Bengkel J2EE/Linux

# APLIKASI ENTERPRISE DENGAN JAVA

By ekobs@developerforce.net
Version 0.1.1
Last update 22 Oct 2003

Untuk release terbaru, question, troubleshoot dan feedback
kami undang Anda bergabung dengan
Masyarakat J2EE/Linux Indonesia
Bisa dilakukan dengan mengirim email ke

**jlinux-subscribe@yahoogroups.com**

PENGANTAR

Bagaimana membuat aplikasi enterprise dengan teknologi Java ?

# Table of Contents

# 1JBoss

JBoss adalah enterprise application server di mana Anda bisa men-deploy EJB. Setelah di-deploy di JBoss, EJB Anda akan siap bekerja di dalam JBoss melayani client.
Untuk bisa menjalankan JBoss, Anda membutuhkan Java Development Kit. Untuk meng-install JBoss, Anda bisa men-download binary dari http://www.jboss.org. Anda bisa men-download installation file dalam format .zip atau .tar.gz. Yang Anda perlu lakukan adalah men-decompress file ini ke sebuah directory.

Directory di mana JBoss di-install dikenal sebagai JBOSS_HOME.

## *CS-081-011*

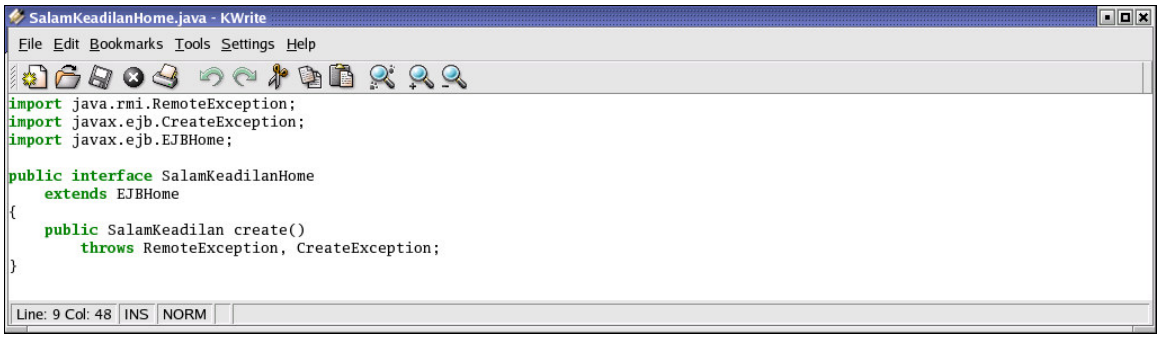
Sebuah praktikum dengan Stateless Session Bean. Bean menyediakan sebuah method yang jika di-invoke akan mengembalikan satu String.

**Persiapan**
Jalankan JBoss jika belum
Buat sebuah directory untuk bekerja
misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis SalamKeadilanHome.java, simpan di-directory yang sudah dipersiapkan.

**`SalamKeadilanHome.java`**

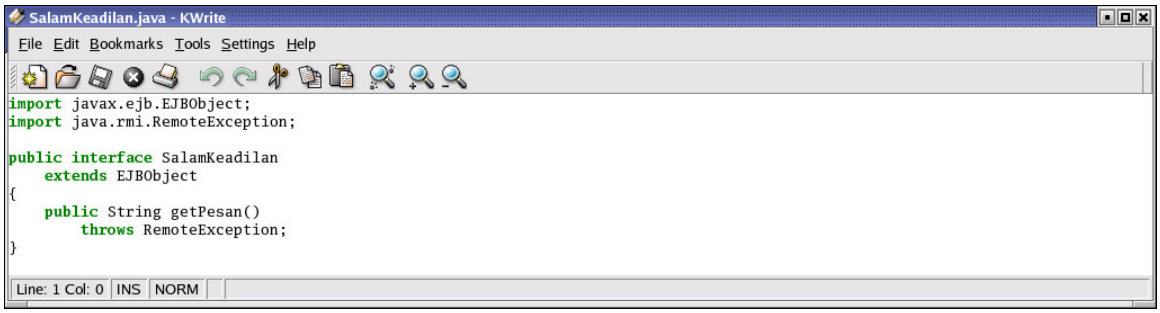
```
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;

public interface SalamKeadilanHome
    extends EJBHome
{
    public SalamKeadilan create()
        throws RemoteException, CreateException;
}
```




**Langkah ke-2**
Tulis SalamKeadilan.java, simpan di directory yang sudah dipersiapkan

**`SalamKeadilan.java`**

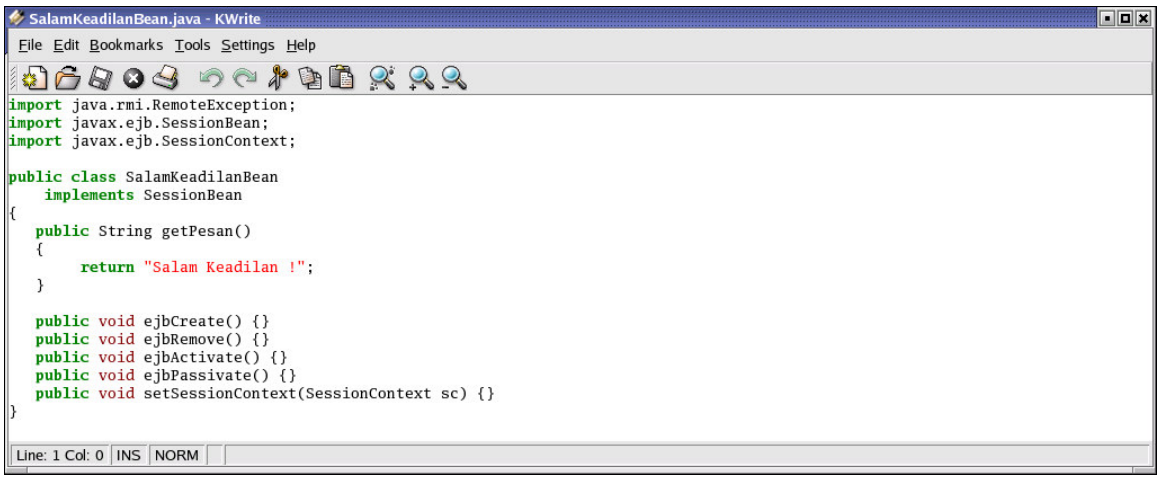
```
import javax.ejb.EJBObject;
import java.rmi.RemoteException;

public interface SalamKeadilan
    extends EJBObject
{
    public String getPesan()
        throws RemoteException;
}
```




**Langkah ke-3**
Tulis SalamKeadilanBean.java, simpan di directory yang sudah disiapkan

**`SalamKeadilanBean.java`**

```
import java.rmi.RemoteException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;

public class SalamKeadilanBean
     implements SessionBean
{
    public String getPesan()
    {
          return "Salam Keadilan !";
    }

    public void ejbCreate() {}
    public void ejbRemove() {}
    public void ejbActivate() {}
    public void ejbPassivate() {}
    public void setSessionContext(SessionContext sc) {}
}
```




**Langkah ke-4**
Buka sebuah terminal dan change directory ke directory di atas

**Langkah ke-5**
Compile ...

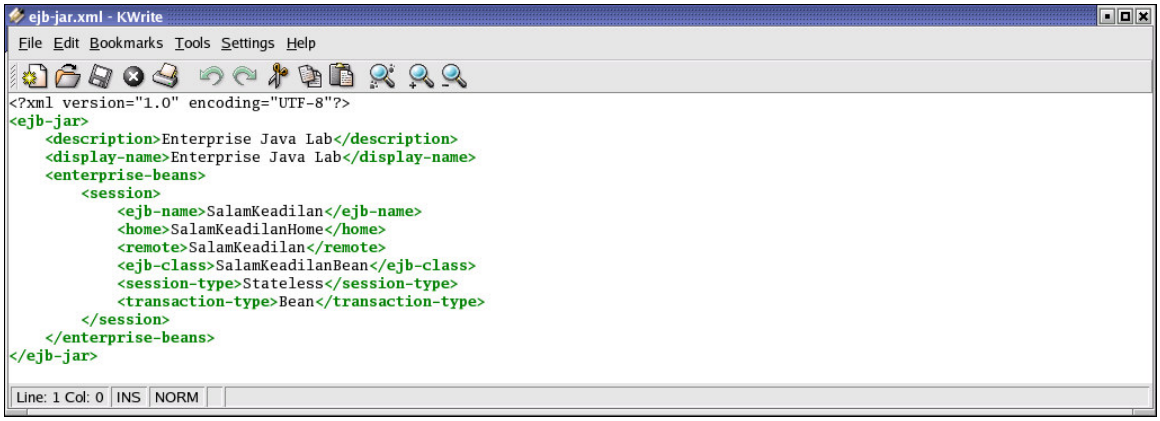
```
$ export CLASSPATH=.:/home/lab/jboss-3.0.4/client/jbossall-client.jar
$ javac *.java

$ ls *.class
SalamKeadilanBean.class  SalamKeadilan.class  SalamKeadilanHome.class
```

**Langkah ke-6**
Buat sub-directory META-INF

**Langkah ke-7**
Tulis ejb-jar.xml, simpan ke sub directory META-INF

**`META-INF/ejb-jar.xml`**

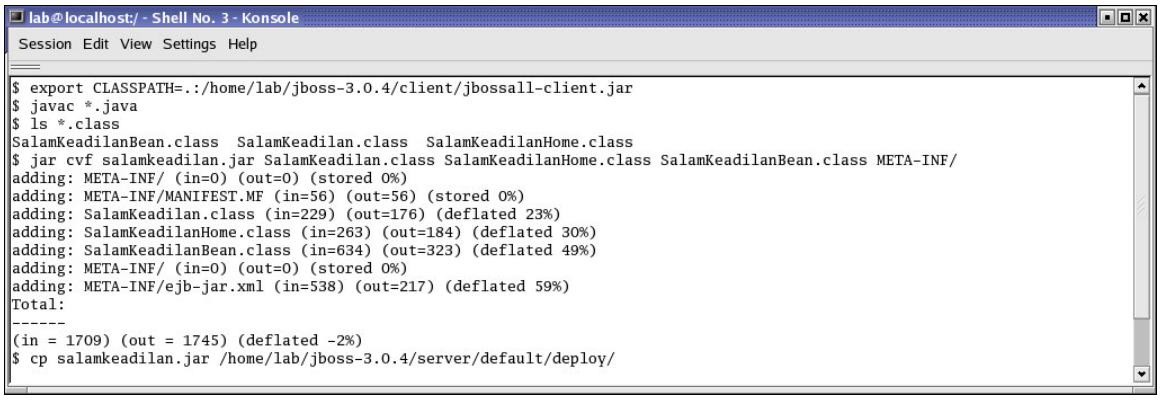
```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar>
    <description>Enterprise Java Lab</description>
    <display-name>Enterprise Java Lab</display-name>
    <enterprise-beans>
        <session>
            <ejb-name>SalamKeadilan</ejb-name>
            <home>SalamKeadilanHome</home>
            <remote>SalamKeadilan</remote>
            <ejb-class>SalamKeadilanBean</ejb-class>
            <session-type>Stateless</session-type>
            <transaction-type>Bean</transaction-type>
        </session>
    </enterprise-beans>
</ejb-jar>
```




**Langkah ke-8**
Gabungkan ke dalam sebuah jar

```
$ jar cvf salamkeadilan.jar SalamKeadilan.class SalamKeadilanHome.class

adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/MANIFEST.MF (in=56) (out=56) (stored 0%)
adding: SalamKeadilan.class (in=229) (out=176) (deflated 23%)
adding: SalamKeadilanHome.class (in=263) (out=184) (deflated 30%)
adding: SalamKeadilanBean.class (in=634) (out=323) (deflated 49%)
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/ejb-jar.xml (in=538) (out=217) (deflated 59%)
Total:
------
```

```
(in = 1709) (out = 1745) (deflated -2%)
```

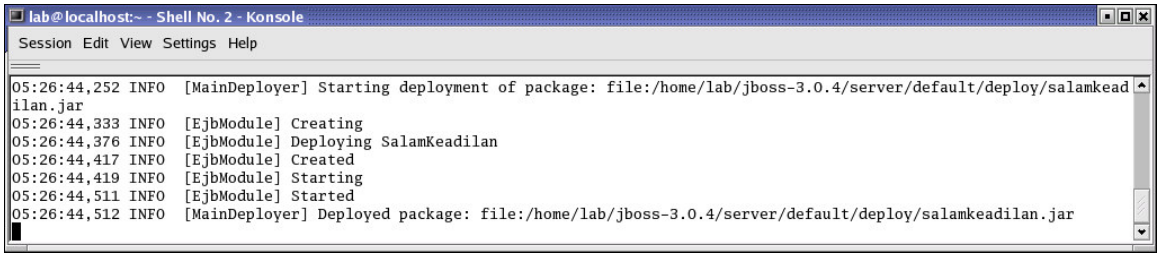



**Langkah ke-9**
Copy ke dalam deployment directory dari jBoss

```
$ cp salamkeadilan.jar /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
05:26:44,252 INFO [MainDeployer] Starting deployment of package:
file:/home/lab/jboss-3.0.4/server/default/deploy/salamkeadilan.jar
05:26:44,333 INFO [EjbModule] Creating
05:26:44,376 INFO [EjbModule] Deploying SalamKeadilan
05:26:44,417 INFO [EjbModule] Created
05:26:44,419 INFO [EjbModule] Starting
05:26:44,511 INFO [EjbModule] Started
05:26:44,512 INFO [MainDeployer] Deployed package:
file:/home/lab/jboss-3.0.4/server/default/deploy/salamkeadilan.jar
```




Module EJB telah di-deploy di atas JBoss enterprise application server dan siap melayani request dari client.

untuk mengakses Stateless Session Bean dari sebuah stand alone Java application ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis SalamKeadilanApp.java, simpan di-directory yang sudah dipersiapkan.

**`SalamKeadilanApp.java`**

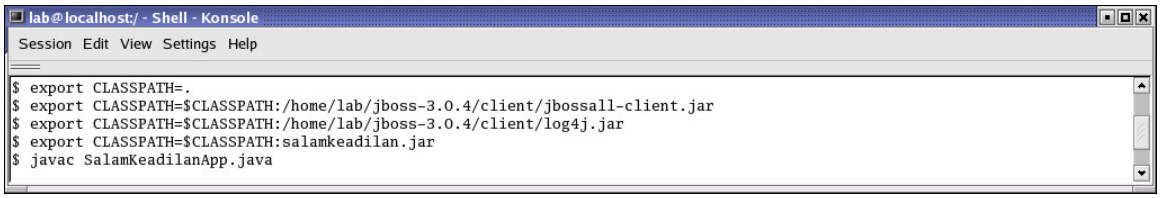
```
import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;

public class SalamKeadilanApp
{
    public static void main(String[] args)
    {
        try
        {
            Properties props = new Properties();
            props.setProperty(
                "java.naming.factory.initial",
                "org.jnp.interfaces.NamingContextFactory"
                );
            props.setProperty(
                "java.naming.provider.url",
                "localhost:1099"
                );
            InitialContext jndiContext
                = new InitialContext(props);
            Object ref  = jndiContext.lookup("SalamKeadilan");
            SalamKeadilanHome home
                = (SalamKeadilanHome)
            PortableRemoteObject.narrow
                (ref, SalamKeadilanHome.class);
            SalamKeadilan ejb = home.create();

            String pesan = ejb.getPesan();

            System.out.println(pesan);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

**Langkah ke-2**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-3**
Compile ...

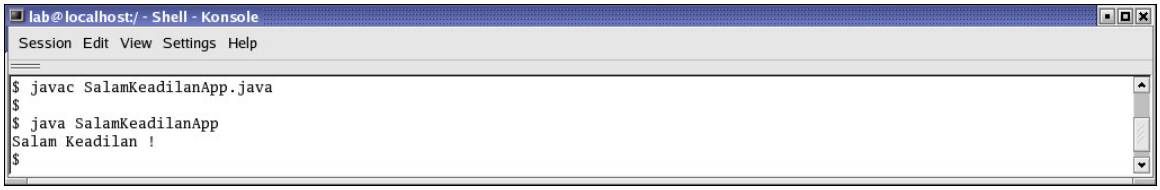
```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$ export CLASSPATH=$CLASSPATH:salamkeadilan.jar
$ javac SalamKeadilanApp.java
```

akan dihasilkan SalamKeadilanApp.class




**Langkah ke-4**
Launch ...

