LAB # 5

**OBJECTIVE:** To become familiar with rmi object serialization

**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)

**Employee class: -**

import java.io.\*;

public class Employee implements Serializable{

private String name;

private int age;

private String jobTitle;

private double salary;

public Employee(String name, int age, String jobTitle, double salary){

this.name = name;

this.age = age;

this.jobTitle = jobTitle;

this.salary = salary;

}

public String getName(){

return name;

}

public void setName(String name){

this.name = name;

}

public int getAge(){

return age;

}

public void setAge(int age){

this.age = age;

}

public String getJobTitle(){

return jobTitle;

}

public void setJobTitle(String jobTitle){

this.jobTitle = jobTitle;

}

public double getSalary(){

return salary;

}

public void setSalary(double salary){

this.salary = salary;

}

}

**Api Iterface: -**

import java.util.List;

import java.rmi.\*;

public interface Api extends Remote{

public void addEmployee(Employee emp)throws RemoteException;

public List<Employee> getEmployee() throws RemoteException;

}

**ApiImpl Class: -**

import java.util.List;

import java.util.ArrayList;

import java.rmi.\*;

import java.rmi.server.\*;

public class ApiImpl extends UnicastRemoteObject implements Api{

List<Employee> employees;

public ApiImpl() throws RemoteException{

super();

employees = new ArrayList<>();

}

public synchronized void addEmployee(Employee emp) throws RemoteException{

employees.add(emp);

}

public synchronized List<Employee> getEmployee() throws RemoteException{

return employees;

}

}

**Server Class: -**

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 = LocateRegistry.createRegistry(PORT);

}

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

registry.bind(name, remoteObj);

System.out.println("Registered");

System.out.println("Server is running...");

}

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

startRegistry();

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

}

}

**Client Class: -**

import java.util.List;

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());

//adding employees data

remoteApi.addEmployee(new Employee("Ahmed", 23, "Accountant", 55000.00));

remoteApi.addEmployee(new Employee("Usman", 27, "Designer", 65000.00));

remoteApi.addEmployee(new Employee("Hamza", 24, "Accountant", 54000.00));

//getting employees data

List<Employee> employees = remoteApi.getEmployee();

System.out.println("------Employees Details------\n");

for(Employee emp: employees){

System.out.println("Name: "+emp.getName());

System.out.println("Age: "+emp.getAge());

System.out.println("Job Title: "+emp.getJobTitle());

System.out.println("Salary: "+emp.getSalary());

System.out.println("------------------------------");

}

}

}

**Output:-**



