.next();
```

```
        CallbackServerInterface serverRMI = (CallbackServerInterface)
```

```
        Naming.lookup( "rmi://localhost:1099/Callbacks");
```

```
        CallbackClientInterface clientobj = new CallbackClientImpl( );
```

```
        serverRMI.Vote(nome,clientobj);
```

```
        }
```

```
        catch (Exception e){System.out.println(e);} 
```

```
    } }
```

L'INTERFACCIA DEL SERVER

```
import java.rmi.*;
```

```
public interface CallbackServerInterface extends Remote  
{ void Vote (String name, CallbackClientInterface cbc) throws  
RemoteException;  
}
```

L'OGGETTO REMOTO SERVER

```
import java.rmi.*;
```

```
import java.rmi.server.*;
```

```
import java.util.*;
```

```
public class CallbackServerImpl extends UnicastRemoteObject  
    implements CallbackServerInterface
```

```
{int vote=0;
```

```
Vector clients;
```

```
public CallbackServerImpl( ) throws RemoteException
```

```
{clients=new Vector();};
```

L'OGGETTO REMOTO SERVER

```
public void Vote (String name, CallbackClientInterface cbc)  
    throws RemoteException  
        { vote++;  
          System.out.println("i voti sono "+vote);  
          for (int i=0; i<clients.size( ); i++)  
            { clients.elementAt(i).notifyVote(name); }  
          clients.addElement(cbc);  
        }  
    }
```


IL SERVER

```
import java.rmi.*;
```

```
public class CallbackServer{
```

```
public static void main(String args[])
```

```
    {try
```

```
    {
```

```
        CallbackServerImpl serverback= new CallbackServerImpl();
```

```
        Naming.rebind("rmi://localhost:1099/Callbacks",serverback);
```

```
        System.out.println("Server Back Registrato");
```

```
    } catch (Exception e){
```

```
        System.out.println(e);} } }
```

L'OGGETTO REMOTO CLIENT

```
import java.rmi.*;
```

```
import java.rmi.server.*;
```

```
public class CallbackClientImpl extends
```

```
    UnicastRemoteObject implements CallbackClientInterface
```

```
{
```

```
    public CallbackClientImpl() throws RemoteException {super();};
```

```
    public void notifyVote(String s)
```

```
    {
```

```
        System.out.println(s);
```

```
    }
```

```
}
```