Membuat Aplikasi Converter Suhu Menggunakan Komponen CORBA

Mudafiq Riyan Pratama
Mudafiq.riyan@yahoo.com
http://dhafiq-san.blogspot.com/

Pendahuluan

CORBA (*Common Object Request Broker Architecture*) adalah sebuah arsitektur software yang berbasis pada teknologi berorientasi obyek atau Object Oriented (OO) dengan paradigma *client-server*. CORBA dapat digunakan untuk pengembangan software dengan berbasis pada rekayasa berkomponen.

CORBA lahir berdasarkan 'kesepakatan' antara sejumlah vendor dan pengembang perangkat lunak terkenal seperti IBM, Hewlett-Packard, dan DEC, yang tergabung dalam sebuah konsorsium bernama **OMG** (*Object Management Group*).

Konsep *Object-Oriented* (OO) melahirkan paradigma *client-server* yang mana pada sebuah obyek berkomunikasi dengan obyek lain dengan cara pengiriman pesan (*message passing*). Konteks komunikasi ini kemudian dipetakan ke dalam model *client-server*: satu obyek berperan sebagai *client* (si pengirim pesan) dan yang lain bertindak sebagai *server* (yang menerima pesan dan memproses pesan yang bersangkutan).

Dengan menerapkan konsep *Object-Oriented* ataupun *Client-Server*, kita akan mencoba untuk membuat sebuah aplikasi sederhana yaitu **Membuat Aplikasi Converter Suhu Menggunakan Komponen CORBA,** yang mana komponen CORBA telah disediakan oleh Java.

Kebutuhan Software:
1. JDK (penulis menggunakan **jdk1.6.0_16**)
2. Notepad
3. Disini penulis mempraktekkan menggunakan sistem operasi Windows XP

PEMBAHASAN

1. **Langkah Awal**
   Buatlah sebuah folder untuk project **ConverterCORBA** yang akan dibuat, pada tahap ini penulis membuat folder **Corba**

2. **Membuat File IDL**
   Berikut isi dari file IDL, dan simpan dengan nama **Convert.idl** dan letakkan pada folder project yang telah dibuat yaitu **Corba**

```
module TempConvertApp
{
       interface Convert
       {
              double cToF(in double c);
       };
};
```

3. **Membuat Package TempConvertApp**
   Pada folder project yang dibuat tadi, buatlah folder baru lagi dengan nama **TempConvertApp** untuk tempat package pengolahan komponen CORBA

4. **Buat File "Convert.java" Dan Letakkan Dalam Folder "TempConvertApp"**

```
package TempConvertApp;

public interface Convert extends org.omg.CORBA.Object {
    double cToF(double c);
}
```

   Dan simpan file **Convert.java** tersebut pada package atau folder **TempConvertApp**

5. **Buat File "ConvertOperations.java" Dan Letakkan Dalam Folder "TempConvertApp"**

   File tersebut yang mendeklarasikan semua operasi

```
package TempConvertApp;

public interface ConvertOperations {
    double cToF(double c);
} // interface ConvertOperations
```