```
$ java SalamKeadilanApp
Salam keadilan !
```



## *CS-081-013*

untuk mengakses Stateless Session Bean dari sebuah aplikasi Swing ....

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis SalamKeadilanJFrame.java, simpan di-directory yang sudah dipersiapkan.

### `SalamKeadilanJFrame.java`

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;

public class SalamKeadilanJFrame
    extends JFrame
    implements ActionListener
{
    JLabel labelPesan;

    public SalamKeadilanJFrame()
    {
        JButton tombolPesan
            = new JButton("Tombol Pesan");
        tombolPesan.addActionListener(this);
        getContentPane()
                    .add(tombolPesan, BorderLayout.NORTH);

        labelPesan = new JLabel();
        getContentPane()
                    .add(labelPesan, BorderLayout.CENTER);

        WindowListener wlSalamKeadilanFrame =
            new WindowAdapter()
        {
            public void windowClosing(WindowEvent we)
            {
                System.exit(0);
            }
        } ;
        addWindowListener(wlSalamKeadilanFrame);
    }

    public void actionPerformed(ActionEvent ae)
    {
            try
            {
                Properties props = new Properties();
                props.setProperty(
                    "java.naming.factory.initial",
                    "org.jnp.interfaces.NamingContextFactory"
                    );
                props.setProperty(
```

```
                    "java.naming.provider.url",
                    "localhost:1099"
                    );
                InitialContext jndiContext
                    = new InitialContext(props);
                Object ref
                    = jndiContext.lookup("SalamKeadilan");
                SalamKeadilanHome home
                    = (SalamKeadilanHome)
                    PortableRemoteObject.narrow
                        (ref, SalamKeadilanHome.class);
                SalamKeadilan ejb = home.create();

                String pesan = ejb.getPesan();
                labelPesan.setText(pesan);
            }
            catch(Exception e)
            {
                e.printStackTrace();
            }

    }
}
```

**Langkah ke-2**
Tulis SalamKeadilanSwingApp.java, simpan di-directory yang sudah dipersiapkan.

### SalamKeadilanSwingApp.java

```
import javax.swing.*;

public class SalamKeadilanSwingApp
{
    public static void main(String[] args)
    {
        JFrame frame = new SalamKeadilanJFrame();

        String lookAndFeel
            = "javax.swing.plaf.metal.MetalLookAndFeel";
        if(args.length>0)
        {
            if(args[0].equals("metal"))
            {
                //
            }
            else
            if(args[0].equals("window"))
            {
                lookAndFeel
                =
                "com.sun.java.swing.plaf.windows.WindowsLookAndFeel";
            }
            else
            if(args[0].equals("motif"))
            {
                lookAndFeel
                = "com.sun.java.swing.plaf.motif.MotifLookAndFeel";
            }
        }

        try
        {
                    UIManager.setLookAndFeel(lookAndFeel);
```

```
                    SwingUtilities.updateComponentTreeUI(frame);
            }
                catch (Exception e)
            {
                e.printStackTrace();
            }

            frame.setLocation(300, 300);
            frame.setSize(300, 200);
            frame.show();
        }
}
```

**Langkah ke-3**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

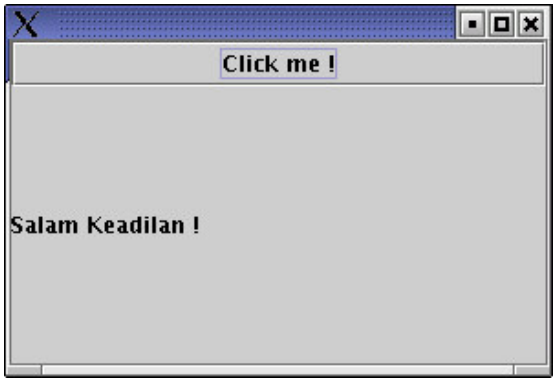
**Langkah ke-4**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$ export CLASSPATH=$CLASSPATH:salamkeadilan.jar
$ javac SalamKeadilanApp.java
```

akan dihasilkan SalamKeadilanApp.class

**Langkah ke-4**
Launch ...

```
$ java SalamKeadilanSwingApp
```



## *CS-071-014*

Mengakses EJB dari Servlet !

**Prasyarat**
Anda harus melakukan praktikum CS-071-011 terlebih dahulu ...

**Langkah**
**Langkah ke-1**
Copy library yang diperlukan ke Context myjava ...

```
$           cp           /home/lab/jboss-3.0.4/client/jbossall-client.jar
/home/lab/myjava/WEB-INF/lib/
$ cp salamkeadilan.jar /home/lab/myjava/WEB-INF/lib/
```

**Langkah ke-2**
Tulis SalamKeadilanServlet.java, simpan di-directory yang sudah dipersiapkan WEB-INF/classes

### `SalamKeadilanServlet.java`

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;

public class SalamKeadilanServlet extends HttpServlet
{

    public void service(HttpServletRequest request,
                        HttpServletResponse response)
        throws IOException, ServletException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        try
        {
            Properties props = new Properties();
            props.setProperty(
                "java.naming.factory.initial",
                "org.jnp.interfaces.NamingContextFactory"
                );
            props.setProperty(
                "java.naming.provider.url",
                "localhost:1099"
                );
            InitialContext jndiContext
                = new InitialContext(props);
            Object ref  = jndiContext.lookup("SalamKeadilan");
            SalamKeadilanHome home
                = (SalamKeadilanHome)
                PortableRemoteObject.narrow
                (ref, SalamKeadilanHome.class);
            SalamKeadilan ejb = home.create();
```

```
            String pesan = ejb.getPesan();
            out.println("<html><body>");
            out.println(pesan);
            out.println("</body></html>");
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }

    }
}
```

**Langkah ke-3**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$          export          CLASSPATH=$CLASSPATH:/home/lab/jakarta-tomcat-
4.1.18/common/lib/servlet.jar
$ export CLASSPATH=$CLASSPATH:salamkeadilan.jar

$ javac WEB-INF/classes/SalamKeadilanServlet.java
```

**Langkah ke-4**
Tulis web.xml yang baru ...

**`WEB-INF/web.xml`**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
     PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>

    <servlet>
        <servlet-name>SalamKeadilanServlet</servlet-name>
        <servlet-class>
           SalamKeadilanServlet
        </servlet-class>
    </servlet>

           <servlet-mapping>
        <servlet-name>SalamKeadilanServlet</servlet-name>
        <url-pattern>/SalamKeadilanServlet</url-pattern>
    </servlet-mapping>
</web-app>
```

**Langkah ke-5**
Terpantau di log dari Tomcat :

```
2003-10-22 07:45:04 StandardContext[/myjava]: Stopping
2003-10-22 07:45:04 StandardContext[/myjava]: Stopping filters
2003-10-22   07:45:04   StandardContext[/myjava]:   Processing   standard
container shutdown
2003-10-22  07:45:04  ContextConfig[/myjava]:  ContextConfig:  Processing
STOP
2003-10-22  07:45:04  StandardContext[/myjava]:  Sending  application  stop
events
2003-10-22 07:45:04 StandardContext[/myjava]: Stopping complete
```

```
2003-10-22 07:45:04 StandardContext[/myjava]: Starting
2003-10-22 07:45:04 StandardContext[/myjava]: Processing start(),
current available=false
2003-10-22 07:45:04 StandardContext[/myjava]: Processing standard
container startup
2003-10-22 07:45:04 WebappLoader[/myjava]: Deploying class repositories
to work directory /home/lab/jakarta-tomcat-
4.1.18/work/Standalone/localhost/myjava
2003-10-22 07:45:04 WebappLoader[/myjava]: Deploy class files /WEB-
INF/classes to /home/lab/myjava/WEB-INF/classes
2003-10-22 07:45:04 WebappLoader[/myjava]: Deploy JAR /WEB-
INF/lib/mm.mysql-2.0.8-bin.jar to /home/lab/myjava/WEB-INF/lib/mm.mysql-
2.0.8-bin.jar
2003-10-22 07:45:04 WebappLoader[/myjava]: Reloading checks are enabled
for this Context
2003-10-22 07:45:04 Authenticator[/myjava]: No SingleSignOn Valve is
present
2003-10-22 07:45:04 ContextConfig[/myjava]: ContextConfig: Processing
START
2003-10-22 07:45:04 StandardContext[/myjava]: Setting deployment
descriptor public ID to '-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN'
2003-10-22 07:45:04 StandardContext[/myjava]: Setting deployment
descriptor public ID to '-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN'
2003-10-22 07:45:04 ContextConfig[/myjava]: Accumulating TLD resource
paths
2003-10-22 07:45:04 ContextConfig[/myjava]: Scanning JAR at resource
path '/WEB-INF/lib/mm.mysql-2.0.8-bin.jar'
2003-10-22 07:45:04 ContextConfig[/myjava]: Pipline Configuration:
2003-10-22 07:45:04 ContextConfig[/myjava]:
org.apache.catalina.authenticator.BasicAuthenticator/1.0
2003-10-22 07:45:04 ContextConfig[/myjava]:
org.apache.catalina.core.StandardContextValve/1.0
2003-10-22 07:45:04 ContextConfig[/myjava]: ======================
2003-10-22 07:45:04 NamingContextListener[/Standalone/localhost/myjava]:
Creating JNDI naming context
2003-10-22 07:45:04 StandardManager[/myjava]: Seeding random number
generator class java.security.SecureRandom
2003-10-22 07:45:04 StandardManager[/myjava]: Seeding of random number
generator has been completed
2003-10-22 07:45:04 StandardContext[/myjava]: Posting standard context
attributes
2003-10-22 07:45:04 StandardContext[/myjava]: Configuring application
event listeners
2003-10-22 07:45:04 StandardContext[/myjava]: Sending application start
events
2003-10-22 07:45:04 StandardContext[/myjava]: Starting filters
2003-10-22 07:45:04 StandardWrapper[/myjava:default]: Loading container
servlet default
2003-10-22 07:45:04 StandardWrapper[/myjava:invoker]: Loading container
servlet invoker
2003-10-22 07:45:04 StandardContext[/myjava]: Starting completed
```

**Langkah ke-6**

Servlet telah di-deploy di atas Tomcat web application server, siap melayani request dari browser. melalui URL :

```
http://localhost:8081/myjava/SalamKeadilanServlet
```


Mozilla
http://localhost:8081/myjava/SalamKeadilanServlet
Search
Salam Keadilan !
Document: Done (0.674 secs)

## *CS-081-015*

Contoh lain tentang Stateless Session Bean

**Persiapan**
Jalankan JBoss jika belum
Buat sebuah directory untuk bekerja
misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis MyCalculatorHome.java, simpan di sub-directory calculator di bawah directory yang sudah dipersiapkan.

### `calculator/MyCalculatorHome.java`

```
package calculator;

import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;

public interface MyCalculatorHome
    extends EJBHome
{
    public MyCalculator create()
        throws RemoteException, CreateException;
}
```

**Langkah ke-2**
Tulis MyCalculator.java, simpan di sub-directory calculator di bawah directory yang sudah dipersiapkan.

### `calculator/MyCalculator.java`

```
package calculator;

import javax.ejb.EJBObject;
import java.rmi.RemoteException;

public interface MyCalculator
    extends EJBObject
{
    public double add(double x, double y)
        throws RemoteException;
    public double subscribe(double x, double y)
        throws RemoteException;
    public double multiply(double x, double y)
        throws RemoteException;
    public double divide(double x, double y)
        throws RemoteException;
}
```

**Langkah ke-3**

Tulis MyCalculatorBean.java, simpan di sub-directory calculator di bawah directory yang sudah dipersiapkan.

**calculator/MyCalculatorBean.java**

```
package calculator;

import java.rmi.RemoteException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;

public class MyCalculatorBean
    implements SessionBean
{
    public double add(double x, double y)
    {
        return x + y;
    }
    public double subscribe(double x, double y)
    {
        return x - y;
    }
    public double multiply(double x, double y)
    {
        return x * y;
    }
    public double divide(double x, double y)
    {
        return x / y;
    }

    public void ejbCreate() {}
    public void ejbRemove() {}
    public void ejbActivate() {}
    public void ejbPassivate() {}
    public void setSessionContext(SessionContext sc) {}
}
```

**Langkah ke-4**
Buka sebuah terminal dan change directory ke directory di atas

**Langkah ke-5**
Compile ...

```
$ export CLASSPATH=.:/home/lab/jboss-3.0.4/client/jbossall-client.jar
$ javac calculator/*.java
```

**Langkah ke-6**
Buat sub-directory META-INF

**Langkah ke-7**
Tulis ejb-jar.xml, simpan ke sub directory META-INF

**META-INF/ejb-jar.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar>
    <description>Enterprise Java Lab</description>
    <display-name>Enterprise Java Lab</display-name>
```

```
    <enterprise-beans>
        <session>
            <ejb-name>MyCalculator</ejb-name>
            <home>calculator.MyCalculatorHome</home>
            <remote>calculator.MyCalculator</remote>
            <ejb-class>calculator.MyCalculatorBean</ejb-class>
            <session-type>Stateless</session-type>
            <transaction-type>Bean</transaction-type>
        </session>
    </enterprise-beans>
</ejb-jar>
```

**Langkah ke-8**
Gabungkan ke dalam sebuah jar

```
$ jar cvf calculator.jar calculator/*.class META-INF
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/MANIFEST.MF (in=56) (out=56) (stored 0%)
adding: calculator/MyCalculator.class (in=304) (out=206) (deflated 32%)
adding: calculator/MyCalculatorBean.class (in=760) (out=358) (deflated
52%)
adding: calculator/MyCalculatorHome.class (in=282) (out=195) (deflated
30%)
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/ejb-jar.xml (in=568) (out=228) (deflated 59%)
Total:
------
(in = 1959) (out = 1892) (deflated 3%)
```

**Langkah ke-9**
Copy ke dalam deployment directory dari jBoss

```
$ cp calculator.jar /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
1:05:06,744 INFO [MainDeployer] Starting deployment of package:
file:/home/lab/jboss-3.0.4/server/default/deploy/calculator
.jar
01:05:06,943 INFO [EjbModule] Creating
01:05:07,049 INFO [EjbModule] Deploying MyCalculator
01:05:07,094 INFO [EjbModule] Created
01:05:07,098 INFO [EjbModule] Starting
01:05:07,261 INFO [EjbModule] Started
01:05:07,264 INFO [MainDeployer] Deployed package:
file:/home/lab/jboss-3.0.4/server/default/deploy/calculator.jar
```

Module EJB telah di-deploy di atas JBoss enterprise application server dan siap melayani request dari client.

untuk mengakses Stateless Session Bean dari sebuah stand alone Java application ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis CalculatorApp.java, simpan di sub-directory app di bawah directory yang sudah dipersiapkan.

**app/CalculatorApp.java**

```
package app;

import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;

import calculator.*;

public class CalculatorApp
{
    public static void main(String[] args)
    {
        try
        {
            Properties props = new Properties();
            props.setProperty(
                "java.naming.factory.initial",
                "org.jnp.interfaces.NamingContextFactory"
                );
            props.setProperty(
                "java.naming.provider.url",
                "localhost:1099"
                );
            InitialContext jndiContext = new InitialContext(props);
            Object ref  = jndiContext.lookup("MyCalculator");
            MyCalculatorHome home = (MyCalculatorHome)
            PortableRemoteObject.narrow (ref, MyCalculatorHome.class);
            MyCalculator calc = home.create();


            double x = 7.5;
            double y = 3.3;

            System.out.println("x     = " + x);
            System.out.println("y     = " + y);
            System.out.println("x + y = " + calc.add(x,y));
            System.out.println("x - y = " + calc.subscribe(x,y));
            System.out.println("x * y = " + calc.multiply(x,y));
            System.out.println("x / y = " + calc.divide(x,y));

        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
```

```
}
```

**Langkah ke-2**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-3**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$ export CLASSPATH=$CLASSPATH:calculator.jar


$ javac app/CalculatorApp.java
```

akan dihasilkan CalculatorApp.class

**Langkah ke-4**
Launch ...

```
$ java app.CalculatorApp
x     = 7.5
y     = 3.3
x + y = 10.8
x - y = 4.2
x * y = 24.75
x / y = 2.272727272727273
```

## *CS-081-017*

untuk mengakses Stateless Session Bean dari sebuah aplikasi Swing ....

