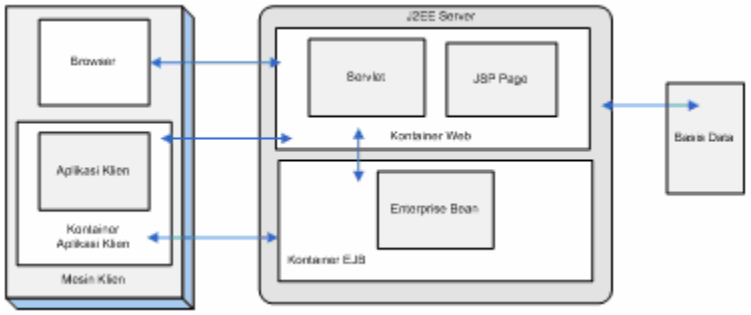
**J2EE dalam Aplikasi Enterprise + CD**
**Pengarang: M. Shalahuddin dan Rosa A. S.**
**Penerbit: INFORMATIKA**
**Rp. 38.000**

---

## 1.3 Java2 Enterprise Edition (J2EE)

J2EE adalah kumpulan teknologi yang cukup kuat dan berada di atas lingkungan J2SE. J2EE berbasis pada Java2 yang berusaha untuk menyediakan sebuah lingkungan aplikasi yang bersifat *reliable* dan stabil serta dapat dijalankan pada beberapa lingkungan sistem operasi. Teknologi enterprise sebagai perkembangan dari lingkungan Java2 difokuskan pada pemenuhan antarmuka yang standar dimana aplikasi J2EE dapat menghasilkan sebuah aplikasi berbasis server yang tangguh (*robust*) dan tidak bergantung pada lingkungan sistem operasi yang digunakan.




**Gambar 1 Arsitektur J2EE**

J2EE server menyediakan dua buah kontainer besar yaitu kontainer EJB dan kontainer web dimana kontainer EJB digunakan untuk mengelola dan mengeksekusi Enterprise bean yang juga disebut dengan bean dan kontainer web digunakan untuk mengelola dan mengeksekusi servlet ( yang akan di bahas lebih lanjut pada bab lima) serta JavaServer Pages atau yang disebut juga dengan JSP, JSP tidak dibahas dalam buku ini.

Enterprise bean terdiri dari tiga jenis bean yang diantaranya adalah :

- ***Session Bean***, yang akan dibahas lebih lanjut pada **Bab 2**,
- ***Entity Bean***, yang akan dibahas lebih lanjut pada **Bab 3**,
- ***Message Driven Bean***, yang akan dibahas lebih lanjut pada **Bab 4**.

*Package* pada J2EE dapat berupa Enterprise Archive (EAR), Java Archive (JAR) yang merupakan kumpulan file dalam sebuah paket, dan Web Archive (WAR) dimana EAR biasanya merupakan gabungan dari file-file JAR yang biasa digunakan oleh bean, sedangkan WAR biasa digunakan oleh servlet dan JSP.

J2EE memiliki beberapa tipe modul yang diantaranya adalah sebagai berikut :

- EJB, terdiri dari file-file *class* dari enterprise bean dan deskriptor dari EJB *deployment*. EJB biasanya menggunakan *package* berupa JAR dengan ekstensi file `.jar`.
- Web, seperti servlet dan JSP. Terdiri dari file-file *class* servlet atau file-file *class* yang dibutuhkan JSP, file gambar, file HTML, dan deskriptor dari web *deployment*. *Package* yang digunakan biasanya adalah WAR dengan ekstensi file `.war`.
- Aplikasi Klien, terdiri dari file class klien dan deskriptor klien. *Package* yang biasanya digunakan adalah JAR dengan ekstensi file `.jar`.