6. Buat File "ConvertHelper.java" Dan Letakkan Dalam Folder "TempConvertApp"

File inilah yang menyediakan fungsi pelengkap seperti method *narrow()* yang diperlukan untuk memanggil object CORBA

```
   package TempConvertApp;

   abstract public class ConvertHelper {
       private static String _id =
"IDL:TempConvertApp/Convert:1.0";

       public static void insert(org.omg.CORBA.Any a,
TempConvertApp.Convert that) {
           org.omg.CORBA.portable.OutputStream out =
a.create_output_stream();
           a.type(type());
           write(out, that);
           a.read_value(out.create_input_stream(), type());
       }
       public static TempConvertApp.Convert
extract(org.omg.CORBA.Any a) {
           return read(a.create_input_stream());
       }
       private static org.omg.CORBA.TypeCode __typeCode =
null;

       synchronized public static org.omg.CORBA.TypeCode
type() {
           if (__typeCode == null) {
               __typeCode =
org.omg.CORBA.ORB.init().create_interface_tc(TempConvertApp.C
onvertHelper.id(), "Convert");
           }
           return __typeCode;
       }

       public static String id() {
           return _id;
       }

       public static TempConvertApp.Convert
read(org.omg.CORBA.portable.InputStream istream) {
           return
narrow(istream.read_Object(_ConvertStub.class));
       }

       public static void
write(org.omg.CORBA.portable.OutputStream ostream,
               TempConvertApp.Convert value) {
           ostream.write_Object((org.omg.CORBA.Object)
value);
       }
```

```
        public static TempConvertApp.Convert
narrow(org.omg.CORBA.Object obj) {
            if (obj == null) {
                return null;
            } else if (obj instanceof TempConvertApp.Convert)
{
                return (TempConvertApp.Convert) obj;
            } else if (!obj._is_a(id())) {
                throw new org.omg.CORBA.BAD_PARAM();
            } else {
                org.omg.CORBA.portable.Delegate delegate =
                        ((org.omg.CORBA.portable.ObjectImpl)
obj)._get_delegate();
                TempConvertApp._ConvertStub stub = new
TempConvertApp._ConvertStub();
                stub._set_delegate(delegate);
                return stub;
            }
        }
    }
```

7. Buat File "ConvertHolder.java" Dan Letakkan Dalam Folder "TempConvertApp"

Memiliki class *Holder* yang digunakan untuk memanggil object CORBA untuk membaca dan menulis sebuah operasi input stream parameter.

```
    package TempConvertApp;

    public final class ConvertHolder implements
            org.omg.CORBA.portable.Streamable
    {
    public TempConvertApp.Convert value = null;
    public ConvertHolder ()
    {
    }
    public ConvertHolder (TempConvertApp.Convert
initialValue)
    {
    value = initialValue;
    }
    public void _read (org.omg.CORBA.portable.InputStream i)
    {
    value = TempConvertApp.ConvertHelper.read (i);
    }
    public void _write (org.omg.CORBA.portable.OutputStream
o)
    {
    TempConvertApp.ConvertHelper.write (o, value);
    }
    public org.omg.CORBA.TypeCode _type ()
```

```
    {
    return TempConvertApp.ConvertHelper.type ();
    }
    }
```

8. Buat File "ConvertPOA.java" Dan Letakkan Dalam Folder "TempConvertApp"

Adalah sebuah kerangka class untuk yang mengimplementasikan server bahwasanya akan mengoperasikan interface dan penggunaan *narrow()* pada class *Helper* sebelum ditampilkan.

```
   package TempConvertApp;

   public abstract class ConvertPOA extends
org.omg.PortableServer.Servant implements
TempConvertApp.ConvertOperations,
           org.omg.CORBA.portable.InvokeHandler {
   // Constructors

       private static java.util.Hashtable _methods = new
java.util.Hashtable();

       static {
           _methods.put("cToF", new java.lang.Integer(0));
       }

       public org.omg.CORBA.portable.OutputStream
_invoke(String $method, org.omg.CORBA.portable.InputStream
in,
               org.omg.CORBA.portable.ResponseHandler $rh) {
           org.omg.CORBA.portable.OutputStream out = null;
           java.lang.Integer __method = (java.lang.Integer)
_methods.get($method);
           if (__method == null) {
               throw new org.omg.CORBA.BAD_OPERATION(0,

org.omg.CORBA.CompletionStatus.COMPLETED_MAYBE);
           }
   // Dispatch method request to its handler
           switch (__method.intValue()) {
               case 0: // TempConvertApp/Convert/cToF
               {
                   double c = in.read_double();
                   double $result = (double) 0;
   //invoke the method
                   $result = this.cToF(c);
   //create an output stream for delivery of the result
                   out = $rh.createReply();
   //Marshal the result via output stream which connects
   //the input stream of client
                   out.write_double($result);
                   break;
               }
```

```
                default:
                    throw new org.omg.CORBA.BAD_OPERATION(0,

org.omg.CORBA.CompletionStatus.COMPLETED_MAYBE);
            }
            return out;
        } // _invoke
    // Type-specific CORBA::Object operations
        private static String[] __ids = {
            "IDL:TempConvertApp/Convert:1.0"};

        public String[]
_all_interfaces(org.omg.PortableServer.POA poa, byte[]
objectId) {
            return (String[]) __ids.clone();
        }

        public Convert _this() {
            return ConvertHelper.narrow(
                    super._this_object());
        }

        public Convert _this(org.omg.CORBA.ORB orb) {
            return ConvertHelper.narrow(
                    super._this_object(orb));
        }
    } // class ConvertPOA
```