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis CalculatorJFrame.java, simpan di sub-directory swing di bawah directory yang sudah dipersiapkan.

### `swing/CalculatorJFrame.java`

```
package swing;

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;

import calculator.*;

public class CalculatorJFrame
        extends JFrame
        implements ActionListener
{
        JLabel resultLbl;
        JTextField xTxt;
        JTextField yTxt;

        public CalculatorJFrame()
        {
            xTxt = new JTextField();
            yTxt = new JTextField();

            JPanel textPnl = new JPanel();
            textPnl.setLayout(new GridLayout(2,2));
            textPnl.add(new JLabel("x = "));
            textPnl.add(xTxt);
            textPnl.add(new JLabel("y = "));
            textPnl.add(yTxt);

            getContentPane().add(textPnl, BorderLayout.NORTH);

            JButton addBtn = new JButton("+");
            JButton subscribeBtn = new JButton("-");
            JButton multiplyBtn = new JButton("*");
            JButton divideBtn = new JButton("/");

            addBtn.addActionListener(this);
            subscribeBtn.addActionListener(this);
            multiplyBtn.addActionListener(this);
            divideBtn.addActionListener(this);

            JPanel buttonPnl = new JPanel();
```

```
            buttonPnl.add(addBtn);
            buttonPnl.add(subscribeBtn);
            buttonPnl.add(multiplyBtn);
            buttonPnl.add(divideBtn);

            getContentPane().add(buttonPnl, BorderLayout.SOUTH);


            resultLbl = new JLabel();
            getContentPane().add(resultLbl, BorderLayout.CENTER);

            WindowListener wlSalamKeadilanFrame =
                    new WindowAdapter()
                    {
                            public void windowClosing(WindowEvent we)
                            {
                                    System.exit(0);
                            }
                    } ;
            addWindowListener(wlSalamKeadilanFrame);
        }

        public void actionPerformed(ActionEvent ae)
        {
            double result = 0;
            try
            {
                String xStr = xTxt.getText();
                double x = Double.parseDouble(xStr);

                String yStr = yTxt.getText();
                double y = Double.parseDouble(yStr);

                Properties props = new Properties();
                props.setProperty(
                    "java.naming.factory.initial",
                    "org.jnp.interfaces.NamingContextFactory"
                    );
                props.setProperty(
                    "java.naming.provider.url",
                    "localhost:1099"
                    );
                InitialContext jndiContext = new InitialContext(props);
                Object ref  = jndiContext.lookup("MyCalculator");
                MyCalculatorHome home = (MyCalculatorHome)
                PortableRemoteObject.narrow (ref,
MyCalculatorHome.class);
                MyCalculator calc = home.create();


                String command = ae.getActionCommand();
                if(command.equals("+"))
                {
                    result =  calc.add(x, y);
                }
                else
                if(command.equals("-"))
                {
                    result =  calc.subscribe(x, y);
                }
                else
                if(command.equals("*"))
                {
                    result =  calc.multiply(x, y);
```

```
                }
                else
                if(command.equals("/"))
                {
                    result =  calc.divide(x, y);
                }


            }
            catch(Exception e)
            {
                e.printStackTrace();
            }

            resultLbl.setText(Double.toString(result));

        }
}
```

**Langkah ke-2**
Tulis CalculatorApp.java, simpan di sub-directory app di bawah directory yang sudah dipersiapkan.

### app/CalculatorSwingApp.java

```
package app;

import javax.swing.*;

import swing.*;

public class CalculatorSwingApp
{
    public static void main(String[] args)
    {
        JFrame frame = new CalculatorJFrame();
        frame.setLocation(300, 300);
        frame.setSize(300, 200);
        frame.show();
    }
}
```

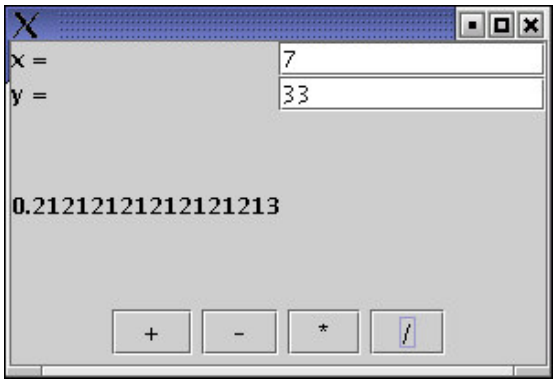
**Langkah ke-3**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-4**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$ export CLASSPATH=$CLASSPATH:calculator.jar
$ javac -sourcepath . app/CalculatorSwingApp.java
```

**Langkah ke-5**
Launch ...

```
$ java app.CalculatorSwingApp
```


x =
7
y =
33
0.21212121212121213
+
-
*
/

## *CS-071-018*

Mengakses EJB dari Servlet !

**Prasyarat**
Anda harus melakukan praktikum CS-071-011 terlebih dahulu ...

**Langkah**
**Langkah ke-1**
Jika belum, copy library yang diperlukan ke Context myjava

```
$           cp           /home/lab/jboss-3.0.4/client/jbossall-client.jar
/home/lab/myjava/WEB-INF/lib/
$ cp calculator.jar /home/lab/myjava/WEB-INF/lib/
```

**Langkah ke-2**
Tulis calculatorForm.html, simpan di-directory yang sudah dipersiapkan.

### `calculatorform.html`

```
<html>
<body>

<script language=JavaScript>

function onAdd()
{
    document.calculatorForm.command.value = '+';
    document.calculatorForm.submit();
}
function onSubscribe()
{
    document.calculatorForm.command.value = '-';
    document.calculatorForm.submit();
}
function onMultiply()
{
    document.calculatorForm.command.value = '*';
    document.calculatorForm.submit();
}
function onDivide()
{
    document.calculatorForm.command.value = '/';
    document.calculatorForm.submit();
}

</script>

<form name=calculatorForm method=POST action=CalculatorServlet>

<input type=hidden name=command>

<table>
    <tr>
    <td>x = </td>
    <td><input type=text name=x>
    </tr>
    <tr>
    <td>y = </td>
```

```
    <td><input type=text name=y>
    </tr>
    <tr>
    <td colspan=2>
    <input type=button value=' + ' onClick="javascript:onAdd()">
    <input type=button value=' - ' onClick="javascript:onSubscribe()">
    <input type=button value=' * ' onClick="javascript:onMultiply()">
    <input type=button value=' / ' onClick="javascript:onDivide()">
    </td>
    </tr>

</table>
</form>

</body>
</html>
```

**Langkah ke-3**
Tulis CalculatorServlet.java, simpan di-directory yang sudah dipersiapkan WEB-INF/classes/servlet

## servlet/CalculationServlet.java

```
package servlet;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;

import calculator.*;

public class CalculatorServlet extends HttpServlet
{

    public void service(HttpServletRequest request,
                        HttpServletResponse response)
        throws IOException, ServletException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        try
        {
            String xStr = request.getParameter("x");
            double x = Double.parseDouble(xStr);

            String yStr = request.getParameter("y");
            double y = Double.parseDouble(yStr);

            Properties props = new Properties();
            props.setProperty(
                "java.naming.factory.initial",
                "org.jnp.interfaces.NamingContextFactory"
                );
            props.setProperty(
                "java.naming.provider.url",
                "localhost:1099"
                );
            InitialContext jndiContext = new InitialContext(props);
```

```
            Object ref  = jndiContext.lookup("MyCalculator");
            MyCalculatorHome home = (MyCalculatorHome)
            PortableRemoteObject.narrow (ref, MyCalculatorHome.class);
            MyCalculator calc = home.create();


            String command = request.getParameter("command");
            double result = 0;
            switch(command.charAt(0))
            {
                case '+' : result = calc.add(x,y); break;
                case '-' : result = calc.subscribe(x,y); break;
                case '*' : result = calc.multiply(x,y); break;
                case '/' : result = calc.divide(x,y); break;
            }

            out.println("<html><body>");
            out.println(result);
            out.println("</body></html>");
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }

    }

}
```

**Langkah ke-4**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$          export          CLASSPATH=$CLASSPATH:/home/lab/jakarta-tomcat-
4.1.18/common/lib/servlet.jar
$ export CLASSPATH=$CLASSPATH:calculator.jar
$           javac           -d           WEB-INF/classes/           WEB-
INF/classes/servlet/CalculatorServlet.java
```

**Langkah ke-5**
Tulis web.xml yang baru ...

**`WEB-INF/web.xml`**

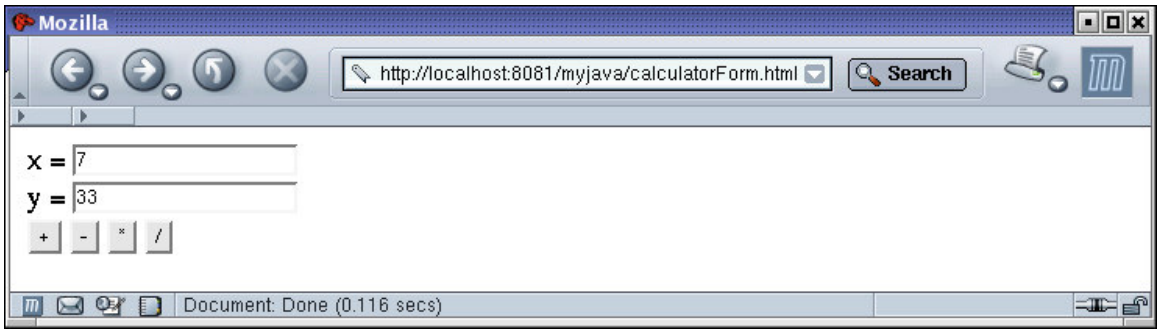
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
     PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>

    <servlet>
        <servlet-name>CalculatorServlet</servlet-name>
        <servlet-class>
           servlet.CalculatorServlet
        </servlet-class>
    </servlet>

           <servlet-mapping>
        <servlet-name>CalculatorServlet</servlet-name>
        <url-pattern>/CalculatorServlet</url-pattern>
```

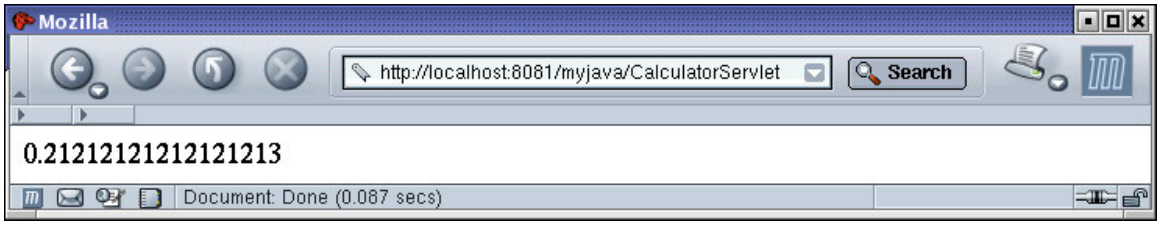
```
    </servlet-mapping>
</web-app>
```

**Langkah ke-6**
Terpantau di log dari Tomcat adanya redeployment ...

**Langkah ke-7**
Servlet telah di-deploy di atas Tomcat web application server, siap melayani request dari browser. melalui URL :

```
http://localhost:8081/myjava/calculatorForm.html
```



# 2Enterprise Java Bean

Enterprise Java Bean (EJB) adalah teknologi untuk mengembangkan komponen di sisi server yang transactional, scalable dan secure untuk aplikasi enterprise.

EJB dirancang untuk mendukung transaction, di mana satu rentetan perubahan data diperlakukan sebagai satu kesatuan. Sebagai contoh, dalam aplikasi e-banking, di mana user melakukan transfer uang dari satu account ke account lain. Dalam operasi ini terdapat dua buah perubahan data setidaknya yaitu pengurangan jumlah uang dari account pengirim, dan penambahan jumlah uang ke account penerima. Dua buah perubahan data ini dipandang sebagai satu kesatuan. Jika salah satu gagal, maka keduanya dibatalkan.

EJB dirancang untuk bisa scalable, yaitu untuk bisa menangani jumlah concurent user yang membesar. Scalability bisa dicapai dengan vertical scalability yaitu dengan menambahkan memory maupun processor dari sebuah mesin, atau dengan membuat cluster di mana EJB di-deploy di beberapa mesin.

EJB juga bisa dirancang untuk secure, dengan menerapkan Role-Based Access Control di mana user-user dengan role tertentu saja yang bisa mengakses komponen-komponen EJB tertentu.

Sebagai sebuah teknologi untuk aplikasi enterprise, EJB juga mendukung location transparancy, di mana EJB bisa di-deploy di enterprise application server yang lokasinya bisa di mana saja. Client dari EJB akan menemukan EJB ini dengan melakukan lookup ke directory server.

Dalam sistem enterprise, bisa terdapat banyak enterprise application server. Sebagai contoh, sebuah server bisa digunakan untuk men-deploy EJB untuk aplikasi finance di divisi keuangan yang berada di New York. Server lain untuk men-deploy EJB untuk aplikasi inventory di cabang di Singapura. Server lain untuk men-deploy EJB untuk aplikasi order management di Jepang.

Untuk mengembangkan EJB, Anda menulis source code, lalu meng-compile-nya. Kemudian Anda menuliskan deployment descriptor. Terakhir Anda men-deploy ke application server.

Berbeda dengan Swing atau JSP yang menyediakan user interface, EJB bekerja di back-end. EJB menyediakan service yang dipanggil oleh client. Client dari EJB bisa berupa stand alone application dengan Swing, ataupun web-based application dengan JSP.

# 3Teknologi Pendukung

RMI
Remote Method Invocation (RMI) digunakan untuk mengembangkan aplikasi terdistribusi dengan Java. Dengan RMI, sebuah obyek yang berada di sebuah JVM, dapat memanggil obyek lain yang berada di JVM yang terpisah.

JNDI
Java Naming and Directory Interface (JNDI) dalam arsitektur EJB digunakan oleh client untuk menemukan komponen EJB.

JDBC
Java Database Connectivity (JDBC) digunakan untuk berinteraksi dengan Database.

JTS
Java Transaction Service (JTS) digunakan untuk menyediakan transaction dalam arsitektur EJB

# 4Tipe EJB

Terdapat 3 tipe utama EJB, yaitu SessionBean, EntityBean dan MessageDriven Bean.

Entity Bean digunakan untuk merepresentasikan business obyek. Contohnya, entity bean digunakan untuk merepresentasikan Product, Order, Student, Course, Employee, dan To Do. Entity Bean tersimpan di database, jika server mengalami crash dia akan survive. Terdapat dua strategi penyimpanan yaitu Bean Managed Persistence dan Container Managed Persistence. Entity Bean adalah di-share oleh banyak client dan transactional.

Session Bean digunakan untuk mereprsentasikan proses, kendali dan alur kerja. Contohnya, session bean digunakan untuk melakukan validasi credit card, mencari ketersediaan jadwal penerbangan, dan menyimpan shopping cart. Session Bean bisa Stateless atau Stateful.
SessionBean tidak disimpan di dalam database, dan tidak akan survive jika server mengalami crash. Session bean hanya diakses oleh satu client, dan transactional.

MessageDriven Bean digunakan untuk menyediakan layanan asynchronous messaging.

# 5Source Code dan Deployment Descriptor

Untuk mengembangkan sebuah Enterprise Java Bean, Anda wajib menuliskan source code untuk Home interface, Remote interface, dan bean implementation. Anda juga wajib menyediakan Deployment descriptor.

Home interface ditulis dengan meng-extends EJBHome. Di home interface, Anda mendeklarasikan method untuk meng-create EJB object. Home interface akan diimplementasikan oleh enterprise application server, dan dikenal sebagai EJB home.

Remote interface ditulis dengan meng-extends EJBObject. Di remote interface, Anda mendeklarasikan method-method yang disediakan untuk melayani client. Di Session Bean, method-method ini bisa berupa business method. Di Entity Bean, method-method ini bisa berupa setter dan getter method.

Remote interface juga akan diimplementasikan oleh enterprise application server. Dalam implementasinya, yang dikenal sebagai EJB object, masing-masing method akan meng-invoke method yang bersesuaian di bean implementation. EJB object menyediakan service-service tambahan seperti security, dan transaction.

Bean implementation adalah Java class di mana Anda mengimplementasikan method-method yang telah Anda deklarasikan di Remote interface, sebagaimana juga di Home interface.

# 6Cara Kerja EJB

Dalam aplikasi EJB, Anda akan bekerja dengan EJB home, EJB object dan bean implementation. EJB home menyediakan method-method create, finder dan remove untuk EJB Object. EJB Object adalah jembatan antara client dengan bean implementation.

Dari sisi client, client tidak bisa meng-create secara langsung bean implementation. Dan client juga tidak pernah berhubungan langsung dengan bean implementation.

Client bisa me-lookup Home object menggunakan Java Naming and Directory Interface. Home object adalah class yang di-generate oleh enterprise application server, yang meng-implement Home interface. Melalui Home object ini, client meng-create EJB Object. Untuk Entity Bean, Home object juga mempunyai finder method dan remove method.

Selanjutnya client bisa meng-invoke method-method di EJB Object. EJB Object adalah class yang di-generate oleh enterprise application server, yang meng-implement Remote interface. Dalam implementasi dari method-method yang dideklarasikan di Remote interface, akan di-invoke method bersesuaian di bean implementation.

# 7Stateless Session Bean

Stateless Session Bean dirancang untuk menyediakan layanan kepada client, tanpa kemampuan memegang conversational state. Layanan disediakan dengan menuliskan method-method yang akan di-invoke oleh client.

Stateless Session Bean disediakan oleh enterprise application server dari sebuah pool. Saat sebuah client meng-create Stateless Session Bean, ia tidak bermakna sebuah instance baru di-create. Enterprise application server bisa saja menggunakan instance yang tersedia di pool. Dan instance ini selalu ada di dalam pool, kecuali saat menjalankan sebuah method untuk sebuah client.

Setiap kali sebuah client memanggil sebuah method dari Stateless Session Bean, maka sebuah instance akan bekerja secara khusus untuk client tsb. Saat method tsb selesai dijalankan, instance tsb akan dikembalikan ke pool. Jika sebuah client memanggil dua method secara berurutan, tidak bisa dipastikan bahwa keduanya dilayani oleh instance yang sama.

Stateless Session Bean tidak akan menyimpan conversational-state. Secara sederhana ini bermakna, instance variable dari Stateless Session Bean tidak bisa digunakan untuk memegang data yang bersifat spesifik untuk sebuah client.

Jika sebuah client memanggil sebuah method, dan di method tersebut memberi nilai ke sebuah instance variable, maka bisa saja terjadi, saat memanggil method berikutnya, client akan menemukan nilai dari instance variable tsb berbeda. Ini bisa dipahami, karena dua method tersebut bisa saja dilayani oleh dua instance yang berbeda.

Untuk mengembangkan sebuah Stateless Session Bean, Anda wajib menuliskan source code untuk Home interface, Remote interface, dan bean implementation. Anda juga wajib menyediakan Deployment descriptor.

## *CS-081-071*

Sebuah praktikum yang menunjukkan tingkah laku Stateless Session Bean

**Persiapan**
Jalankan JBoss jika belum
Buat sebuah directory untuk bekerja
misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis MyFruitHome.java, simpan di-directory yang sudah dipersiapkan.

**`MyFruitHome.java`**

