

Aplikasi Enterprise dan Web dengan Java J2EE

Eko Budhi Suprasetiawan

ekobs@developerforce.net

Lisensi Dokumen:

Copyright © 2003-2006 IlmuKomputer.Com

Seluruh dokumen di IlmuKomputer.Com dapat digunakan, dimodifikasi dan disebarkan secara bebas untuk tujuan bukan komersial (nonprofit), dengan syarat tidak menghapus atau merubah atribut penulis dan pernyataan copyright yang disertakan dalam setiap dokumen. Tidak diperbolehkan melakukan penulisan ulang, kecuali mendapatkan ijin terlebih dahulu dari IlmuKomputer.Com.

1. Instalasi Tomcat

Pengantar

JavaServlet dan Java Server Page merupakan solusi Java untuk pengembangan aplikasi berbasis Web. Untuk dapat bekerja, JavaServlet dan Java Server Page membutuhkan engine. Di antara engine yang tersedia sebagai open source adalah Tomcat. Tomcat dikembangkan sebagai bagian dari Project Jakarta yang bernaung di dalam Apache Software Foundation. Website tentang Tomcat dapat dijangkau melalui <http://jakarta.apache.org>

Proses Installation

Mendapatkan installation file

Installation file bisa di-download dari jakarta.apache.org. Anda dapat men-download source file atau binary file. File tersedia dalam compressed file ber-extension .tar dan .zip.

Installation dengan ZIP file

Untuk meng-install Java Development Kit dari installation file ber-format ZIP, Anda dapat meng-unzipnya.

Misalkan di atas Linux, installation file yang Anda download bernama jakarta-tomcat-3.2.1.zip. Tentukan di mana Anda akan meng-install. Misalkan Anda akan meng-install ke directory /home/lab. Pindahkan file jakarta-tomcat-3.2.1.zip ke directory pilihan Anda tersebut, selanjutnya jalankan unzip melalui console dari directory tersebut.

```
$ unzip jakarta-tomcat-3.2.1.zip
Maka installation file akan di unzip :
Archive:  jakarta-tomcat-3.2.1.zip
  creating: jakarta-tomcat-3.2.1/
```

```
inflating: jakarta-tomcat-3.2.1/LICENSE
creating: jakarta-tomcat-3.2.1/webapps/
inflating: jakarta-tomcat-3.2.1/webapps/ROOT.war
inflating: jakarta-tomcat-3.2.1/webapps/test.war
inflating: jakarta-tomcat-3.2.1/webapps/examples.war
inflating: jakarta-tomcat-3.2.1/webapps/admin.war
```

sehingga akhir :

```
creating: jakarta-tomcat-3.2.1/src/org/apache/tomcat/startup/
inflating:
jakarta-tomcat-3.2.1/src/org/apache/tomcat/startup/EmbeddedTomcat.java
inflating:
jakarta-tomcat-3.2.1/src/org/apache/tomcat/startup/HostConfig.java
inflating:
jakarta-tomcat-3.2.1/src/org/apache/tomcat/startup/Tomcat.java
creating: jakarta-tomcat-3.2.1/logs/
```

Anda mendapatkan sebuah sub directory jakarta-tomcat-3.2.1 di bawah directory /home/lab. Directory /home/lab/jakarta-tomcat-3.2.1 ini disebut TOMCAT_HOME. Selesailah proses instalasi Tomcat, dan Anda siap bekerja dengan JavaServlet dan Java Server Pages.

Struktur Directory Di Bawah TOMCAT_HOME

Di dalam directory TOMCAT_HOME terdapat beberapa sub directory, di antaranya :

- bin, di mana script untuk menjalankan dan menghidupkan Tomcat berada.
- conf, di mana file-file konfigurasi berada.
- lib, di mana file-file library ber-extension .jar berada.
- webapps, di mana, secara default, Anda dapat meletakkan JavaServlet dan JSP.

Proses Menjalankan Tomcat

Untuk menjalankan Tomcat :

1. Set variabel lingkungan PATH agar memuat directory dimana java berada

Contoh :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
```

2. Change directory ke TOMCAT_HOME/bin

Misalnya :

```
$ cd /home/lab/jakarta-tomcat-3.2.1/bin
```

3. Jalankan startup.bat

Misalnya :

```
$ ./startup.sh
```

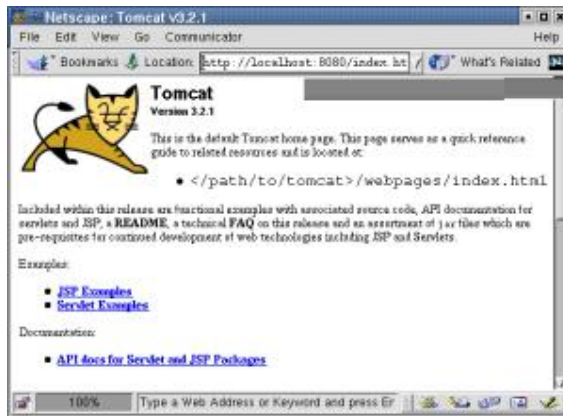
4. Tomcat akan berjalan

Misalnya :

```
Guessing TOMCAT_HOME from tomcat.sh to ./...
Setting TOMCAT_HOME to ./...
Using classpath:
./../lib/ant.jar:./../lib/jasper.jar:./../lib/jaxp.jar:./../lib/
parser.jar:
./../lib/servlet.jar:./../lib/test:./../lib/webserver.jar:/home/
lab/jdk1.3.1_01/bin/./lib/tools.jar
[lab@localhost bin]$ 2002-06-03 09:57:41 - ContextManager: Adding
context Ctx( /examples )
2002-06-03 09:57:41 - ContextManager: Adding context Ctx( /admin )
Starting tomcat. Check logs/tomcat.log for error messages
2002-06-03 09:57:41 - ContextManager: Adding context Ctx( )
2002-06-03 09:57:41 - ContextManager: Adding context Ctx( /test )
2002-06-03 09:57:41 - PoolTcpConnector: Starting
HttpConnectionHandler on 8080
2002-06-03 09:57:41 - PoolTcpConnector: Starting
Ajpl2ConnectionHandler on 8007
```

5. Uji melalui Web browser

Anda dapat melihat Tomcat dengan mem-browse ke <http://localhost:8080>



Proses Menghentikan Tomcat

Untuk menjalankan Tomcat :

1. Set variabel lingkungan PATH agar memuat directory di mana java berada

Contoh :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
```

2. Change directory ke TOMCAT_HOME/bin

Misalnya :

```
$ cd /home/lab/jakarta-tomcat-3.2.1/bin
```

3. Jalankan startup.bat

Misalnya :

```
$ ./shutdown.sh
```

4. Tomcat akan berhenti

Misalnya :

```
Guessing TOMCAT_HOME from tomcat.sh to ./..  
Setting TOMCAT_HOME to ./..  
Using classpath:  
./../lib/ant.jar:./../lib/jasper.jar:./../lib/jaxp.jar:./../lib/  
parser.jar:  
./../lib/servlet.jar:./../lib/test:./../lib/webserver.jar:/home/  
lab/jdk1.3.1_01/bin/./lib/tools.jar  
Stop tomcat
```

Troubleshoot

Beberapa trouble yang lumrah terjadi saat menjalankan Tomcat :

```
Cannot find JAVA. Please set your PATH.
```

Anda perlu menge-set variabel lingkungan dengan benar agar memuat directory di mana java berada.

```
FATAL:java.net.BindException: Address already in use  
java.net.BindException: Address already in use  
    at java.net.PlainSocketImpl.socketBind(Native Method)  
    at java.net.PlainSocketImpl.bind(PlainSocketImpl.java:405)  
    at java.net.ServerSocket.(ServerSocket.java:170)  
    at java.net.ServerSocket.(ServerSocket.java:121)  
    at  
org.apache.tomcat.net.DefaultServerSocketFactory.createSocket(Default  
ServerSocketFactory.java:97)  
    at  
org.apache.tomcat.service.PoolTcpEndpoint.startEndpoint(PoolTcpEndp  
oint.java:239)  
    at  
org.apache.tomcat.service.PoolTcpConnector.start(PoolTcpConnector.j  
ava:188)  
    at  
org.apache.tomcat.core.ContextManager.start(ContextManager.java:527  
)  
    at org.apache.tomcat.startup.Tomcat.execute(Tomcat.java:202)  
    at org.apache.tomcat.startup.Tomcat.main(Tomcat.java:235)
```

Sebuah aplikasi lain atau sebuah Tomcat sudah berjalan di port 8080. Anda dapat menghentikan aplikasi lain tsb, atau mengubah file konfigurasi server.xml di bawah directory TOMCAT_HOME/conf

2. Instalasi Axis

Pengantar

Axis adalah project open source dalam lingkungan Apache.org yang dikembangkan untuk menyediakan wahana bagi web service dengan Java.

Persiapan

Untuk dapat bekerja dengan Axis, Anda membutuhkan

1. Java Development Kit
2. Web application server/container
3. XML parser

Dalam tutorial ini :

1. JDK 1.3 telah di-instal di home/lab/jdk1.3.1_01
2. Tomcat 4.0.4 telah di-instal di /home/lab/jakarta-tomcat-4.0.4
3. Xerces 2.0.2 telah di-instal di /home/lab/xerces-2_0_2

Proses Installation

Proses instalasi Axis mencakup :

1. Download installation file
2. Extract
3. Deploy ke web application server/container, seperti Tomcat

Download installation file

File instalasi Axis dapat di-download dari <http://xml.apache.org/axis>

Extract

Misalkan di atas Linux, installation file yang Anda download bernama xml-axis-beta2.tar.gz Tentukan di mana Anda akan meng-install. Misalkan Anda akan meng-install ke directory /home/lab. Pindahkan file xml-axis-beta2.tar.gz ke directory pilihan Anda tersebut, selanjutnya jalankan unzip melalui console dari directory tersebut.

```
$ gunzip xml-axis-beta2.tar.gz
$ tar -xvf xml-axis-beta2.tar
Maka installation file akan di-extract :
xml-axis-beta2/
xml-axis-beta2/LICENSE
xml-axis-beta2/README
xml-axis-beta2/release-notes.html
xml-axis-beta2/docs/
xml-axis-beta2/docs/architecture-guide.html
```

sehingga akhir :

```
xml-axis-beta2/webapps/axis/WEB-INF/lib/  
xml-axis-beta2/webapps/axis/WEB-INF/lib/axis.jar  
xml-axis-beta2/webapps/axis/WEB-INF/lib/commons-logging.jar  
xml-axis-beta2/webapps/axis/WEB-INF/lib/jaxrpc.jar  
xml-axis-beta2/webapps/axis/WEB-INF/lib/log4j-core.jar  
xml-axis-beta2/webapps/axis/WEB-INF/lib/tt-bytecode.jar  
xml-axis-beta2/webapps/axis/WEB-INF/lib/wsd14j.jar  
Anda mendapatkan sebuah sub directory xml-axis-beta2 di bawah directory /home/lab.
```

Struktur Directory Di Bawah AXIS_HOME

Di dalam directory AXIS_HOME terdapat beberapa sub directory, di antaranya :

- lib, di mana file-file library ber-extension .jar berada.
- webapps, di mana webapp Axis berada

Buat keputusan tentang nama webapp

Axis dapat dipandang sebagai sebuah aplikasi web dan untuk menghidupkannya dibutuhkan web application server/container seperti Tomcat. Sebelum men-deploy Axis, Anda perlu menetapkan nama webapp dimana Axis akan dideploy. Dalam tutorial ini, nama webapp-nya adalah mahakam.

Deployment ke atas Tomcat

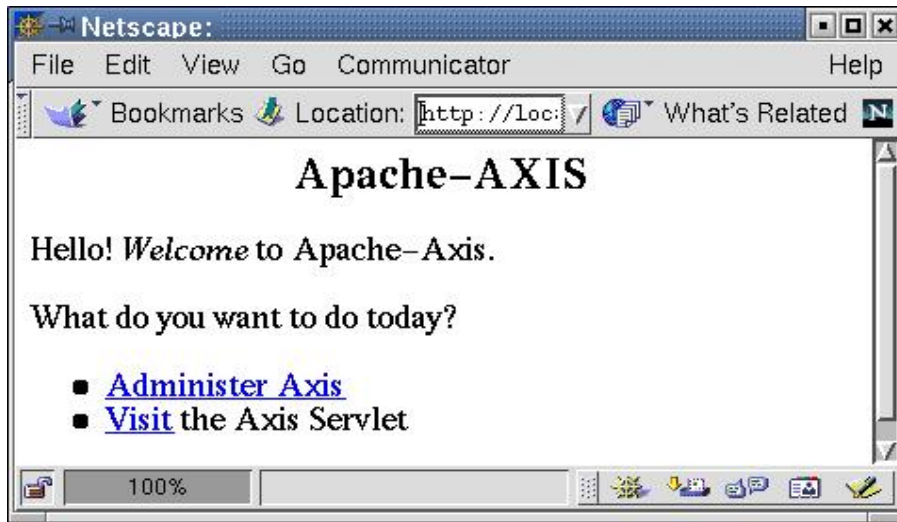
Untuk menjadikan Axis sebagai webapp yang hidup dalam Tomcat, Anda hanya meng-copy directory AXIS_HOME/webapps/axis ke TOMCAT_HOME/webapps dengan mengubah nama directory axis menjadi nama directory pilihan Anda.

```
$ cp -R /home/lab/xml-axis-beta2/webapps/axis  
/home/lab/jakarta-tomcat-3.2.1/webapps/mahakam
```

Menjalankan Tomcat

```
$ export JAVA_HOME=/home/lab/jdk1.3.1_01  
$ /home/lab/jakarta-tomcat-4.0.4/bin/startup.sh  
Using CATALINA_BASE: /home/lab/jakarta-tomcat-4.0.4  
Using CATALINA_HOME: /home/lab/jakarta-tomcat-4.0.4  
Using CATALINA_TMPDIR: /home/lab/jakarta-tomcat-4.0.4/temp  
Using JAVA_HOME: /home/lab/jdk1.3.1_01
```

Mengunjungi Axis home



Menguji instalasi Axis

Untuk menguji apakah proses instalasi yang Anda lakukan berhasil, Anda dapat mencoba sample yang disediakan Axis.

Untuk memulai, persiapkan dulu PATH dan CLASSPATH

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
$ export CLASSPATH=/home/lab/xerces-2_0_2/xmlParserAPIs.jar:/home/lab/xerces-2_0_2/xercesImpl.jar:/home/lab/xml-axis-beta2/lib/axis.jar:/home/lab/xml-axis-beta2/lib/jaxrpc.jar:/home/lab/xml-axis-beta2/lib/log4j-core.jar:/home/lab/xml-axis-beta2/lib/commons-logging.jar:/home/lab/xml-axis-beta2
```

Selanjutnya, lakukan deployment sebuah webservice :

```
$ java org.apache.axis.client.AdminClient
-lhttp://localhost:8080/mahakam/services/AdminService
/home/lab/xml-axis-beta2/samples/stock/deploy.wsdd
Jika berjalan sukses, akan muncul log berikut :
- Processing file /home/lab/xml-axis-beta2/samples/stock/deploy.wsdd
<Admin>Done processing</Admin>
```

Akhirnya, lakukan pemanggilan web service tersebut melalui client

```
$ java samples.stock.GetQuote
-lhttp://localhost:8080/mahakam/servlet/AxisServlet -uuser1 -wpass1 XXX
Jika berjalan sukses, akan muncul log berikut :
```

XXX: 55.25

3. Instalasi jBoss

Pengantar

jBoss adalah sebuah application server yang dikembangkan oleh masyarakat open source. Anda dapat menggunakan jBoss untuk menjadi EJB server/container.

Proses Installation

Mendapatkan installation file

Anda dapat mendapatkan installation file dari <http://www.jboss.org>. Misalkan Anda men-download jboss-3.0.0.zip

Installation dengan ZIP file

Misalkan di atas Linux, installation file yang Anda download bernama jboss-3.0.0.zip. Tentukan di mana Anda akan meng-install. Misalkan Anda akan meng-install ke directory /home/lab. Pindahkan file jboss-3.0.0.zip ke directory pilihan Anda tersebut, selanjutnya jalankan unzip melalui console dari directory tersebut.

```
$ unzip jboss-3.0.0.zip
```

Maka installation file akan di unzip :

```
Archive:  jboss-3.0.0.zip
  creating: jboss-3.0.0/
  creating: jboss-3.0.0/lib/
  creating: jboss-3.0.0/docs/
  creating: jboss-3.0.0/docs/dtd/
  creating: jboss-3.0.0/docs/examples/
  creating: jboss-3.0.0/docs/examples/jca/
  creating: jboss-3.0.0/client/
  creating: jboss-3.0.0/bin/
  creating: jboss-3.0.0/server/
sehingga akhir :
  inflating: jboss-3.0.0/server/minimal/conf/jndi.properties
  inflating: jboss-3.0.0/server/minimal/conf/log4j.xml
  inflating: jboss-3.0.0/server/minimal/conf/jboss-service.xml
  inflating: jboss-3.0.0/server/minimal/lib/jnet.jar
  inflating: jboss-3.0.0/server/minimal/lib/log4j.jar
  inflating: jboss-3.0.0/server/minimal/lib/jnpserver.jar
```

```
$ unzip jakarta-tomcat-3.2.1.zip
```

Anda mendapatkan sebuah sub directory jboss-3.0.0 di bawah directory /home/lab. Directory /home/lab/jboss-3.0.0 ini disebut JBOSS_HOME. Selesailah proses instalasi jBoss, dan Anda siap bekerja dengan Enterprise Java Bean dengan jBoss.

Struktur Directory Di Bawah JBOSS_HOME

Di dalam directory JBOSS_HOME terdapat beberapa sub directory, di antaranya :

- bin, di mana script untuk menjalankan dan menghidupkan jBoss berada.
- client, di mana file-file library ber-extension .jar yang dibutuhkan untuk client.
- lib, di mana file-file library ber-extension .jar berada.

- server, di antaranya mempunyai sub directory default yang memuat file-file konfigurasi, jar yang dijalankan jBoss dalam default.

Proses Menjalankan jBoss

Untuk menjalankan jBoss, set variabel lingkungan JAVA_HOME agar memuat directory dimana java berada

Contoh :

```
$ export JAVA_HOME=/home/lab/jdk1.3.1_01
```

Jalankan run.sh di dalam directory JBOSS_HOME/bin/

Misalnya :

```
$ sh /home/lab/jboss-3.0.0/bin/run.sh
```

jBoss akan berjalan ...