9. Buat File "_ConvertStub.java" Dan Letakkan Dalam Folder "TempConvertApp"

```
    package TempConvertApp;

    public class _ConvertStub extends
org.omg.CORBA.portable.ObjectImpl implements
TempConvertApp.Convert {

        public double cToF(double c) {
            org.omg.CORBA.portable.InputStream $in = null;
            try {
    //create a request via an output stream
                org.omg.CORBA.portable.OutputStream $out =
                        _request("cToF", true);
    //marshal the arguments
                $out.write_double(c);
    //method invication via output stream and
    //connect to a input stream
                $in = _invoke($out);
    //unmarshal the return result
                double $result = $in.read_double();
                return $result;
```

```
            } catch
(org.omg.CORBA.portable.ApplicationException $ex) {
                $in = $ex.getInputStream();
                String _id = $ex.getId();
                throw new org.omg.CORBA.MARSHAL(_id);
            } catch
(org.omg.CORBA.portable.RemarshalException $rm) {
                return cToF(c);
            } finally {
                _releaseReply($in);
            }
        } // cToF
    // Type-specific CORBA::Object operations
        private static String[] __ids =
{"IDL:TempConvertApp/Convert:1.0"};

        public String[] _ids() {
            return (String[]) __ids.clone();
        }

        private void readObject(java.io.ObjectInputStream s)
throws
                java.io.IOException {
            String str = s.readUTF();
            String[] args = null;
            java.util.Properties props = null;
            org.omg.CORBA.Object obj =
org.omg.CORBA.ORB.init(args,
                    props).string_to_object(str);
            org.omg.CORBA.portable.Delegate delegate =
                    ((org.omg.CORBA.portable.ObjectImpl)
obj)._get_delegate();
            _set_delegate(delegate);
        }

        private void writeObject(java.io.ObjectOutputStream
s)
                throws java.io.IOException {
            String[] args = null;
            java.util.Properties props = null;
            String str = org.omg.CORBA.ORB.init(args,
                    props).object_to_string(this);
            s.writeUTF(str);
        }
    } // class _ConvertStub
```

10. Buat File "ConvertServer.java" Dan Letakkan Di Dalam Folder Project "Corba" (Bukan package "TempConvertApp")

Merupakan file server yang memiliki 2 class: class *ConvertImpl* yang mewarisi atau meng-*extends* class *ConvertPOA* (kerangka CORBA) dan class satunya merupakan class untuk *main()* method.

```
// ConvertServer.java
import TempConvertApp.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;
import org.omg.CORBA.*;
import org.omg.PortableServer.*;
import java.util.Properties;

class ConvertImpl extends ConvertPOA {

    private ORB orb;

    public void setORB(ORB orb_val) {
        orb = orb_val;
    }
// implement cToF() method

    public double cToF(double c) {
        return (c * 9. / 5 + 32);
    }
}

public class ConvertServer {

    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
            ConvertImpl convertImpl = new ConvertImpl();
            convertImpl.setORB(orb);
//get object reference from servant
            org.omg.CORBA.Object ref =

rootpoa.servant_to_reference(convertImpl);
            Convert href = ConvertHelper.narrow(ref);
//get naming context
            org.omg.CORBA.Object objRef =

orb.resolve_initial_references("NameService");
// cast the generic object reference to a proper type
            NamingContextExt ncRef =

NamingContextExtHelper.narrow(objRef);
//bind the name "Convert" with naming service
            NameComponent path[] =
ncRef.to_name("Convert");
```

```
            ncRef.rebind(
                    path,
                    href);
// wait for invocations from client
            orb.run();
        } catch (Exception ex) {
            System.err.println("ERROR: " + ex);
            ex.printStackTrace(System.out);
        }
    }
}
```