```
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;

public interface MyFruitHome
    extends EJBHome
{
    public MyFruit create()
        throws RemoteException, CreateException;
}
```

**Langkah ke-2**
Tulis MyFruit.java, simpan di directory yang sudah dipersiapkan

**`MyFruit.java`**

```
import javax.ejb.EJBObject;
import java.rmi.RemoteException;

public interface MyFruit
    extends EJBObject
{
    public void setFruit(String fruit)
        throws RemoteException;
    public String getFruit()
        throws RemoteException;
}
```

**Langkah ke-3**
Tulis MyFruitBean.java, simpan di directory yang sudah disiapkan

**`MyFruitBean.java`**

```
import java.rmi.RemoteException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;

public class MyFruitBean
    implements SessionBean
{
   public String fruit;
   public void setFruit(String fruit)
```

```
    {
            this.fruit = fruit;
    }

    public String getFruit()
    {
            return this.fruit;
    }

    public void ejbCreate() {}
    public void ejbRemove() {}
    public void ejbActivate() {}
    public void ejbPassivate() {}
    public void setSessionContext(SessionContext sc) {}
}
```

**Langkah ke-4**
Buka sebuah terminal dan change directory ke directory di atas

**Langkah ke-5**
Compile ...

```
$ export CLASSPATH=.:/home/lab/jboss-3.0.4/client/jbossall-client.jar
$ javac *.java
```

**Langkah ke-6**
Buat sub-directory META-INF

**Langkah ke-7**
Tulis ejb-jar.xml, simpan ke sub directory META-INF

**`META-INF/ejb-jar.xml`**

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar>
    <description>Enterprise Java Lab</description>
    <display-name>Enterprise Java Lab</display-name>
    <enterprise-beans>
        <session>
            <ejb-name>MyStatelessFruit</ejb-name>
            <home>MyFruitHome</home>
            <remote>MyFruit</remote>
            <ejb-class>MyFruitBean</ejb-class>
            <session-type>Stateless</session-type>
            <transaction-type>Bean</transaction-type>
        </session>
    </enterprise-beans>
</ejb-jar>
```

**Langkah ke-8**
Gabungkan ke dalam sebuah jar

```
$ jar cvf myfruit.jar MyFruitHome.class MyFruit.class MyFruitBean.class
META-INF/

adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/MANIFEST.MF (in=56) (out=56) (stored 0%)
adding: MyFruitHome.class (in=245) (out=180) (deflated 26%)
adding: MyFruit.class (in=270) (out=186) (deflated 31%)
```

```
adding: MyFruitBean.class (in=732) (out=352) (deflated 51%)
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/ejb-jar.xml (in=524) (out=224) (deflated 57%)
Total:
------
(in = 1816) (out = 1751) (deflated 3%)
```

**Langkah ke-9**
Copy ke dalam deployment directory dari jBoss

```
$ cp myfruit.jar /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
04:46:41,925 INFO [MainDeployer] Starting deployment of package:
file:/home/lab/jboss-3.0.4/server/default/deploy/myfruit.jar
04:46:42,071 INFO [EjbModule] Creating
04:46:42,167 INFO [EjbModule] Deploying MyStatelessFruit
04:46:42,239 INFO [EjbModule] Created
04:46:42,240 INFO [EjbModule] Starting
04:46:42,445 INFO [EjbModule] Started
04:46:42,446 INFO [MainDeployer] Deployed package:
file:/home/lab/jboss-3.0.4/server/default/deploy/myfruit.jar
```

Module EJB telah di-deploy di atas JBoss enterprise application server dan siap melayani request dari client.

untuk mengamati tingkah laku Stateless Session Bean ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis FruitApp.java, simpan di-directory yang sudah dipersiapkan.

**`FruitApp.java`**

```
public class FruitApp
{
    public static void main(String[] args)
    {
        for(int i=0;i<1000;i++)
        {
            MyThread thread = new MyThread(i);
            thread.start();
        }
    }
}
```

**Langkah ke-2**
Tulis MyThread.java, simpan di-directory yang sudah dipersiapkan.

**`MyThread.java`**

```
import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;


public class MyThread extends Thread
{
    private static final String[] fruitArray
        = { "Apple", "Avocado", "Banana", "Durian", "Grape",
            "Lychee", "Mango", "Orange", "Starfruit", "Soursop" };
    private int threadNo;

    public MyThread(int threadNo)
    {
        super("Thread-" + threadNo);
        this.threadNo = threadNo;
    }

    public void run()
    {

        try
        {
            Properties props = new Properties();
            props.setProperty(
                "java.naming.factory.initial",
                "org.jnp.interfaces.NamingContextFactory"
                );
```

```
            props.setProperty(
                "java.naming.provider.url",
                "localhost:1099"
                );
            InitialContext jndiContext = new InitialContext(props);
            Object ref  = jndiContext.lookup("MyStatelessFruit");
            MyFruitHome home = (MyFruitHome)
            PortableRemoteObject.narrow (ref, MyFruitHome.class);

            MyFruit ejb = home.create();

            String fruitInThread = fruitArray[threadNo%10];

            ejb.setFruit(fruitInThread);

            yield();

            try
            {
                int t = (int)(100*Math.random());
                sleep(t);
            }
            catch(InterruptedException ie)
            {
                ie.printStackTrace();
            }

            String fruitInEJB = ejb.getFruit();

            System.out.println(getName()
                + " : in Thread value = " + fruitInThread
                + " in EJB value = " + fruitInEJB
                + (fruitInEJB.equals(fruitInThread) ?
                    "":" FRUIT IS CHANGED"));
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

**Langkah ke-3**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-4**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$ export CLASSPATH=$CLASSPATH:myfruit.jar
$ javac FruitApp.java
```

akan dihasilkan FruitApp.class  dan MyThread.class

**Langkah ke-5**
Launch ...

```
$ java FruitApp
```

```
Thread-3 : in Thread value = Durian in EJB value = Durian
Thread-5 : in Thread value = Lychee in EJB value = Durian FRUIT IS
CHANGED
Thread-2 : in Thread value = Banana in EJB value = Banana
Thread-82 : in Thread value = Banana in EJB value = Grape FRUIT IS
CHANGED
Thread-84 : in Thread value = Grape in EJB value = Grape
Thread-85 : in Thread value = Lychee in EJB value = Grape FRUIT IS
CHANGED
Thread-83 : in Thread value = Durian in EJB value = Starfruit FRUIT IS
CHANGED
Thread-88 : in Thread value = Starfruit in EJB value = Starfruit
Thread-87 : in Thread value = Orange in EJB value = Grape FRUIT IS
CHANGED
Thread-72 : in Thread value = Banana in EJB value = Banana
```

# 8Statefull Session Bean

Statefull Session Bean dirancang untuk menyediakan layanan kepada client, dengan kemampuan memegang converstational state. Layanan disediakan dengan menuliskan method-method yang akan di-invoke oleh client. Conversational state bisa ditangani melalui member variable di bean implementation.

Statefull Session Bean tidak dikelola oleh enterprise application server dalam sebuah pool. Sebuah Statefull Session Bean selalu di-create untuk sebuah client, dan hanya diperuntukkan untuk sebuah client tertentu. Ia akan di-destroy jika dibuang oleh client.

Saat client memanggil method dari Statefull Session Bean ia bisa menyimpan data ke member variable. Dan data ini akan tetap dipegang oleh Stateful Session Bean.

Untuk mengembangkan sebuah Statefull Session Bean, Anda wajib menuliskan source code untuk Home interface, Remote interface, dan bean implementation. Anda juga wajib menyediakan Deployment descriptor.

## *CS-081-081*

Sebuah praktikum yang menunjukkan tingkah laku Statefull Session Bean

**Persiapan**
Jalankan JBoss jika belum
Buat sebuah directory untuk bekerja
misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis MyFruitHome.java, simpan di-directory yang sudah dipersiapkan.

**`MyFruitHome.java`**

```
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;

public interface MyFruitHome
    extends EJBHome
{
    public MyFruit create()
        throws RemoteException, CreateException;
}
```

**Langkah ke-2**
Tulis MyFruit.java, simpan di directory yang sudah dipersiapkan

**`MyFruit.java`**

```
import javax.ejb.EJBObject;
import java.rmi.RemoteException;

public interface MyFruit
    extends EJBObject
{
    public void setFruit(String fruit)
        throws RemoteException;
    public String getFruit()
        throws RemoteException;
}
```

**Langkah ke-3**
Tulis MyFruitBean.java, simpan di directory yang sudah disiapkan

**`MyFruitBean.java`**

```
import java.rmi.RemoteException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;

public class MyFruitBean
    implements SessionBean
{
   public String fruit;
   public void setFruit(String fruit)
```

```
    {
            this.fruit = fruit;
    }

    public String getFruit()
    {
            return this.fruit;
    }

    public void ejbCreate() {}
    public void ejbRemove() {}
    public void ejbActivate() {}
    public void ejbPassivate() {}
    public void setSessionContext(SessionContext sc) {}
}
```

**Langkah ke-4**
Buka sebuah terminal dan change directory ke directory di atas

**Langkah ke-5**
Compile ...

```
$ export CLASSPATH=.:/home/lab/jboss-3.0.4/client/jbossall-client.jar
$ javac *.java
```

**Langkah ke-6**
Buat sub-directory META-INF

**Langkah ke-7**
Tulis ejb-jar.xml, simpan ke sub directory META-INF

**`META-INF/ejb-jar.xml`**

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar>
    <description>Enterprise Java Lab</description>
    <display-name>Enterprise Java Lab</display-name>
    <enterprise-beans>
        <session>
            <ejb-name>MyStatefulFruit</ejb-name>
            <home>MyFruitHome</home>
            <remote>MyFruit</remote>
            <ejb-class>MyFruitBean</ejb-class>
            <session-type>Stateful</session-type>
            <transaction-type>Bean</transaction-type>
        </session>
    </enterprise-beans>
</ejb-jar>
```

**Langkah ke-8**
Gabungkan ke dalam sebuah jar

```
$ jar cvf myfruit.jar MyFruitHome.class MyFruit.class MyFruitBean.class
META-INF/

adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/MANIFEST.MF (in=56) (out=56) (stored 0%)
adding: MyFruitHome.class (in=245) (out=180) (deflated 26%)
adding: MyFruit.class (in=270) (out=186) (deflated 31%)
```

```
adding: MyFruitBean.class (in=732) (out=352) (deflated 51%)
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/ejb-jar.xml (in=522) (out=224) (deflated 57%)
Total:
------
(in = 1814) (out = 1751) (deflated 3%)
```

**Langkah ke-9**
Copy ke dalam deployment directory dari jBoss

```
$ cp myfruit.jar /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
04:51:53,440 INFO [MainDeployer] Starting deployment of package:
file:/home/lab/jboss-3.0.4/server/default/deploy/myfruit.jar
04:51:53,512 INFO [EjbModule] Creating
04:51:53,546 INFO [EjbModule] Deploying MyStatefulFruit
04:51:53,873 INFO [EjbModule] Created
04:51:53,874 INFO [EjbModule] Starting
04:51:54,056 INFO [EjbModule] Started
04:51:54,057 INFO [MainDeployer] Deployed package:
file:/home/lab/jboss-3.0.4/server/default/deploy/myfruit.jar
```

Module EJB telah di-deploy di atas JBoss enterprise application server dan siap melayani request dari client.

untuk mengamati tingkah laku Statefull Session Bean ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis FruitApp.java, simpan di-directory yang sudah dipersiapkan.

**`FruitApp.java`**

```
public class FruitApp
{
    public static void main(String[] args)
    {
        for(int i=0;i<1000;i++)
        {
            MyThread thread = new MyThread(i);
            thread.start();
        }
    }
}
```

**Langkah ke-2**
Tulis MyThread.java, simpan di-directory yang sudah dipersiapkan.

**`MyThread.java`**

```
import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;


public class MyThread extends Thread
{
    private static final String[] fruitArray
        = { "Apple", "Avocado", "Banana", "Durian", "Grape",
            "Lychee", "Mango", "Orange", "Starfruit", "Soursop" };
    private int threadNo;

    public MyThread(int threadNo)
    {
        super("Thread-" + threadNo);
        this.threadNo = threadNo;
    }

    public void run()
    {

        try
        {
            Properties props = new Properties();
            props.setProperty(
                "java.naming.factory.initial",
                "org.jnp.interfaces.NamingContextFactory"
                );
            props.setProperty(
```

```
                "java.naming.provider.url",
                "localhost:1099"
                );
            InitialContext jndiContext = new InitialContext(props);
            Object ref  = jndiContext.lookup("MyStatefulFruit");
            MyFruitHome home = (MyFruitHome)
            PortableRemoteObject.narrow (ref, MyFruitHome.class);

            MyFruit ejb = home.create();

            String fruitInThread = fruitArray[threadNo%10];

            ejb.setFruit(fruitInThread);

            yield();

            try
            {
                int t = (int)(100*Math.random());
                sleep(t);
            }
            catch(InterruptedException ie)
            {
                ie.printStackTrace();
            }

            String fruitInEJB = ejb.getFruit();

            System.out.println(getName()
                + " : in Thread value = " + fruitInThread
                + " in EJB value = " + fruitInEJB
                + (fruitInEJB.equals(fruitInThread) ?
                    "":" FRUIT IS CHANGED"));
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

**Langkah ke-3**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-4**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$ export CLASSPATH=$CLASSPATH:myfruit.jar
$ javac FruitApp.java
```

akan dihasilkan FruitApp.class  dan MyThread.class

**Langkah ke-5**
Launch ...

