JBoss [http://www.jboss.org]

Weiqi Gao
Principal Software Engineer
Object Computing, Inc. http://www.ociweb.com

Thursday July 10, 2003
St. Louis Java Users Group http://www.ociweb.com [http://www.ociweb.com/javasig]

Agenda

Today’s Agenda focuses on the practical, "HOWTO", side.

- **What Is JBoss?**
- **Setup JBoss**
- **Ant, XDoclet, Etc.**
- **Data Sources**
- **Start a Project**
- **Generated Code**

Agenda (Continued)

- **JBoss Class Loading**
- **Configure Logging**

What Is JBoss?

- A popular open source Java application server that supports the J2EE 1.3 specifications
- Runs under any J2SE 1.3 or later Java virtual machine
- Based on an JMX core where other pieces of the system are plugged in
- Supports JNDI, Servlet/JSP (Tomcat or Jetty), EJB, JTS/JTA, JCA, JMS
- Also supports Clustering (JavaGroups), Web Services (Axis), and IIOP integration (JacORB)

What Is JBoss? (Continued)

- **Production release version is 3.2.1. Developer release version is 4.0.0DR2**
- **Released under the LGPL**

Setup JBoss

The JBoss project is hosted on Source Forge [http://sourceforge.net/p

The Files [http://sourceforge.net/project/showfiles.php?group_id=228 section contains all JBoss releases since 2.2.

The files page contains sections named `JBoss`, `JBoss-Jetty`, `JBoss-Tomcat`.

For this talk we will use `jboss-3.2.1_tomcat-4.1.24.zip`.

Installation is easy:

```
$ cd /opt
$ unzip /download/jboss-3.2.1_tomcat-4.1.24.zip
```

Setup JBoss (Continued)

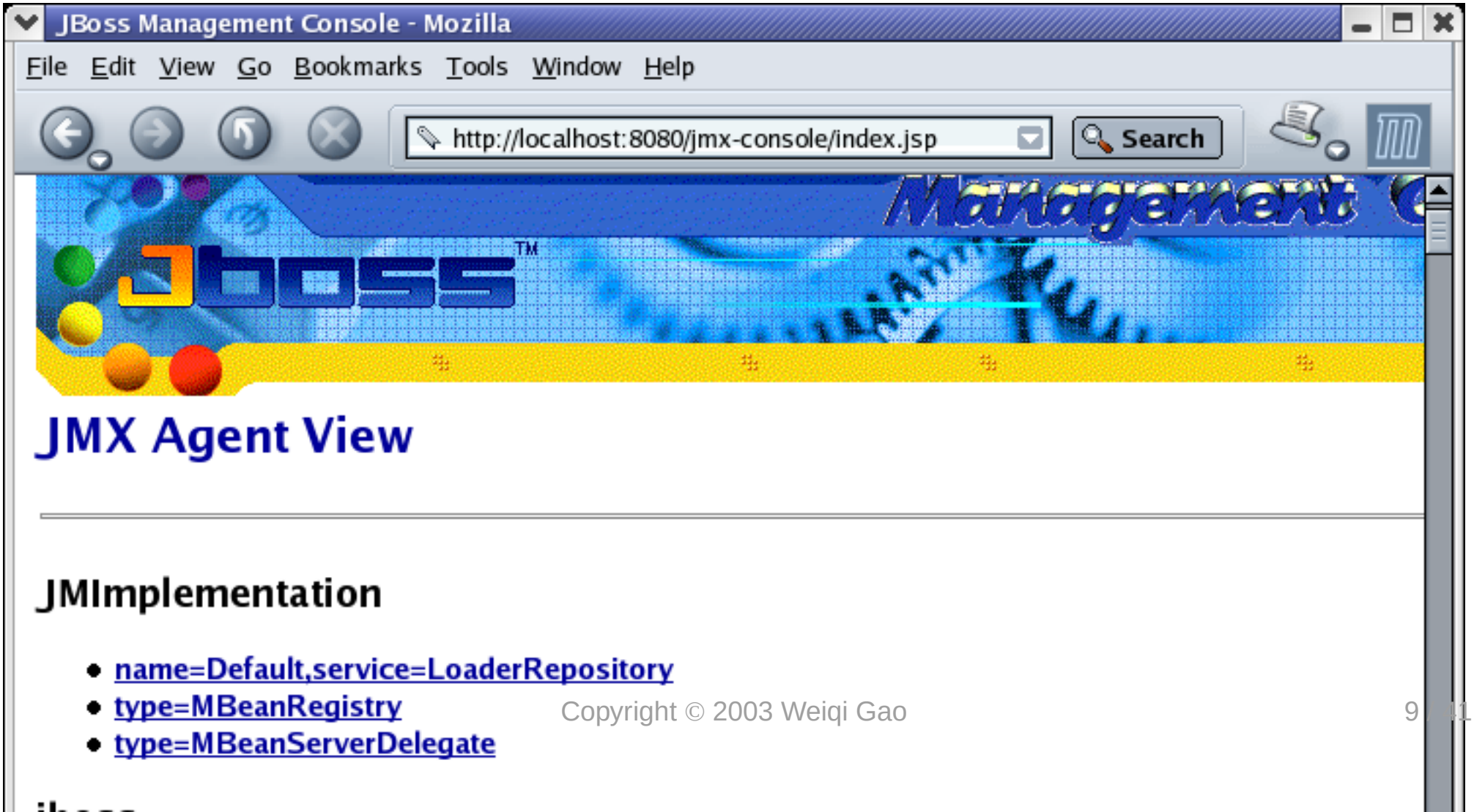
This creates the JBoss directory structure [inspect-dirs.txt]. The `server` subdirectory houses the `minimal`, `default`, and `all` configuration sets.