11. Buat File "ConvertClient.java" Dan Letakkan Di Dalam Folder Project "Corba" (Bukan package "TempConvertApp")

Adalah sebuah class untuk CORBA GUI Client yang mengakses komponen CORBA dari server.

```
import TempConvertApp.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;
import org.omg.CORBA.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;

public class ConvertClient extends JFrame {

    static Convert convertImpl;
    static JTextField input;
    static JTextField output;
    static ConvertClient a;
    static JButton submit, clear;

    public ConvertClient() { //layout the GUI
        Container contentPane = getContentPane();
        contentPane.setLayout(new FlowLayout());
        JLabel l1 = new JLabel("C input:");
        JLabel l2 = new JLabel("F output");
        input = new JTextField(10);
        output = new JTextField(10);
        submit = new JButton("SUBMIT");
        clear = new JButton("Clear");
        submit.addActionListener(new ActionHandler());
        clear.addActionListener(new ActionHandler());
        contentPane.add(l1);
        contentPane.add(input);
        contentPane.add(l2);
        contentPane.add(output);
        contentPane.add(submit);
        contentPane.add(clear);
        setTitle("Client Access");
```

```
            setSize(540, 250);
            show();
        }

        public static void main(String args[]) {
            try {
                a = new ConvertClient();
   // 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 = "Convert";
                convertImpl =
ConvertHelper.narrow(ncRef.resolve_str(name));
            } catch (Exception e) {
                System.out.println("ERROR : " + e);
                e.printStackTrace(System.out);
            }
        }

        class ActionHandler implements ActionListener {

            public void actionPerformed(ActionEvent e) {
                try {
                    if (e.getSource() == submit) {
                        System.out.println("Obtained a handle
on server object:");
                        String temp = input.getText();
                        double a = Double.parseDouble(temp);
                        double result = convertImpl.cToF(a);
                        output.setText("" + result);
                    } else if (e.getSource() == clear) {
                        input.setText("");
                        output.setText("");
                    }
                } catch (Exception ex) {
                    System.out.println("ERROR : " + e);
                    ex.printStackTrace(System.out);
                }
            }
        }
    }
```

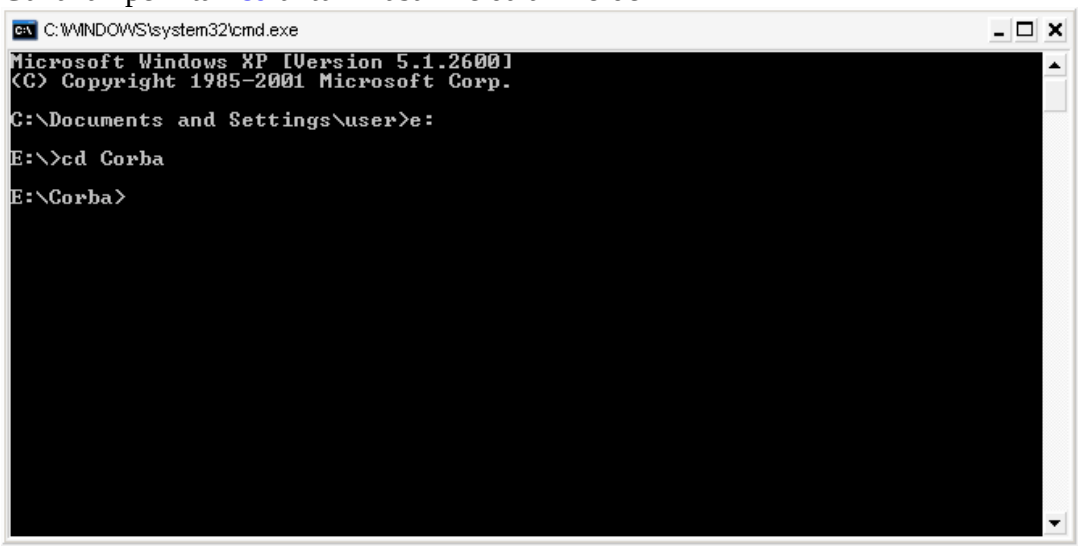
12. Setelah Membuat Code-nya, Jalankan CommandPrompt atau cmd