```
$ sh /home/lab/jboss-3.0.0/bin/shutdown.sh
```

```
Shutting down server localhost:8082
```

```
Shutdown complete
```

```
=====
```

```
JBoss Bootstrap Environment
```

```
JBOSS_HOME: /home/lab/jboss-3.0.0
```

```
JAVA: /home/lab/jdk1.3.1_01/bin/java
```

```
JAVA_OPTS: -server -Dprogram.name=run.sh
```

```
CLASSPATH:
```

```
/home/lab/jboss-3.0.0/bin/run.jar:/home/lab/jdk1.3.1_01/lib/tools.jar
```

```
=====
```

```
23:49:25,532 INFO [Server] JBoss Release: JBoss-3.0.0 CVSTag=JBoss_3_0_0
```

```
23:49:25,615 INFO [Server] Home Dir: /home/lab/jboss-3.0.0
```

```
23:49:25,616 INFO [Server] Home URL: file:/home/lab/jboss-3.0.0/
```

```
23:49:25,617 INFO [Server] Library URL: file:/home/lab/jboss-3.0.0/lib/
```

```
23:49:25,630 INFO [Server] Patch URL: null
```

```
23:49:25,631 INFO [Server] Server Name: default
```

```
23:49:25,632 INFO [Server] Server Home Dir: /home/lab/jboss-3.0.0/server/default
```

```
23:49:25,634 INFO [Server] Server Home URL: file:/home/lab/jboss-3.0.0/server/default/
```

```
23:49:25,634 INFO [Server] Server Data Dir: /home/lab/jboss-3.0.0/server/default/db
```

```
23:49:25,635 INFO [Server] Server Temp Dir: /home/lab/jboss-3.0.0/server/default/tmp
```

```
23:49:25,636 INFO [Server] Server Config URL:
file:/home/lab/jboss-3.0.0/server/default/conf/
23:49:25,637 INFO [Server] Server Library URL:
file:/home/lab/jboss-3.0.0/server/default/lib/
23:49:25,638 INFO [Server] Root Deployment Filename: jboss-service.xml
23:49:25,655 INFO [Server] Starting General Purpose Architecture
(GPA)...
...
hingga akhirnya ...
23:50:11,296 INFO [MainDeployer] Starting deployment of package: file:
/home/lab/jboss-3.0.0/server/default/deploy/ejb-management.jar
23:50:11,963 INFO [EjbModule] Creating
23:50:12,117 INFO [EjbModule] Deploying MEJB
23:50:12,300 INFO [EjbModule] Created
23:50:12,301 INFO [EjbModule] Starting
23:50:12,442 INFO [EjbModule] Started
23:50:12,443 INFO [MainDeployer] Successfully completed deployment of
package:
file:/home/lab/jboss-3.0.0/server/default/deploy/ejb-management.jar
23:50:14,177 INFO [URLDeploymentScanner] Started
23:50:14,178 INFO [MainDeployer] Successfully completed deployment of
package:
file:/home/lab/jboss-3.0.0/server/default/conf/jboss-service.xml
23:50:14,179 INFO [Server] JBoss (MX MicroKernel) [3.0.0
Date:200205311035] Started in 0m:48s:524msd
```

Proses Menghentikan jBoss

Untuk menghentikan jBoss, bukanlah console baru. Set variabel lingkungan JAVA_HOME agar memuat directory dimana java berada

Contoh :

```
$ export JAVA_HOME=/home/lab/jdk1.3.1_01
```

Jalankan shutdown.sh di dalam directory JBOSS_HOME/bin/

Misalnya :

```
$ sh /home/lab/jboss-3.0.0/bin/shutdown.sh
Shutting down server localhost:8082
Shutdown complete
```

Di console dimana jBoss dijalankan, akan terlihat log :

```
05:12:31,258 INFO [Server] Shutting down
05:12:31,264 INFO [Server] Shutting down the JVM now!
05:12:31,266 INFO [Server] undeploying all packages
05:12:31,267 INFO [MainDeployer] Undeploying
file:/home/lab/jboss-3.0.0/server/default/deploy/ejb-management.jar
05:12:31,269 INFO [EjbModule] Stopping
05:12:31,270 INFO [EjbModule] Stopped
05:12:31,271 INFO [EjbModule] Destroying
05:12:31,331 INFO [EjbModule] Remove JSR-77 EJB Module:
jboss.management.single:J2EEApplication=
```

```
,J2EEServer=Single,j2eeType=EJBModule,name=ejb-management.jar  
05:12:31,333 INFO [EjbModule] Destroyed
```

...

Sehingga :

...

```
23:53:32,077 INFO [MainDeployer] Undeployed 61 deployed packages  
23:53:32,078 INFO [Server] Shutting down all services  
Shutting down  
23:53:32,080 INFO [ServiceController] Stopping 3 services  
23:53:32,081 INFO [SARDeployer] Stopping  
23:53:32,082 INFO [MainDeployer] Removing deployer:  
org.jboss.deployment.SARDeployer@237368  
23:53:32,083 INFO [SARDeployer] Stopped  
23:53:32,084 INFO [JARDeployer] Stopping  
23:53:32,085 INFO [JARDeployer] Stopped  
23:53:32,085 INFO [MainDeployer] Stopping  
23:53:32,086 INFO [MainDeployer] Stopped  
23:53:32,086 INFO [ServiceController] Stopped 3 services  
23:53:32,087 INFO [Server] Shutdown complete  
Shutdown complete
```

4. J2EE Technology

JDBC

Java Database Connectivity (JDBC) adalah API yang mendeskripsikan library yang standard dalam Java untuk mengakses data source, terutama database relasional, yang menggunakan SQL.

Servlet

Servlet adalah program Java yang berjalan di sebuah Web server.

JSP

Di permukaan, Java Server Pages dapat dipandang sebagai server-side scripting dalam bahasa Java. Dalam siklus kehidupannya, JSP akan ditulis ulang dan di-compile menjadi JavaServlet dan dijalankan serta bertingkah laku seperti JavaServlet.

RMI

Remote Method Invocation (RMI) adalah sebuah mekanisme yang membolehkan sebuah obyek yang berada di sebuah Java Virtual Machine untuk meng-invoke metoda dari obyek lain yang berada di sebuah Java Virtual Machine yang lain.

JNDI

Java Naming and Directory Interface (JNDI) adalah sebuah API yang mendeskripsikan library Java yang standar untuk mengakses layanan naming dan directory seperti Domain Naming Service (DNS), dan Lightweight Directory Access Protocol (LDAP)

EJB

Enterprise Java Bean (EJB) adalah spesifikasi untuk komponen yang hidup di server, dimaksudkan untuk mendefinisikan cara yang standard dalam menciptakan komponen yang berpartisipasi dalam aplikasi berorientasi obyek yang terdistribusi.

JMS

Java Messaging Service (JMS) adalah sebuah kumpulan Java interface yang merupakan spesifikasi tentang messaging dalam Java, tepatnya untuk berinteraksi dengan Message-Oriented Middleware (MOM)

5. Latihan RMI : SalamKeadilan

Tujuan

Dalam bagian ini Anda akan bekerja dengan aplikasi terdistribusi yang berkomunikasi dengan teknik Remote Method Invocation.

Rancangan

Sebuah aplikasi, SalamKeadilanRMIServerApp, yang berjalan di sebuah JVM, berperan sebagai server yang berjalan di sebuah JVM, dan sebuah aplikasi lain, SalamKeadilanRMIClientApp berjalan di JVM terpisah akan berperan sebagai client.

Pembekalan

Remote Method Invocation (RMI) adalah sebuah teknik yang membolehkan sebuah obyek yang berada dalam sebuah JVM untuk meng-invoke method dari obyek lain yang berada dalam JVM yang terpisah.

Persiapan

- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Buatlah sebuah directory untuk latihan Anda, misalnya /home/lab/climb

Mengembangkan Server

Langkah 1 : Dengan text editor tuliskan SalamKeadilanRemote.java

```
import java.rmi.*;

public interface SalamKeadilanRemote
    extends Remote
{
    public String getPesan()
        throws RemoteException;
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai SalamKeadilanRemote.java

Langkah 2 : Dengan text editor tuliskan SalamKeadilanRemoteImpl.java

```
import java.rmi.*;
import java.rmi.server.*;

public class SalamKeadilanRemoteImpl
    extends UnicastRemoteObject
    implements SalamKeadilanRemote
{
    public SalamKeadilanRemoteImpl()
        throws RemoteException
    {
        super();
    }
}
```

```
public String getPesan()  
{  
    return "Salam Keadilan !";  
}  
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai SalamKeadilanRemoteImpl.java

Langkah 3 : Dengan text editor tulislah SalamKeadilanRMIServerApp.java

```
import java.rmi.*;  
import java.net.*;  
  
public class SalamKeadilanRMIServerApp  
{  
    public static void main(String args[])  
    {  
        System.setSecurityManager(  
            new RMISecurityManager());  
        try  
        {  
            SalamKeadilanRemote remote = new SalamKeadilanRemoteImpl();  
            Naming.rebind("SalamKeadilan", remote);  
        }  
        catch(RemoteException re)  
        {  
            re.printStackTrace();  
        }  
        catch(MalformedURLException murle)  
        {  
            murle.printStackTrace();  
        }  
    }  
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai SalamKeadilanRMIServerApp.java

Mengembangkan Client

Langkah 4 : Dengan text editor tulislah SalamKeadilanRMIClientApp.java

```
import java.rmi.*;  
import java.net.*;  
  
public class SalamKeadilanRMIClientApp  
{  
    public static void main(String args[])  
    {  
        System.setSecurityManager(  
            new RMISecurityManager());  
        try
```

```
        {  
            SalamKeadilanRemote remote = (SalamKeadilanRemote)  
Naming.lookup("SalamKeadilan");  
            String pesan = remote.getPesan();  
            System.out.println(pesan);  
        }  
        catch(RemoteException re)  
        {  
            re.printStackTrace();  
        }  
        catch(NotBoundException nbe)  
        {  
            nbe.printStackTrace();  
        }  
        catch(MalformedURLException murle)  
        {  
            murle.printStackTrace();  
        }  
    }  
}
```

Penyiapan Lingkungan

Langkah 5 : Bukalah sebuah console

Untuk menghidupkan RMI Registry, bukalah sebuah console, dan jalankan rmiregistry.

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH  
$ rmiregistry
```

Langkah 6 : Menuliskan security policy

```
grant  
{  
    permission java.net.SocketPermission "*:1024-65535",  
        "connect,accept";  
    permission java.io.FilePermission  
        "/home/lab/climb/-", "read";  
};
```

Simpanlah sebagai climb.policy di directory yang dipersiapkan sebelumnya.

Meng-compile dan Meluncurkan Server

Langkah 6 : Bukalah sebuah console

Anda akan meng-compile dan meluncurkan aplikasi dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan PATH harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan PATH :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
```

Variabel lingkungan CLASSPATH harus memuat directory di mana file .class dari aplikasi Anda berada. Contoh untuk menge-set variabel lingkungan CLASSPATH :

```
$ export CLASSPATH=/home/lab/climb
```

Langkah 6 : Meng-compile SalamKeadilanRMIServerApp.java

Melalui console, jalankan javac :

```
$ javac SalamKeadilanRMIServerApp.java
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file SalamKeadilanRemote.class, SalamKeadilanRemoteImpl.class dan SalamKeadilanRMIServerApp.class di dalam directory yang sama.

Langkah 7 : Men-generate stub dan skeleton menggunakan rmic

Melalui console, jalankan rmic :

```
$ rmic -d . SalamKeadilanRemoteImpl
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file SalamKeadilanRemoteImpl_Skel.class dan SalamKeadilanRemote_Stub.class di dalam directory yang sama.

Langkah 7 : Meluncurkan aplikasi SalamKeadilanRMIServerApp

Untuk meluncurkan aplikasi Anda, melalui console, jalankan java :

```
$ java -Djava.rmi.server.codebase=file:/home/lab/climb/  
-Djava.rmi.server.hostname=localhost  
-Djava.security.policy=climb.policy SalamKeadilanRMIServerApp
```

Meng-compile dan Meluncurkan Client

Langkah 8 : Bukalah sebuah console lainnya

Anda akan meng-compile dan meluncurkan aplikasi dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan PATH harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan PATH :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
```

Variabel lingkungan CLASSPATH harus memuat directory di mana file .class dari aplikasi Anda berada. Contoh untuk menge-set variabel lingkungan CLASSPATH :

```
$ export CLASSPATH=/home/lab/climb
```

Langkah 9: Meng-compile SalamKeadilanClientApp.java

Melalui console, jalankan javac :

```
$ javac SalamKeadilanRMIClientApp.java
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file SalamKeadilanClientApp.class.

Langkah 10 : Meluncurkan aplikasi SalamKeadilanRMIClientApp

Untuk meluncurkan aplikasi Anda, melalui console, jalankan java :

```
$ java -Djava.rmi.server.hostname=localhost  
-Djava.security.policy=climb.policy  
SalamKeadilanRMIClientApp
```

6. Program Client-Server menggunakan JDBC, DAO dan RMI

Tujuan

Anda akan mengembangkan sebuah aplikasi client server, di mana sebuah Data Access Object yang berada di sisi server melayani client yang berada dalam JVM yang berbeda untuk mengakses sebuah database.

Rancangan

Anda mengembangkan sebuah obyek yang mengakses sebuah table dalam database, yaitu CatalogueDAO yang mengakses table CATALOGUETBL dalam database AMAZONDB. CatalogueDAO mempunyai method-method untuk menyimpan, membaca dan mengubah data dalam table tsb melalui JDBC.

Anda akan menjadikan CatalogueDAO dapat berinteraksi dengan obyek lain yang berada dalam JVM yang terpisah dengan menggunakan teknik RMI. Untuk ini Anda akan mengembangkan class-class yang dibutuhkan untuk membungkus CatalogueDAO menjadi sebuah obyek yang dapat dijalankan di sisi server. Class-class ini adalah CatalogueRemote dan CatalogueRemoteImpl. Selanjutnya Anda menulis CatalogueRMIServerApp sebagai aplikasi stand alone yang bertindak sebagai server untuk CatalogueRemote.

Di sisi client, Anda akan menulis CatalogueRMITestClientApp untuk berinteraksi dengan obyek CatalogueRemote.

Pembekalan

?

Persiapan

- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Buatlah sebuah directory untuk latihan Anda, misalnya /home/lab/kinabalu dan dibawah directory ini buatlah beberapa sub-directory :
 - net/developerforce/amazon/catalogue/bean
 - net/developerforce/amazon/catalogue/dao
 - net/developerforce/amazon/catalogue/rmi
- Anda memerlukan mySQL server yang berjalan
- Di mySQL server Anda persiapkan sebuah database yaitu AMAZONDB, Di dalam database AMAZONDB persiapkan sebuah table yaitu CATALOGUETBL.

```
mysql> desc CATALOGUETBL;
```

Field	Type	Null	Key	Default	Extra
ISBN	varchar(20)		PRI		
TITLE	varchar(255)	YES		NULL	
AUTHOR	varchar(255)	YES		NULL	
PUBLISHER	varchar(50)	YES		NULL	
CITY	varchar(30)	YES		NULL	

```
• | YEAR | varchar(4) | YES | | NULL | |
• | LASTUPDATEDON | varchar(8) | YES | | NULL | |
• +-----+-----+-----+-----+-----+-----+
• 7 rows in set (0.00 sec)
```

Mengembangkan Class Pembantu : CatalogueBean

Langkah 1 : Dengan text editor tulislah CatalogueBean.java

```
package net.developerforce.amazon.catalogue.bean;
```

```
import java.io.Serializable;
import java.util.Date;

public class CatalogueBean
    implements Serializable
{
    public CatalogueBean(String isbn)
    {
        this.isbn = isbn;
    }

    private String isbn;
    public void setIsbn(String isbn)
    {
        this.isbn = isbn;
    }

    public String getIsbn()
    {
        return this.isbn;
    }

    private String title;
    public void setTitle(String title)
    {
        this.title = title;
    }

    public String getTitle()
    {
        return this.title;
    }

    private String author;
    public void setAuthor(String author)
    {
        this.author = author;
    }

    public String getAuthor()
    {
        return this.author;
    }

    private String publisher;
    public void setPublisher(String publisher)
```

```
{
    this.publisher = publisher;
}

public String getPublisher()
{
    return this.publisher;
}

private String city;
public void setCity(String city)
{
    this.city = city;
}

public String getCity()
{
    return this.city;
}

private String year;
public void setYear(String year)
{
    this.year = year;
}

public String getYear()
{
    return this.year;
}

private Date lastUpdatedOn;
public void setLastUpdatedOn(Date lastUpdatedOn)
{
    this.lastUpdatedOn = lastUpdatedOn;
}

public Date getLastUpdatedOn()
{
    return this.lastUpdatedOn;
}
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai CatalogueBean.java di sub-directory net/developerforce/amazon/catalogue/bean

Mengembangkan Class Utama : CatalogueDAO

Langkah 2 : Dengan text editor tulislah CatalogueDAO.java

```
package net.developerforce.amazon.catalogue.dao;

import java.sql.*;
import java.text.*;
import java.util.Date;
import net.developerforce.amazon.catalogue.bean.CatalogueBean;
```

```
public class CatalogueDAO
{
    private Connection conn;
    public CatalogueDAO(Connection conn)
        throws NullPointerException
    {
        if(conn==null)
        {
            throw new NullPointerException();
        }

        this.conn = conn;
    }

    public CatalogueBean selectCatalogueBean(String isbn)
        throws SQLException
    {
        CatalogueBean bean = null;

        String sqlSelect =
            "SELECT * FROM CATALOGUETBL "
            + "WHERE ISBN = '" + isbn + "'";
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery(sqlSelect);

        if(rs.next())
        {
            bean = new CatalogueBean(isbn);
            bean.setTitle(rs.getString("TITLE"));
            bean.setAuthor(rs.getString("AUTHOR"));
            bean.setPublisher(rs.getString("PUBLISHER"));
            bean.setCity(rs.getString("CITY"));
            bean.setYear(rs.getString("YEAR"));