To run a configuration set, say `all`, simply do:

```
$ cd /opt/jboss-3.2.1_tomcat-4.1.24/bin
$ ./run.sh -c all
```

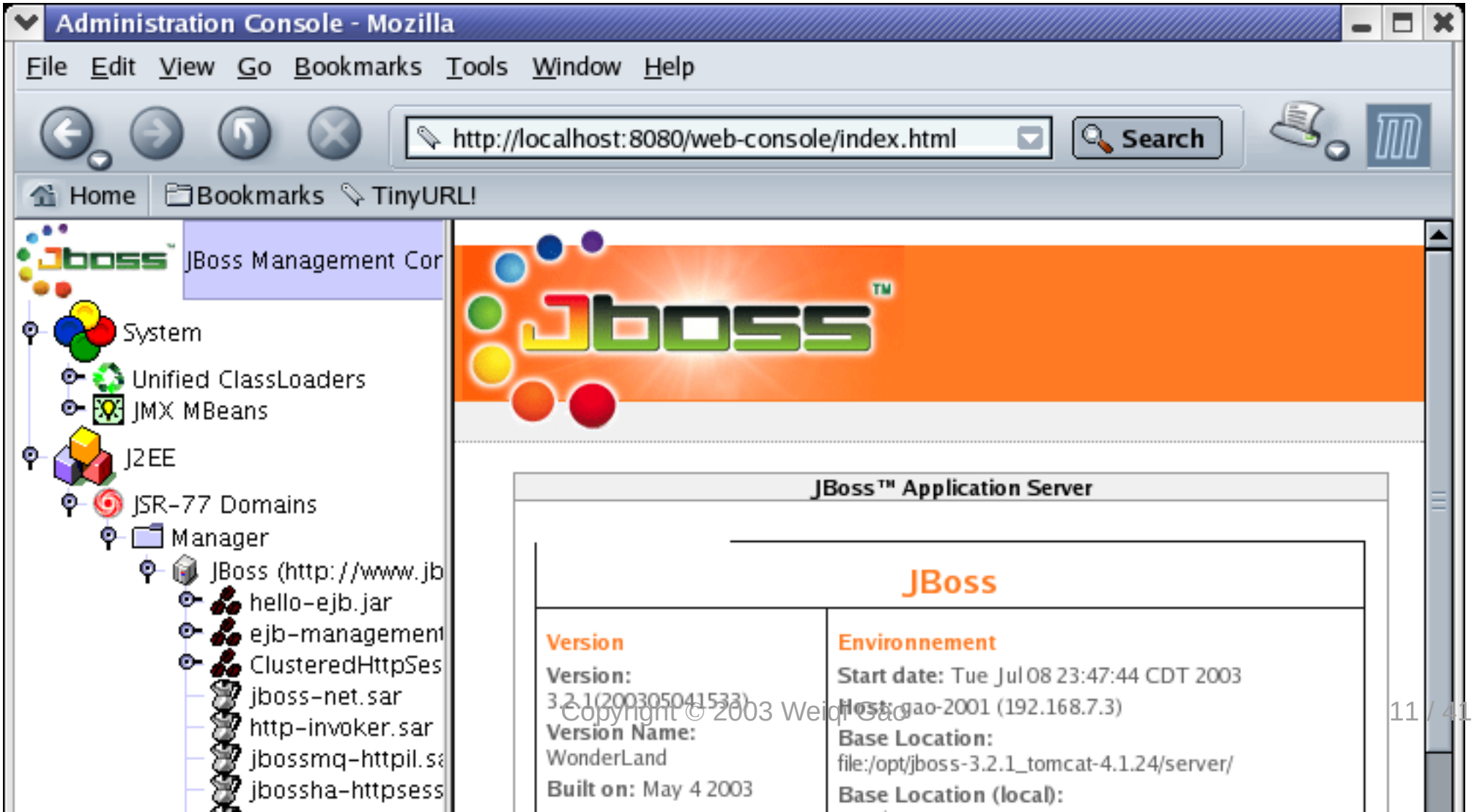
JMX Console

JMX Console (Continued)