Contoh-contoh modul bean dapat didapatkan pada `http://java.sun.com/j2ee/1.4/download.html#tutorial` dan mengkopinya pada komputer. Di dalam direktori tersebut terdapat file-file pada direktori common yang diperlukan untuk melakukan kompilasi terhadap modul-modul bean yang akan dibuat dengan *script* asant (dengan `path j2eetutorial\examples\common`, `j2eetutorial\examples\ejb\common` dan `j2eetutorial\examples\ejb\commonweb`). Direktori `j2eetutorial\examples\common` terdiri dari tiga file yaitu `admin-password.txt`, `build.properties`, `targets.xml`. Pada file `build.properties` dapat diset tempat direktori kerja agar dapat menggunakan perintah asant build pada direktori kerja yang digunakan. Pengesetan itu dapat dilakukan pada `j2ee.home=C:/Sun/AppServer` dimana path `C:/Sun/AppServer` dapat diganti dengan tempat menginstal server J2EE, dan pada `j2ee.tutorial.home=C:/Projects/j2eetutorial`, path `C:/Projects/j2eetutorial` dapat diganti dengan tempat direktori kerja, `admin.password.file=${j2ee.tutorial.home}/examples/common/admin-password.txt` merupakan path tempat dimana file `admin-password.txt` disimpan.

File `build.properties` dalam direktori `j2eetutorial\examples\common` adalah sebagai berikut :

```
j2ee.home=C:/Sun/AppServer
j2ee.tutorial.home=C:/Projects/j2eetutorial
sunone.home=${j2ee.home}
admin.password.file=${j2ee.tutorial.home}/examples/common/admin-password.txt
admin.host=localhost
admin.user=admin
admin.port=4848
https.port=8181
domain.resources="domain.resources"
domain.resources.port=8080
db.root=${j2ee.home}/pointbase
db.driver=com.pointbase.jdbc.jdbcUniversalDriver
db.host=localhost
db.port=9092
db.sid=sun-appserv-samples
db.url=jdbc:pointbase:server://${db.host}:${db.port}/${db.sid}
db.user=pbpublic
db.pwd=pbpublic
url.prop=DatabaseName
ds.class=com.pointbase.jdbc.jdbcDataSource
db.jvmargs=-ms16m -mx32m
```

Pada bagian di bawah ini pada file build.properties digunakan untuk mendefinisikan server J2EE yang digunakan.

```
admin.host=localhost
admin.user=admin
admin.port=4848
https.port=8181
domain.resources="domain.resources"
domain.resources.port=8080
```

Bagian `admin.host=localhost` digunakan untuk mendefiniskan `host` server yang digunakan untuk membangun server J2EE, `admin.user=admin` untuk mendefinisikan nama admin, dan `domain.resources.port=8080` adalah port server yang digunakan untuk mengakses aplikasi J2EE dari klien.

Sedangkan bagian file `build.properties` di bawah ini digunakan untuk mendefinisikan basis data yang nantinya digunakan pada aplikasi J2EE.

```
db.root=${j2ee.home}/pointbase
db.driver=com.pointbase.jdbc.jdbcUniversalDriver
db.host=localhost
db.port=9092
db.sid=sun-appserv-samples
db.url=jdbc:pointbase:server://${db.host}:${db.port}/${db.sid}
db.user=pbpublic
```

```
db.pwd=pbpublic
url.prop=DatabaseName
ds.class=com.pointbase.jdbc.jdbcDataSource
db.jvmargs=-ms16m -mx32m
```

File `admin-password.txt` dalam direktori `j2eetutorial\examples\common` adalah sebagai berikut :

```
AS_ADMIN_PASSWORD=adminpassword
```

