LAB:NO:05

OBJECT: TO BECOME FAMILIAR WITH RMI OBJECT SERIALIZATION.

CLASS TASK: create a program in which server sharing an object via RMI and a client calling the shared instance.

Program: Interface.java:

import java.rmi.\*;

public interface Api extends Remote{

public strdata increementCounter(strdata str)

throws RemoteException;

}

RemoteObjectClass.java:

import java.rmi.\*;

import java.rmi.server.\*;

public class ApiImpl extends UnicastRemoteObject implements Api{

private int counter=0;

public ApiImpl() throws RemoteException{

super();

}

public synchronized strdata increementCounter(strdata ob)throws RemoteException{

counter+=ob.getValue();

return new strdata(counter);

}

}

Serializable class:

import java.io.\*;

public class strdata implements Serializable{

private int str;

public strdata(int str){

this.str=str;

}

public int getValue(){

return str;

}

public void setValue(int str){

this.str=str;

}

}

Server.java:

import java.rmi.\*;

import java.rmi.registry.\*;

public class server{

private static final int PORT=1099;

private static Registry registry;

public static void startRegistry() throws RemoteException{

registry= java.rmi.registry.LocateRegistry.createRegistry(PORT);

}

public static void registerObject(String name,Remote remoteObject)throws RemoteException,AlreadyBoundException{

registry.bind(name,remoteObject);

System.out.println("registered");

}

public static void main(String args[]) throws Exception{

startRegistry();

registerObject(Api.class.getSimpleName(),new ApiImpl());

}

}

Client.java:

import java.rmi.registry.\*;

public class Client{

private static final String HOST="localhost";

private static final int PORT=1099;

private static Registry registry;

public static void main(String args[])throws Exception{

registry=LocateRegistry.getRegistry(HOST,PORT);

Api remoteApi=(Api)registry.lookup(Api.class.getSimpleName());

for(int i=1;i<=100;i++){

System.out.println(i+remoteApi.increementCounter(new strdata(i)).getValue());

Thread.sleep(1000);

}

}}

Output:



TASK: Develop any simple application to demonstrate object passing in RMI. (Expl: Make a class of your own and use its objects as arguments and return types for RMI).

Program: Interface.java:

import java.rmi.\*;

public interface marsh extends Remote{

public Student studentDetail(Student detail)throws RemoteException;

}

RemoteObjectClass.java:

import java.rmi.\*;

import java.rmi.server.\*;

public class marshImpl extends UnicastRemoteObject implements marsh{

private String counter="student info:";

public marshImpl() throws RemoteException{

super();

}

public synchronized Student studentDetail(Student ob)throws RemoteException{

counter=ob.getDetail();

return new Student(counter);

}

}

Serializable class:

import java.io.\*;

public class Student implements Serializable{

private String detail;

public Student(String detail){

this.detail=detail;

}

public String getDetail(){

return detail;}

public void setDetail(String detail){

this.detail=detail;

}

}

Server.java:

import java.rmi.\*;

import java.rmi.registry.\*;

public class marshServer{

private static final int PORT=1099;

private static Registry registry;

public static void startRegistry() throws RemoteException{

registry=java.rmi.registry.LocateRegistry.createRegistry(PORT);

}

public static void registerObject(String name,Remote remoteObj)throws RemoteException,AlreadyBoundException{

registry.bind(name,remoteObj);

System.out.println("registered");

}

public static void main (String args[])throws Exception{

startRegistry();

registerObject(marsh.class.getSimpleName(),new marshImpl());

}

}

Client.java:

import java.rmi.registry.\*;

import java.util.ArrayList;

public class marshClient{

private static final String HOST="localhost";

private static final int PORT=1099;

private static Registry registry;

public static void main(String args[])throws Exception{

registry=LocateRegistry.getRegistry(HOST,PORT);

marsh remoteApi=(marsh)registry.lookup(marsh.class.getSimpleName());

ArrayList<String> info=new ArrayList<String>();

info.add("ahsan");

info.add("16sw16");

info.add("SWE");

info.add("4th year");

info.add("Hyd");

info.add("MUET");

System.out.println("array size"+info.size());

for(int i=0;i<info.size();i++){

System.out.println("Student detail:"+remoteApi.studentDetail(new Student(info.get(i))).getDetail()+"/n");

Thread.sleep(100);

}

}

}

Output:

