TP 5 RMI
Safa YAHI

Exercice 1 : Considérons le serveur RMI suivant :

```
package interfaces;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface HelloInterface extends Remote{

	String getHello(String user) throws RemoteException;

}
```

```
package serveur;


import interfaces.HelloInterface;

import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;


public class HelloImpl extends UnicastRemoteObject implements HelloInterface {

	 protected HelloImpl() throws RemoteException {
		super();
	 }

	 public String getHello(String user) throws RemoteException {

	     String msg = "Coucou " + user + ":)";
		return msg ;
	 }

}
```

```
package serveur;

import java.net.InetAddress;
import java.net.NetworkInterface;
import java.net.SocketException;
import java.net.UnknownHostException;
import java.rmi.NotBoundException;
import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.util.Enumeration;
```

```
import java.util.Properties;

public class HelloServer {

      public static String getAdresseLocale() throws SocketException {
            InetAddress adr = null ;
            Enumeration<InetAddress> listAdr =
NetworkInterface.getByName("eth0").getInetAddresses();

             while (listAdr.hasMoreElements())
             adr = listAdr.nextElement();

             return adr.getHostAddress() ;

      }

            public static void main(String[] args) throws RemoteException,
                  InterruptedException, NotBoundException, UnknownHostException,
                  SocketException {

            // Pour communiquer entre client et serveur sur machines
            // différentes, il faut positionner le hostname du serveur à son @
            // IP. Pour avoir cette @ IP, on peut utiliser la valeur de ifconfig
            // comme on peut la reccupérer automatiquement avec la fonction
            // getAdresseLocale()

            Properties  prop = System.getProperties();

            prop.put("java.rmi.server.hostname", getAdresseLocale());


            // Créer une instance de l'objet distant
            HelloImpl obj = new HelloImpl();

            // avoir une reference pour le registre RMI sur la machine locale et
            // sur le port 1099

            Registry registre;

            registre = LocateRegistry.createRegistry(1099);

            // publier l'objet distant sur le registre avec la string "hello"
            // et se mettre en attente des requêtes.

            registre.rebind("hello", obj);

            System.out.println("Serveur hello pret !");

      }
}
```

1) Qu'est-ce qu'un objet distant en RMI?
2) Quel est l'objet distant de ce serveur ?
3) De quelle classe dérive-t-il ?
4) Quelle(s) interface(s) implémente-t-il ?
5) Qu'est-ce qu'une interface distante ?

6) Rappelez les étapes à suivre pour publier un objet distant.
7) Regardez le main() de la classe ServerHello et faîtes le lien avec les étapes de la question précédente.
8) Quelle sont les informations que doit avoir un client RMI ?
9) Écrivez le client RMI de cette application et testez le avec le serveur RMI indiqué par votre enseignant(e).

Exercice 2 :

Écrivez une application RMI qui permet à un client de demander au serveur de lui indiquer la classe, l'adresse réseau ou l'adresse de diffusion d'une adresse IP donnée. Vous pouvez utiliser les fonctions de la classe CAdresseIP du TP précédent.