`adminpassword` dapat diganti dengan password admin saat mengistal server J2EE. File `targets.xml` dalam direktori `j2eetutorial\examples\common` adalah sebagai berikut :

```
  <path id="classpath">
    <fileset dir="${j2ee.home}/lib">
      <include name="j2ee.jar"/>
    </fileset>
  </path>

  <target name="clean" >
    <delete dir="${build}" />
    <delete dir="${dist}" />
    <delete dir="${assemble}" />
  </target>

  <path id="db.classpath">
    <fileset dir="${db.root}/lib">
      <include name="*.jar"/>
    </fileset>
  </path>

  <target name="create-db_common" depends="init"
     description="Create database tables and populate database." >
  <java classname="com.pointbase.tools.toolsCommander" fork="yes" >
      <jvmarg line="${db.jvmargs}" />
      <arg line="${db.driver} ${db.url} ${sql.script} ${db.user} ${db.pwd}" />
      <classpath refid="db.classpath" />
    </java>
  </target>

  <target name="admin_command_common">
    <echo message="Doing admin task ${admin.command}"/>
    <sun-appserv-admin
      command="${admin.command}"
      user="${admin.user}"
      passwordfile="${admin.password.file}"
      host="${admin.host}"
      port="${admin.port}"
      asinstalldir="${j2ee.home}" />
  </target>

  <target name="create-jdbc-resource_common">
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="create-jdbc-resource
       --connectionpoolid ${conpool.name} ${jdbc.resource.name}" />
    </antcall>
  </target>

  <target name="delete-jdbc-resource_common">
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="delete-jdbc-resource ${jdbc.resource.name}" />
    </antcall>
  </target>

  <target name="deploy-war">
    <antcall target="admin_command_common">
```

```
      <param name="admin.command"
        value="deploy ${war.file}" />
    </antcall>
  </target>

  <target name="undeploy-war">
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="undeploy ${example}" />
    </antcall>
  </target>

  <property environment="env" />

  <target name="listprops"
     description="Displays values of some of the properties of this build file">
     <property file="../../common/admin-password.txt" />

     <echo message="Path information" />
     <echo message="j2ee.home = ${j2ee.home}" />
     <echo message="j2ee.tutorial.home = ${j2ee.tutorial.home}" />
     <echo message="j2ee.home = ${j2ee.home}" />
     <echo message="env.Path = ${env.Path}" />
     <echo message="env.PATH = ${env.PATH}" />
     <echo message="" />
     <echo message="Classpath information" />
     <echo message="classpath = ${env.CLASSPATH}" />
     <echo message="" />
     <echo message="Admin information" />
     <echo message="admin.password = ${AS_ADMIN_PASSWORD}" />
     <echo message="admin.password.file = ${admin.password.file}" />
     <echo message="admin.host = ${admin.host}" />
     <echo message="admin.user = ${admin.user}" />
     <echo message="admin.port = ${admin.port}" />
     <echo message="https.port = ${https.port}" />
     <echo message="" />
     <echo message="Domain information" />
     <echo message="domain.resources = ${domain.resources}" />
     <echo message="domain.resources.port = ${domain.resources.port}" />
     <echo message="" />
     <echo message="Database information" />
     <echo message="db.root = ${db.root}" />
     <echo message="db.driver = ${db.driver}" />
     <echo message="db.host = ${db.host}" />
     <echo message="db.port = ${db.port}" />
     <echo message="db.sid = ${db.sid}" />
     <echo message="db.url = ${db.url}" />
     <echo message="db.user = ${db.user}" />
     <echo message="db.pwd = ${db.pwd}" />
     <echo message="url.prop = ${url.prop}" />
     <echo message="ds.class = ${ds.class}" />
     <echo message="db.jvmargs = ${db.jvmargs}" />
  </target>
```

Bagian berikut

```
<path id="classpath">
    <fileset dir="${j2ee.home}/lib">
      <include name="j2ee.jar"/>
    </fileset>
  </path>
```

merupakan pendefinisian atau pemetaan *path* dari `j2ee.home` yang berisi *library* yang dibutuhkan untuk membangun aplikasi J2EE dimana *library* tersebut dipaket dalam sebuah file .JAR bernama `j2ee.jar`.