```
$ java FruitApp
```

```
Thread-233 : in Thread value = Durian in EJB value = Durian
Thread-9 : in Thread value = Soursop in EJB value = Soursop
Thread-235 : in Thread value = Lychee in EJB value = Lychee
Thread-11 : in Thread value = Avocado in EJB value = Avocado
Thread-241 : in Thread value = Avocado in EJB value = Avocado
Thread-237 : in Thread value = Orange in EJB value = Orange
Thread-240 : in Thread value = Apple in EJB value = Apple
Thread-61 : in Thread value = Avocado in EJB value = Avocado
```

# 9Bean Managed Persistence Entity Bean

Entity Bean dikembangkan untuk merepresentasikan obyek bisnis. Obyek bisnis ini lumrahnya disimpan di dalam database. Sebuah Entity Bean merepresentasikan sebuah table dalam database. Untuk Bean Managed Persistence, Entity Bean bertanggung jawab untuk menyimpan dan membaca data dari database, sehingga data yang dipegang oleh Entity Bean selalu synch dengan yang ada di database.
Untuk mengembangkan sebuah Statefull Session Bean, Anda wajib menuliskan source code untuk Home interface, Remote interface, dan bean implementation. Anda juga wajib menyediakan Deployment descriptor.

Di dalam bean implementation, mesti dituliskan code-code untuk mengakses dengan database.

## *CS-081-091*

Sebuah praktikum dengan Entity Bean yang menggunakan Bean Managed Persistence ...

**Persiapan**
Jalankan JBoss jika belum
Buat sebuah directory untuk bekerja
misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis EmployeeHome.java, simpan di sub-directory employee di bawah directory yang sudah dipersiapkan.

### `employee/EmployeeHome.java`

```
package employee;

import java.util.*;

import java.rmi.RemoteException;
import javax.ejb.*;

public interface EmployeeHome
    extends EJBHome
{
    public Employee create(String id,
        String name, String department, String jobTitle,
        Date hireDate, boolean isPermanentEmployee, double salary)
        throws RemoteException, CreateException;

    public Employee create(String id, String name)
        throws RemoteException, CreateException;

    public Employee findByPrimaryKey(String id)
        throws RemoteException, FinderException;

    public Enumeration findByDepartment(String department)
        throws RemoteException, FinderException;
}
```

**Langkah ke-2**
Tulis Employee.java, simpan di sub-directory employee di bawah directory yang sudah dipersiapkan.

### `employee/Employee.java`

```
package employee;

import java.util.*;

import java.rmi.*;
import javax.ejb.*;

public interface Employee
    extends EJBObject
{
    public String getId()
```

```
        throws RemoteException;
    public void setName(String name)
        throws RemoteException;
    public String getName()
        throws RemoteException;
    public void setDepartment(String department)
        throws RemoteException;
    public String getDepartment()
        throws RemoteException;
    public void setJobTitle(String jobTitle)
        throws RemoteException;
    public String getJobTitle()
        throws RemoteException;
    public void setHireDate(Date hireDate)
        throws RemoteException;
    public Date getHireDate()
        throws RemoteException;
    public void setPermanentEmployee(boolean permanentEmployee)
        throws RemoteException;
    public boolean getPermanentEmployee()
        throws RemoteException;
    public void setSalary(double salary)
        throws RemoteException;
    public double getSalary()
        throws RemoteException;
}
```

**Langkah ke-3**
Tulis EmployeeBean.java, simpan di sub-directory employee di bawah directory yang sudah dipersiapkan.

**employee/EmployeeBean.java**

```
package employee;

import javax.ejb.*;
import java.rmi.*;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.sql.ResultSet;
import java.util.*;
import javax.sql.*;
import javax.naming.*;

public class EmployeeBean
    implements EntityBean
{
    /* DATA */

    private String id;
    private String name;
    private String department;
    private String jobTitle;
    private Date hireDate;
    private boolean permanentEmployee;
    private double salary;

    public String getId()
    {
        return this.id;
    }
```

```
    public void setName(String name)
    {
        this.name = name;
    }
    public String getName()
    {
        return this.name;
    }

    public void setDepartment(String department)
    {
        this.department = department;
    }
    public String getDepartment()
    {
        return this.department;
    }

    public void setJobTitle(String jobTitle)
    {
        this.jobTitle = jobTitle;
    }
    public String getJobTitle()
    {
        return this.jobTitle;
    }

    public void setHireDate(Date hireDate)
    {
        this.hireDate = hireDate;
    }
    public Date getHireDate()
    {
        return this.hireDate;
    }
    public void setPermanentEmployee(boolean permanentEmployee )
    {
        this.permanentEmployee = permanentEmployee;
    }
    public boolean getPermanentEmployee()
    {
        return this.permanentEmployee;
    }
    public void setSalary(double salary)
    {
        this.salary = salary;
    }
    public double getSalary()
    {
        return this.salary;
    }

          /* PROCESS */

    public String ejbCreate(String id,
                    String name)
        throws CreateException
    {
                     return ejbCreate(id, name, null, null, null, false,
0);
          }

          public String ejbCreate(String id,
                    String name,
```

```
                    String department, String jobTitle,
                    Date hireDate, boolean permanentEmployee, double
salary)
        throws CreateException
    {
        this.id = id;
        this.name = name;
        this.department = department;
        this.jobTitle = jobTitle;
        this.hireDate=hireDate;
        this.permanentEmployee = permanentEmployee;
        this.salary = salary;

        Connection conn = null;
        try
        {
            conn = getConnection();
            String sqlInsert = "INSERT INTO EMPLOYEE_TBL
VALUES(?,?,?,?,?,?,?)";
            PreparedStatement stmt
                = conn.prepareStatement(sqlInsert);

            stmt.setString(1, this.id);
            stmt.setString(2, this.name);
            stmt.setString(3, this.department);
            stmt.setString(4, this.jobTitle);
            stmt.setDate(5, new java.sql.Date(this.hireDate.getTime()));
            stmt.setBoolean(6, this.permanentEmployee);
            stmt.setDouble(7, this.salary);

            stmt.executeUpdate();

            return new String(id);

        }
        catch(Exception e)
        {
            throw new CreateException(e.toString());
        }
        finally
        {
            try
            {
                if(conn!=null)conn.close();
            }
            catch(SQLException sqle)
            {
                sqle.printStackTrace();
            }

        }
    }
    public void ejbPostCreate(String id,
                    String name)
        throws CreateException
    {
          }
    public void ejbPostCreate(String id,
                    String name,
                    String department, String jobTitle,
                    Date hireDate, boolean permanentEmployee, double
salary)
        throws CreateException
    {
```

```
 }

public String ejbFindByPrimaryKey(String id)
     throws FinderException
 {
     String pk = null;
     Connection conn = null;
     try
     {
         conn = getConnection();
         String sqlSelect =
             "SELECT * FROM EMPLOYEE_TBL "
             + "WHERE ID = ?";
         PreparedStatement stmt = conn.prepareStatement(sqlSelect);

         stmt.setString(1, id);
         ResultSet rs = stmt.executeQuery();

         if(rs.next())
         {
             pk = rs.getString("ID");
         }
         return pk;
     }
     catch(Exception e)
     {
         throw new FinderException(e.toString());
     }
     finally
     {
         try
         {
             if(conn!=null)conn.close();
         }
         catch(SQLException sqle)
         {
             sqle.printStackTrace();
         }
     }
 }

public Enumeration ejbFindByDepartment(String department)
     throws FinderException
 {
    Vector pks = new Vector();

     Connection conn = null;
     try
     {
         conn = getConnection();
         String sqlSelect =
             "SELECT * FROM EMPLOYEE_TBL "
             + "WHERE DEPARTMENT = ?";
         PreparedStatement stmt = conn.prepareStatement(sqlSelect);

         stmt.setString(1, department);
         ResultSet rs = stmt.executeQuery();

         while(rs.next())
         {
             String pk = rs.getString("ID");
                                     pks.addElement(pk);
         }
         return pks.elements();
```

```
        }
        catch(Exception e)
        {
            throw new FinderException(e.toString());
        }
        finally
        {
            try
            {
                if(conn!=null)conn.close();
            }
            catch(SQLException sqle)
            {
                sqle.printStackTrace();
            }
        }
    }

      public void ejbRemove()
        throws RemoveException
    {
        String pk = (String)entityContext.getPrimaryKey();
        Connection conn = null;
        try
        {
            conn = getConnection();
            String sqlDelete =
                "DELETE FROM EMPLOYEE_TBL "
                + "WHERE "
                + "ID = ?";
            PreparedStatement stmt = conn.prepareStatement(sqlDelete);

            stmt.setString(1, pk);
            stmt.executeUpdate();
        }
        catch(Exception e)
        {
            throw new RemoveException(e.toString());
        }
        finally
        {
            try
            {
                if(conn!=null)conn.close();
            }
            catch(SQLException sqle)
            {
                sqle.printStackTrace();
            }
        }
    }

    public void ejbStore()
        throws RemoteException
    {
        String pk = (String)entityContext.getPrimaryKey();
        Connection conn = null;
        try
        {
            conn = getConnection();
            String sqlUpdate =
                "UPDATE EMPLOYEE_TBL "
                + "SET "
                + "NAME = ?,"
```

```
                + "DEPARTMENT = ?,"
                + "JOB_TITLE = ?,"
                + "HIRE_DATE = ?,"
                + "PERMANENT_EMPLOYEE = ?,"
                + "SALARY = ? "
                + "WHERE "
                + "ID = ?";
            PreparedStatement stmt = conn.prepareStatement(sqlUpdate);

            stmt.setString(7, pk);
            stmt.setString(1, this.name);
            stmt.setString(2, this.department);
            stmt.setString(3, this.jobTitle);
            stmt.setDate(4, new java.sql.Date(this.hireDate.getTime()));
            stmt.setBoolean(5, this.permanentEmployee);
            stmt.setDouble(6, this.salary);

            stmt.executeUpdate();

        }
        catch(Exception e)
        {
            throw new RemoteException(e.toString());
        }
        finally
        {
            try
            {
                if(conn!=null)conn.close();
            }
            catch(SQLException sqle)
            {
                sqle.printStackTrace();
            }
        }
    }

    public void ejbLoad()
        throws RemoteException
    {
        String id = (String)entityContext.getPrimaryKey();
        Connection conn = null;
        try
        {
            conn = getConnection();
            String sqlSelect =
                "SELECT * FROM EMPLOYEE_TBL "
                + "WHERE ID = ?";
            PreparedStatement stmt = conn.prepareStatement(sqlSelect);

            stmt.setString(1, id);
            ResultSet rs = stmt.executeQuery();

            if(rs.next())
            {
                this.id = rs.getString("ID");
                this.name = rs.getString("NAME");
                this.department = rs.getString("DEPARTMENT");
                this.jobTitle = rs.getString("JOB_TITLE");
                this.hireDate = rs.getDate("HIRE_DATE");
                this.permanentEmployee =
rs.getBoolean("PERMANENT_EMPLOYEE");
                this.salary = rs.getDouble("SALARY");
            }
```

```
        }
        catch(Exception e)
        {
            throw new RemoteException(e.toString());
        }
        finally
        {
            try
            {
                if(conn!=null)conn.close();
            }
            catch(SQLException sqle)
            {
                sqle.printStackTrace();
            }
        }
    }

    /* CONTROL */

    public void ejbActivate() {}
    public void ejbPassivate() {}

    transient private EntityContext entityContext = null;
    transient private DataSource dataSource = null;

    private Connection getConnection()
        throws SQLException
    {
        Connection conn = dataSource.getConnection();
        return conn;
    }

    public void setEntityContext(EntityContext entityContext)
    {
        this.entityContext = entityContext;
        try
        {
            Context context = new InitialContext();
            dataSource      =
(DataSource)context.lookup("java:/MySqlDS");
        }
        catch (NamingException e)
        {
            e.printStackTrace();
        }

    }

    public void unsetEntityContext()
    {
        this.entityContext = null;
        this.dataSource = null;
    }
}
```

**Langkah ke-4**
Buka sebuah terminal dan change directory ke directory di atas

**Langkah ke-5**
Compile ...

```
$ export CLASSPATH=.:/home/lab/jboss-3.0.4/client/jbossall-client.jar
```

```
$ javac employee/*
```

**Langkah ke-6**
Buat sub-directory META-INF

**Langkah ke-7**
Tulis ejb-jar.xml, simpan ke sub directory META-INF

**`META-INF/ejb-jar.xml`**

```
<?xml version="1.0"?>

<!DOCTYPE ejb-jar PUBLIC
'-//Sun Microsystems, Inc.//DTD Enterprise JavaBeans 1.1//EN'
'http://java.sun.com/j2ee/dtds/ejb-jar_1_1.dtd'>

<ejb-jar>
    <description>Enterprise Java Lab</description>
    <display-name>Enterprise Java Lab</display-name>
    <enterprise-beans>
        <entity>
            <ejb-name>Employee</ejb-name>
            <home>employee.EmployeeHome</home>
            <remote>employee.Employee</remote>
            <ejb-class>employee.EmployeeBean</ejb-class>
            <persistence-type>Bean</persistence-type>
            <prim-key-class>java.lang.String</prim-key-class>
            <reentrant>False</reentrant>
        </entity>
    </enterprise-beans>
</ejb-jar>
```

**Langkah ke-8**
Gabungkan ke dalam sebuah jar

```
$ jar cvf employee.jar employee/*.class META-INF/

adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/MANIFEST.MF (in=56) (out=56) (stored 0%)
adding: employee/EmployeeBean.class (in=5662) (out=2763) (deflated 51%)
adding: employee/Employee.class (in=711) (out=341) (deflated 52%)
adding: employee/EmployeeHome.class (in=728) (out=312) (deflated 57%)
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/ejb-jar.xml (in=717) (out=325) (deflated 54%)
Total:
------
(in = 7863) (out = 4610) (deflated 41%)
```

**Langkah ke-9**
Copy ke dalam deployment directory dari jBoss

```
$ cp employee.jar /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
22:12:08,768 INFO  [MainDeployer] Starting deployment of package:
file:/home/lab/jboss-3.0.4/server/default/deploy/employee.jar
22:12:08,922 INFO  [EjbModule] Creating
22:12:08,952 INFO  [EjbModule] Deploying Employee
22:12:09,085 INFO  [EjbModule] Created
22:12:09,086 INFO  [EjbModule] Starting
```

```
22:12:09,176 INFO  [EjbModule] Started
22:12:09,177     INFO          [MainDeployer]     Deployed     package:
file:/home/lab/jboss-3.0.4/server/default/deploy/employee.jar
```

Module EJB telah di-deploy di atas JBoss enterprise application server dan siap melayani request dari client.

untuk mendeploy Datasource ke mySQL ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Buka sebuah terminal. Lalu melalui terminal tersebut jalankan mysql client

```
$ mysql -uekobs -pj2ee MYAPP_DB
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 3.23.54

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

**Langkah ke-2**
melalui mysql client, create table yang dibutuhkan ...

```
mysql> create table EMPLOYEE_TBL(ID varchar(10) primary key, NAME
varchar(40), DEPARTMENT varchar(20),JOB_TITLE varchar(30), HIRE_DATE
date, PERMANENT_EMPLOYEE bool, SALARY float);

Query OK, 0 rows affected (0.00 sec)
```

**Langkah ke-3**
Tulis mysql-service.xml

### mysql-service.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<server>

  <mbean
code="org.jboss.resource.connectionmanager.LocalTxConnectionManager"
      name="jboss.jca:service=LocalTxCM,name=MySqlDS">

    <depends optional-attribute-name="ManagedConnectionFactoryName">
      <!--embedded mbean-->
      <mbean code="org.jboss.resource.connectionmanager.RARDeployment"
          name="jboss.jca:service=LocalTxDS,name=MySqlDS">

        <attribute name="JndiName">MySqlDS</attribute>

        <attribute name="ManagedConnectionFactoryProperties">
          <properties>
            <config-property name="ConnectionURL"
type="java.lang.String">
            jdbc:mysql://localhost:3306/MYAPP_DB
            </config-property>
            <config-property name="DriverClass" type="java.lang.String">
```

```
            org.gjt.mm.mysql.Driver
            </config-property>
            <config-property name="UserName" type="java.lang.String">
            ekobs
            </config-property>
            <config-property name="Password" type="java.lang.String">
            j2ee
            </config-property>
          </properties>

        </attribute>
        <depends optional-attribute-name="OldRarDeployment">
            jboss.jca:service=RARDeployment,name=JBoss LocalTransaction
JDBC Wrapper
        </depends>
      </mbean>
    </depends>
    <depends optional-attribute-name="ManagedConnectionPool">

      <mbean
code="org.jboss.resource.connectionmanager.JBossManagedConnectionPool"
          name="jboss.jca:service=LocalTxPool,name=MySqlDS">

        <attribute name="MinSize">0</attribute>
        <attribute name="MaxSize">50</attribute>
        <attribute name="BlockingTimeoutMillis">5000</attribute>
        <attribute name="IdleTimeoutMinutes">15</attribute>
        <attribute name="Criteria">ByContainer</attribute>
      </mbean>

    </depends>
    <depends optional-attribute-name="CachedConnectionManager">
        jboss.jca:service=CachedConnectionManager
    </depends>

    <attribute
name="TransactionManager">java:/TransactionManager</attribute>

  </mbean>


</server>
```

**Langkah ke-4**
Copy ke dalam deployment directory dari jBoss, termasuk JDBC driver yang dibutuhkan ...