Web Console

Web Console (Continued)



A Closer Look

`bin` [inspect-bin.txt] contains `run.sh`, `shutdown.sh`, and bootstrapping jar files.

`client` [inspect-client.txt] contains jar files that must be specified when compiling J2EE applications. Note that `servlet.jar` is in `server/default/lib`. The server does not use this directory at run time.

`docs` [inspect-docs.txt] contains DTDs for deployment descriptors as well as data source configuration scripts for supported RDBMSs.

A Closer Look (Continued)

`lib` [inspect-lib.txt] contains jar files needed to by JBoss. Do not put user jar files into this directory.

`server` [inspect-server.txt] contains the "factory" configuration sets. New configuration sets can be created here, starting with a copy of one of the preconfigured sets.

For each configuration set, `conf` [inspect-conf.txt] contains configuration files, `deploy` [inspect-deploy.txt] contain deployed system resources and hot-deployable user applications, and `lib` [inspect-server-lib.txt] contain non-hot-deployable jar files used by the configuration set.

A Closer Look (Continued)

The `data`, `log`, and `tmp` directories are created if necessary when the server is started. The `log/server.log` [server.log] file contains lots of useful information.

Ant, XDoclet, Etc.

Ant [http://ant.apache.org] is an indispensable build tool for Java development.

XDoclet [http://xdoclet.sourceforge.net] is a must have for EJB development.

JUnit [http://junit.org] is necessary for test-first design and development.

Download the latest releases and install by unpacking, and update your `PATH` and `CLASSPATH`:

Ant, XDoclet, Etc. (Continued)

```
$ cd /opt
$ tar jxvf /download/apache-ant-1.5.3-1-bin.tar.bz2
$ unzip /download/junit3.8.1.zip
$ tar zxvf /download/xjavadoc-1.0.tar.gz
$ mkdir xdoclet-1.2b3; cd xdoclet-1.2b3
$ tar zxvf /download/xdoclet-bin-1.2b2.tar.gz
```

We also use CVS, MySQL, and PostgreSQL, which comes with Red Hat Linux 9.0. CVS and PostgreSQL is also available as part of Cygwin [http://cygwin.com] for Windows.

JBoss as a Service

The Java Service Wrapper [http://wrapper.tanukisoftware.org] can install JBoss as a Windows service or Unix daemon. Its documentation uses JBoss as an example ;).

It consists of:

- **a native executable `wrapper` (`Wrapper.Exe`)**
- **a native shared library `libwrapper.so` (`Wrapper.Dll`)**
- **a Java library `wrapper.jar`**
- **a few shell scripts (batch files)**
- **and a configuration file `wrapper.conf`**

JBoss as a Service (Continued)

All of these files are copied to JBoss's `bin` directory. The wrapper.conf file is modified to suit JBoss's needs.

On Windows, batch files are provided to install and uninstall JBoss as a service. On Red Hat Linux, the shell script can be softlinked to `/etc/rc.d/init.d` to serve as init-script.

Data Sources

JBoss uses JCA to configure JDBC data sources. To configure a data source, we need:

- **The JDBC driver jar file and its java.sql.Driver class name**
- **The JDBC URL**
- **The user name and password**

`docs/examples/jcs` contains data source templates: `oracle-ds.xml`, [oracle-ds.xml] `mysql-ds.xml`, [mysql-ds.xml] `postgres-ds.xml`, etc.

Data Sources (Continued)

To configure a data source available to a configuration set, we simply:

- **Copy the JDBC driver to its `lib` directory**
- **Copy the appropriate `*-ds.xml` to its `deploy` directory**
- **Modify the `*-ds.xml` to reflect the correct JDBC URL, user name and password**

For Oracle, `transaction-service.xml` also needs modification.

Start a Project

There are many starting points for a project. From a physical point of view. The first step should be to setup the environment and a build loop.