Bagian berikut

```
  <path id="db.classpath">
    <fileset dir="${db.root}/lib">
      <include name="*.jar"/>
    </fileset>
  </path>
```

merupakan pendefinisian atau pemetaan *path* dari *library* basis data yang digunakan sebagai pendukung proses pengaksesan data.

Bagian yang menggunakan tag `<target>` adalah pendefinisian perintah yang nantinya akan dikompilasi dengan menggunakan *asant*. Sedangkan yang menggunakan tag `<echo>` adalah bagian yang akan mengeluarkan informasi mengenai pemetaan berbagai hal yang didefinisikan dalam file `targets.xml`.

Seperti misalnya pada bagian

```
<target name="clean" >
    <delete dir="${build}" />
    <delete dir="${dist}" />
    <delete dir="${assemble}" />
</target>
```

pada file `targets.xml` adalah pendefinisian perintah `clean` yang digunakan untuk menghapus hal-hal yang digenerasi yang sudah tidak digunakan lagi. Penulisan file seperti `targets.xml` sebenarnya sama dengan penulisan pada file .xml yang biasa digunakan pada Apache Ant, sebuah *tool* berbasis Java untuk membangun aplikasi Java sekaligus melakukan kompilasi (keterangan lebih lanjut dapat dilihat di `http://ant.apache.org/manual/`).

File `targets.xml` merupakan pemetaan perintah, pendefinisian *library* dan *resource* yang digunakan yang dipergunakan pada aplikasi J2EE.

Dalam direktori `j2eetutorial\examples\ejb\common` terdapat dua file yaitu `build.properties` dan `targets.xml` sedangkan dalam direktori `commonweb` juga terdapat dua file dengan nama yang sama. Direktori `j2eetutorial\examples\ejb\common` dibutuhkan untuk membangun ejb sedangkan direktori `j2eetutorial\examples\ejb\commonweb` dibutuhkan untuk membangun servlet.

File `build.properties` dalam direktori `j2eetutorial\examples\ejb\common` adalah sebagai berikut :

```
# general
build=build
sql.script=create.sql

# database
conpool.name=PointBasePool
jdbc.resource.name=jdbc/ejbTutorialDB

# mail
mailhost=mail_server_ip
mailuser=user_id
fromaddress=email_address
mailjndi=mail/MySession

# cmp
pm.resource.name=jdo/cmp-roster
pm.factory.class=com.sun.jdo.spi.persistence.support.sqlstore.impl.PersistenceManagerFact
oryImpl
db.schema=PBPUBLIC
```

File `build.properties` digunakan untuk memetakan segala properti yang dibutuhkan untuk membangun aplikasi J2EE, properti itu antara lain properti yang dibutuhkan untuk secara umum, properti

basis data, properti pelayanan mail, dan properti pelayanan terhadap CMP yang nanti akan dibahas detail pada bab tiga.