```
$  cp  /home/lab/mm.mysql-2.0.8/mm.mysql-2.0.8-bin.jar  /home/lab/jboss-
3.0.4/server/default/deploy/
$ cp mysql-service.xml /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
22:27:19,923  INFO    [MainDeployer]  Starting  deployment  of  package:
file:/home/lab/jboss-3.0.4/server/default/deploy/mysql-ser
vice.xml
22:27:19,950            WARN                          [ServiceController]
jboss.jca:service=LocalTxDS,name=MySqlDS does not implement any Service
methods
22:27:19,951 INFO  [JBossManagedConnectionPool] Creating
22:27:19,951 INFO  [JBossManagedConnectionPool] Created
22:27:19,952 INFO  [LocalTxConnectionManager] Creating
22:27:19,958 INFO  [LocalTxConnectionManager] Created
22:27:19,959 INFO  [JBossManagedConnectionPool] Starting
```

```
22:27:19,959 INFO  [JBossManagedConnectionPool] Started
22:27:19,960 INFO  [LocalTxConnectionManager] Starting
22:27:19,975 INFO   [MySqlDS] Bound connection factory for resource
adapter 'JBoss LocalTransaction JDBC Wrapper' to JNDI nam
e 'java:/MySqlDS'
22:27:19,976 INFO  [LocalTxConnectionManager] Started
22:27:19,976     INFO          [MainDeployer]     Deployed     package:
file:/home/lab/jboss-3.0.4/server/default/deploy/mysql-service.xml
```

untuk mengakses Entity Bean ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis EmployeeApp.java, simpan di sub-directory app di bawah directory yang sudah dipersiapkan.

**app/EmployeeApp.java**

```
package app;

import employee.*;

import java.rmi.*;
import java.util.*;

import javax.ejb.*;
import javax.naming.*;
import javax.rmi.PortableRemoteObject;

import java.io.*;
import java.text.*;

public class EmployeeApp
{
    public static void main(String[] args)
    {
        try
        {
            EmployeeApp app
                = new EmployeeApp();

            char command;
            do
            {
                System.out.println();
                System.out.println("Press Enter to continue ...");
                app.readLine();
                System.out.println("Available command : ");
                System.out.println("C Create a new record");
                System.out.println("I View by Id");
                System.out.println("D View by Department");
                System.out.println("M Modify a record");
                System.out.println("R Remove a record");
                System.out.println("X Exit");
                System.out.println();

                command
                    = app.readLine("Press C, I, D, M, R or X")
                        .toUpperCase().charAt(0);

                System.out.println();

                switch(command)
                {
```

```
                case 'C' : app.create();break;
                case 'I' : app.viewById();break;
                case 'D' : app.viewByDepartment();break;
                case 'M' : app.modify();break;
                case 'R' : app.remove();break;
                case 'X' : break;
                default :
                    System.out.println("You press invalid key");
            }
        }
        while(command != 'X');
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
}
private EmployeeHome home;

public EmployeeApp()
    throws CreateException, NamingException, RemoteException
{
    Properties props = new Properties();
    props.setProperty(
            "java.naming.factory.initial",
            "org.jnp.interfaces.NamingContextFactory"
            );
    props.setProperty(
            "java.naming.provider.url",
            "localhost:1099"
            );

    InitialContext jndiContext = new InitialContext(props);
    Object ref  = jndiContext.lookup("Employee");
    home = (EmployeeHome)
        PortableRemoteObject.narrow (ref, EmployeeHome.class);

    in = new BufferedReader(new InputStreamReader(System.in));
}

private SimpleDateFormat format
    = new SimpleDateFormat("ddMMyyyy");

private BufferedReader in;

private void readLine()
    throws IOException
{
    in.readLine();
}

private String readLine(String parameterName)
    throws IOException
{
    System.out.print(parameterName + " : ");
    return in.readLine();
}

public void create()
    throws IOException, ParseException,
    CreateException, RemoteException
{
    System.out.println("Creating ... ");
```

```
        Employee employee = home.create (
            readLine("Id"),
            readLine("Name"),
            readLine("Department"),
            readLine("Job Title"),
            format.parse(readLine("Hire Date (ddMMyyyy)")),
            Boolean.valueOf(
                readLine("Permanent Employee (True/False)"))
                .booleanValue(),
            Double.parseDouble(readLine("Salary"))
            );
    }
    public void modify()
        throws IOException, ParseException, FinderException,
RemoteException
    {
        String id = readLine("Id");
        Employee employee = home.findByPrimaryKey(id);

        employee.setName(readLine("Name"));
        employee.setDepartment(readLine("Department"));
        employee.setJobTitle(readLine("Job Title"));
        employee.setHireDate(format.parse(readLine("Hire Date
(ddMMyyyy)")));
        employee.setPermanentEmployee(
            Boolean.valueOf(
                readLine("Permanent Employee (True/False)"))
                    .booleanValue());
        employee.setSalary(Double.parseDouble(readLine("Salary")));
    }

    public void remove()
        throws IOException, ParseException,
            FinderException, RemoteException, RemoveException
    {
        String id = readLine("Id");
        Employee employee = home.findByPrimaryKey(id);
        employee.remove();
    }

    public void viewById()
        throws IOException, FinderException, RemoteException
    {
        String id = readLine("Id");
        Employee employee = home.findByPrimaryKey(id);

        System.out.println("Name : " + employee.getName());
        System.out.println("Department : " + employee.getDepartment());
        System.out.println("Job Title : " + employee.getJobTitle());
        System.out.println("Hire Date : " + employee.getHireDate());
        System.out.println("Permanent Employee : "
            + employee.getPermanentEmployee());
        System.out.println("Salary        : " + employee.getSalary());
    }

    public void viewByDepartment()
        throws IOException, FinderException, RemoteException
    {
        String department = readLine("Department");
        Enumeration employees = home.findByDepartment(department);

        while(employees.hasMoreElements())
```

```
        {
            Employee employee = (Employee) employees.nextElement();

            System.out.print("Name : " + employee.getName());
            System.out.print("    ");
            System.out.println("Job Title  : "
                + employee.getJobTitle());
        }
    }
}
```

**Langkah ke-2**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-3**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$ export CLASSPATH=$CLASSPATH:employee.jar
$ javac app/EmployeeApp.java
```

akan dihasilkan app/EmployeeApp.class

**Langkah ke-4**
Launch ...

```
$ java app.EmployeeApp

Press Enter to continue ...

Available command :
C Create a new record
I View by Id
D View by Department
M Modify a record
R Remove a record
X Exit

Press C, I, D, M, R or X :
```

# 10Container Managed Persistence Entity Bean

Untuk Bean Managed Persistence, EJB container bertanggung jawab untuk menyimpan dan membaca data dari database, sehingga data yang dipegang oleh Entity Bean selalu synch dengan yang ada di database.

Untuk mengembangkan sebuah CMP Entity Bean, Anda wajib menuliskan source code untuk Home interface, Remote interface, dan bean implementation. Anda juga wajib menyediakan Deployment descriptor.

## *CS-081-101*

Sebuah praktikum dengan Entity Bean yang menggunakan Container Managed Persistence ...

**Persiapan**
Jalankan JBoss jika belum
Buat sebuah directory untuk bekerja
misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis EmployeeHome.java, simpan di sub-directory employee di bawah directory yang sudah dipersiapkan.

**`employee/EmployeeHome.java`**

```
package employee;

import java.util.*;

import java.rmi.RemoteException;
import javax.ejb.*;

public interface EmployeeHome
    extends EJBHome
{
    public Employee create(String id,
        String name, String department, String jobTitle,
         Date hireDate, boolean isPermanentEmployee, double salary)
        throws RemoteException, CreateException;

    public Employee create(String id, String name)
        throws RemoteException, CreateException;

    public Employee findByPrimaryKey(String id)
        throws RemoteException, FinderException;

    public Enumeration findByDepartment(String department)
        throws RemoteException, FinderException;
}
```

**Langkah ke-2**
Tulis Employee.java, simpan di sub-directory employee di bawah directory yang sudah dipersiapkan.

**`employee/Employee.java`**

```
package employee;

import java.util.*;

import java.rmi.*;
import javax.ejb.*;

public interface Employee
    extends EJBObject
{
    public String getId()
```

```
        throws RemoteException;
    public void setName(String name)
        throws RemoteException;
    public String getName()
        throws RemoteException;
    public void setDepartment(String department)
        throws RemoteException;
    public String getDepartment()
        throws RemoteException;
    public void setJobTitle(String jobTitle)
        throws RemoteException;
    public String getJobTitle()
        throws RemoteException;
    public void setHireDate(Date hireDate)
        throws RemoteException;
    public Date getHireDate()
        throws RemoteException;
    public void setPermanentEmployee(boolean permanentEmployee)
        throws RemoteException;
    public boolean getPermanentEmployee()
        throws RemoteException;
    public void setSalary(double salary)
        throws RemoteException;
    public double getSalary()
        throws RemoteException;
}
```

**Langkah ke-3**
Tulis EmployeeBean.java, simpan di sub-directory employee di bawah directory yang sudah dipersiapkan.

**employee/EmployeeBean.java**

```
package employee;

import javax.ejb.*;
import java.rmi.*;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.sql.ResultSet;
import java.util.*;
import javax.sql.*;
import javax.naming.*;

public abstract class EmployeeBean
    implements EntityBean
{
    /* DATA */
    public abstract void setId(String id);
    public abstract String getId();
    public abstract void setName(String name);
    public abstract String getName();
    public abstract void setDepartment(String department);
    public abstract String getDepartment();
    public abstract void setJobTitle(String jobTitle);
    public abstract String getJobTitle();
    public abstract void setHireDate(Date hireDate);
    public abstract Date getHireDate();
    public abstract void
        setPermanentEmployee(boolean permanentEmployee );
    public abstract boolean getPermanentEmployee();
```

```
    public abstract void setSalary(double salary);
    public abstract double getSalary();

    /* PROCESS */

    public String ejbCreate(String id,
                    String name)
        throws CreateException
    {
        return ejbCreate(id, name, null, null, null, false, 0);
    }

    public String ejbCreate(String id,
                    String name,
                    String department, String jobTitle,
                    Date hireDate,
                    boolean permanentEmployee, double salary)
        throws CreateException
    {
        setId(id);
        setName(name);
        setDepartment(department);
        setJobTitle(jobTitle);
        setHireDate(hireDate);
        setPermanentEmployee(permanentEmployee);
        setSalary(salary);
        return new String(id);

    }
    public void ejbPostCreate(String id,
                    String name)
        throws CreateException
    {
    }
    public void ejbPostCreate(String id,
                    String name,
                    String department, String jobTitle,
                    Date hireDate,
                    boolean permanentEmployee, double salary)
        throws CreateException
    {
    }
    public void ejbRemove() {}
    public void ejbLoad() {}
    public void ejbStore() {}
    public void ejbActivate() {}
    public void ejbPassivate() {}
    public void setEntityContext(EntityContext entityContext){}
    public void unsetEntityContext(){}
}
```

**Langkah ke-4**
Buka sebuah terminal dan change directory ke directory di atas

**Langkah ke-5**
Compile ...

```
$ export CLASSPATH=.:/home/lab/jboss-3.0.4/client/jbossall-client.jar
$ javac employee/*
```

**Langkah ke-6**

Buat sub-directory META-INF

**Langkah ke-7**
Tulis ejb-jar.xml, simpan ke sub directory META-INF

**`META-INF/ejb-jar.xml`**

```
<?xml version="1.0"?>

<!DOCTYPE ejb-jar PUBLIC
'-//Sun Microsystems, Inc.//DTD Enterprise JavaBeans 2.0//EN'
'http://java.sun.com/j2ee/dtd/ejb-jar_2_0.dtd'>

<ejb-jar>
    <description>Enterprise Java Lab</description>
    <display-name>Enterprise Java Lab</display-name>
    <enterprise-beans>
        <entity>
            <ejb-name>Employee</ejb-name>
            <home>employee.EmployeeHome</home>
            <remote>employee.Employee</remote>
            <ejb-class>employee.EmployeeBean</ejb-class>
            <persistence-type>Container</persistence-type>
            <prim-key-class>java.lang.String</prim-key-class>
            <reentrant>False</reentrant>

            <cmp-field><field-name>id</field-name></cmp-field>
            <cmp-field><field-name>name</field-name></cmp-field>
            <cmp-field><field-name>department</field-name></cmp-field>
            <cmp-field><field-name>jobTitle</field-name></cmp-field>
            <cmp-field><field-name>hireDate</field-name></cmp-field>
            <cmp-field><field-name>permanentEmployee</field-name></cmp-
field>
            <cmp-field><field-name>salary</field-name></cmp-field>
            <primkey-field><field-name>id</field-name></primkey-field>
        </entity>
    </enterprise-beans>
</ejb-jar>
```

**Langkah ke-8**
Tulis jbosscmp-jdbc.xml, simpan ke sub directory META-INF

**`META-INF/jbosscmp-jdbc.xml`**

```
<?xml version="1.0"?>

<!DOCTYPE jbosscmp-jdbc PUBLIC
'-//JBoss//DTD JBOSSCMP-JDBC 3.0//EN'
'http://www.jboss.org/j2ee/dtd/jbosscmp-jdbc_3_0.dtd'>

<jbosscmp-jdbc>
    <enterprise-beans>
        <entity>
            <ejb-name>Employee</ejb-name>
            <datasource>java:/MySqlDS</datasource>
            <datasource-mapping>mySQL</datasource-mapping>
            <table-name>EMPLOYEE_TBL</table-name>
            <cmp-field>
                <field-name>id</field-name>
                <column-name>ID</column-name>
```

```
            </cmp-field>
            <cmp-field>
                <field-name>name</field-name>
                <column-name>NAME</column-name>
            </cmp-field>
            <cmp-field>
                <field-name>department</field-name>
                <column-name>DEPARTMENT</column-name>
            </cmp-field>
            <cmp-field>
                <field-name>jobTitle</field-name>
                <column-name>JOB_TITLE</column-name>
            </cmp-field>
            <cmp-field>
                <field-name>hireDate</field-name>
                <column-name>HIRE_DATE</column-name>
            </cmp-field>
            <cmp-field>
                <field-name>permanentEmployee</field-name>
                <column-name>PERMANENT_EMPLOYEE</column-name>
            </cmp-field>
            <cmp-field>
                <field-name>salary</field-name>
                <column-name>SALARY</column-name>
            </cmp-field>
        </entity>
    </enterprise-beans>
</jbosscmp-jdbc>
```

**Langkah ke-9**
Gabungkan ke dalam sebuah jar

```
$ jar cvf employee.jar employee/*.class META-INF/

adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/MANIFEST.MF (in=56) (out=56) (stored 0%)
adding: employee/EmployeeBean.class (in=1839) (out=754) (deflated 58%)
adding: employee/Employee.class (in=712) (out=339) (deflated 52%)
adding: employee/EmployeeHome.class (in=633) (out=292) (deflated 53%)
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/ejb-jar.xml (in=1275) (out=416) (deflated 67%)
adding: META-INF/jbosscmp-jdbc.xml (in=1523) (out=402) (deflated 73%)
Total:
------
(in = 6027) (out = 3200) (deflated 46%)
```

**Langkah ke-9**
Copy ke dalam deployment directory dari jBoss

```
$ cp employee.jar /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
05:19:11,472 INFO  [MainDeployer] Starting deployment of package:
file:/home/lab/jboss-3.0.4/server/default/deploy/employee.jar
05:19:11,629 WARN  [EntityMetaData] Employee: The abstract-schema-name
Element must be specified for CMP 2.x Beans
05:19:11,631 WARN  [EntityMetaData] Employee: The abstract-schema-name
must be a valid Java Identifier 'null'
05:19:11,763 INFO  [EJBDeployer]
Bean   : Employee
Method : public abstract Enumeration findByDepartment(String) throws
```

```
RemoteException, FinderException
Section: 10.5.6
Warning: Every finder method except findByPrimaryKey(key) must be
associated with a query element in the deployment descriptor.

05:19:11,955 INFO  [EjbModule] Creating
05:19:12,016 INFO  [EjbModule] Deploying Employee
05:19:12,144 INFO  [EjbModule] Created
05:19:12,146 INFO  [EjbModule] Starting
05:19:14,901 INFO  [Employee] Table 'EMPLOYEE_TBL' already exists
05:19:14,963 INFO  [EjbModule] Started
05:19:14,964     INFO          [MainDeployer]     Deployed     package:
file:/home/lab/jboss-3.0.4/server/default/deploy/employee.jar
```

Module EJB telah di-deploy di atas JBoss enterprise application server dan siap melayani request dari client.

