**calculator.idl**

module CalcApp

{

interface Calc

{

exception DivisionByZero {};

float sum(in float a, in float b);

float div(in float a, in float b) raises (DivisionByZero);

float mul(in float a, in float b);

float sub(in float a, in float b);

};

};

**CalcServer.java**

import CalcApp.\*;

import CalcApp.CalcPackage.DivisionByZero;

import org.omg.CosNaming.\*;

import org.omg.CosNaming.NamingContextPackage.\*;

import org.omg.CORBA.\*;

import org.omg.PortableServer.\*;

import java.util.Properties;

class CalcImpl extends CalcPOA {

    @Override

    public float sum(float a, float b) {

        return a + b;

    }

    @Override

    public float div(float a, float b) throws DivisionByZero {

        if (b == 0) {

            throw new CalcApp.CalcPackage.DivisionByZero();

        } else {

            return a / b;

        }

    }

    @Override

    public float mul(float a, float b) {

        return a \* b;

    }

    @Override

    public float sub(float a, float b) {

        return a - b;

    }

    private ORB orb;

    public void setORB(ORB orb\_val) {

        orb = orb\_val;

    }

}

public class CalcServer {

    public static void main(String args[]) {

        try {

            // create and initialize the ORB

            ORB orb = ORB.init(args, null);

            // get reference to rootpoa & activate the POAManager

            POA rootpoa = POAHelper.narrow(orb.resolve\_initial\_references("RootPOA"));

            rootpoa.the\_POAManager().activate();

            // create servant and register it with the ORB

            CalcImpl helloImpl = new CalcImpl();

            helloImpl.setORB(orb);

            // get object reference from the servant

            org.omg.CORBA.Object ref = rootpoa.servant\_to\_reference(helloImpl);

            Calc href = CalcHelper.narrow(ref);

            // get the root naming context

            // NameService invokes the name service

            org.omg.CORBA.Object objRef = orb.resolve\_initial\_references("NameService");

            // Use NamingContextExt which is part of the Interoperable

            // Naming Service (INS) specification.

            NamingContextExt ncRef = NamingContextExtHelper.narrow(objRef);

            // bind the Object Reference in Naming

            String name = "Calc";

            NameComponent path[] = ncRef.to\_name(name);

            ncRef.rebind(path, href);

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

            // wait for invocations from clients

            orb.run();

        } catch (Exception e) {

            System.err.println("ERROR: " + e);

            e.printStackTrace(System.out);

        }

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

    }

}

**CalcClient.java**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import CalcApp.\*;

import CalcApp.CalcPackage.DivisionByZero;

import org.omg.CosNaming.\*;

import org.omg.CosNaming.NamingContextPackage.\*;

import org.omg.CORBA.\*;

import static java.lang.System.out;

public class CalcClient {

    static Calc calcImpl;

    static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

    public static void main(String args[]) {

        try {

            // create and initialize the ORB

            ORB orb = ORB.init(args, null);

            // get the root naming context

            org.omg.CORBA.Object objRef = orb.resolve\_initial\_references("NameService");

            // Use NamingContextExt instead of NamingContext. This is

            // part of the Interoperable naming Service.

            NamingContextExt ncRef = NamingContextExtHelper.narrow(objRef);

            // resolve the Object Reference in Naming

            String name = "Calc";

            calcImpl = CalcHelper.narrow(ncRef.resolve\_str(name));

//          System.out.println(calcImpl);

            while (true) {

                out.println("1. Sum");

                out.println("2. Sub");

                out.println("3. Mul");

                out.println("4. Div");

                out.println("5. exit");

                out.println("--");

                out.println("choice: ");

                try {

                    String opt = br.readLine();

                    if (opt.equals("5")) {

                        break;

                    } else if (opt.equals("1")) {

                        out.println("a+b= " + calcImpl.sum(getFloat("a"), getFloat("b")));

                    } else if (opt.equals("2")) {

                        out.println("a-b= " + calcImpl.sub(getFloat("a"), getFloat("b")));

                    } else if (opt.equals("3")) {

                        out.println("a\*b= " + calcImpl.mul(getFloat("a"), getFloat("b")));

                    } else if (opt.equals("4")) {

                        try {

                            out.println("a/b= " + calcImpl.div(getFloat("a"), getFloat("b")));

                        } catch (DivisionByZero de) {

                            out.println("Division by zero!!!");

                        }

                    }

                } catch (Exception e) {

                    out.println("===");

                    out.println("Error with numbers");

                    out.println("===");

                }

                out.println("");

            }

            //calcImpl.shutdown();

        } catch (Exception e) {

            System.out.println("ERROR : " + e);

            e.printStackTrace(System.out);

        }

    }

    static float getFloat(String number) throws Exception {

        out.print(number + ": ");

        return Float.parseFloat(br.readLine());

    }

}