File `targets.xml` dalam direktori `j2eetutorial\examples\ejb\common` adalah sebagai berikut :

```
  <target name="init">
      <tstamp />
  </target>

  <path id="capture.schema.classpath">
       <fileset dir="${j2ee.home}/lib">
              <include name="appserv-cmp.jar"/>
       </fileset>
  </path>

  <target name="prepare" depends="init"
       description="Create build directories.">
    <mkdir dir="${build}" />
  </target>

  <target name="build" depends="prepare"
     description="Compile source code" >
    <javac srcdir="src" destdir="${build}">
       <include name="**/*.java" />
     <classpath refid="classpath"/>
    </javac>
  </target>

  <!-- ================= JavaMail ====================== -->

  <target name="create-javamail-resource">
    <antcall target="admin_command_common">
      <param name="admin.command" value="create-javamail-resource
       --mailhost ${mailhost} --mailuser ${mailuser}
       --fromaddress ${fromaddress} --storeprotocol=imap
       --storeprotocolclass=com.sun.mail.imap.IMAPStore --transprotocol=smtp
       --transprotocolclass=com.sun.mail.smtp.SMTPTransport
       --debug=false --enabled=true ${mailjndi}"
      />
    </antcall>
  </target>

  <target name="delete-javamail-resource">
    <antcall target="admin_command_common">
      <param name="admin.command" value="delete-javamail-resource ${mailjndi}" />
    </antcall>
  </target>

  <!--======================= CMP ==================== -->

  <target name="create-persistence-resource">
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="create-persistence-resource" />
    </antcall>
    <antcall target="admin_command_common">
      <param name="admin.command" value="set
        ${domain.resources}.persistence-manager-factory-
resource.${pm.resource.name}.factory_class=${pm.factory.class}"/>
    </antcall>
    <antcall target="admin_command_common">
      <param name="admin.command" value="set
        ${domain.resources}.persistence-manager-factory-
resource.${pm.resource.name}.jdbc_resource_jndi_name=${jdbc.resource.name}"/>
    </antcall>
  </target>

  <target name="delete-persistence-resource">
    <antcall target="admin_command_common">
```

```
      <param name="admin.command"
        value="delete-persistence-resource" />
    </antcall>
  </target>

       <target name="capture-db-schema" depends="prepare">
              <java     classname="com.sun.jdo.api.persistence.mapping.ejb.CaptureSchema"
fork="yes" >
              <jvmarg line="${db.jvmargs}" />
              <arg  line="-dburl  ${db.url}  -username  ${db.user}  -password  ${db.pwd}
${db.table.args}      -schemaname      ${db.schema}      -driver      ${db.driver}      -out
${build}/${db.schema.file}" />
              <classpath refid="capture.schema.classpath" />
                    <classpath refid="db.classpath" />
              </java>
       </target>

  <!-- ====================== JMS ==================== -->

  <target name="create-qcf"
    description="Create queue connection factory.">
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="create-jms-resource       --restype       javax.jms.QueueConnectionFactory
jms/QueueConnectionFactory" />
    </antcall>
  </target>

  <target name="create-queue"
    description="Create physical queue and queue resource.">
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="create-jmsdest --desttype queue PhysicalQueue" />
    </antcall>
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="create-jms-resource       --restype       javax.jms.Queue       --property
Name=PhysicalQueue jms/Queue" />
    </antcall>
  </target>

  <target name="create-resources"
    depends="create-qcf,create-queue">
  </target>

  <target name="delete-qcf"
    description="Delete queue connection factory.">
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="delete-jms-resource jms/QueueConnectionFactory" />
    </antcall>
  </target>

  <target name="delete-queue"
    description="Delete physical queue and queue resource.">
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="delete-jmsdest --desttype queue PhysicalQueue" />
    </antcall>
    <antcall target="admin_command_common">
      <param name="admin.command"
        value="delete-jms-resource jms/Queue" />
    </antcall>
  </target>

  <target name="delete-resources"
    depends="delete-qcf,delete-queue">
  </target>

  <!-- ================================================ -->
```

```
  <target name="listprops-ejb" depends="init,listprops"
        description="List property values">
    <echo message="sql.script = ${sql.script}"/>
    <echo message="conpool.name = ${conpool.name}"/>
    <echo message="jdbc.resource.name = ${jdbc.resource.name}"/>
    <echo message="mailhost = ${mailhost}"/>
    <echo message="mailuser = ${mailuser}"/>
    <echo message="fromaddress = ${fromaddress}"/>
    <echo message="mailjndi = ${mailjndi}"/>
    <echo message="pm.resource.name = ${pm.resource.name}"/>
    <echo message="pm.factory.class = ${pm.factory.class}"/>
    <echo message="rr = ${rr}"/>
  </target>
```

File `targets.xml` di atas merupakan pemetaan dari layanan yang dibutuhkan seperti JavaMail, JMS, dan CMP. JavaMail adalah sebuah kumpulan *interface* Java yang merupakan spesifikasi tentang *mail* dalam Java, JMS (*Java Messaging Service*) adalah sebuah kumpulan *interface* Java yang merupakan spesifikasi tentang *messaging* dalam Java, dan CMP(*Container Managed Persistence*) akan dibahas lebih lanjut pada bab tiga.