            String lastUpdatedOn = rs.getString("LASTUPDATEDON");
            SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");
            try
            {
                Date date = format.parse(lastUpdatedOn);
                bean.setLastUpdatedOn(date);
            }
            catch(ParseException pe)
            {
                pe.printStackTrace();
            }
        }
        return bean;
    }

    public int insertCatalogueBean(CatalogueBean bean)
        throws SQLException
    {
        String lastUpdatedOn = null;
        Date date = bean.getLastUpdatedOn();
        if(date != null)
        {

```

```
        SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");
        lastUpdatedOn = format.format(date);
    }

    Statement stmt = conn.createStatement();
    String sqlInsert =
        "INSERT INTO CATALOGUETBL "
        + "VALUES ("
        + "'" + bean.getIsbn() + "',"
        + "'" + bean.getTitle() + "',"
        + "'" + bean.getAuthor() + "',"
        + "'" + bean.getPublisher() + "',"
        + "'" + bean.getCity() + "',"
        + "'" + bean.getYear() + "',"
        + "'" + lastUpdatedOn + "'"
        + ")"
        ;
    return stmt.executeUpdate(sqlInsert);
}

public int updateCatalogueBean(CatalogueBean bean)
    throws SQLException
{
    String lastUpdatedOn = null;
    Date date = bean.getLastUpdatedOn();
    if(date != null)
    {
        SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");
        lastUpdatedOn = format.format(date);
    }

    String sqlUpdate =
        "UPDATE CATALOGUETBL "
        + "SET "
        + "TITLE = '" + bean.getTitle() + "',"
        + "AUTHOR = '" + bean.getAuthor() + "',"
        + "PUBLISHER = '" + bean.getPublisher() + "',"
        + "CITY = '" + bean.getCity() + "',"
        + "YEAR = '" + bean.getYear() + "',"
        + "LASTUPDATEDON = '" + lastUpdatedOn + "'"
        + "WHERE "
        + "ISBN = '" + bean.getIsbn() + "'";

    Statement stmt = conn.createStatement();
    return stmt.executeUpdate(sqlUpdate);
}

public int deleteCatalogueBean(String isbn)
    throws SQLException
{
    String sqlDelete =
        "DELETE FROM CATALOGUETBL "
        + "WHERE "
        + "ISBN = '" + isbn + "'";
    Statement stmt = conn.createStatement();
    return stmt.executeUpdate(sqlDelete);
}
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai CatalogueDAO.java di sub-directory net/developerforce/amazon/catalogue/dao

Mengembangkan Server

Langkah 3 : Dengan text editor tulislah CatalogueRemote.java

```
package net.developerforce.amazon.catalogue.rmi;

import java.rmi.*;
import java.sql.*;

import net.developerforce.amazon.catalogue.bean.CatalogueBean;

public interface CatalogueRemote
    extends Remote
{
    public CatalogueBean findCatalogueBean(String isbn)
        throws RemoteException, SQLException;

    public int createCatalogueBean(CatalogueBean bean)
        throws RemoteException, SQLException;

    public int modifyCatalogueBean(CatalogueBean bean)
        throws RemoteException, SQLException;

    public int removeCatalogueBean(String isbn)
        throws RemoteException, SQLException;
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai CatalogueRemote.java di sub-directory net/developerforce/amazon/catalogue/rmi

Langkah 4 : Dengan text editor tulislah CatalogueRemoteImpl.java

```
package net.developerforce.amazon.catalogue.rmi;

import java.rmi.*;
import java.rmi.server.*;
import java.sql.*;

import net.developerforce.amazon.catalogue.bean.CatalogueBean;
import net.developerforce.amazon.catalogue.dao.CatalogueDAO;

public class CatalogueRemoteImpl
    extends UnicastRemoteObject
    implements CatalogueRemote
{
    public CatalogueRemoteImpl()
        throws RemoteException
    {
    }

    public Connection getConnection()
        throws ClassNotFoundException, SQLException
    {
    }
}
```

```
String jdbcDriver
    = "org.gjt.mm.mysql.Driver";
Class.forName(jdbcDriver);

String url = "jdbc:mysql://localhost:3306/AMAZONDB";
String user = "ekobs";
String pwd = "j2ee";

Connection conn
    = DriverManager.getConnection(
        url, user, pwd);
return conn;
}

public CatalogueBean findCatalogueBean(String isbn)
    throws RemoteException
{
    Connection conn = null;
    try
    {
        conn = getConnection();
        CatalogueDAO dao = new CatalogueDAO(conn);
        return dao.selectCatalogueBean(isbn);
    }
    catch(Exception e)
    {
        throw new RemoteException(e.toString());
    }
    finally
    {
        try
        {
            if(conn != null) conn.close();
        }
        catch(SQLException sqle)
        {
            sqle.printStackTrace();
        }
    }
}

public int createCatalogueBean(CatalogueBean bean)
    throws RemoteException
{
    Connection conn = null;
    try
    {
        conn = getConnection();
        CatalogueDAO dao = new CatalogueDAO(conn);
        return dao.insertCatalogueBean(bean);
    }
    catch(Exception e)
    {
        throw new RemoteException(e.toString());
    }
    finally
    {
    }
}
```



```
        {
            try
            {
                if(conn != null) conn.close();
            }
            catch(SQLException sqle)
            {
                sqle.printStackTrace();
            }
        }
    }

    public int modifyCatalogueBean(CatalogueBean bean)
        throws RemoteException
    {
        Connection conn = null;
        try
        {
            conn = getConnection();
            CatalogueDAO dao = new CatalogueDAO(conn);
            return dao.updateCatalogueBean(bean);
        }
        catch(Exception e)
        {
            throw new RemoteException(e.toString());
        }
        finally
        {
            try
            {
                if(conn != null) conn.close();
            }
            catch(SQLException sqle)
            {
                sqle.printStackTrace();
            }
        }
    }

    public int removeCatalogueBean(String isbn)
        throws RemoteException
    {
        Connection conn = null;
        try
        {
            conn = getConnection();
            CatalogueDAO dao = new CatalogueDAO(conn);
            return dao.deleteCatalogueBean(isbn);
        }
        catch(Exception e)
        {
            throw new RemoteException(e.toString());
        }
        finally
        {
            try
            {

```

```
        if(conn != null) conn.close();
    }
    catch(SQLException sqle)
    {
        sqle.printStackTrace();
    }
}
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai CatalogueDAORemoteImpl.java di sub-directory net/developerforce/amazon/catalogue/rmi

Langkah 5 : Dengan text editor tulislah CatalogueRMIServerApp.java

```
package net.developerforce.amazon.catalogue.app;

import java.net.*;
import java.rmi.*;
import java.sql.*;

import net.developerforce.amazon.catalogue.rmi.*;

public class CatalogueRMIServerApp
{
    public static void main(String args[])
    {
        System.setSecurityManager(
            new RMISecurityManager());
        try
        {
            CatalogueRemote remote = new CatalogueRemoteImpl();
            Naming.rebind("Catalogue", remote);
        }

        catch(RemoteException re)
        {
            re.printStackTrace();
        }
        catch(MalformedURLException murle)
        {
            murle.printStackTrace();
        }
        catch(NullPointerException npe)
        {
            npe.printStackTrace();
        }
    }
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai CatalogueRMIServerApp.java di sub-directory net/developerforce/amazon/catalogue/app

Mengembangkan Client

Langkah 6 : Dengan text editor tulislah CatalogueRMITestClientApp.java

```
package net.developerforce.amazon.catalogue.app;

import java.net.*;
import java.rmi.*;
import java.sql.*;
import java.util.Date;

import net.developerforce.amazon.catalogue.bean.CatalogueBean;
import net.developerforce.amazon.catalogue.rmi.CatalogueRemote;

public class CatalogueRMITestClientApp
{
    public static void main(String[] args)
    {
        System.setSecurityManager(new RMISecurityManager());

        try
        {
            CatalogueRMITestClientApp app = new
CatalogueRMITestClientApp();
            app.create();
            app.view();
            app.modify();
            app.view();
            app.remove();
        }
        catch(MalformedURLException murle)
        {
            murle.printStackTrace();
        }
        catch(NotBoundException ne)
        {
            ne.printStackTrace();
        }
        catch(RemoteException re)
        {
            re.printStackTrace();
        }
        catch(SQLException sqle)
        {
            sqle.printStackTrace();
        }
    }

    private CatalogueRemote remote;

    public CatalogueRMITestClientApp()
        throws MalformedURLException, NotBoundException,
RemoteException
    {
        remote = (CatalogueRemote)Naming.lookup("Catalogue");
    }
}
```

```
}

public void create()
    throws RemoteException, SQLException
{
    System.out.println("Creating ... ");

    String isbn = "0-201-47967-2";
    CatalogueBean bean = new CatalogueBean(isbn);
    bean.setTitle("Customer-centered Growth :
5 Proven Strategies For Building Competitive Advantage");
    bean.setAuthor("Richard Whiteley");
    bean.setPublisher("Addison Wesley Publishing Company");
    bean.setCity("Massachusetts");
    bean.setYear("1996");
    bean.setLastUpdatedOn(new Date());

    remote.createCatalogueBean(bean);
}

public void modify()
    throws RemoteException, SQLException
{
    System.out.println("Modifying ... ");

    String isbn = "0-201-47967-2";
    CatalogueBean bean = remote.findCatalogueBean(isbn);
    bean.setCity("Reading, Massachusetts");

    remote.modifyCatalogueBean(bean);
}

public void remove()
    throws RemoteException, SQLException
{
    System.out.println("Removing ... ");

    String isbn = "0-201-47967-2";
    remote.removeCatalogueBean(isbn);
}

public void view()
    throws RemoteException, SQLException, NullPointerException
{
    System.out.println("Displaying ... ");

    String isbn = "0-201-47967-2";
    CatalogueBean bean = remote.findCatalogueBean(isbn);

    System.out.println("ISBN           : " + bean.getIsbn());
    System.out.println("Title           : " + bean.getTitle());
    System.out.println("Author          : " + bean.getAuthor());
    System.out.println("Publisher       : " + bean.getPublisher());
    System.out.println("City            : " + bean.getCity());
    System.out.println("Year            : " + bean.getYear());
    System.out.println("Last Updated On : " + bean.getLastUpdatedOn());
}
```

```
}
```

Penyiapan Lingkungan

Langkah 7 : Bukalah sebuah console

Untuk menghidupkan RMI Registry, bukalah sebuah console, dan jalankan rmiregistry.

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
$ rmiregistry
```

Langkah 8 : Menuliskan security policy

```
grant
{
    permission java.net.SocketPermission "*:1024-65535",
        "connect,accept";
    permission java.io.FilePermission
        "/home/lab/kinabalu/-", "read";
};
```

Simpanlah sebagai kinabalu.policy di directory yang dipersiapkan sebelumnya.

Meng-compile dan Meluncurkan Server

Langkah 9 : Bukalah sebuah console

Anda akan meng-compile dan meluncurkan aplikasi dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan PATH harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan PATH :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
Variabel lingkungan CLASSPATH harus memuat directory di mana file .class dari aplikasi Anda berada. Contoh untuk menge-set variabel lingkungan CLASSPATH :
$ export CLASSPATH=/home/lab/kinabalu
Dan Anda perlu memasukkan JDBC terkait, yaitu JDBC ke mySQL, ke dalam CLASSPATH :
$ export CLASSPATH=/home/lab/lib/mm.mysql-2.0.8/mm.mysql-2.0.8-bin.jar:$CLASSPATH
```

Langkah 10 : Meng-compile CatalogueDAORMIServerApp.java

Melalui console, jalankan javac :

```
$ javac -sourcepath /home/lab/kinabalu
net/developerforce/amazon/catalogue/app/CatalogueRMIServerApp.java
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file CatalogueBean.class, CatalogueDAO.class, CatalogueRemote.class, CatalogueRemoteImpl.class dan CatalogueRMIServerApp.class

Langkah 11 : Men-generate stub dan skeleton menggunakan rmic

Melalui console, jalankan rmic :

```
$ rmic -d . net.developerforce.amazon.catalogue.rmi.CatalogueRemoteImpl
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file CatalogueRemoteImpl_Skel.class dan CatalogueRemoteImpl_Stub.class di dalam directory /home/lab/kinabalu/net/developerforce/amazon/catalogue/rmi

Langkah 12 : Meluncurkan aplikasi CatalogueDAORMIServerApp

Untuk meluncurkan aplikasi Anda, melalui console, jalankan java :

```
$ java -Djava.rmi.server.codebase=file:/home/lab/kinabalu/
-Djava.rmi.server.hostname=localhost
-Djava.security.policy=kinabalu.policy
net.developerforce.amazon.catalogue.app.CatalogueRMIServerApp
```

Meng-compile dan Meluncurkan Client

Langkah 13 : Bukalah sebuah console lainnya

Anda akan meng-compile dan meluncurkan aplikasi dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan PATH harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan PATH :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
```

Variabel lingkungan CLASSPATH harus memuat directory di mana file .class dari aplikasi Anda berada. Contoh untuk menge-set variabel lingkungan CLASSPATH :

```
$ export CLASSPATH=/home/lab/kinabalu
```

Langkah 14 : Meng-compile CatalogueRMITestClientApp.java

Melalui console, jalankan javac :

```
$ javac -sourcepath /home/lab/kinabalu
net/developerforce/amazon/catalogue/app/
CatalogueRMITestClientApp.java
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file CatalogueRMITestClientApp.class.

Langkah 15 : Meluncurkan aplikasi CatalogueRMITestClientApp

Untuk meluncurkan aplikasi Anda, melalui console, jalankan java :

```
$ java -Djava.rmi.server.hostname=localhost
-Djava.security.policy=kinabalu.policy
net.developerforce.amazon.catalogue.app.CatalogueRMITestClientApp
```

Creating ...
Displaying ...
ISBN : 0-201-47967-2
Title : Customer-centered Growth :
5 Proven Strategies For Building Competitive Advantage
Author : Richard Whiteley

Publisher : Addison Wesley Publishing Company
City : Massachusetts
Year : 1996
Last Updated On : Sat Aug 17 00:00:00 SGT 2002
Modifying ...
Displaying ...
ISBN : 0-201-47967-2
Title : Customer-centered Growth :
5 Proven Strategies For Building Competitive Advantage
Author : Richard Whiteley
Publisher : Addison Wesley Publishing Company
City : Reading, Massachusetts
Year : 1996
Last Updated On : Sat Aug 17 00:00:00 SGT 2002
Removing ...

7. Latihan Stateless Session Bean dengan Salam Keadilan

Tujuan

Dalam bagian ini Anda akan mulai melakukan sentuhan dengan Enterprise Java Bean.

Rancangan

Sebuah komponen EJB akan di-deploy ke application server sebagai stateless session bean. Komponen ini akan menyediakan sebuah method yang jika dipanggil akan mengembalikan pesan "Salam Keadilan !"

Pembekalan

?

Persiapan

- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Anda perlu meng-install terlebih dahulu jBoss Application Server
- Buatlah sebuah directory untuk latihan Anda, misalnya /home/lab/kinabalu

Mengembangkan Server

Langkah 1 : Dengan text editor tulislah 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;
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai SalamKeadilanHome.java

Langkah 2 : Dengan text editor tulislah SalamKeadilan.java

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

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

Simpanlah di directory yang telah Anda persiapkan sebagai SalamKeadilan.java

Langkah 3 : Dengan text editor tulislah 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) {}
}
```

Simpanlah di directory yang telah Anda persiapkan sebagai SalamKeadilanBean.java

Meng-compile SalamKeadilan EJB

Langkah 6 : Bukalah sebuah console

Anda akan meng-compile dan meluncurkan aplikasi dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan PATH harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan PATH :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
```

Untuk dapat meng-compile dengan sukses, Anda CLASSPATH perlu memuat directory atau jar dimana EJB API berada. Contoh untuk menge-set variabel lingkungan CLASSPATH :

```
$ export CLASSPATH=/home/lab/jboss-3.0.0/client/jboss-j2ee.jar:/home/lab/kinaba
lu
```

Langkah 7 : Meng-compile EJB

Anda akan meng-compile dengan

```
$ javac SalamKeadilan*.java
```

Men-deploy

Langkah 8 : Persiapkan deployment descriptor

```
<?xml version="1.0" encoding="Cp1252"?>
<ejb-jar>
    <description>jBoss test application</description>
    <display-name>Test</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>
```

Simpan sebagai ejb-jar.xml di /home/lab/kinabalu/META-INF/

Langkah 9 : Gabungkan semua .class dan ejb-jar.xml sebagai sebuah jar

Yaitu dengan :

```
$ jar -cvf salamkeadilan.jar SalamKeadilanHome.class SalamKeadilan.class
SalamKeadilanBean.class META-INF/ejb-jar.xml
```

Log yang dapat dipantau :

added manifest

adding: SalamKeadilanHome.class(in = 263) (out= 183)(deflated 30%)

adding: SalamKeadilan.class(in = 229) (out= 176)(deflated 23%)

adding: SalamKeadilanBean.class(in = 634) (out= 323)(deflated 49%)

adding: META-INF/ejb-jar.xml(in = 527) (out= 225)(deflated 57%)

Jika Anda berminat mengetahui muatan salamkeadilan.jar :

```
$ jar tvf salamkeadilan.jar
```

Hasilnya akan seperti ini :

```
0 Fri Aug 02 05:31:30 SGT 2002 META-INF/
71 Fri Aug 02 05:31:30 SGT 2002 META-INF/MANIFEST.MF
263 Fri Aug 02 05:29:38 SGT 2002 SalamKeadilanHome.class
229 Fri Aug 02 05:29:38 SGT 2002 SalamKeadilan.class
634 Fri Aug 02 05:29:38 SGT 2002 SalamKeadilanBean.class
527 Thu Aug 01 23:07:06 SGT 2002 META-INF/ejb-jar.xml
```

Langkah 10 : Men-deploy ke jBoss

Deployment ke jBoss adalah sangat sederhana. Anda cukup meng-copy .jar ke directory deploy :

```
$ cp salamkeadilan.jar /home/lab/jboss-3.0.0/server/default/deploy/
```

Di console dimana jBoss dijalankan :