Proses *compile* dan *running* aplikasi menggunakan *console* MS-DOS karena melalui Operating System Windows XP, jika melalui Linux, gunakanlah terminal.

13. Masuk Ke Alamat Direktori Project Corba

Pada percobaan yang saya lakukan, project Corba saya letakkan di alamat direktori E:\Corba

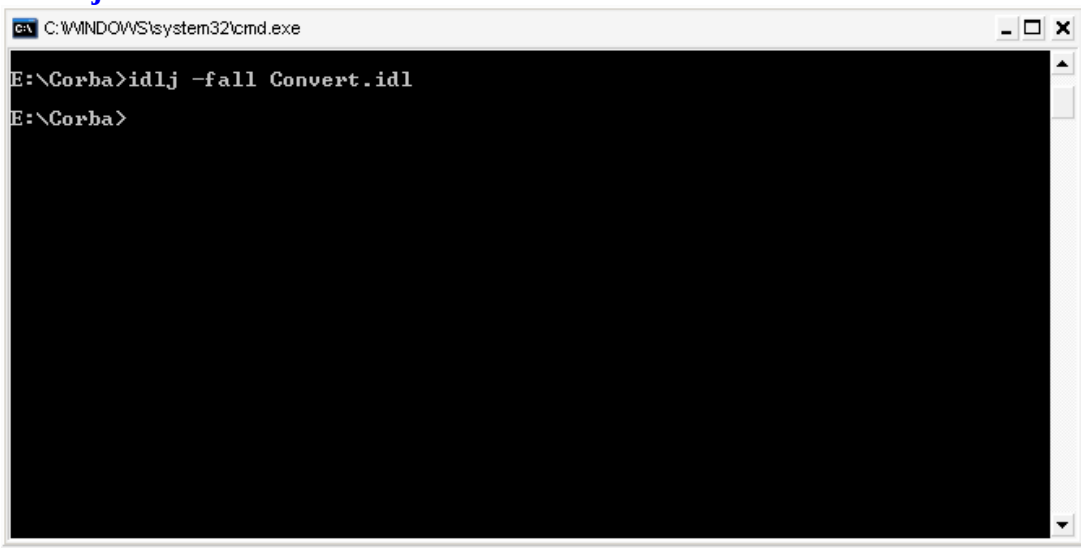
Gunakan perintah **cd** untuk masuk ke dalam folder