# 11Message Driven Bean

Message Driven Bean adalah komponen EJB yang dirancang untuk bekerja dalam asynchronous messaging.

Untuk mengembangkan sebuah Message Driven Bean ...

# 12Life cycle

Life cycle EJB adalah perjalanan bean instance, mulai di-create, di-invoke oleh client, dan sampai di-destroy. Life cycle EJB berbeda-beda untuk masing-masing tipe.

# 13Environment Variable

Untuk variable-variable yang hanya bisa diketahui saat deployment, atau bisa diubah bergantung kepada lingkungan deployment, Anda bisa memanfaatkan kemampuan EJB untuk membaca environment variable. Environment variable ditulis di dalam deployment descriptor.

# 14Transaction

Sebuah transaction adalah suatu rentetan operasi database yang bersifat ACID, yaitu Atomic, Consistency, Isolated dan Durable.

Sebuah transaction adalah atomic, bermakna perubahan-perubahan yang terjadi dipandang sebagai satu kesatuan, semuanya disimpan dengan commit atau semuanya dibatalkan dengan rollback.

Sebuah transaction adalah consistent, bermakna ia tidak melanggar aturan-aturan tertentu.

Sebuah transaction adalah isolated, bermakna satu transaction terpisah dari transaction lainnya.

Sebuah transaction adalah durable, bermakna setelah perubahan-perubahan yang terjadi akan tersimpan sesuai transaction berakhir.

Di dalam EJB, Anda bisa mendeklarasikan transaction sebagai salah satu dari : TX_SUPPORTS, TX_REQUIRED, TX_REQUIRES_NEW, TX_MANDATORY, TX_NOT_SUPPORTED, TX_BEAN_MANAGED.

Sebuah transaction harus mempunyai isolation, karena sebuah transaction akan mengubah data dimana transaction lain juga mengaksesnya. Di dalam EJB, dapat dideklarasikan transaction level salah satu dari ReadUncommitted, ReadCommited, RepeatableRead dan Serializable.

Client ... no transaction access a mandatory ...

access to required new ...
access to required
access to support
access to notsupported

Client with transaction to mandatory
access to required new ...
access to required
access to support
To notsupported

# 15Security

Sebuah enterpri application dapat dilindungi dalam satu security rule. Di mana EJB tertentu hanya bisa diakses oleh user dengan role tertentu.

Experiment : SalamKeadilan dgn user ...
test dari client tanpa account ...

Experiment 2 : method memanggil method lain ...

## *CS-081-151*

Sebuah praktikum untuk mengamankan EJB dengan security constraint ...

**Persiapan**
Buat sebuah directory untuk bekerja
misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis login-config.xml simpan di /home/lab/jboss-3.0.4/server/default/conf

**`JBOSS_HOME/server/default/conf/login-config.xml`**

```
<?xml version='1.0'?>
<!DOCTYPE policy PUBLIC
"-//JBoss//DTD JBOSS Security Config 3.0//EN"
"http://www.jboss.org/j2ee/dtd/security_config.dtd">

<policy>
    <application-policy name = "myDomain">
       <authentication>
             <login-module code =
"org.jboss.security.auth.spi.UsersRolesLoginModule"
               flag = "required">
               <module-option name
="usersProperties">users.properties</module-option>
               <module-option name
="rolesProperties">roles.properties</module-option>
                <module-option
name="unauthenticatedIdentity">guest</module-option>
             </login-module>
       </authentication>
    </application-policy>
</policy>
```

**Langkah ke-2**
Restart JBoss

**Langkah ke-3**
Tulis SalamKeadilanHome.java, simpan di-directory yang sudah dipersiapkan.

**`SalamKeadilanHome.java`**

```
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;

public interface SalamKeadilanHome
    extends EJBHome
{
    public SalamKeadilan create()
        throws RemoteException, CreateException;
}
```

**Langkah ke-4**
Tulis SalamKeadilan.java, simpan di directory yang sudah dipersiapkan

**SalamKeadilan.java**

```
import javax.ejb.EJBObject;
import java.rmi.RemoteException;

public interface SalamKeadilan
    extends EJBObject
{
    public String getPesan()
        throws RemoteException;
}
```

**Langkah ke-5**
Tulis SalamKeadilanBean.java, simpan di directory yang sudah disiapkan

**SalamKeadilanBean.java**

```
import java.rmi.RemoteException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;

public class SalamKeadilanBean
    implements SessionBean
{
   public String getPesan()
   {
        return "Salam Keadilan !";
   }

   public void ejbCreate() {}
   public void ejbRemove() {}
   public void ejbActivate() {}
   public void ejbPassivate() {}
   public void setSessionContext(SessionContext sc) {}
}
```

**Langkah ke-6**
Buka sebuah terminal dan change directory ke directory di atas

**Langkah ke-7**
Compile ...

```
$ export CLASSPATH=.:/home/lab/jboss-3.0.4/client/jbossall-client.jar
$ javac *.java

$ ls *.class
SalamKeadilanBean.class  SalamKeadilan.class  SalamKeadilanHome.class
```

**Langkah ke-8**
Buat sub-directory META-INF

**Langkah ke-9**
Tulis ejb-jar.xml, simpan ke sub directory META-INF

**META-INF/ejb-jar.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar>
    <description>Enterprise Java Lab</description>
    <display-name>Enterprise Java Lab</display-name>
    <enterprise-beans>
        <session>
            <ejb-name>SalamKeadilan</ejb-name>
            <home>SalamKeadilanHome</home>
            <remote>SalamKeadilan</remote>
            <ejb-class>SalamKeadilanBean</ejb-class>
            <session-type>Stateless</session-type>
            <transaction-type>Bean</transaction-type>
            <security-role-ref>
                <role-name>consultant</role-name>
                <role-link>consultant</role-link>
            </security-role-ref>
        </session>
    </enterprise-beans>
    <assembly-descriptor>
        <security-role>
            <description>consultant</description>
            <role-name>consultant</role-name>
        </security-role>
        <method-permission>
            <role-name>consultant</role-name>
            <method>
                <ejb-name>SalamKeadilan</ejb-name>
                <method-name>*</method-name>
            </method>
        </method-permission>
    </assembly-descriptor>
</ejb-jar>
```

**Langkah ke-10**
Tulis jboss.xml, simpan ke sub directory META-INF

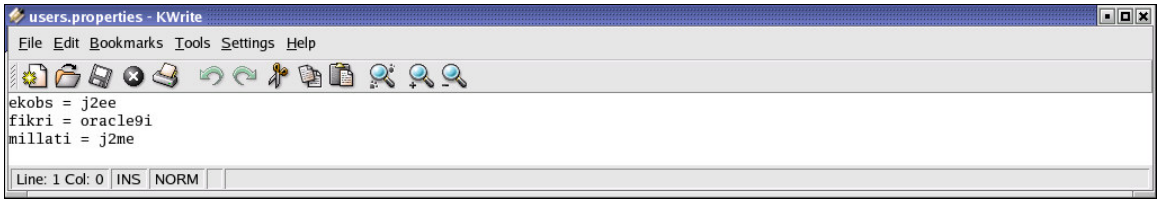
**META-INF/jboss.xml**

```
<?xml version="1.0" encoding="UTF-8"?>

<jboss>
   <!-- All bean containers use this security manager by default -->
   <container-configurations>
        <container-configuration>
            <container-name>
            Standard Stateless SessionBean
            </container-name>
            <security-domain>
            java:/jaas/myDomain
            </security-domain>
        </container-configuration>
   </container-configurations>
</jboss>
```

**Langkah ke-11**
Tulis users.properties, simpan di-directory yang sudah dipersiapkan.

**users.properties**

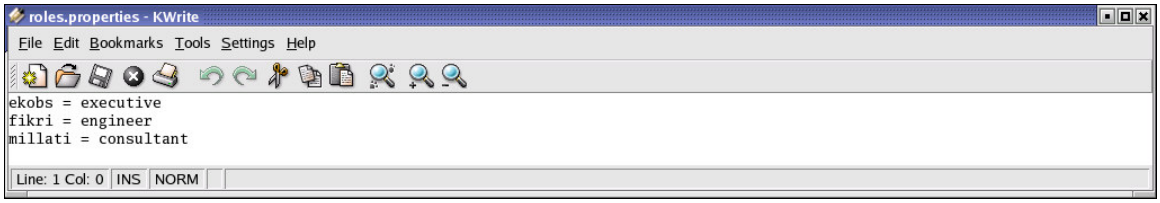
```
ekobs = j2ee
fikri = oracle9i
millati = j2me
```




**Langkah ke-12**
Tulis roles.properties, simpan di-directory yang sudah dipersiapkan.

**roles.properties**

```
ekobs = executive
fikri = engineer
millati = consultant
```




**Langkah ke-13**
Gabungkan ke dalam sebuah jar

```
$ jar cvf salamkeadilan.jar *.class *.properties META-INF/
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/MANIFEST.MF (in=56) (out=56) (stored 0%)
adding: SalamKeadilanBean.class (in=634) (out=324) (deflated 48%)
adding: SalamKeadilan.class (in=229) (out=176) (deflated 23%)
adding: SalamKeadilanHome.class (in=263) (out=184) (deflated 30%)
adding: roles.properties (in=55) (out=52) (deflated 5%)
adding: users.properties (in=45) (out=42) (deflated 6%)
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/ejb-jar.xml (in=1144) (out=334) (deflated 70%)
adding: META-INF/jboss.xml (in=451) (out=205) (deflated 54%)
Total:
------
(in = 2866) (out = 2490) (deflated 13%)
```

**Langkah ke-14**
Copy ke dalam deployment directory dari jBoss

```
$ cp salamkeadilan.jar /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
15:19:01,235 INFO  [EjbModule] Creating
15:19:01,263 INFO  [EjbModule] Deploying SalamKeadilan
15:19:01,305 INFO  [EjbModule] Created
15:19:01,306 INFO  [EjbModule] Starting
15:19:01,530 INFO  [EjbModule] Started
15:19:01,570     INFO          [MainDeployer]     Deployed     package:
file:/home/lab/jboss-3.0.4/server/default/deploy/salamkeadilan.jar
```

Module EJB telah di-deploy di atas JBoss enterprise application server dan siap melayani request dari client.

untuk menunjukkan bahwa tanpa authentication dan authorization, client tidak bisa mengakses EJB ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis SalamKeadilanApp.java, simpan di-directory yang sudah dipersiapkan.

**`SalamKeadilanApp.java`**

```
import java.util.Properties;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;

public class SalamKeadilanApp
{
    public static void main(String[] args)
    {
        try
        {
            Properties props = new Properties();
            props.setProperty(
                "java.naming.factory.initial",
                "org.jnp.interfaces.NamingContextFactory"
                );
            props.setProperty(
                "java.naming.provider.url",
                "localhost:1099"
                );
            InitialContext jndiContext = new InitialContext(props);
            Object ref  = jndiContext.lookup("SalamKeadilan");
            SalamKeadilanHome home = (SalamKeadilanHome)
            PortableRemoteObject.narrow (ref, SalamKeadilanHome.class);
            SalamKeadilan ejb = home.create();

            String pesan = ejb.getPesan();

            System.out.println(pesan);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

**Langkah ke-2**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-3**
Compile ...

```
$ export CLASSPATH=.
```

```
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$ export CLASSPATH=$CLASSPATH:salamkeadilan.jar
$ javac SalamKeadilanApp.java
```

akan dihasilkan SalamKeadilanApp.class

**Langkah ke-4**
Launch ...

```
$ java SalamKeadilanApp
java.rmi.ServerException:  RemoteException  occurred  in  server  thread;
nested exception is:
        java.rmi.ServerException: EJBException:; nested exception is:
        javax.ejb.EJBException:                checkSecurityAssociation;
CausedByException is:
        Authentication exception, principal=null
```

## *CS-081-153*

mengakses EJB yang diamankan ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis SalamKeadilanApp.java, simpan di-directory yang sudah dipersiapkan.

### `SalamKeadilanApp.java`

```
import java.util.Properties;
import javax.naming.*;
import javax.rmi.PortableRemoteObject;
import javax.security.auth.callback.*;
import javax.security.auth.login.*;

public class SalamKeadilanApp
{
    public static void main(String[] args)
    {
        try
        {
            Properties props = new Properties();
            props.setProperty(
                Context.INITIAL_CONTEXT_FACTORY,
                //"org.jnp.interfaces.NamingContextFactory"
                "org.jboss.security.jndi.LoginInitialContextFactory"
                );
            props.setProperty(
                Context.PROVIDER_URL,
                "localhost:1099"
                );

            String userName = null;
            String password = null;
            if(args.length >= 1)
            {
                userName = args[0];
                props.setProperty(
                    Context.SECURITY_PRINCIPAL,
                    userName
                    );
            }

            if(args.length >= 2)
            {
                password = args[1];
                props.setProperty(
                    Context.SECURITY_CREDENTIALS,
                    password
                    );
            }

            InitialContext jndiContext = new InitialContext(props);
            Object ref  = jndiContext.lookup("SalamKeadilan");
            SalamKeadilanHome home = (SalamKeadilanHome)
            PortableRemoteObject.narrow (ref, SalamKeadilanHome.class);
            SalamKeadilan ejb = home.create();
```

```
            String pesan = ejb.getPesan();

            System.out.println(pesan);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

**Langkah ke-2**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-3**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$              export              CLASSPATH=$CLASSPATH:/home/lab/jboss-
3.0.4/server/default/lib/jbosssx.jar
$ export CLASSPATH=$CLASSPATH:salamkeadilan.jar
$ javac SalamKeadilanApp.java
```

akan dihasilkan SalamKeadilanApp.class

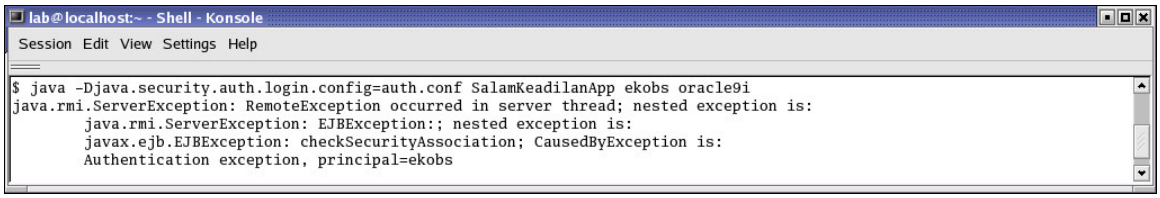
**Langkah ke-4**
Tulis auth.conf, simpan di directory yang sudah dipersiapkan ...