File `build.properties` dalam direktori `j2eetutorial\examples\ejb\commonweb` adalah sebagai berikut:

```
build=build
sql.script=books.sql
war.file=${example}.war
assemble=assemble
assemble.war=${assemble}/war
```

File `targets.xml` dalam direktori `j2eetutorial\examples\ejb\commonweb` adalah sebagai berikut:

```
  <target name="prepare" depends="init"
  description="Create build directories.">
    <mkdir dir="${build}" />
  </target>


  <target name="copy" depends="prepare"
     description="Copy HTML and JSP pages" >
    <copy todir="${build}">
      <fileset dir="web">
        <include name="**/*.html" />
        <include name="**/*.jsp" />
        <include name="**/*.jspf" />
        <include name="**/*.jspx" />
        <include name="**/*.gif" />
        <include name="**/*.xml" />
        <include name="**/*.tld" />
        <include name="**/*.tag" />
        <include name="**/*.jpg" />
        <include name="**/*.css" />
      </fileset>
    </copy>
  </target>

  <target name="create-war" depends="build"
       description="Packages the WAR file">
     <echo message="Creating the WAR...."/>
     <delete file="${assemble.war}/${war.file}" />
     <delete dir="${assemble.war}/WEB-INF" />
     <copy todir="${assemble.war}/WEB-INF">
       <fileset dir=".">
         <include name="*.xml" />
         <exclude name="build.xml" />
         <exclude name="web.xml" />
```

```
      </fileset>
    </copy>
    <copy todir="${assemble.war}/WEB-INF/classes/">
      <fileset dir="${build}">
        <include name="**/*.class" />
      </fileset>
    </copy>
    <copy todir="${assemble.war}/WEB-INF/tags">
      <fileset dir="${build}">
        <include name="*.tag" />
      </fileset>
    </copy>
    <copy todir="${assemble.war}/WEB-INF">
      <fileset dir="${build}">
        <include name="*.tld" />
      </fileset>
    </copy>
    <copy todir="${assemble.war}">
      <fileset dir="${build}">
        <include name="*.jsp" />
        <include name="*.gif" />
      </fileset>
    </copy>
    <war destfile="${assemble.war}/${war.file}"
         webxml="./web.xml" filesonly="true" >
      <fileset dir="${assemble.war}" includes="WEB-INF/**, *.jsp, *.gif" />
    </war>
    <copy file="${assemble.war}/${war.file}" todir="." />
  </target>



  <target name="copy-clock" depends="build" if="clock.exists"
       description="Copies clock class for bookstore2">
    <copy file="${build}/clock/DigitalClock.class"
          todir="${assemble.war}/clock/" />
  </target>

  <target name="create-bookstore-war" depends="copy-clock"
       description="Packages the WAR file">
    <echo message="Creating the WAR...."/>
    <delete file="${assemble.war}/${war.file}" />
    <delete dir="${assemble.war}/WEB-INF" />
    <copy todir="${assemble.war}/WEB-INF">
      <fileset dir=".">
        <include name="*.xml" />
        <exclude name="build.xml" />
        <exclude name="web.xml" />
      </fileset>
    </copy>
    <copy todir="${assemble.war}/WEB-INF/classes/">
      <fileset dir="${build}">
        <include name="**/*.class" />
        <include name="**/*.properties" />
        <exclude name="clock/*.class" />
      </fileset>
    </copy>
    <copy todir="${assemble.war}/WEB-INF/lib/"
          file="../bookstore/dist/bookstore.jar" />
    <copy todir="${assemble.war}/WEB-INF/tags">
      <fileset dir="${build}">
        <include name="*.tag" />
      </fileset>
    </copy>
    <copy todir="${assemble.war}/WEB-INF">
      <fileset dir="${build}">
        <include name="*.tld" />
      </fileset>
    </copy>
    <copy todir="${assemble.war}">
      <fileset dir="${build}">
```

```
        <include name="**/*.jsp" />
        <include name="**/*.jspx" />
        <include name="**/*.jspf" />
        <include name="**/*.gif" />
        <include name="**/*.jpg" />
        <include name="**/*.css" />
        <include name="*.html" />
      </fileset>
    </copy>
    <war destfile="${assemble.war}/${war.file}"
         webxml="./web.xml" filesonly="true" >
      <fileset dir="${assemble.war}"
        includes="WEB-INF/**,  clock/*.class,  **/*.jsp,  **/*.jspx,  **/*.jspf,  **/*.gif,
*.html, **/*.jpg, **/*.css" />
    </war>
    <copy file="${assemble.war}/${war.file}" todir="." />
  </target>

  <target name="listprops-web" depends="init,listprops"
       description="List property values">
    <echo message="sql.script = ${sql.script}"/>
    <echo message="conpool.name = ${conpool.name}"/>
    <echo message="jdbc.resource.name = ${jdbc.resource.name}"/>
  </target>
```