```
05:35:20,828 INFO [MainDeployer] Starting deployment of package:
file:/home/lab/jboss-3.0.0/server/default/deploy/salamkeadilan.jar
05:35:21,059 INFO [EjbModule] Creating
05:35:21,127 INFO [EjbModule] Deploying SalamKeadilan
05:35:21,576 INFO [EjbModule] Created
05:35:21,578 INFO [EjbModule] Starting
05:35:21,669 INFO [EjbModule] Started
05:35:21,670 INFO [MainDeployer] Successfully completed deployment of
package:
file:/home/lab/jboss-3.0.0/server/default/deploy/salamkeadilan.jar
```

Mengembangkan Client

Langkah 11 : Dengan text editor tulislah SalamKeadilanEJBClient.java

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

public class SalamKeadilanEJBClientApp
{
    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 salamKeadilan = home.create();

            String pesan = salamKeadilan.getPesan();

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

Simpanlah sebagai SalamKeadilanEJBClientApp.java.

Langkah 12 : Bukalah sebuah console untuk client

Anda akan meng-compile dan meluncurkan aplikasi client dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan PATH harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan PATH :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
Contoh untuk menge-set variabel lingkungan CLASSPATH :
$ export CLASSPATH=/home/lab/jboss-3.0.0/client/jboss-j2ee.jar:
/home/lab/kinabalu/salamkeadilan.jar:
/home/lab/jboss-3.0.0/client/jnp-client.jar:/home/lab/jboss-3.0.0/clie
```

```
nt/jnet.jar
:/home/lab/jboss-3.0.0/client/jboss-client.jar:
/home/lab/jboss-3.0.0/client/jbosssx-client.jar:
/home/lab/jboss-3.0.0/client/jboss-common-client.jar:
/home/lab/jboss-3.0.0/client/log4j.jar:.
```

Langkah 13 : Meng-compile SalamKeadilanEJBClientApp.java

Melalui console, jalankan javac :

```
$ javac SalamKeadilanEJBClientApp.java
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file SalamKeadilanEJBClientApp.class di dalam directory yang sama.

Langkah 14 : Meluncurkan aplikasi client

Untuk meluncurkan aplikasi Anda, melalui console, jalankan java :

```
$ java SalamKeadilanEJBClientApp
```

8. Berinteraksi dengan DB melalui Stateless SessionBean

Tujuan

Dalam bagian ini Anda akan mencoba menggunakan SLSB untuk berinteraksi dengan DB.

Rancangan

Pembekalan

?

Persiapan

- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Anda perlu meng-install terlebih dahulu jBoss Application Server
- Anda perlu menjalankan mySQL Server
- Buatlah sebuah directory untuk latihan Anda, misalnya /home/lab/kinabalu. Di bawah directory ini persiapkan directory sbb. :
 - net/developerforce/amazon/catalogue/app
 - net/developerforce/amazon/catalogue/bean
 - net/developerforce/amazon/catalogue/dao
 - net/developerforce/amazon/catalogue/slsb

Mengembangkan Class Pembantu

Langkah 1 : Dengan text editor tulislah CatalogueBean.java

```
package net.developerforce.amazon.catalogue.bean;
```

```
import java.io.Serializable;  
import java.util.Date;
```

```
public class CatalogueBean  
    implements Serializable  
{  
    public CatalogueBean(String isbn)  
    {  
        this.isbn = isbn;  
    }  
  
    private String isbn;  
    public void setIsbn(String isbn)  
    {  
        this.isbn = isbn;  
    }  
  
    public String getIsbn()  
    {  
        return this.isbn;  
    }  
  
    private String title;  
    public void setTitle(String title)  
    {
```

```
        this.title = title;
    }

    public String getTitle()
    {
        return this.title;
    }

    private String author;
    public void setAuthor(String author)
    {
        this.author = author;
    }

    public String getAuthor()
    {
        return this.author;
    }

    private String publisher;
    public void setPublisher(String publisher)
    {
        this.publisher = publisher;
    }

    public String getPublisher()
    {
        return this.publisher;
    }

    private String city;
    public void setCity(String city)
    {
        this.city = city;
    }

    public String getCity()
    {
        return this.city;
    }

    private String year;
    public void setYear(String year)
    {
        this.year = year;
    }

    public String getYear()
    {
        return this.year;
    }

    private Date lastUpdatedOn;
    public void setLastUpdatedOn(Date lastUpdatedOn)
    {
        this.lastUpdatedOn = lastUpdatedOn;
    }
}
```

```
    public Date getLastUpdatedOn()  
    {  
        return this.lastUpdatedOn;  
    }  
}
```

Simpanlah di sub directory net/developforce/amazon/catalogue/bean yang telah Anda persiapkan sebagai CatalogueBean.java

Langkah 2 : Dengan text editor tulislah CatalogueDAO.java

```
package net.developforce.amazon.catalogue.dao;  
  
import java.sql.*;  
import java.text.*;  
import java.util.Date;  
import net.developforce.amazon.catalogue.bean.CatalogueBean;  
  
public class CatalogueDAO  
{  
    private Connection conn;  
    public CatalogueDAO(Connection conn)  
        throws NullPointerException  
    {  
        if(conn==null)  
        {  
            throw new NullPointerException();  
        }  
  
        this.conn = conn;  
    }  
  
    public CatalogueBean selectCatalogueBean(String isbn)  
        throws SQLException  
    {  
        CatalogueBean bean = null;  
  
        String sqlSelect =  
            "SELECT * FROM CATALOGUETBL "  
            + "WHERE ISBN = '" + isbn + "'";  
        Statement stmt = conn.createStatement();  
        ResultSet rs = stmt.executeQuery(sqlSelect);  
  
        if(rs.next())  
        {  
            bean = new CatalogueBean(isbn);  
            bean.setTitle(rs.getString("TITLE"));  
            bean.setAuthor(rs.getString("AUTHOR"));  
            bean.setPublisher(rs.getString("PUBLISHER"));  
            bean.setCity(rs.getString("CITY"));  
            bean.setYear(rs.getString("YEAR"));  
  
            String lastUpdatedOn = rs.getString("LASTUPDATEDON");  
            SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");  
            try  
            {  
                Date date = format.parse(lastUpdatedOn);  
            }  
        }  
    }  
}
```

```
        bean.setLastUpdatedOn(date);
    }
    catch(ParseException pe)
    {
        pe.printStackTrace();
    }
}
return bean;
}

public int insertCatalogueBean(CatalogueBean bean)
    throws SQLException
{
    String lastUpdatedOn = null;
    Date date = bean.getLastUpdatedOn();
    if(date != null)
    {
        SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");
        lastUpdatedOn = format.format(date);
    }

    Statement stmt = conn.createStatement();
    String sqlInsert =
        "INSERT INTO CATALOGUETBL "
        + "VALUES ("
        + "'" + bean.getIsbn() + "',"
        + "'" + bean.getTitle() + "',"
        + "'" + bean.getAuthor() + "',"
        + "'" + bean.getPublisher() + "',"
        + "'" + bean.getCity() + "',"
        + "'" + bean.getYear() + "',"
        + "'" + lastUpdatedOn + "'"
        + ")"
        ;
    return stmt.executeUpdate(sqlInsert);
}

public int updateCatalogueBean(CatalogueBean bean)
    throws SQLException
{
    String lastUpdatedOn = null;
    Date date = bean.getLastUpdatedOn();
    if(date != null)
    {
        SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");
        lastUpdatedOn = format.format(date);
    }

    String sqlUpdate =
        "UPDATE CATALOGUETBL "
        + "SET "
        + "TITLE = '" + bean.getTitle() + "',"
        + "AUTHOR = '" + bean.getAuthor() + "',"
        + "PUBLISHER = '" + bean.getPublisher() + "',"
        + "CITY = '" + bean.getCity() + "',"
        + "YEAR = '" + bean.getYear() + "',"

```



```
        + "LASTUPDATEDON = '" + lastUpdatedOn + "'"
        + "WHERE "
        + "ISBN = '" + bean.getIsbn() + "'";

        Statement stmt = conn.createStatement();
        return stmt.executeUpdate(sqlUpdate);
    }
    public int deleteCatalogueBean(String isbn)
        throws SQLException
    {
        String sqlDelete =
            "DELETE FROM CATALOGUETBL "
            + "WHERE "
            + "ISBN = '" + isbn + "'";
        Statement stmt = conn.createStatement();
        return stmt.executeUpdate(sqlDelete);
    }
}
```

Simpanlah di sub directory net/developerforce/amazon/catalogue/dao yang telah Anda persiapkan sebagai CatalogueDAO.java

Mengembangkan SessionBean

Langkah 3 : Dengan text editor tulislah CatalogueSessionHome.java

```
package net.developerforce.amazon.catalogue.slsb;

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

public interface CatalogueSessionHome
    extends EJBHome
{
    public CatalogueSession create()
        throws RemoteException, CreateException;
}
```

Simpanlah di sub directory net/developerforce/amazon/catalogue/slsb yang telah Anda persiapkan sebagai CatalogueSessionHome.java

Langkah 4 : Dengan text editor tulislah CatalogueSession.java

```
package net.developerforce.amazon.catalogue.slsb;

import java.rmi.*;
import java.sql.*;

import javax.ejb.*;

import net.developerforce.amazon.catalogue.bean.CatalogueBean;

public interface CatalogueSession
    extends EJBObject
{
}
```

```
public CatalogueBean findCatalogueBean(String isbn)
    throws RemoteException;

public int createCatalogueBean(CatalogueBean bean)
    throws RemoteException;

public int modifyCatalogueBean(CatalogueBean bean)
    throws RemoteException;

public int removeCatalogueBean(String isbn)
    throws RemoteException;

}
```

Simpanlah di sub directory net/developforce/amazon/catalogue/slsb yang telah Anda persiapkan sebagai CatalogueSession.java

Langkah 5 : Dengan text editor tulislah CatalogueSessionBean.java

```
package net.developforce.amazon.catalogue.slsb;

import javax.ejb.*;
import java.rmi.*;
import java.sql.*;
import java.util.*;
import javax.sql.*;
import javax.naming.*;

import net.developforce.amazon.catalogue.bean.CatalogueBean;
import net.developforce.amazon.catalogue.dao.CatalogueDAO;

public class CatalogueSessionBean
    implements SessionBean
{
    public CatalogueBean findCatalogueBean(String isbn)
        throws RemoteException
    {
        CatalogueBean bean = null;
        Connection conn = null;
        try
        {
            conn = getConnection();
            CatalogueDAO dao = new CatalogueDAO(conn);
            return dao.selectCatalogueBean(isbn);
        }
        catch(Exception e)
        {
            throw new RemoteException(e.toString());
        }
        finally
        {
            try
            {
                if(conn!=null)conn.close();
            }
            catch(SQLException sqle)
            {
                sqle.printStackTrace();
            }
        }
    }
}
```

```
    }  
  }  
}  
  
public int createCatalogueBean(CatalogueBean bean)  
    throws RemoteException  
{  
    Connection conn = null;  
    try  
    {  
        conn = getConnection();  
        CatalogueDAO dao = new CatalogueDAO(conn);  
        return dao.insertCatalogueBean(bean);  
    }  
    catch(Exception e)  
    {  
        throw new RemoteException(e.toString());  
    }  
    finally  
    {  
        try  
        {  
            if(conn!=null)conn.close();  
        }  
        catch(SQLException sqle)  
        {  
            sqle.printStackTrace();  
        }  
    }  
}  
  
public int modifyCatalogueBean(CatalogueBean bean)  
    throws RemoteException  
{  
    Connection conn = null;  
    try  
    {  
        conn = getConnection();  
        CatalogueDAO dao = new CatalogueDAO(conn);  
        return dao.updateCatalogueBean(bean);  
    }  
    catch(Exception e)  
    {  
        throw new RemoteException(e.toString());  
    }  
    finally  
    {  
        try  
        {  
            if(conn!=null)conn.close();  
        }  
        catch(SQLException sqle)  
        {  
            sqle.printStackTrace();  
        }  
    }  
}
```

```
public int removeCatalogueBean(String isbn)
    throws RemoteException
{
    Connection conn = null;
    try
    {
        conn = getConnection();
        CatalogueDAO dao = new CatalogueDAO(conn);
        return dao.deleteCatalogueBean(isbn);
    }
    catch(Exception e)
    {
        throw new RemoteException(e.toString());
    }
    finally
    {
        try
        {
            if(conn!=null)conn.close();
        }
        catch(SQLException sqle)
        {
            sqle.printStackTrace();
        }
    }
}

public Connection getConnection()
    throws NamingException, SQLException
{
    Connection conn          = dataSource.getConnection();
    return conn;
}

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

private transient SessionContext sessionContext;
private transient DataSource dataSource;
public void setSessionContext(SessionContext sessionContext)
{
    this.sessionContext = sessionContext;
    try
    {
        Context context = new InitialContext();
        dataSource      =
(DataSource)context.lookup("java:/MySqlDS");
    }
    catch (NamingException e)
    {
        e.printStackTrace();
    }
}
}
```

Simpanlah di sub directory net/developerforce/amazon/catalogue/slsb yang telah Anda persiapkan sebagai CatalogueSessionBean.java

Meng-compile Catalogue SessionBean

Langkah 6 : Bukalah sebuah console

Anda akan meng-compile dan meluncurkan aplikasi dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan PATH harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan PATH :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
Untuk dapat meng-compile dengan sukses, Anda CLASSPATH perlu memuat directory atau jar
dimana EJB API berada. Contoh untuk menge-set variabel lingkungan CLASSPATH :
$ export
CLASSPATH=/home/lab/jboss-3.0.0/client/jboss-j2ee.jar:/home/lab/kinabalu
```

Langkah 7 : Meng-compile EJB

Anda akan meng-compile dengan

```
$ javac -sourcepath /home/lab/kinabalu/
net/developerforce/amazon/catalogue/slsb/*.java
```

Men-deploy

Langkah 8 : Persiapkan deployment descriptor

```
<?xml version="1.0" encoding="Cp1252"?>
<ejb-jar>
  <description>jBoss test application</description>
  <display-name>Test</display-name>
  <enterprise-beans>
    <session>
      <ejb-name>CatalogueSession</ejb-name>

<home>net.developerforce.amazon.catalogue.slsb.CatalogueSession</home>

<remote>net.developerforce.amazon.catalogue.slsb.CatalogueSession</remote>

<ejb-class>net.developerforce.amazon.catalogue.slsb.CatalogueSession</ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Bean</transaction-type>
  </session>
</enterprise-beans>
</ejb-jar>
```

Simpan sebagai ejb-jar.xml di /home/lab/kinabalu/META-INF/

Langkah 9 : Gabungkan semua .class dan ejb-jar.xml sebagai sebuah jar

Yaitu dengan :

```
$ jar cvf catalogueslsb.jar
net/developerforce/amazon/catalogue/slsb/*.class
added manifest
adding:
net/developerforce/amazon/catalogue/slsb/CatalogueSessionBean.class
(in = 1897) (out= 838)(deflated 55%)
adding: net/developerforce/amazon/catalogue/slsb/CatalogueSession.class
(in = 582) (out= 285)(deflated 51%)
adding:
net/developerforce/amazon/catalogue/slsb/CatalogueSessionHome.class
(in = 354) (out= 221)(deflated 37%)

$ jar uvf catalogueslsb.jar
net/developerforce/amazon/catalogue/bean/*.class
adding: net/developerforce/amazon/catalogue/bean/CatalogueBean.class
(in = 1415) (out= 598)(deflated 57%)

$ jar uvf catalogueslsb.jar
net/developerforce/amazon/catalogue/dao/*.class
adding: net/developerforce/amazon/catalogue/dao/CatalogueDAO.class
(in = 3146) (out= 1535)(deflated 51%)

$ jar uvf catalogueslsb.jar META-INF/ejb-jar.xml
adding: META-INF/ejb-jar.xml(in = 632) (out= 254)(deflated 59%)
```

Jika Anda berminat mengetahui muatan catalogueslsb.jar :

```
$ jar tvf catalogueslsb.jar
```

Hasilnya akan seperti ini :

```
0 Fri Aug 16 21:00:38 SGT 2002 META-INF/
71 Fri Aug 16 21:00:38 SGT 2002 META-INF/MANIFEST.MF
1897 Fri Aug 16 20:53:12 SGT 2002 net/developerforce/amazon/
catalogue/slsb/CatalogueSessionBean.class
582 Fri Aug 16 20:53:12 SGT 2002 net/developerforce/amazon/
catalogue/slsb/CatalogueSession.class
354 Fri Aug 16 20:53:12 SGT 2002 net/developerforce/amazon/
catalogue/slsb/CatalogueSessionHome.class
1415 Fri Aug 16 06:25:36 SGT 2002 net/developerforce/amazon/
catalogue/bean/CatalogueBean.class
3146 Fri Aug 16 05:11:50 SGT 2002 net/developerforce/amazon/
catalogue/dao/CatalogueDAO.class
632 Fri Aug 16 05:16:20 SGT 2002 META-INF/ejb-jar.xml
```

Langkah 10 : Jalankan jBoss

```
$ export JAVA_HOME=/home/lab/jdk1.3.1_01
$ sh /home/lab/jboss-3.0.0/bin/run.sh
=====
=====
```

JBoss Bootstrap Environment

JBoss_HOME: /home/lab/jboss-3.0.0

JAVA: /home/lab/jdk1.3.1_01/bin/java

JAVA_OPTS: -server -Dprogram.name=run.sh

CLASSPATH:

/home/lab/jboss-3.0.0/bin/run.jar:/home/lab/jdk1.3.1_01/lib/tools.jar

=====
=====

Langkah 11 : Persiapkan DataSource untuk mySQL

Anda dapat menggunakan template yang disediakan di JBoss_HOME/docs/examples/jca, yaitu mysql-service.xml. Yang perlu Anda lakukan adalah mengubah ConnectionURL, UserName dan Password, sehingga sesuai dengan lingkungan komputasi Anda.

```
<?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">
      <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/AMAZONDB</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>
  </mbean>
</server>
```

```
<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>

<depends optional-attribute-name="JaasSecurityManagerService">
jboss.security:name=JaasSecurityManager</depends>

<attribute
name="TransactionManager">java:/TransactionManager</attribute>
<depends>jboss.jca:service=RARDeployer</depends>

</mbean>

</server>
```

Simpan file ini di directory Anda, misalnya /home/lab/kinabalu

Langkah 12 : Men-deploy mySQL datasource ke jBoss

Deployment ke jBoss adalah sangat sederhana. Anda cukup meng-copy .jar ke directory deploy :

```
$ cp mysql-service.xml /home/lab/jboss-3.0.0/server/default/deploy/
Di console dimana jBoss dijalankan :
21:22:21,680 INFO [MainDeployer] Starting deployment of package:
  file:/home/lab/jboss-3.0.0/server/default/deploy/mysql-service.xml
21:22:22,335 WARN [ServiceController] jboss.jca:service=LocalTxDS,
name=MySqlDS does not implement any Service methods
21:22:22,337 INFO [JBossManagedConnectionPool] Creating
21:22:22,338 INFO [JBossManagedConnectionPool] Created
21:22:22,339 INFO [LocalTxConnectionManager] Creating
21:22:22,349 INFO [LocalTxConnectionManager] Created
21:22:22,351 INFO [JBossManagedConnectionPool] Starting
21:22:22,351 INFO [JBossManagedConnectionPool] Started
21:22:22,352 INFO [LocalTxConnectionManager] Starting
21:22:23,036 INFO [MySqlDS] Bound connection factory for resource adapter
'JBoss LocalTransaction JDBC Wrapper' to JNDI name 'java:/MySqlDS'
21:22:23,037 INFO [LocalTxConnectionManager] Started
21:22:23,038 INFO [MainDeployer] Successfully completed deployment of
package:
  file:/home/lab/jboss-3.0.0/server/default/deploy/mysql-service.xml
```


Langkah 13 : Men-deploy ke jBoss

Deployment ke jBoss adalah sangat sederhana. Anda cukup meng-copy .jar ke directory deploy :

```
$ cp catalogueslsb.jar /home/lab/jboss-3.0.0/server/default/deploy/  
Di console dimana jBoss dijalankan :  
21:26:39,995 INFO [EjbModule] Creating  
21:26:40,025 INFO [EjbModule] Deploying CatalogueSession  
21:26:40,101 INFO [EjbModule] Created  
21:26:40,102 INFO [EjbModule] Starting  
21:26:40,169 INFO [EjbModule] Started  
21:26:40,170 INFO [MainDeployer] Successfully completed deployment of  
package:  
file:/home/lab/jboss-3.0.0/server/default/deploy/catalogueslsb.jar
```

Mengembangkan Client

Langkah 14 : Dengan text editor tulislah CatalogueSessionBeanTestClientApp.java

```
package net.developerforce.amazon.catalogue.app;  
  
import java.rmi.*;  
import java.sql.SQLException;  
import java.util.Date;  
import java.util.Properties;  
  
import javax.ejb.*;  
import javax.naming.*;  
import javax.rmi.PortableRemoteObject;  
  
import net.developerforce.amazon.catalogue.bean.CatalogueBean;  
import net.developerforce.amazon.catalogue.slsb.*;  
  
public class CatalogueSessionBeanTestClientApp  
{  
    public static void main(String[] args)  
    {  
        try  
        {  
            CatalogueSessionBeanTestClientApp app  
                = new CatalogueSessionBeanTestClientApp();  
  
            app.create();  
            app.display();  
            app.modify();  
            app.display();  
            app.remove();  
        }  
        catch(CreateException ce)  
        {  
            ce.printStackTrace();  
        }  
        catch(NamingException ne)  
        {  

```

```
        ne.printStackTrace();
    }
    catch(RemoteException re)
    {
        re.printStackTrace();
    }
}

private CatalogueSession session;

public CatalogueSessionBeanTestClientApp()
    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("CatalogueSession");
    CatalogueSessionHome home = (CatalogueSessionHome)
        PortableRemoteObject.narrow(ref,
CatalogueSessionHome.class);
    session = home.create();
}

public void create()
    throws RemoteException
{
    System.out.println("Creating ... ");

    String isbn = "0-201-47967-2";
    CatalogueBean bean = new CatalogueBean(isbn);
    bean.setTitle("Customer-centered Growth :
5 Proven Strategies For Building Competitive Advantage");
    bean.setAuthor("Richard Whiteley");
    bean.setPublisher("Addison Wesley Publishing Company");
    bean.setCity("Massachusetts");
    bean.setYear("1996");
    bean.setLastUpdatedOn(new Date());

    session.createCatalogueBean(bean);
}

public void modify()
    throws RemoteException
{
    System.out.println("Modifying ... ");

    String isbn = "0-201-47967-2";
    CatalogueBean bean = session.findCatalogueBean(isbn);
    bean.setCity("Reading, Massachusetts");
}
```

```
        session.modifyCatalogueBean(bean);
    }

    public void remove()
        throws RemoteException
    {
        System.out.println("Removing ... ");

        String isbn = "0-201-47967-2";
        session.removeCatalogueBean(isbn);
    }

    public void display()
        throws RemoteException
    {
        System.out.println("Displaying ... ");

        String isbn = "0-201-47967-2";
        CatalogueBean bean = session.findCatalogueBean(isbn);

        System.out.println("ISBN           : " + bean.getIsbn());
        System.out.println("Title           : " + bean.getTitle());
        System.out.println("Author          : " + bean.getAuthor());
        System.out.println("Publisher       : " + bean.getPublisher());
        System.out.println("City            : " + bean.getCity());
        System.out.println("Year            : " + bean.getYear());
        System.out.println("Last Updated On : " + bean.getLastUpdatedOn());
    }
}
```

Simpanlah di sub directory `net/developerforce/amazon/catalogue/app` yang telah Anda persiapkan sebagai `CatalogueSessionBeanTestClientApp.java`.

Langkah 15 : Bukalah sebuah console untuk client

Anda akan meng-compile dan meluncurkan aplikasi client dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan `PATH` harus memuat directory dimana perintah `java` dan `javac`. Contoh untuk menge-set variabel lingkungan `PATH` :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
Contoh untuk menge-set variabel lingkungan CLASSPATH :
$ export CLASSPATH=/home/lab/jboss-3.0.0/client/jboss-j2ee.jar:
/home/lab/kinabalu/catalogueslsb.jar:
/home/lab/jboss-3.0.0/client/jnp-client.jar:
/home/lab/jboss-3.0.0/client/jnet.jar:
/home/lab/jboss-3.0.0/client/jboss-client.jar:
/home/lab/jboss-3.0.0/client/jbosssx-client.jar:
/home/lab/jboss-3.0.0/client/jboss-common-client.jar:
/home/lab/jboss-3.0.0/client/log4j.jar:.
```

Langkah 16 : Meng-compile CatalogueSessionBeanTestClientApp.java

Melalui console, jalankan javac :

```
$ javac  
net/developerforce/amazon/catalogue/app/CatalogueSessionBeanTestClient  
App.java
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file CatalogueSessionBeanTestClientApp.class di dalam directory yang sama.

Langkah 17 : Meluncurkan aplikasi client

Untuk meluncurkan aplikasi Anda, melalui console, jalankan java :

```
$ java  
net.developerforce.amazon.catalogue.app.CatalogueSessionBeanTestClient  
App  
Creating ...  
Displaying ...  
ISBN : 0-201-47967-2  
Title : Customer-centered Growth :  
5 Proven Strategies For Building Competitive Advantage  
Author : Richard Whiteley  
Publisher : Addison Wesley Publishing Company  
City : Massachusetts  
Year : 1996  
Last Updated On : Sat Aug 17 00:00:00 SGT 2002  
Modifying ...  
Displaying ...  
ISBN : 0-201-47967-2  
Title : Customer-centered Growth :  
5 Proven Strategies For Building Competitive Advantage  
Author : Richard Whiteley  
Publisher : Addison Wesley Publishing Company  
City : Reading, Massachusetts  
Year : 1996  
Last Updated On : Sat Aug 17 00:00:00 SGT 2002  
Removing ...
```

9. Berinteraksi dengan DB melalui BMP EntityBean

Tujuan

Dalam bagian ini Anda akan mencoba menggunakan Bean Managed Persistence Entity Bean untuk berinteraksi dengan DB.

Rancangan

Pembekalan

?

Persiapan

- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Anda perlu meng-install terlebih dahulu jBoss Application Server
- Anda perlu menjalankan mySQL Server
- Buatlah sebuah directory untuk latihan Anda, misalnya /home/lab/kinabalu. Di bawah directory ini persiapkan directory sbb. :
 - net/developerforce/amazon/catalogue/app
 - net/developerforce/amazon/catalogue/bean
 - net/developerforce/amazon/catalogue/bmp
 - net/developerforce/amazon/catalogue/dao

Mengembangkan Class Pembantu

Langkah 1 : Dengan text editor tulislah CatalogueBean.java

```
package net.developerforce.amazon.catalogue.bean;
```

```
import java.io.Serializable;  
import java.util.Date;
```

```
public class CatalogueBean  
    implements Serializable  
{  
    public CatalogueBean(String isbn)  
    {  
        this.isbn = isbn;  
    }  
  
    private String isbn;  
    public void setIsbn(String isbn)  
    {  
        this.isbn = isbn;  
    }  
  
    public String getIsbn()  
    {  
        return this.isbn;  
    }  
  
    private String title;  
    public void setTitle(String title)  
    {
```

```
        this.title = title;
    }

    public String getTitle()
    {
        return this.title;
    }

    private String author;
    public void setAuthor(String author)
    {
        this.author = author;
    }

    public String getAuthor()
    {
        return this.author;
    }

    private String publisher;
    public void setPublisher(String publisher)
    {
        this.publisher = publisher;
    }

    public String getPublisher()
    {
        return this.publisher;
    }

    private String city;
    public void setCity(String city)
    {
        this.city = city;
    }

    public String getCity()
    {
        return this.city;
    }

    private String year;
    public void setYear(String year)
    {
        this.year = year;
    }

    public String getYear()
    {
        return this.year;
    }

    private Date lastUpdatedOn;
    public void setLastUpdatedOn(Date lastUpdatedOn)
    {
        this.lastUpdatedOn = lastUpdatedOn;
    }
}
```

```
        public Date getLastUpdatedOn()  
        {  
            return this.lastUpdatedOn;  
        }  
    }  
}
```

Simpanlah di sub directory net/developforce/amazon/catalogue/bean yang telah Anda persiapkan sebagai CatalogueBean.java

Langkah 2 : Dengan text editor tulislah CatalogueDAO.java

```
package net.developforce.amazon.catalogue.dao;  
  
import java.sql.*;  
import java.text.*;  
import java.util.Date;  
import net.developforce.amazon.catalogue.bean.CatalogueBean;  
  
public class CatalogueDAO  
{  
    private Connection conn;  
    public CatalogueDAO(Connection conn)  
        throws NullPointerException  
    {  
        if(conn==null)  
        {  
            throw new NullPointerException();  
        }  
  
        this.conn = conn;  
    }  
  
    public CatalogueBean selectCatalogueBean(String isbn)  
        throws SQLException  
    {  
        CatalogueBean bean = null;  
  
        String sqlSelect =  
            "SELECT * FROM CATALOGUETBL "  
            + "WHERE ISBN = '" + isbn + "'";  
        Statement stmt = conn.createStatement();  
        ResultSet rs = stmt.executeQuery(sqlSelect);  
  
        if(rs.next())  
        {  
            bean = new CatalogueBean(isbn);  
            bean.setTitle(rs.getString("TITLE"));  
            bean.setAuthor(rs.getString("AUTHOR"));  
            bean.setPublisher(rs.getString("PUBLISHER"));  
            bean.setCity(rs.getString("CITY"));  
            bean.setYear(rs.getString("YEAR"));  
  
            String lastUpdatedOn = rs.getString("LASTUPDATEDON");  
            SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");  
            try  
            {  
                Date date = format.parse(lastUpdatedOn);  
            }  
        }  
    }  
}
```

```
        bean.setLastUpdatedOn(date);
    }
    catch(ParseException pe)
    {
        pe.printStackTrace();
    }
}
return bean;
}

public int insertCatalogueBean(CatalogueBean bean)
    throws SQLException
{
    String lastUpdatedOn = null;
    Date date = bean.getLastUpdatedOn();
    if(date != null)
    {
        SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");
        lastUpdatedOn = format.format(date);
    }

    Statement stmt = conn.createStatement();
    String sqlInsert =
        "INSERT INTO CATALOGUETBL "
        + "VALUES ("
        + "'" + bean.getIsbn() + "',"
        + "'" + bean.getTitle() + "',"
        + "'" + bean.getAuthor() + "',"
        + "'" + bean.getPublisher() + "',"
        + "'" + bean.getCity() + "',"
        + "'" + bean.getYear() + "',"
        + "'" + lastUpdatedOn + "'"
        + ")"
        ;
    return stmt.executeUpdate(sqlInsert);
}

public int updateCatalogueBean(CatalogueBean bean)
    throws SQLException
{
    String lastUpdatedOn = null;
    Date date = bean.getLastUpdatedOn();
    if(date != null)
    {
        SimpleDateFormat format = new SimpleDateFormat("yyyyMMdd");
        lastUpdatedOn = format.format(date);
    }

    String sqlUpdate =
        "UPDATE CATALOGUETBL "
        + "SET "
        + "TITLE = '" + bean.getTitle() + "',"
        + "AUTHOR = '" + bean.getAuthor() + "',"
        + "PUBLISHER = '" + bean.getPublisher() + "',"
        + "CITY = '" + bean.getCity() + "',"
        + "YEAR = '" + bean.getYear() + "',"
```



```
        + "LASTUPDATEDON = '" + lastUpdatedOn + "'"
        + "WHERE "
        + "ISBN = '" + bean.getIsbn() + "'";

        Statement stmt = conn.createStatement();
        return stmt.executeUpdate(sqlUpdate);
    }

    public int deleteCatalogueBean(String isbn)
        throws SQLException
    {
        String sqlDelete =
            "DELETE FROM CATALOGUETBL "
            + "WHERE "
            + "ISBN = '" + isbn + "'";
        Statement stmt = conn.createStatement();
        return stmt.executeUpdate(sqlDelete);
    }
}
```

Simpanlah di sub directory net/developerforce/amazon/catalogue/dao yang telah Anda persiapkan sebagai CatalogueDAO.java

Mengembangkan Server

Langkah 3 : Dengan text editor tulislah CatalogueEntityHome.java

```
package net.developerforce.amazon.catalogue.bmp;

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

import net.developerforce.amazon.catalogue.bean.CatalogueBean;

public interface CatalogueEntityHome
    extends EJBHome
{
    public CatalogueEntity create(CatalogueBean bean)
        throws RemoteException, CreateException;
    public CatalogueEntity findByPrimaryKey(CatalogueEntityPK pk)
        throws RemoteException, FinderException;
}
```

Simpanlah di sub directory net/developerforce/amazon/catalogue/bmp yang telah Anda persiapkan sebagai CatalogueEntityHome.java

Langkah 4 : Dengan text editor tulislah CatalogueEntity.java

```
package net.developerforce.amazon.catalogue.bmp;

import java.rmi.*;
import java.sql.*;

import javax.ejb.*;
```

```
import net.developerforce.amazon.catalogue.bean.CatalogueBean;

public interface CatalogueEntity
    extends EJBObject
{
    public void setCatalogueBean(CatalogueBean bean)
        throws RemoteException;
    public CatalogueBean getCatalogueBean()
        throws RemoteException;
}
```

Simpanlah di sub directory net/developerforce/amazon/catalogue/bmp yang telah Anda persiapkan sebagai CatalogueEntity.java

Langkah 5 : Dengan text editor tulislah CatalogueEntityBean.java

```
package net.developerforce.amazon.catalogue.bmp;

import javax.ejb.*;
import java.rmi.*;
import java.sql.*;
import java.util.*;
import javax.sql.*;
import javax.naming.*;

import net.developerforce.amazon.catalogue.bean.CatalogueBean;
import net.developerforce.amazon.catalogue.dao.CatalogueDAO;

public class CatalogueEntityBean
    implements EntityBean
{
    public CatalogueBean bean;

    public CatalogueEntityPK ejbCreate(CatalogueBean bean)
        throws CreateException
    {
        Connection conn = null;
        try
        {
            conn = getConnection();
            CatalogueDAO dao = new CatalogueDAO(conn);

            this.bean = bean;
            dao.insertCatalogueBean(bean);
            return new CatalogueEntityPK(bean.getIsbn());
        }
        catch(Exception e)
        {
            throw new CreateException(e.toString());
        }
        finally
        {
            try
            {
                if(conn!=null)conn.close();
            }
            catch(SQLException sqle)
            {
            }
        }
    }
}
```

```
        sqle.printStackTrace();
    }
}

public void ejbPostCreate(CatalogueBean bean)
    throws CreateException
{
}

public CatalogueEntityPK ejbFindByPrimaryKey(CatalogueEntityPK pk)
    throws FinderException
{
    Connection conn = null;
    try
    {
        conn = getConnection();
        CatalogueDAO dao = new CatalogueDAO(conn);

        bean = dao.selectCatalogueBean(pk.getIsbn());
        return new CatalogueEntityPK(bean.getIsbn());
    }
    catch(Exception e)
    {
        throw new FinderException(e.toString());
    }
    finally
    {
        try
        {
            if(conn!=null)conn.close();
        }
        catch(SQLException sqle)
        {
            sqle.printStackTrace();
        }
    }
}

public void ejbRemove()
    throws RemoteException
{
    CatalogueEntityPK pk =
(CatalogueEntityPK)entityContext.getPrimaryKey();
    String isbn = pk.getIsbn();
    Connection conn = null;
    try
    {
        conn = getConnection();
        CatalogueDAO dao = new CatalogueDAO(conn);

        dao.deleteCatalogueBean(isbn);
    }
    catch(Exception e)
    {
        throw new RemoteException(e.toString());
    }
    finally
    {

```

```
        try
        {
            if(conn!=null)conn.close();
        }
        catch(SQLException sqle)
        {
            sqle.printStackTrace();
        }
    }
}

public void ejbStore()
    throws RemoteException
{
    Connection conn = null;
    try
    {
        conn = getConnection();
        CatalogueDAO dao = new CatalogueDAO(conn);
        dao.updateCatalogueBean(bean);
    }
    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
{
    CatalogueEntityPK pk =
(CatalogueEntityPK)entityContext.getPrimaryKey();
    String isbn = pk.getIsbn();
    Connection conn = null;
    try
    {
        conn = getConnection();
        CatalogueDAO dao = new CatalogueDAO(conn);
        bean = dao.selectCatalogueBean(isbn);
    }
    catch(Exception e)
    {
        throw new RemoteException(e.toString());
    }
    finally
    {
        try
```

```
        {
            if(conn!=null)conn.close();
        }
        catch(SQLException sqle)
        {
            sqle.printStackTrace();
        }
    }
}

public void setCatalogueBean(CatalogueBean bean)
{
    this.bean = bean;
}

public CatalogueBean getCatalogueBean()
{
    return this.bean;
}

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

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

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

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;
}
}
```