**`auth.conf`**

```
// The default client login module configuration that infects the
// EJB transport layer with the application caller
other {

   // JBoss LoginModule
   org.jboss.security.ClientLoginModule  required
          ;

   // Put your login modules that need JBoss here
};
```

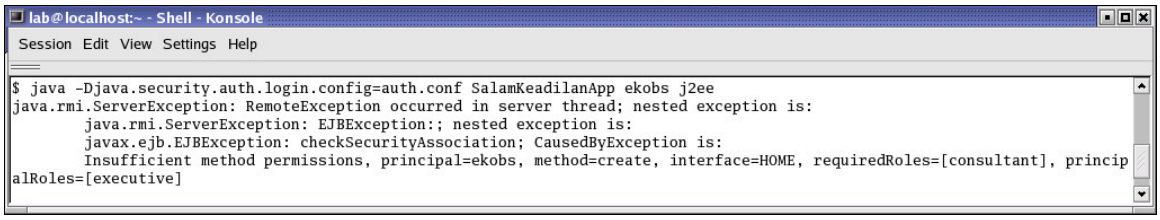
**Langkah ke-5**
Launch ... untuk kasus salah user/password ...

```
$   java   -Djava.security.auth.login.config=auth.conf   SalamKeadilanApp
ekobs oracle9i
java.rmi.ServerException:  RemoteException  occurred  in  server  thread;
nested exception is:
        java.rmi.ServerException: EJBException:; nested exception is:
        javax.ejb.EJBException:                checkSecurityAssociation;
CausedByException is:
        Authentication exception, principal=ekobs
```



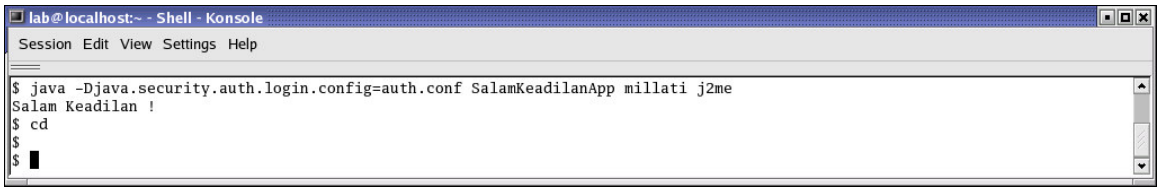

**Langkah ke-5**
Launch ... untuk kasus tanpa access right ...

```
$ java -Djava.security.auth.login.config=auth.conf SalamKeadilanApp
ekobs j2ee
java.rmi.ServerException: RemoteException occurred in server thread;
nested exception is:
        java.rmi.ServerException: EJBException:; nested exception is:
        javax.ejb.EJBException: checkSecurityAssociation;
CausedByException is:
        Insufficient method permissions, principal=ekobs, method=create,
interface=HOME, requiredRoles=[consultant], principalRoles=[executive]
```




**Langkah ke-6**
Launch ... untuk kasus dengan access right ...

```
$ java -Djava.security.auth.login.config=auth.conf SalamKeadilanApp
millati j2me
Salam Keadilan !
```



## *CS-081-154*

Sebuah praktikum untuk mengamankan EJB dengan security constraint. Dengan menyimpan Access Control List di dalam database ...

**Persiapan**
Buat sebuah directory untuk bekerja
misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis login-config.xml simpan di /home/lab/jboss-3.0.4/server/default/conf

**`JBOSS_HOME/server/default/conf/login-config.xml`**

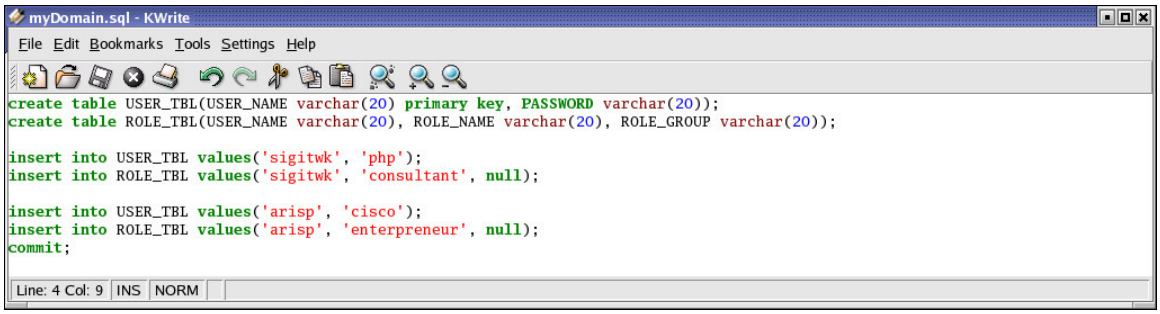
```
<?xml version='1.0'?>
<!DOCTYPE policy PUBLIC
      "-//JBoss//DTD JBOSS Security Config 3.0//EN"
      "http://www.jboss.org/j2ee/dtd/security_config.dtd">

<policy>
    <application-policy name = "myDomain">
       <authentication>
             <login-module code =
"org.jboss.security.auth.spi.DatabaseServerLoginModule"
                flag = "required">
                <module-option name =
"dsJndiName">java:/MySqlDS</module-option>
                <module-option name = "principalsQuery">select PASSWORD
from USER_TBL where USER_NAME=?</module-option>
                <module-option name = "rolesQuery">select ROLE_NAME,
ROLE_GROUP from ROLE_TBL where USER_NAME=?</module-option>
                <module-option name = "debug">1</module-option>
             </login-module>
       </authentication>
    </application-policy>
</policy>
```

**Langkah ke-2**
Restart JBoss

**Langkah ke-3**
Tulis myDomain.sql untuk meng-create dan meng-insert data :

**`myDomain.sql`**

```
create table USER_TBL(USER_NAME varchar(20) primary key, PASSWORD
varchar(20));
create table ROLE_TBL(USER_NAME varchar(20), ROLE_NAME varchar(20),
ROLE_GROUP varchar(20));

insert into USER_TBL values('sigitwk', 'php');
insert into ROLE_TBL values('sigitwk', 'consultant', null);

insert into USER_TBL values('arisp', 'cisco');
insert into ROLE_TBL values('arisp', 'enterpreneur', null);
commit;
```




**Langkah ke-4**
Buka sebuah terminal. Lalu melalui terminal tersebut jalankan mysql client untuk meng-execute myDomain.sql

```
$ mysql –uekobs –pj2ee MYAPP_DB < myDomain.sql
```

**Langkah ke-5**
Tulis SalamKeadilanHome.java, simpan di-directory yang sudah dipersiapkan.

**`SalamKeadilanHome.java`**

```
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import javax.ejb.EJBHome;

public interface SalamKeadilanHome
    extends EJBHome
{
    public SalamKeadilan create()
        throws RemoteException, CreateException;
}
```

**Langkah ke-6**
Tulis SalamKeadilan.java, simpan di directory yang sudah dipersiapkan

**`SalamKeadilan.java`**

```
import javax.ejb.EJBObject;
import java.rmi.RemoteException;

public interface SalamKeadilan
    extends EJBObject
{
    public String getPesan()
        throws RemoteException;
}
```

**Langkah ke-7**
Tulis SalamKeadilanBean.java, simpan di directory yang sudah disiapkan

**`SalamKeadilanBean.java`**

```
import java.rmi.RemoteException;
import javax.ejb.SessionBean;
```

```
import javax.ejb.SessionContext;

public class SalamKeadilanBean
    implements SessionBean
{
   public String getPesan()
   {
        return "Salam Keadilan !";
   }

   public void ejbCreate() {}
   public void ejbRemove() {}
   public void ejbActivate() {}
   public void ejbPassivate() {}
   public void setSessionContext(SessionContext sc) {}
}
```

**Langkah ke-8**
Buka sebuah terminal dan change directory ke directory di atas

**Langkah ke-9**
Compile ...

```
$ export CLASSPATH=.:/home/lab/jboss-3.0.4/client/jbossall-client.jar
$ javac *.java

$ ls *.class
SalamKeadilanBean.class  SalamKeadilan.class  SalamKeadilanHome.class
```

**Langkah ke-10**
Buat sub-directory META-INF

**Langkah ke-11**
Tulis ejb-jar.xml, simpan ke sub directory META-INF

**`META-INF/ejb-jar.xml`**

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar>
    <description>Enterprise Java Lab</description>
    <display-name>Enterprise Java Lab</display-name>
    <enterprise-beans>
        <session>
            <ejb-name>SalamKeadilan</ejb-name>
            <home>SalamKeadilanHome</home>
            <remote>SalamKeadilan</remote>
            <ejb-class>SalamKeadilanBean</ejb-class>
            <session-type>Stateless</session-type>
            <transaction-type>Bean</transaction-type>
            <security-role-ref>
                <role-name>consultant</role-name>
                <role-link>consultant</role-link>
            </security-role-ref>
        </session>
    </enterprise-beans>
    <assembly-descriptor>
        <security-role>
            <description>consultant</description>
```

```
            <role-name>consultant</role-name>
        </security-role>
        <method-permission>
            <role-name>consultant</role-name>
            <method>
                <ejb-name>SalamKeadilan</ejb-name>
                <method-name>*</method-name>
            </method>
        </method-permission>
    </assembly-descriptor>
</ejb-jar>
```

**Langkah ke-12**
Tulis jboss.xml, simpan ke sub directory META-INF

**`META-INF/jboss.xml`**

```
<?xml version="1.0" encoding="UTF-8"?>

<jboss>
   <!-- All bean containers use this security manager by default -->
   <container-configurations>
        <container-configuration>
            <container-name>
            Standard Stateless SessionBean
            </container-name>
            <security-domain>
            java:/jaas/myDomain
            </security-domain>
        </container-configuration>
   </container-configurations>
</jboss>
```

**Langkah ke-13**
Gabungkan ke dalam sebuah jar

```
$ jar cvf salamkeadilan.jar *.class  META-INF/
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/MANIFEST.MF (in=56) (out=56) (stored 0%)
adding: SalamKeadilanBean.class (in=634) (out=324) (deflated 48%)
adding: SalamKeadilan.class (in=229) (out=176) (deflated 23%)
adding: SalamKeadilanHome.class (in=263) (out=184) (deflated 30%)
adding: META-INF/ (in=0) (out=0) (stored 0%)
adding: META-INF/ejb-jar.xml (in=1144) (out=334) (deflated 70%)
adding: META-INF/jboss.xml (in=369) (out=203) (deflated 44%)
Total:
------
(in = 2684) (out = 2178) (deflated 18%)
```

**Langkah ke-14**
Copy ke dalam deployment directory dari jBoss

```
$ cp salamkeadilan.jar /home/lab/jboss-3.0.4/server/default/deploy/
```

Terpantau di log dari JBoss :

```
07:28:55,709  INFO    [MainDeployer]  Starting  deployment  of  package:
file:/home/lab/jboss-3.0.4/server/default/deploy/salamkeadilan.jar
07:28:55,845 INFO  [EjbModule] Creating
07:28:55,907 INFO  [EjbModule] Deploying SalamKeadilan
```

```
07:28:55,997 INFO  [EjbModule] Created
07:28:55,998 INFO  [EjbModule] Starting
07:28:56,334 INFO  [EjbModule] Started
07:28:56,335     INFO          [MainDeployer]     Deployed     package:
file:/home/lab/jboss-3.0.4/server/default/deploy/salamkeadilan.jar
```

Module EJB telah di-deploy di atas JBoss enterprise application server dan siap melayani request dari client.

mengakses EJB yang diamankan ...

**Persiapan**
Buat sebuah directory untuk bekerja, misalnya /home/lab/myjava

**Langkah**
**Langkah ke-1**
Tulis SalamKeadilanApp.java, simpan di-directory yang sudah dipersiapkan.

**`SalamKeadilanApp.java`**

```
import java.util.Properties;
import javax.naming.*;
import javax.rmi.PortableRemoteObject;
import javax.security.auth.callback.*;
import javax.security.auth.login.*;

public class SalamKeadilanApp
{
    public static void main(String[] args)
    {
        try
        {
            Properties props = new Properties();
            props.setProperty(
                Context.INITIAL_CONTEXT_FACTORY,
                //"org.jnp.interfaces.NamingContextFactory"
                "org.jboss.security.jndi.LoginInitialContextFactory"
                );
            props.setProperty(
                Context.PROVIDER_URL,
                "localhost:1099"
                );

            String userName = null;
            String password = null;
            if(args.length >= 1)
            {
                userName = args[0];
                props.setProperty(
                    Context.SECURITY_PRINCIPAL,
                    userName
                    );
            }

            if(args.length >= 2)
            {
                password = args[1];
                props.setProperty(
                    Context.SECURITY_CREDENTIALS,
                    password
                    );
            }

            InitialContext jndiContext = new InitialContext(props);
            Object ref  = jndiContext.lookup("SalamKeadilan");
            SalamKeadilanHome home = (SalamKeadilanHome)
```

```
            PortableRemoteObject.narrow (ref, SalamKeadilanHome.class);
            SalamKeadilan ejb = home.create();

            String pesan = ejb.getPesan();

            System.out.println(pesan);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

**Langkah ke-2**
Buka sebuah terminal, dan change directory ke directory kerja tersebut

**Langkah ke-3**
Compile ...

```
$ export CLASSPATH=.
$   export   CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/jbossall-
client.jar
$ export CLASSPATH=$CLASSPATH:/home/lab/jboss-3.0.4/client/log4j.jar
$              export              CLASSPATH=$CLASSPATH:/home/lab/jboss-
3.0.4/server/default/lib/jbosssx.jar
$ export CLASSPATH=$CLASSPATH:salamkeadilan.jar
$ javac SalamKeadilanApp.java
```

akan dihasilkan SalamKeadilanApp.class

**Langkah ke-4**
Tulis auth.conf, simpan di directory yang sudah dipersiapkan ...

**`auth.conf`**

```
// The default client login module configuration that infects the
// EJB transport layer with the application caller
other {

   // JBoss LoginModule
   org.jboss.security.ClientLoginModule  required
          ;

   // Put your login modules that need JBoss here
};
```

**Langkah ke-5**
Launch ... untuk kasus salah user/password ...

```
$   java   -Djava.security.auth.login.config=auth.conf   SalamKeadilanApp
millati j2me
java.rmi.ServerException:  RemoteException  occurred  in  server  thread;
nested exception is:
        java.rmi.ServerException: EJBException:; nested exception is:
        javax.ejb.EJBException:                checkSecurityAssociation;
CausedByException is:
        Authentication exception, principal=millati
        at
sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:292)
```

```
        at sun.rmi.transport.Transport$1.run(Transport.java:148)
        at java.security.AccessController.doPrivileged(Native Method)
        at sun.rmi.transport.Transport.serviceCall(Transport.java:144)
        at
sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:460)
        at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(TCPTransport.ja
va:701)
        at java.lang.Thread.run(Thread.java:536)
```

**Langkah ke-5**

Launch ... untuk kasus tanpa access right ...

```
$  java  -Djava.security.auth.login.config=auth.conf  SalamKeadilanApp
arisp cisco
java.rmi.ServerException: RemoteException occurred in server thread;
nested exception is:
        java.rmi.ServerException: EJBException:; nested exception is:
        javax.ejb.EJBException:               checkSecurityAssociation;
CausedByException is:
        Insufficient method permissions, principal=arisp, method=create,
interface=HOME,                              requiredRoles=[consultant],
principalRoles=[enterpreneur]
        at
sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:292)
        at sun.rmi.transport.Transport$1.run(Transport.java:148)
        at java.security.AccessController.doPrivileged(Native Method)
        at sun.rmi.transport.Transport.serviceCall(Transport.java:144)
        at
sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:460)
        at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(TCPTransport.ja
va:701)
        at java.lang.Thread.run(Thread.java:536)
```

**Langkah ke-6**

Launch ... untuk kasus dengan access right ...

```
$  java  -Djava.security.auth.login.config=auth.conf  SalamKeadilanApp
sigitwk php
Salam Keadilan !
```