14. Compile File IDL

Jalankan perintah berikut untuk mengcompile file **Convert.idl**

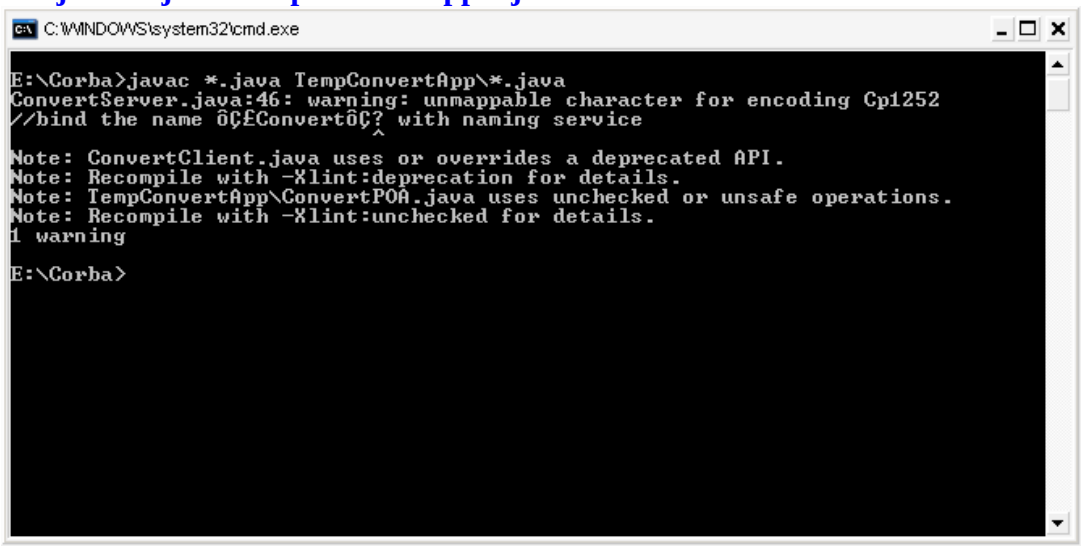
```
idlj -fall Convert.idl
```



15. Compile Semua Java File

Ikuti perintah berikut untuk mengcompile semua file .java yang ada pada project Corba ini

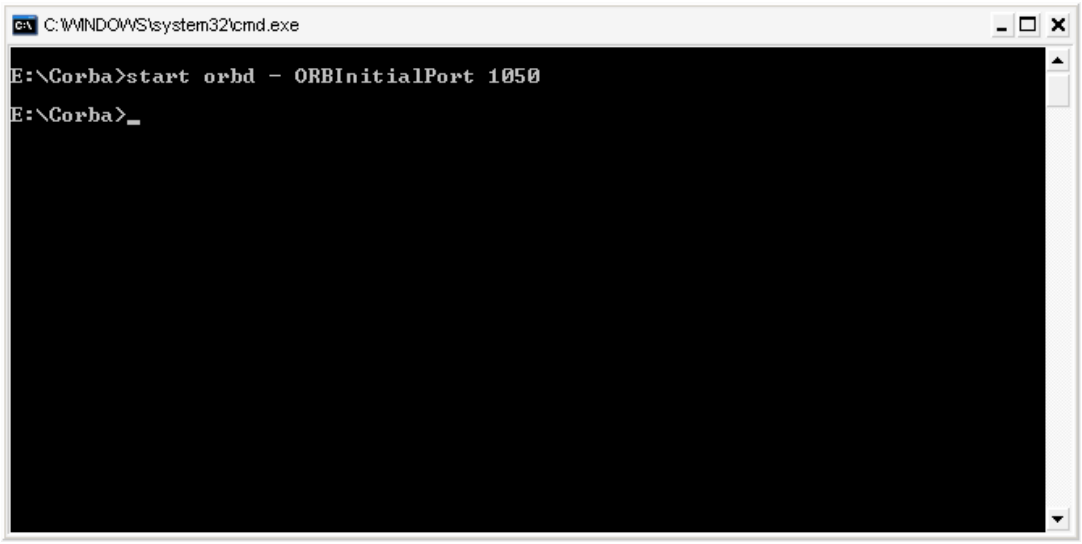
javac *.java TempConvertApp*.java



Terlihat seperti error, ini hanya warning, tapi tidak apa-apa, hiraukan saja. Lanjutkan untuk step berikutnya.