Simpanlah di sub directory net/developerforce/amazon/catalogue/bmp yang telah Anda persiapkan sebagai CatalogueEntityBean.java

Langkah 6 : Dengan text editor tulislah CatalogueEntityPK.java

```
package net.developerforce.amazon.catalogue.bmp;

public class CatalogueEntityPK
    implements java.io.Serializable
{
    private String isbn;

    public CatalogueEntityPK(String isbn)
    {
        this.isbn = isbn;
    }

    public String getIsbn()
    {
        return this.isbn;
    }

    public boolean equals(Object o)
    {
        {
            if(o instanceof CatalogueEntityPK)
            {
                CatalogueEntityPK pk = (CatalogueEntityPK) o;
                return this.isbn.equals(pk.getIsbn());
            }
            else
                return false;
        }

        public int hashCode()
        {
            return this.isbn.hashCode();
        }
    }
}
```

Simpanlah di sub directory net/developerforce/amazon/catalogue/bmp yang telah Anda persiapkan sebagai CatalogueEntityPK.java

Meng-compile Catalogue EJB

Langkah 7 : Bukalah sebuah console

Anda akan meng-compile dan meluncurkan aplikasi dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan PATH harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan PATH :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
```

Untuk dapat meng-compile dengan sukses, Anda CLASSPATH perlu memuat directory atau jar

dimana EJB API berada. Contoh untuk menge-set variabel lingkungan CLASSPATH :

```
$ export  
CLASSPATH=/home/lab/jboss-3.0.0/client/jboss-j2ee.jar:/home/lab/kinaba  
lu
```

Langkah 8 : Meng-compile EJB

Anda akan meng-compile dengan

```
$ javac -sourcepath /home/lab/kinabalu/  
net/developerforce/amazon/catalogue/bmp/*.java
```

Men-deploy

Langkah 9 : Persiapkan deployment descriptor

```
<?xml version="1.0" encoding="Cp1252"?>  
<ejb-jar>  
  <description>jBoss test application</description>  
  <display-name>Test</display-name>  
  <enterprise-beans>  
    <entity>  
      <ejb-name>CatalogueEntity</ejb-name>  
      <home>net.developerforce.amazon.catalogue.bmp.  
CatalogueEntityHome</home>  
      <remote>net.developerforce.amazon.catalogue.bmp.  
CatalogueEntity</remote>  
      <ejb-class>net.developerforce.amazon.catalogue.bmp.  
CatalogueEntityBean</ejb-class>  
      <persistence-type>Bean</persistence-type>  
      <prim-key-class>net.developerforce.amazon.catalogue.bmp.  
CatalogueEntityPK</prim-key-class>  
      <reentrant>False</reentrant>  
    </entity>  
  </enterprise-beans>  
</ejb-jar>
```

Simpan sebagai ejb-jar.xml di /home/lab/kinabalu/META-INF/

Langkah 10 : Gabungkan semua .class dan ejb-jar.xml sebagai sebuah jar

Yaitu dengan :

```
$ jar cvf cataloguebmp.jar  
net/developerforce/amazon/catalogue/bmp/*.class  
added manifest  
adding:  
net/developerforce/amazon/catalogue/bmp/CatalogueEntityBean.class  
(in = 3216) (out= 1321)(deflated 58%)  
adding: net/developerforce/amazon/catalogue/bmp/CatalogueEntity.class  
(in = 362) (out= 226)(deflated 37%)  
adding:  
net/developerforce/amazon/catalogue/bmp/CatalogueEntityHome.class  
(in = 629) (out= 292)(deflated 53%)
```

```
adding: net/developerforce/amazon/catalogue/bmp/CatalogueEntityPK.class
(in = 407) (out= 280)(deflated 31%)
```

```
$          jar          uvf          cataloguebmp.jar
net/developerforce/amazon/catalogue/bean/*.class
adding: net/developerforce/amazon/catalogue/bean/CatalogueBean.class
(in = 1415) (out= 598)(deflated 57%)
```

```
$          jar          uvf          cataloguebmp.jar
net/developerforce/amazon/catalogue/dao/*.class
adding: net/developerforce/amazon/catalogue/dao/CatalogueDAO.class
(in = 3146) (out= 1535)(deflated 51%)
```

```
$ jar uvf cataloguebmp.jar META-INF/ejb-jar.xml
adding: META-INF/ejb-jar.xml(in = 746) (out= 278)(deflated 62%)
```

Jika Anda berminat mengetahui muatan cataloguebmp.jar :

```
$ jar tvf cataloguebmp.jar
```

Hasilnya akan seperti ini :

```
0 Sat Aug 17 00:38:14 SGT 2002 META-INF/
71 Sat Aug 17 00:38:14 SGT 2002 META-INF/MANIFEST.MF
1486 Sat Aug 17 00:33:14 SGT 2002 net/developerforce/amazon/
catalogue/bmp/CatalogueEntityBean.class
418 Sat Aug 17 00:28:46 SGT 2002 net/developerforce/amazon/
catalogue/bmp/CatalogueEntity.class
629 Sat Aug 17 00:28:46 SGT 2002 net/developerforce/amazon/
catalogue/bmp/CatalogueEntityHome.class
407 Sat Aug 17 00:28:46 SGT 2002 net/developerforce/amazon/
catalogue/bmp/CatalogueEntityPK.class
1415 Fri Aug 16 06:25:36 SGT 2002 net/developerforce/amazon/
catalogue/bean/CatalogueBean.class
3146 Fri Aug 16 05:11:50 SGT 2002 net/developerforce/amazon/
catalogue/dao/CatalogueDAO.class
746 Sat Aug 17 00:09:06 SGT 2002 META-INF/ejb-jar.xml
```

Langkah 11 : Jalankan jBoss

```
$ export JAVA_HOME=/home/lab/jdk1.3.1_01
```

```
$ sh /home/lab/jboss-3.0.0/bin/run.sh
```

```
=====
=====
```

```
JBoss Bootstrap Environment
```

```
JBoss_HOME: /home/lab/jboss-3.0.0
```

```
JAVA: /home/lab/jdk1.3.1_01/bin/java
```

```
JAVA_OPTS: -server -Dprogram.name=run.sh
```

```
CLASSPATH:
```

```
/home/lab/jboss-3.0.0/bin/run.jar:/home/lab/jdk1.3.1_01/lib/tools.jar
```

```
=====
=====
```


Langkah 12 : Persiapkan DataSource untuk mySQL

Anda dapat menggunakan template yang disediakan di JBOSS_HOME/docs/examples/jca, yaitu mysql-service.xml. Yang perlu Anda lakukan adalah mengubah ConnectionURL, UserName dan Password, sehingga sesuai dengan lingkungan komputasi Anda.

```
<?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">
      <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/AMAZONDB</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>
  </mbean>
</server>
```

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

<depends optional-attribute-name="JaasSecurityManagerService">
jboss.security:name=JaasSecurityManager</depends>

<attribute
name="TransactionManager">java:/TransactionManager</attribute>
<depends>jboss.jca:service=RARDeployer</depends>

</mbean>

</server>
```

Simpan file ini di directory Anda, misalnya /home/lab/kinabalu

Langkah 13 : Men-deploy mySQL datasource ke jBoss

Deployment ke jBoss adalah sangat sederhana. Anda cukup meng-copy .jar ke directory deploy :

```
$ cp mysql-service.xml /home/lab/jboss-3.0.0/server/default/deploy/
Di console dimana jBoss dijalankan :
21:22:21,680 INFO [MainDeployer] Starting deployment of package:
file:/home/lab/jboss-3.0.0/server/default/deploy/mysql-service.xml
21:22:22,335 WARN [ServiceController] jboss.jca:service=LocalTxDS,
name=MySqlDS does not implement any Service methods
21:22:22,337 INFO [JBossManagedConnectionPool] Creating
21:22:22,338 INFO [JBossManagedConnectionPool] Created
21:22:22,339 INFO [LocalTxConnectionManager] Creating
21:22:22,349 INFO [LocalTxConnectionManager] Created
21:22:22,351 INFO [JBossManagedConnectionPool] Starting
21:22:22,351 INFO [JBossManagedConnectionPool] Started
21:22:22,352 INFO [LocalTxConnectionManager] Starting
21:22:23,036 INFO [MySqlDS] Bound connection factory for resource adapter
'JBoss LocalTransaction JDBC Wrapper' to JNDI name 'java:/MySqlDS'
21:22:23,037 INFO [LocalTxConnectionManager] Started
21:22:23,038 INFO [MainDeployer] Successfully completed deployment
of package:
file:/home/lab/jboss-3.0.0/server/default/deploy/mysql-service.xml
```

Langkah 14 : Men-deploy ke jBoss

Deployment ke jBoss adalah sangat sederhana. Anda cukup meng-copy .jar ke directory deploy :

```
$ cp cataloguebmp.jar /home/lab/jboss-3.0.0/server/default/deploy/
Di console dimana jBoss dijalankan :
00:53:42,396 INFO [MainDeployer] Starting deployment
of package:
file:/home/lab/jboss-3.0.0/server/default/deploy/cataloguebmp.jar
00:53:42,582 INFO [EjbModule] Creating
00:53:42,610 INFO [EjbModule] Deploying CatalogueEntity
```

```
00:53:42,702 INFO [EjbModule] Created
00:53:42,703 INFO [EjbModule] Starting
00:53:42,773 INFO [EjbModule] Started
00:53:42,773 INFO [MainDeployer] Successfully completed deployment
of package:
file:/home/lab/jboss-3.0.0/server/default/deploy/cataloguebmp.jar
```

Mengembangkan Client

Langkah 15 : Dengan text editor tulislah CatalogueEntityBeanTestClientApp.java

```
package net.developerforce.amazon.catalogue.app;

import java.rmi.*;
import java.sql.SQLException;
import java.util.Date;
import java.util.Properties;

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

import net.developerforce.amazon.catalogue.bean.CatalogueBean;
import net.developerforce.amazon.catalogue.bmp.*;

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

            app.create();
            app.display();
            app.modify();
            app.display();
            app.remove();
        }
        catch(CreateException ce)
        {
            ce.printStackTrace();
        }
        catch(FinderException fe)
        {
            fe.printStackTrace();
        }
        catch(NamingException ne)
        {
            ne.printStackTrace();
        }
        catch(RemoteException re)
        {
            re.printStackTrace();
        }
        catch(RemoveException re)
        {
            re.printStackTrace();
        }
    }
}
```

```
        {
            re.printStackTrace();
        }
    }

    private CatalogueEntityHome home;

    public CatalogueEntityBeanTestClientApp()
        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("CatalogueEntity");
        home = (CatalogueEntityHome)
            PortableRemoteObject.narrow(ref,
CatalogueEntityHome.class);
    }

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

        String isbn = "0-201-47967-2";
        CatalogueBean bean = new CatalogueBean(isbn);
        bean.setTitle("Customer-centered Growth :
5 Proven Strategies For Building Competitive Advantage");
        bean.setAuthor("Richard Whiteley");
        bean.setPublisher("Addison Wesley Publishing Company");
        bean.setCity("Massachusetts");
        bean.setYear("1996");
        bean.setLastUpdatedOn(new Date());

        home.create(bean);
    }

    public void modify()
        throws FinderException, RemoteException
    {
        System.out.println("Modifying ... ");

        String isbn = "0-201-47967-2";
        CatalogueEntityPK pk = new CatalogueEntityPK(isbn);
        CatalogueEntity entity = home.findByPrimaryKey(pk);

        CatalogueBean bean = entity.getCatalogueBean();
        bean.setCity("Reading, Massachusetts");
        entity.setCatalogueBean(bean);
        entity = null;
    }
}
```

```
}

public void remove()
    throws FinderException, RemoteException, RemoveException
{
    System.out.println("Removing ... ");

    String isbn = "0-201-47967-2";

    CatalogueEntityPK pk = new CatalogueEntityPK(isbn);
    CatalogueEntity entity = home.findByPrimaryKey(pk);
    entity.remove();
    entity = null;
}

public void display()
    throws FinderException, RemoteException
{
    System.out.println("Displaying ... ");

    String isbn = "0-201-47967-2";
    CatalogueEntityPK pk = new CatalogueEntityPK(isbn);
    CatalogueEntity entity = home.findByPrimaryKey(pk);
    CatalogueBean bean = entity.getCatalogueBean();

    System.out.println("ISBN           : " + bean.getIsbn());
    System.out.println("Title           : " + bean.getTitle());
    System.out.println("Author          : " + bean.getAuthor());
    System.out.println("Publisher       : " + bean.getPublisher());
    System.out.println("City            : " + bean.getCity());
    System.out.println("Year            : " + bean.getYear());
    System.out.println("Last Updated   On      : " +
bean.getLastUpdatedOn());
    bean = null;
}
}
```

Simpanlah di sub directory `net/developerforce/amazon/catalogue/app` yang telah Anda persiapkan sebagai `CatalogueEntityBeanTestClientApp.java`.

Langkah 16 : Bukalah sebuah console untuk client

Anda akan meng-compile dan meluncurkan aplikasi client dari console.

Untuk dapat sukses melakukan kompilasi dan meluncurkan aplikasi, Anda harus menge-set setidaknya dua buah variabel lingkungan dalam Operating System Anda, yaitu : Variabel lingkungan `PATH` harus memuat directory dimana perintah java dan javac. Contoh untuk menge-set variabel lingkungan `PATH` :

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
Contoh untuk menge-set variabel lingkungan CLASSPATH :
$ export CLASSPATH=/home/lab/jboss-3.0.0/client/jboss-j2ee.jar:
/home/lab/kinabalu/cataloguebmp.jar:
/home/lab/jboss-3.0.0/client/jnp-client.jar:
/home/lab/jboss-3.0.0/client/jnet.jar:
```

```
/home/lab/jboss-3.0.0/client/jboss-client.jar:  
/home/lab/jboss-3.0.0/client/jbosssx-client.jar:  
/home/lab/jboss-3.0.0/client/jboss-common-client.jar:  
/home/lab/jboss-3.0.0/client/log4j.jar:.
```

Langkah 17 : Meng-compile CatalogueEntityBeanTestClientApp.java

Melalui console, jalankan javac :

```
$ javac  
net.developerforce.amazon.catalogue.app.CatalogueEntityBeanTestClientA  
pp.java
```

Jika Anda menjalani langkah-langkah dengan benar, Anda dapat menemukan file CatalogueEntityBeanTestClientApp.class di dalam directory yang sama.

Langkah 18 : Meluncurkan aplikasi client

Untuk meluncurkan aplikasi Anda, melalui console, jalankan java :

```
$ java  
net.developerforce.amazon.catalogue.app.CatalogueEntityBeanTestClientA  
pp  
Creating ...  
Displaying ...  
ISBN : 0-201-47967-2  
Title : Customer-centered Growth :  
5 Proven Strategies For Building Competitive Advantage  
Author : Richard Whiteley  
Publisher : Addison Wesley Publishing Company  
City : Massachusetts  
Year : 1996  
Last Updated On : Sat Aug 17 23:18:48 SGT 2002  
Modifying ...  
Displaying ...  
ISBN : 0-201-47967-2  
Title : Customer-centered Growth :  
5 Proven Strategies For Building Competitive Advantage  
Author : Richard Whiteley  
Publisher : Addison Wesley Publishing Company  
City : Reading, Massachusetts  
Year : 1996  
Last Updated On : Sat Aug 17 23:18:48 SGT 2002  
Removing ...
```

10. Proyek 1 : Membuat SalamKeadilanServlet

Tujuan

Proyek SalamKeadilanServlet dirancang sebagai sentuhan pertama Anda dengan JavaServlet.

Rancangan

Anda akan mengembangkan sebuah Java Servlet yang menampilkan pesan "Salam Keadilan !" di atas browser.

Pembekalan

Secara umum, tahap untuk dapat mengembangkan aplikasi dengan JavaServlet, adalah :

1. Menulis kode sumber JavaServlet sebagaimana lumrahnya menulis kode sumber Java lainnya, dan menyimpannya sebagai file berekstension .java
2. Menulis, dan meng-compile class-class pembantu yang dibutuhkan
3. Meng-compile kode sumber JavaServlet untuk mendapatkan file .class dari JavaServlet Anda.
4. Men-deploy ke atas JavaServlet/JSP engine.

Persiapan

- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Persiapkan sebuah directory untuk meletakkan file .java Anda, misalkan /home/lab/process
- Anda perlu meng-install terlebih dahulu Tomcat
- Di bawah directory TOMCAT_HOME/webapps, buatlah sebuah directory untuk aplikasi Anda. Lalu di bawah directory ini buatlah directory WEB-INF, dan kemudian di bawah directory WEB-INF, buatlah directory classes. Misalkan TOMCAT_HOME adalah /home/lab/jakarta-tomcat-3.2.1 dan Anda memilih sub directory bernama process untuk aplikasi Anda, maka Anda akan mempunyai /home/lab/jakarta-tomcat-3.2.1/process/WEB-INF/classes.

Langkah

Langkah 1 : Luncurkan text editor pilihan Anda

Anda dapat menggunakan sembarang text editor untuk menuliskan kode sumber Anda.

Langkah 2 : Menulis kode sumber JavaServlet

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

public class SalamKeadilanServlet extends HttpServlet
{
    public void service(HttpServletRequest request,
```

```
        HttpServletResponse response)
        throws IOException, ServletException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        out.println("<html><body>");
        out.println("Salam keadilan !");
        out.println("</body></html>");
    }
}
```

Langkah 3 : Simpan sebagai SalamKeadilanServlet.java

Simpanlah sebagai SalamKeadilanServlet.java di directory yang telah dipersiapkan. Dalam contoh di atas, simpanlah sebagai /home/lab/process/SalamKeadilanServlet.java

Langkah 4 : Compile SalamKeadilanServlet.java dengan mengarahkan hasilnya ke dalam Tomcat

Sebelum meng-compile Anda perlu menge-set PATH dan CLASSPATH. Untuk dapat meng-compile JavaServlet, CLASSPATH harus memuat sebuah library, lumrahnya dalam bentuk JAR file, yang memuat JavaServlet API. Di dalam Tomcat, tersedia JavaServlet API sebagai TOMCAT_HOME/lib/servlet.jar

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
$ export CLASSPATH=/home/lab/jakarta-tomcat-3.2.1/lib/servlet.jar
```

Untuk memadukan proses compilation dan deployment ke atas Tomcat, Anda dapat meng-compile SalamKeadilanServlet.java menggunakan javac dengan option -d. Option ini akan menunjukkan kepada javac di mana file .class akan disimpan.