Untuk setiap pembuatan bean diperlukan file `build.xml` yang dimasukkan dalam direktori bean yang akan dibuat. File `build.xml` sebagai berikut:

```
<!DOCTYPE project [
  <!ENTITY targets SYSTEM "../../common/targets.xml">
  <!ENTITY ejbtargets SYSTEM "../common/targets.xml">
]>

<project name="j2ee-tutorial-ejb" default="build" basedir=".">

  <property file="../../common/build.properties"/>
  <property file="../common/build.properties"/>


  &targets;
  &ejbtargets;

</project>
```

File `build.xml` merupakan pemetaan letak file properti yang dibutuhkan untuk membangun modul EJB atau servlet yang dibuat. Bagian di bawah ini digunakan untuk mendefinisikan proyek, `name` diisi dengan nama proyek, `default` diisi dengan `build`, isi dari default akan memanggil fungsi `build` yang telah dipetakan pada file `targets.xml`. Fungsi `build` merupakan perintah untuk mengkompilasi *source code*.

```
<project name="j2ee-tutorial-ejb" default="build" basedir=".">
```

Bagian di bawah ini digunakan untuk mengacu pada file build.properties yang digunakan pada aplikasi J2EE.

```
  <property file="../../common/build.properties"/>
  <property file="../common/build.properties"/>
```

Sedangkan untuk pembuatan servlet diperlukan file `build.xml` sebagai berikut :

```
<!DOCTYPE project [
  <!ENTITY targets SYSTEM "../../common/targets.xml">
  <!ENTITY webtargets SYSTEM "../commonweb/targets.xml">
]>
```

```
<project name="autodebet" default="build" basedir=".">
  <target name="init">
      <tstamp/>
  </target>

  <!-- Configure the context path for this application -->
  <property name="example" value="hello2" />

  <!-- Configure properties -->
  <property file="../../common/build.properties"/>
  <property file="../commonweb/build.properties"/>

  &targets;
  &webtargets;

  <target name="build" depends="copy"
     description="Compile app Java files" >
    <javac srcdir="src" destdir="${build}">
       <include name="**/*.java" />
       <classpath refid="classpath"/>
    </javac>
  </target>

</project>
```

Bagian di bawah ini merupakan contoh pendefinisian perintah yang ditulis di file `build.xml` (*inline*).

```
<target name="build" depends="copy"
     description="Compile app Java files" >
    <javac srcdir="src" destdir="${build}">
       <include name="**/*.java" />
       <classpath refid="classpath"/>
    </javac>
  </target>
```