16. Jalankan orbd

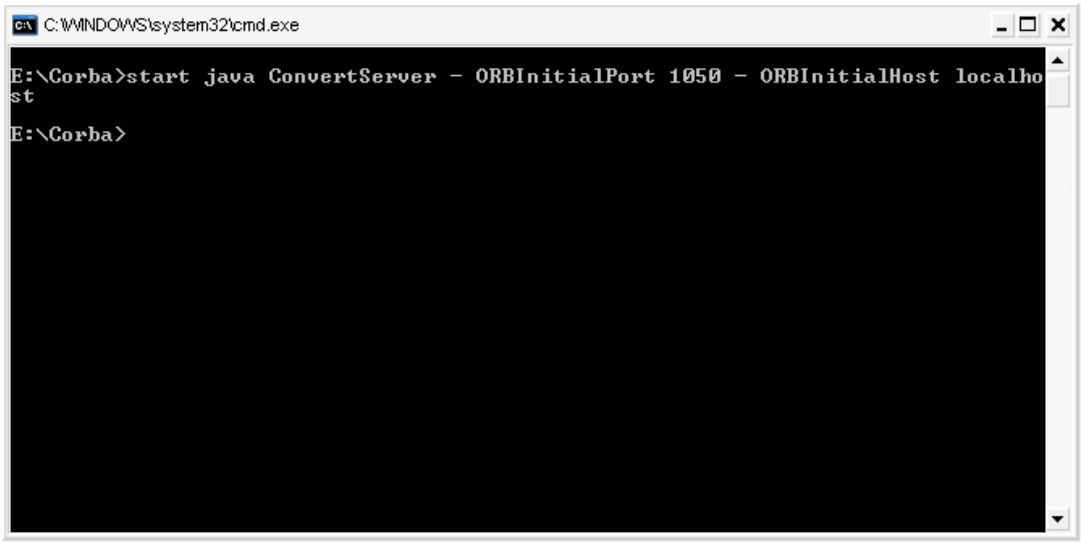
start orbd - ORBInitialPort 1050



Maka akan muncul DOS baru yang tidak menampilkan apa-apa, ini menyatakan bahwa **orbd** sedang dalam keadaan *running*. Jangan tutup jendela ini.

17. Jalankan ConvertServer

start java ConvertServer - ORBInitialPort 1050 - ORBInitialHost localhost



Akan muncul DOS baru, menyatakan bahwa java sedang running dengan memproses **ConvertServer.java**

18. Jalankan Aplikasi Client

java ConvertClient - ORBInitialPort 1050 - ORBInitialHost localhost

19. Akan Muncul Frame Aplikasi Client

20. Silahkan Lakukan Eksekusi Convert Suhu Celcius ke Fahrenheit

DOWNLOAD PROJECT

Silahkan download source code project ConverterCORBA tersebut di link berikut:
http://ilmukomputer.org/wp-content/uploads/2010/12/ConverterCORBA.zip

RESOURCE

[1] Andy Ju An Wang and Kai Qian, "*Component-Oriented Programming*," Wiley-Interscience. Southern Polytechnic State University, Marietta, Georgia, 2005.

=0=0=0=0=0=0=0=0= Selamat Mencoba =0=0=0=0=0=0=0=0=

Thank's to:

- Rasa syukur selalu pantas dipanjatkan kepada ALLAH SWT yang telah memberikan saya kesempatan untuk tetap menuntut ilmu sebanyak-banyaknya.

- Kepada orang tuaku yang terus mendoakan dan membimbingku. Semoga aku bisa membalas kebaikan orang tuaku. AMIN...!!!

- Dosen-dosen yang telah membimbingku

Biografi Penulis

Mudafiq Riyan Pratama. Lahir di Jember pada tanggal 9 Mei 1989. Memulai pendidikan TK dan SD di Jenggawah. Menempuh SMP di SMPN 6 Jember yang kemudian dilanjutkan ke SMAN 2 Jember. Dan saat ini sedang menempuh kuliah S1 jurusan Teknik Informatika di Universitas Muhammadiyah Malang angkatan 2007.

Didunia maya, penulis lebih sering memakai nama **Dhafiq Sagara**.
YM : mudafiq.riyan
FB : ray_dafier@yahoo.co.id