```
$ javac -d
/home/lab/jakarta-tomcat-3.2.1/webapps/process/WEB-INF/classes
SalamKeadilanServlet.java
```

Kini di bawah directory /home/lab/jakarta-tomcat-3.2.1/webapps/process/WEB-INF/classes akan terdapat SalamKeadilanServlet.class

Langkah 5 : Jalankan Tomcat

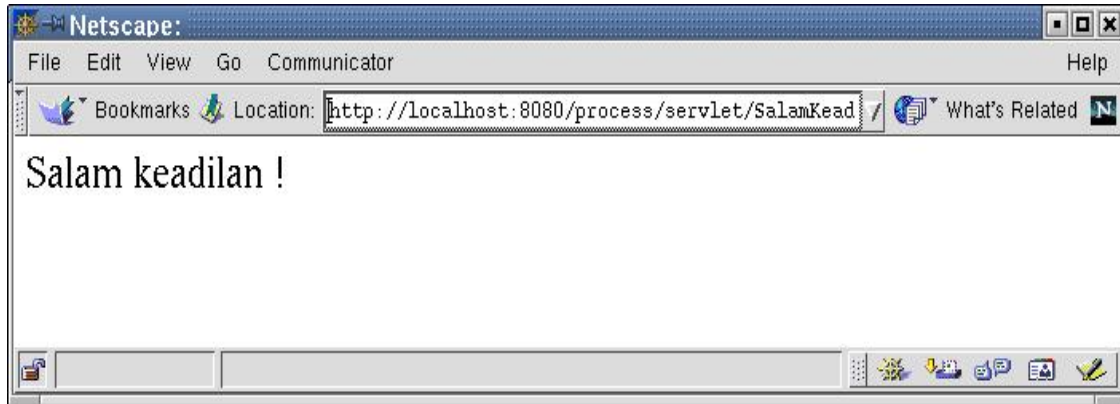
Jalankan Tomcat, dengan memanggil script startup.bat yang terletak di bawah directory TOMCAT_HOME/bin, dengan terlebih dahulu menge-set PATH agar memuat java.

Maka Anda akan mendapatkan adanya context baru :

```
2002-06-03 10:27:21 - ContextManager: Adding context Ctx( /process )
```


Langkah 6 : Uji dengan browser

Anda dapat mem-browse ke <http://localhost:8080/process/servlet/SalamKeadilanServlet>



Troubleshooting

- `package javax.servlet does not exist`
- `import javax.servlet.*;`
- Anda perlu menge-set CLASSPATH agar memuat library JavaServlet API, lumrahnya disimpan sebagai file berformat JAR.
- Not Found (404)
- Dapat terjadi karena Anda :
 - Salah meletakkan file .class, tidak pada directory yang ditentukan. Misalkan seharusnya diletakkan di bawah `/home/lab/jakarta-tomcat-3.2.1/webapps/process/WEB-INF/classes`, tetapi Anda meletakkanya di bawah `/home/lab/jakarta-tomcat-3.2.1/webapps/process`.
 - Salah dalam URL. Misalkan seharusnya Anda mem-browse ke `http://localhost:8080/process/servlet/SalamKeadilanServlet`, tetapi Anda mem-browse ke `http://localhost:8080/process/servlets/SalamKeadilanServlet` atau <http://localhost:8080/process/servlet/SalamKeadilanServlet.class>

11. Proyek 2 : Membuat SalamKeadilan.jsp

Tujuan

Proyek SalamKeadilan.jsp dirancang sebagai sentuhan pertama Anda dengan Java Server Pages.

Rancangan

Anda akan mengembangkan sebuah JSP yang menampilkan pesan "Salam Keadilan !" di atas browser.

Pembekalan

Secara umum, tahap untuk dapat mengembangkan aplikasi dengan JSP

1. Menulis JSP
2. Menulis, dan meng-compile class-class yang dibutuhkan
3. Men-deploy ke atas JavaServlet/JSP engine.

Persiapan

- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Anda perlu meng-install terlebih dahulu Tomcat
- Di bawah directory TOMCAT_HOME/webapps, buatlah sebuah directory untuk meletakkan file jsp Anda Misalkan TOMCAT_HOME adalah /home/lab/jakarta-tomcat-3.2.1 dan Anda memilih sub directory bernama process maka Anda akan mempunyai /home/lab/jakarta-tomcat-3.2.1/process.

Langkah

Langkah 1 : Luncurkan text editor pilihan Anda

Anda dapat menggunakan sembarang text editor untuk menuliskan kode sumber Anda.

Langkah 2 : Menulis kode sumber JSP

```
<html>
<body>
<%
out.println("Salam keadilan !");
%>
</body>
</html>
```

Langkah 3 : Simpan sebagai SalamKeadilan.jsp

Simpanlah sebagai SalamKeadilan.jsp di sub directory yang telah dipersiapkan. Dalam contoh di atas, simpanlah sebagai /home/lab/jakarta-tomcat-3.2.1/webapps/process/SalamKeadilan.jsp

Langkah 4 : Jalankan Tomcat

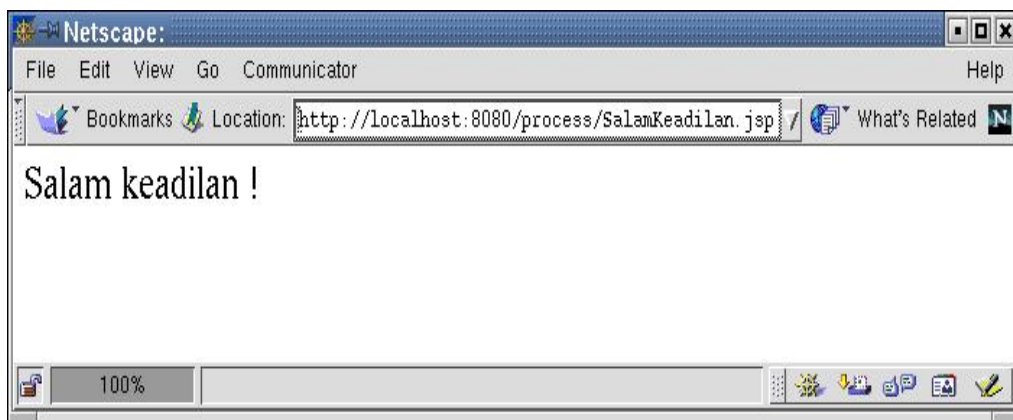
Jalankan Tomcat, dengan memanggil script startup.bat yang terletak di bawah directory TOMCAT_HOME/bin, dengan terlebih dahulu menge-set PATH agar memuat java.

Maka Anda akan mendapatkan adanya context baru :

```
2002-06-03 10:27:21 - ContextManager: Adding context Ctx( /process )
```

Langkah 5 : Uji dengan browser

Anda dapat mem-browse ke <http://localhost:8080/process/SalamKeadilan.jsp>



12. Membaca Parameter dari HTML Form di JavaServlet

Tujuan

Latihan ini dirancang untuk mencoba membaca parameter yang dikirimkan dari HTML Form ke sebuah Servlet

Rancangan

Anda akan menulis sebuah HTML yang mempunyai form untuk diisi dan di-submit ke sebuah Servlet. HTML Anda disimpan sebagai `LatFormDgnServlet.html`, memuat form isian yang dibangun dari beberapa tipe input yaitu text, checkbox, radiobutton dan select. Servlet Anda akan bernama `LatFormServlet.java`, akan membaca parameter yang dikirimkan dari HTML tsb dan menuliskannya di browser.

Pembekalan

Parameter dari HTML Form dapat dibaca melalui interaksi dengan obyek `HttpServletRequest`, yaitu dengan menggunakan method `getParameter()`. Sebagai contoh :

```
String name = request.getParameter("NAME");  
String sex  = request.getParameter("SEX");  
String email = request.getParameter("EMAIL");
```

Persiapan

- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Persiapkan sebuah directory untuk meletakkan file .java Anda, misalkan `/home/lab/process`
- Anda perlu meng-install terlebih dahulu Tomcat
- Di bawah directory `TOMCAT_HOME/webapps`, buatlah sebuah directory untuk aplikasi Anda. Lalu di bawah directory ini buatlah directory `WEB-INF`, dan kemudian di bawah directory `WEB-INF`, buatlah directory `classes`. Misalkan `TOMCAT_HOME` adalah `/home/lab/jakarta-tomcat-3.2.1` dan Anda memilih sub directory bernama `process` untuk aplikasi Anda, maka Anda akan mempunyai `/home/lab/jakarta-tomcat-3.2.1/process/WEB-INF/classes`.

Langkah

Langkah 1 : Dengan text editor tulislah `LatFormDgnServlet.html`

Anda dapat menggunakan sembarang text editor untuk menuliskan halaman HTML.

```
<html>  
<head>  
  <title></title>  
  <meta content="">  
  <style></style>  
</head>  
<body bgcolor=white>  
  
<form action=servlet/LatFormServlet method=post>
```

```

<table border=1>
  <tr>
    <td colspan=2>DATA</td>
  </tr>
  <tr>
    <td>Name :</td>
    <td><input type=text name=NAME></td>
  </tr>
  <tr>
    <td>Sex :</td>
    <td><input type=radio name=SEX value=M checked>Male <input
type=radio
name=SEX value=F>Female </td>
  </tr>
  <tr>
    <td>Email :</td>
    <td><input type=text name=EMAIL></td>
  </tr>
  <tr>
    <td>Competency :</td>
    <td>
      <select name=COMPETENCY>
        <option value="Programming :: Java">Programming ::
J2EE</option>
        <option value="Programming :: .NET">Programming ::
NET</option>
        <option value="Programming :: PHP">Programming ::
PHP</option>
        <option value="OS :: Linux">OS :: Linux</option>
        <option value="OS :: Solaris">OS :: Solaris</option>
        <option value="OS :: Windows">OS :: Windows</option>
        <option value="DB :: mySQL">DB :: mySQL</option>
        <option value="DB :: Oracle">DB :: Oracle</option>
        <option value="DB :: PostgreSQL">DB ::
PostgreSQL</option>
      </select>
    </td>
  </tr>
  <tr>
    <td>Interest :</td>
    <td>
      <input type=checkbox name=INTEREST0 value=Anthropology>
Anthropology
      <input type=checkbox name=INTEREST1 value=Astronomy>
Astronomy
      <input type=checkbox name=INTEREST2 value=Business> Business
      <input type=checkbox name=INTEREST3 value=Politics> Politics
      <input type=checkbox name=INTEREST4 value=Sport> Sport
    </td>
  </tr>
  <tr>
    <td> </td>
    <td><input type=submit value=Proceed></td>
  </tr>
</table>
</form>
</body>
</html>

```

Simpanlah sebagai LatFormDgnServlet.html di bawah directory
TOMCAT_HOME/webapps/process/

Langkah 2 : Dengan text editor tulislah LatFormServlet.java

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

public class LatFormServlet extends HttpServlet
{
    public void service(HttpServletRequest request,
                        HttpServletResponse response)
        throws IOException, ServletException
    {
        String name = request.getParameter("NAME");
        String sex = request.getParameter("SEX");
        String email = request.getParameter("EMAIL");
        String competency = request.getParameter("COMPETENCY");
        String interest = "";

        for(int i=0;i<5;i++)
        {
            String interestI = request.getParameter("INTEREST" + i);
            if(interestI != null)
            {
                interest += " " + interestI;
            }
        }

        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        out.println("<html>");
        out.println("<head>");
        out.println("<title></title>");
        out.println("    <meta content=\"\\\">");
        out.println("    <style></style>");
        out.println("</head>");
        out.println("<body bgcolor=white>");
        out.println("<table border=1>");
        out.println("    <tr>");
        out.println("        <td colspan=2>DATA</td>");
        out.println("    </tr>");
        out.println("    <tr>");
        out.println("        <td>Name :</td>");
        out.println("        <td>" + name + " </td>");
        out.println("    </tr>");
        out.println("    <tr>");
        out.println("        <td>Sex :</td>");
        out.println("        <td>" + sex + " </td>");
        out.println("    </tr>");
        out.println("    <tr>");
        out.println("        <td>Email :</td>");
        out.println("        <td>" + email + " </td>");
        out.println("    </tr>");
```

```
        out.println("        <tr>");  
        out.println("                <td>Competency :</td>");  
        out.println("                <td>" + competency + "</td>");  
        out.println("        </tr>");  
        out.println("        <tr>");  
        out.println("                <td>Interest :</td>");  
        out.println("                <td>" + interest + "</td>");  
        out.println("        </tr>");  
        out.println("</table>");  
        out.println("</form>");  
        out.println("</body>");  
        out.println("</html>");  
    }  
}
```

Simpanlah sebagai LatFormServlet.java di directory yang telah dipersiapkan. Dalam contoh di atas, simpanlah sebagai /home/lab/process/LatFormServlet.java

Langkah 4 : Compile LatFormServlet.java dengan mengarahkan hasilnya ke dalam Tomcat

Sebelum meng-compile Anda perlu menge-set PATH dan CLASSPATH. Untuk dapat meng-compile JavaServlet, CLASSPATH harus memuat sebuah library, lumrahnya dalam bentuk JAR file, yang memuat JavaServlet API. Di dalam Tomcat, tersedia JavaServlet API sebagai TOMCAT_HOME/lib/servlet.jar

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH  
$ export CLASSPATH=/home/lab/jakarta-tomcat-3.2.1/lib/servlet.jar
```

Untuk memadukan proses compilation dan deployment ke atas Tomcat, Anda dapat meng-compile LatFormServlet.java menggunakan javac dengan option -d. Option ini akan menunjukkan kepada javac di mana file .class akan disimpan.

```
$                                javac                                -d  
/home/lab/jakarta-tomcat-3.2.1/webapps/process/WEB-INF/classes  
LatFormServlet.java
```

Kini di bawah directory /home/lab/jakarta-tomcat-3.2.1/webapps/process/WEB-INF/classes akan terdapat LatFormServlet.class

Langkah 5 : Jalankan Tomcat

Jalankan Tomcat, dengan memanggil script startup.bat yang terletak di bawah directory TOMCAT_HOME/bin, dengan terlebih dahulu menge-set PATH agar memuat java.

Maka Anda akan mendapatkan adanya context baru :

```
2002-06-03 10:27:21 - ContextManager: Adding context Ctx( /process )
```

Langkah 6 : Dengan browser bukalah

<http://localhost:8080/process/LatFormDgnServlet.html>

Netscape: File Edit View Go Communicator Help

Location: <http://localhost:8080/process/LatFormDgnServlet.html> What's Related

DATA

Name :	Millati Fitri S
Sex :	<input checked="" type="radio"/> Male <input type="radio"/> Female
Email :	mfs@eramuslim.com
Competency :	OS :: Solaris
Interest :	<input type="checkbox"/> Anthropology <input type="checkbox"/> Astronomy <input type="checkbox"/> Business <input type="checkbox"/> Politics <input type="checkbox"/> Sport
<input type="button" value="Proceed"/>	

100%

Langkah 6 : Dengan menekan tombol Proceed ...

Netscape: File Edit View Go Communicator Help

Location: <http://localhost:8080/process/servlet> What's Related

DATA

Name :	Millati Fitri S
Sex :	F
Email :	mfs@eramuslim.com
Competency :	OS :: Solaris
Interest :	Anthropology Business Sport

13. Membaca Parameter dari HTML Form di JSP

Tujuan

Latihan ini dirancang untuk mencoba membaca parameter yang dikirimkan dari HTML Form ke sebuah JSP

Rancangan

Anda akan menulis sebuah HTML yang mempunyai form untuk diisi dan di-submit ke sebuah JSP. HTML Anda disimpan sebagai LatFormDgnJsp.html, memuat form isian yang dibangun dari beberapa tipe input yaitu text, checkbox, radiobutton dan select. JSP Anda akan bernama LatForm.jsp, akan membaca parameter yang dikirimkan dari HTML tsb dan menuliskannya di browser.

Pembekalan

Parameter dari HTML Form dapat dibaca melalui interaksi dengan obyek `HttpServletRequest`, yaitu dengan menggunakan method `getParameter()`. Sebagai contoh :

```
String name = request.getParameter("NAME");  
String sex  = request.getParameter("SEX");  
String email = request.getParameter("EMAIL");
```

Persiapan

-
- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Anda perlu meng-install terlebih dahulu Tomcat
- Di bawah directory `TOMCAT_HOME/webapps`, buatlah sebuah directory untuk meletakkan file jsp Anda Misalkan `TOMCAT_HOME` adalah `/home/lab/jakarta-tomcat-3.2.1` dan Anda memilih sub directory bernama process maka Anda akan mempunyai `/home/lab/jakarta-tomcat-3.2.1/process`.

Langkah

Langkah 1 : Dengan text editor tulislah LatFormDgnJsp.html

Anda dapat menggunakan sembarang text editor untuk menuliskan halaman HTML.

```
<html>  
<head>  
    <title></title>  
    <meta content="">  
    <style></style>  
</head>  
<body bgcolor=white>  
  
<form action=LatForm.jsp method=post>  
<table border=1>  
    <tr>  
        <td colspan=2>DATA</td>
```

```

        </tr>
        <tr>
            <td>Name :</td>
            <td><input type=text name=NAME></td>
        </tr>
        <tr>
            <td>Sex :</td>
            <td><input type=radio name=SEX value=M checked>Male <input
type=radio
name=SEX value=F>Female </td>
        </tr>
        <tr>
            <td>Email :</td>
            <td><input type=text name=EMAIL></td>
        </tr>
        <tr>
            <td>Competency :</td>
            <td>
                <select name=COMPETENCY>
                    <option value="Programming :: Java">Programming ::
J2EE</option>
                    <option value="Programming :: .NET">Programming ::
NET</option>
                    <option value="Programming :: PHP">Programming ::
PHP</option>
                    <option value="OS :: Linux">OS :: Linux</option>
                    <option value="OS :: Solaris">OS :: Solaris</option>
                    <option value="OS :: Windows">OS :: Windows</option>
                    <option value="DB :: mySQL">DB :: mySQL</option>
                    <option value="DB :: Oracle">DB :: Oracle</option>
                    <option value="DB :: PostgreSQL">DB ::
PostgreSQL</option>
                </select>
            </td>
        </tr>
        <tr>
            <td>Interest :</td>
            <td>
                <input type=checkbox name=INTEREST0 value=Anthropology>
Anthropology
                <input type=checkbox name=INTEREST1 value=Astronomy>
Astronomy
                <input type=checkbox name=INTEREST2 value=Business> Business
                <input type=checkbox name=INTEREST3 value=Politics> Politics
                <input type=checkbox name=INTEREST4 value=Sport> Sport
            </td>
        </tr>
        <tr>
            <td> </td>
            <td><input type=submit value=Proceed></td>
        </tr>
    </table>
</form>
</body>
</html>

```

Simpanlah sebagai LatFormDgnJsp.html di directory TOMCAT_HOME/webapps/process

Langkah 2 : Menulis kode sumber JSP

```
<%
    String name  = request.getParameter("NAME");
    String sex   = request.getParameter("SEX");
    String email = request.getParameter("EMAIL");
    String competency = request.getParameter("COMPETENCY");
    String interest = "";

    for(int i=0;i<5;i++)
    {
        String interestI = request.getParameter("INTEREST" + i);
        if(interestI != null)
        {
            interest += "    " + interestI;
        }
    }
%>

<html>
<head>
    <title></title>
    <meta content="">
    <style></style>
</head>
<body bgcolor=white>
<table border=1>
    <tr>
        <td colspan=2>DATA</td>
    </tr>
    <tr>
        <td>Name :</td>
        <td><%=name%></td>
    </tr>
    <tr>
        <td>Sex :</td>
        <td><%=sex%></td>
    </tr>
    <tr>
        <td>Email :</td>
        <td><%=email%></td>
    </tr>
    <tr>
        <td>Competency :</td>
        <td>
            <%=competency%>
        </td>
    </tr>
    <tr>
        <td>Interest :</td>
        <td>
            <%=interest%>
        </td>
    </tr>
</table>
</form>
</body>
</html>
```

Simpanlah sebagai LatForm.jsp di sub directory yang telah dipersiapkan. Dalam contoh di atas, simpanlah sebagai /home/lab/jakarta-tomcat-3.2.1/webapps/process/LatForm.jsp

Langkah 3: Jalankan Tomcat

Jalankan Tomcat, dengan memanggil script startup.bat yang terletak di bawah directory TOMCAT_HOME/bin, dengan terlebih dahulu menge-set PATH agar memuat java.

Maka Anda akan mendapatkan adanya context baru :

```
2002-06-03 10:27:21 - ContextManager: Adding context Ctx( /process )
```

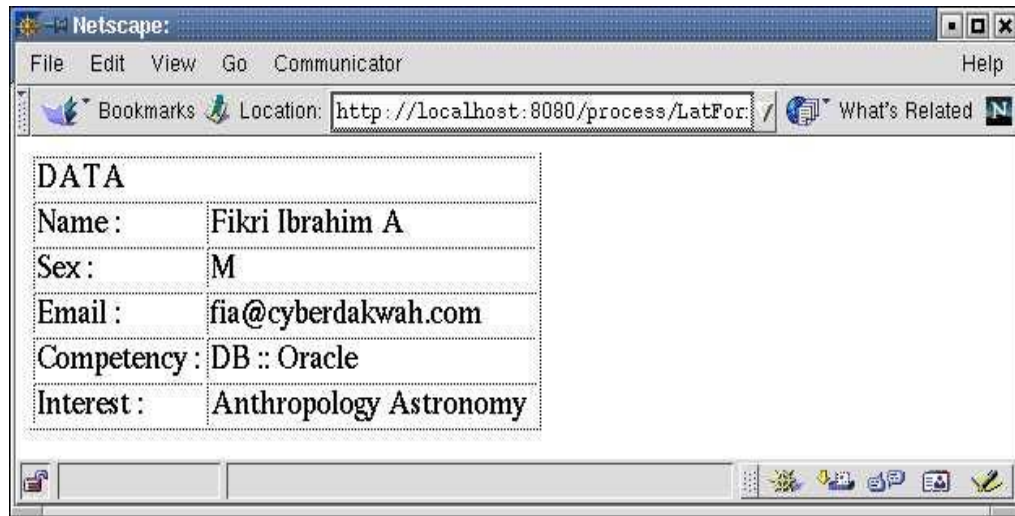
Langkah 4 : Dengan browser bukalah <http://localhost:8080/process/LatFormDgnJsp.html>



The screenshot shows a Netscape browser window with the address bar set to <http://localhost:8080/process/LatFormDgnJsp.html>. The page displays a form titled "DATA" with the following fields:

Name :	<input type="text" value="Fikri Ibrahim A"/>
Sex :	<input checked="" type="radio"/> Male <input type="radio"/> Female
Email :	<input type="text" value="fia@cyberdakwah.com"/>
Competency :	<input type="text" value="DB :: Oracle"/>
Interest :	<input type="checkbox"/> Anthropology <input type="checkbox"/> Astronomy <input type="checkbox"/> Business <input type="checkbox"/> Politics <input type="checkbox"/> Sport
<input type="button" value="Proceed"/>	

Langkah 5 : Dengan menekan tombol Proceed



14. Mengakses Database Dari JavaServlet

Tujuan

Latihan ini dirancang untuk mencoba bagaimana mengakses database dari JavaServlet.

Rancangan

Anda akan mengembangkan sebuah JavaServlet yang mengakses sebuah table dan menampilkan data yang ada dalam kolom tersebut. Database server yang digunakan adalah mySQL, dengan database bernama RELIEVEDB dengan table bernama PATIENTTBL yang memuat data pasien.

Pembekalan

JavaServlet adalah sebuah Java class yang mempunyai tata cara pengembangan khusus untuk dijalankan oleh JavaServlet container dan bekerja untuk aplikasi berbasis Web. Akses database melalui Servlet dapat dilakukan sebagaimana dapat dilakukan dalam Java class lainnya.

Persiapan

- Anda perlu mempunyai mySQL, dan database serta table yang dibutuhkan. Database bernama RELIEVEDB. Table yang dibutuhkan bernama PATIENTTBL dengan kolom PIN, NAME, ADDRESS, IC.
- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Anda perlu meng-install terlebih dahulu Tomcat
- Di bawah directory TOMCAT_HOME/webapps, buatlah sebuah directory untuk meletakkan file jsp Anda. Misalkan TOMCAT_HOME adalah /home/lab/jakarta-tomcat-3.2.1 dan Anda memilih sub directory bernama process maka Anda akan mempunyai /home/lab/jakarta-tomcat-3.2.1/webapps/process. Dan di bawah directory ini buatlah directory WEB-INF/classes.
- Anda membutuhkan JDBC Driver untuk mySQL

Langkah

Langkah 1 : Luncurkan text editor pilihan Anda

Anda dapat menggunakan sembarang text editor untuk menuliskan kode sumber Anda.

Langkah 2 : Menulis kode sumber JavaServlet

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

public class LatJDBCServlet extends HttpServlet
{
    public void service(HttpServletRequest request,
                        HttpServletResponse response)
        throws IOException, ServletException
```

```
{
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    out.println( "<html>"
        + "<head><title>Latihan JDBC dengan Servlet</title></head>"
        + "<body>"
        + "<h1>PATIENT TABLE</h1>" );

    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;

    try
    {
        String jdbcDriver
            = "org.gjt.mm.mysql.Driver";
        Class.forName(jdbcDriver);

        String url = "jdbc:mysql://localhost:3306/RELIEVEDB";
        String user = "ekobs";
        String pwd = "j2ee";

        conn = DriverManager.getConnection(
            url, user, pwd);
        stmt = conn.createStatement();

        String selectSQL =
            "SELECT PIN, NAME, IC, ADDRESS FROM PATIENTTBL";
        rs = stmt.executeQuery(selectSQL);

        out.println("<table border=1>");
        out.println(
            "<tr>"
            + "<th>Patient Identification Number</th>"
            + "<th>Name</th>"
            + "<th>Identification Card</th>"
            + "<th>Address</th>"
            + "</tr>");

        while(rs.next())
        {
            String pin = rs.getString("PIN");
            String name = rs.getString("NAME");
            String ic = rs.getString("IC");
            String address = rs.getString("ADDRESS");

            out.println("<tr>");
            out.println("<td>" + pin + "</td>");
            out.println("<td>" + name + "</td>");
            out.println("<td>" + ic + "</td>");
            out.println("<td>" + address + "</td>");
            out.println("</tr>");
        }
        out.println("</table>");
    }
    catch(ClassNotFoundException cnfe)
    {
    }
}
```

```
        out.println(cnfe.toString());
    }
    catch(SQLException sqle)
    {
        out.println(sqle.toString());
    }
    out.println("</body></html>");
}
}
```

Langkah 3 : Simpan sebagai LatJDBCServlet.java

Simpanlah sebagai LatJDBCServlet.java di directory yang telah dipersiapkan. Dalam contoh di atas, simpanlah sebagai /home/lab/process/LatJDBCServlet.java

Langkah 4 : Compile LatJDBCServlet.java dengan mengarahkan hasilnya ke dalam Tomcat

Sebelum meng-compile Anda perlu menge-set PATH dan CLASSPATH. Untuk dapat meng-compile JavaServlet, CLASSPATH harus memuat sebuah library, lumrahnya dalam bentuk JAR file, yang memuat JavaServlet API. Di dalam Tomcat, tersedia JavaServlet API sebagai TOMCAT_HOME/lib/servlet.jar

```
$ export PATH=/home/lab/jdk1.3.1_01/bin:$PATH
$ export CLASSPATH=/home/lab/jakarta-tomcat-3.2.1/lib/servlet.jar
```

Untuk memadukan proses compilation dan deployment ke atas Tomcat, Anda dapat meng-compile LatJDBCServlet.java menggunakan javac dengan option -d. Option ini akan menunjukkan kepada javac di mana file .class akan disimpan.

```
$ javac -d
/home/lab/jakarta-tomcat-3.2.1/webapps/process/WEB-INF/classes
LatJDBCServlet.java
```

Kini di bawah directory
/home/lab/jakarta-tomcat-3.2.1/webapps/process/WEB-INF/classes akan terdapat
LatJDBCServlet.class

Langkah 5 : Jalankan Tomcat

Jalankan Tomcat, dengan memanggil script startup.bat yang terletak di bawah directory TOMCAT_HOME/bin, dengan terlebih dahulu menge-set PATH agar memuat java.

Maka Anda akan mendapatkan adanya context baru :

```
2002-06-03 10:27:21 - ContextManager: Adding context Ctx( /process )
```


Langkah 6 : Uji dengan browser

Anda dapat mem-browse ke <http://localhost:8080/process/servlet/LatJDBCServlet>



The screenshot shows a Netscape browser window titled "Netscape: Latihan JDBC dengan Servlet". The address bar displays "http://localhost:8080/process/servlet/LatJDBCServlet". The main content area shows a table titled "PATIENT TABLE". The table has four columns: "Patient Identification Number", "Name", "Identification Card", and "Address". There are two data rows. The first row shows "M201", "Muhammad Ilham", "KTP 209/8989/876/200969", and "Jl Cimanggis 3 No 77, Jakarta Selatan 12000 Indonesia". The second row shows "K099", "Kwik Tan Tong", "Passport S908080D", and "Block 998L #20-2, Red Hill 98962 Singapore".

Patient Identification Number	Name	Identification Card	Address
M201	Muhammad Ilham	KTP 209/8989/876/200969	Jl Cimanggis 3 No 77, Jakarta Selatan 12000 Indonesia
K099	Kwik Tan Tong	Passport S908080D	Block 998L #20-2, Red Hill 98962 Singapore

Troubleshooting

- `java.lang.ClassNotFoundException: org.gjt.mm.mysql.Driver`
- Anda perlu meletakkan JDBC driver terkait di bawah TOMCAT_HOME/lib.

15. Mengakses Database dari JSP

Tujuan

Latihan ini dirancang untuk mencoba bagaimana mengakses database dari JSP.

Rancangan

Anda akan mengembangkan sebuah JSP yang mengakses sebuah table dan menampilkan data yang ada dalam kolom tersebut. Database server yang digunakan adalah MySQL, dengan database bernama RELIEVEDB dengan table bernama PATIENTTBL yang memuat data pasien.

Pembekalan

Dalam JSP, program Java dapat ditulis di antara baris-baris HTML. Tidak ada batasan tentang program Java yang boleh ditulis dalam JSP. Pada akhirnya, apa yang bisa Anda tulis dengan JavaServlet dapat ditulis dengan JSP.

Kemampuan JSP yang berdaya guna ini bagaimanapun harus dikembalikan kepada perbedaan peran yang semestinya dijalankan oleh JavaServlet dan JSP. Dalam desain yang cantik, JavaServlet bertanggung jawab atas logika bisnis, sedangkan JSP hanya bertanggung jawab untuk menampilkan data.

Latihan mengakses database dari JSP menunjukkan bagaimana JSP sangat berdaya guna, tetapi dalam penggunaannya, Anda perlu menyadari bahwa ini adalah contoh penggunaan JSP yang tidak cantik.

Persiapan

- Anda perlu mempunyai MySQL, dan database serta table yang dibutuhkan. Database bernama RELIEVEDB. Table yang dibutuhkan bernama PATIENTTBL dengan kolom PIN, NAME, ADDRESS, IC.
- Anda perlu meng-install terlebih dahulu Java Development Kit (JDK)
- Anda perlu meng-install terlebih dahulu Tomcat
- Di bawah directory TOMCAT_HOME/webapps, buatlah sebuah directory untuk meletakkan file jsp Anda. Misalkan TOMCAT_HOME adalah /home/lab/jakarta-tomcat-3.2.1 dan Anda memilih sub directory bernama process maka Anda akan mempunyai /home/lab/jakarta-tomcat-3.2.1/webapps/process.
- Anda membutuhkan JDBC Driver untuk MySQL

Langkah

Langkah 1 : Luncurkan text editor pilihan Anda

Anda dapat menggunakan sembarang text editor untuk menuliskan kode sumber Anda.

Langkah 2 : Menulis kode sumber JSP

```
<html>
<title>Latihan JDBC dengan JSP</title>
<body>
```

```
<h1>PATIENT TABLE</h1>

<%@ page import="java.sql.*"%>
<%
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;

    String jdbcDriver
        = "org.gjt.mm.mysql.Driver";
        Class.forName(jdbcDriver);

    String url = "jdbc:mysql://localhost:3306/RELIEVEDB";
    String user = "ekobs";
    String pwd = "j2ee";

    conn = DriverManager.getConnection(
        url, user, pwd);
    stmt = conn.createStatement();

    String selectSQL =
        "SELECT PIN, NAME, IC, ADDRESS FROM PATIENTTBL";
    rs = stmt.executeQuery(selectSQL);

%>

<table border=1>
    <tr>
        <th>Patient Identification Number</th>
        <th>Name</th>
        <th>Identification Card</th>
        <th>Address</th>
    </tr>
    <%
        while(rs.next())
        {
            String pin = rs.getString("PIN");
            String name = rs.getString("NAME");
            String ic = rs.getString("IC");
            String address = rs.getString("ADDRESS");

%>
            <tr>
                <td><%=pin%></td>
                <td><%=name%></td>
                <td><%=ic%></td>
                <td><%=address%></td>
            </tr>
        <%
        }
    %>
</table>

</body>
</html>
```

Langkah 3 : Simpan sebagai LatJDBC.jsp

Simpanlah sebagai LatJDBC.jsp di sub directory yang telah dipersiapkan. Dalam contoh di atas, simpanlah sebagai /home/lab/jakarta-tomcat-3.2.1/webapps/process/LatJDBC.jsp

Langkah 4 : Jalankan Tomcat

Jalankan Tomcat, dengan memanggil script startup.bat yang terletak di bawah directory TOMCAT_HOME/bin, dengan terlebih dahulu menge-set PATH agar memuat java.

Maka Anda akan mendapatkan adanya context baru :

```
2002-06-03 10:27:21 - ContextManager: Adding context Ctx( /process )
```

Langkah 5 : Uji dengan browser

Anda dapat mem-browse ke <http://localhost:8080/process/LatJDBC.jsp>



The screenshot shows a Netscape browser window titled "Netscape: Latihan JDBC dengan JSP". The address bar shows "http://localhost:8080/process/LatJDBC.jsp". The main content area displays a table titled "PATIENT TABLE". The table has four columns: "Patient Identification Number", "Name", "Identification Card", and "Address". There are two rows of data. The first row shows "M201", "Muhammad Ilham", "KTP 209/8989/876/200969", and "Jl Cimanggis 3 No 77, Jakarta Selatan 12000 Indonesia". The second row shows "K099", "Kwik Tan Tong", "Passport S908080D", and "Block 998L #20-2, Red Hill 98962 Singapore".

Patient Identification Number	Name	Identification Card	Address
M201	Muhammad Ilham	KTP 209/8989/876/200969	Jl Cimanggis 3 No 77, Jakarta Selatan 12000 Indonesia
K099	Kwik Tan Tong	Passport S908080D	Block 998L #20-2, Red Hill 98962 Singapore

Biografi dan Profil



Eko Budhi Suprasetiawan. Lahir 22 Juli 1975 di Jakarta. Menyelesaikan S1 di Institut Teknologi Bandung (ITB) pada tahun 1998. Pendiri dan pengelola situs DeveloperForce.Net. Minat dan bidang keahlian adalah pada J2EE, mySQL, dan Linux. Aktif dalam berbagai organisasi kemahasiswaan selama di ITB, termasuk didalamnya: Pusat Studi Ilmu Kemasyarakatan, Pusat Teknologi Tepat Guna Masjid Salman, Pusat Ilmu Komputer dan Sistem Informasi ITB, Jaringan Informasi Islam, dsb. Saat ini juga aktif di organisasi MIFTA

Berpengalaman kerja di Fujitsu System Indonesia, Sun Microsystems Indonesia, iGine Ptd Ltd. Eko Budhi Suprasetiawan adalah pengelola milis: jlinux@yahoogroups.com.

Informasi lebih lanjut tentang penulis ini bisa didapat melalui:

Email: ekobs@developerforce.net