The `build.xml` of the hello project started out like this:

```
<project default="junit">
  <target name="junit">
    <junit printsummary="on" fork="false"
      haltonfailure="false"
      showoutput="true">
      <classpath refid="cp"/>
```

Start a Project (Continued)

```
    <formatter type="brief" usefile="false"/>
    <batchtest>
      <fileset dir="src"
        includes="**/Test*.java"/>
    </batchtest>
  </junit>
 </target>
</project>
```

Project Structure

Slight elaboration on the starter build.xml file will give us a full build.xml file that is capable to generate code, compile, package, deploy and run the entire project.

Here's a pictorial depiction [graphics/dependency_graph.png] of the full build.xml.

As the `build.xml` takes shape, the directory structure of the project also become clear:

`src`, `resources`	**The source files, resource files, deployment descriptors, configuration files.**

Project Structure (Continued)

`gen`, `classes`, `staging`, `dist` **Directories for generated code, compiled code, a staging area for intermediate jars (the EJB jar that will go into the EAR), and the final deliverables (the EAR, and the client program.**

A Session Bean

```
...

/**
 * @ejb.bean name="Hello"
 *          type="Stateless"
 *          view-type="remote"
 *          transaction-type="Container"
 *
 * @ejb.transaction type="Required"
 */
public abstract class HelloBean
   implements SessionBean {
```

A Session Bean (Continued)

```
  /**
   * @ejb.interface-method view-type="remote"
   */
  public String sayHello(String message) {
...
```

A Entity Bean

```
/**
 * @ejb.bean name="Item"
 *         type="CMP"
 *         view-type="both"
 *         primkey-field="item"
 *
 * @ejb.transaction type="Required"
 *
 * @ejb.finder
 *   signature "java.util.Collection findAll()"
 *
 * @jboss.persistence
```

A Entity Bean (Continued)

```
 *   datasource="java:/OracleDS"
 *   datasource-mapping="Oracle9i"
 */
public abstract class ItemBean
   implements EntityBean {
```

Persistent Fields

```
/**
 * @ejb.create-method
 */
public Integer ejbCreate(Integer item)
   throws CreateException {
   setItem(item);
   return null;
}
/**
 * @ejb.interface-method view-type="both"
 * @ejb.persistence
 * @ejb.pk-field
```

Persistent Fields (Continued)

```
 */
public abstract Integer getItem();
```

A Message-Driven Bean

```
/**
 * @ejb.bean name="Note"
 *          destination-type="javax.jms.Queue"
 *          acknowledge-mode="Auto-acknowledge"
 *          subscription-durability="NonDurable"
 *          transaction-type="Container"
 *
 * @jboss.destination-jndi-name name="Note"
 */
public class NoteBean implements
   MessageDrivenBean, MessageListener {
   private Context context;
```

A Message-Driven Bean (Continued)

```
public void onMessage(Message message) {
```

Generated Code

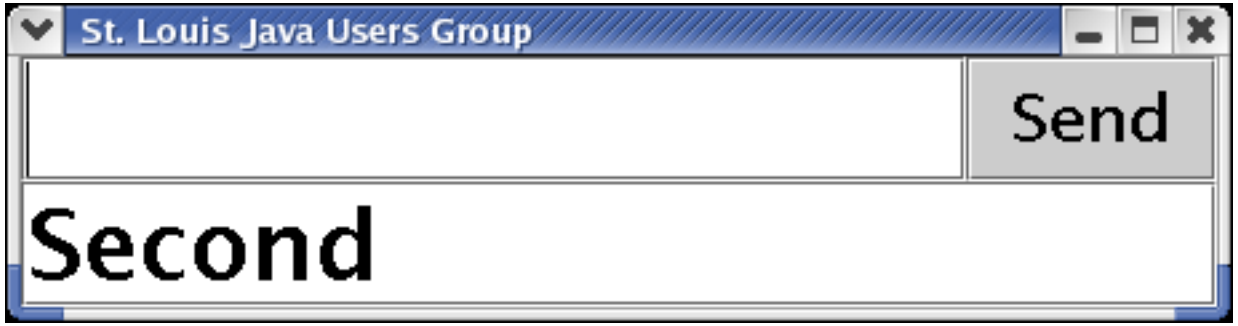
This snippet of Ant script is responsible for generating all the EJB interfaces and deployment descriptors:

```
<taskdef name="ejbdoclet"
   classname="xdoclet.modules.ejb.EjbDocletTask">
  <classpath refid="cp"/>
</taskdef>
<ejbdoclet ejbspec="2.0" destdir="gen"
  <fileset dir="src"
    includes="**/foo/server/*Bean.java"/>
  <homeinterface/>
  <remoteinterface/>
```

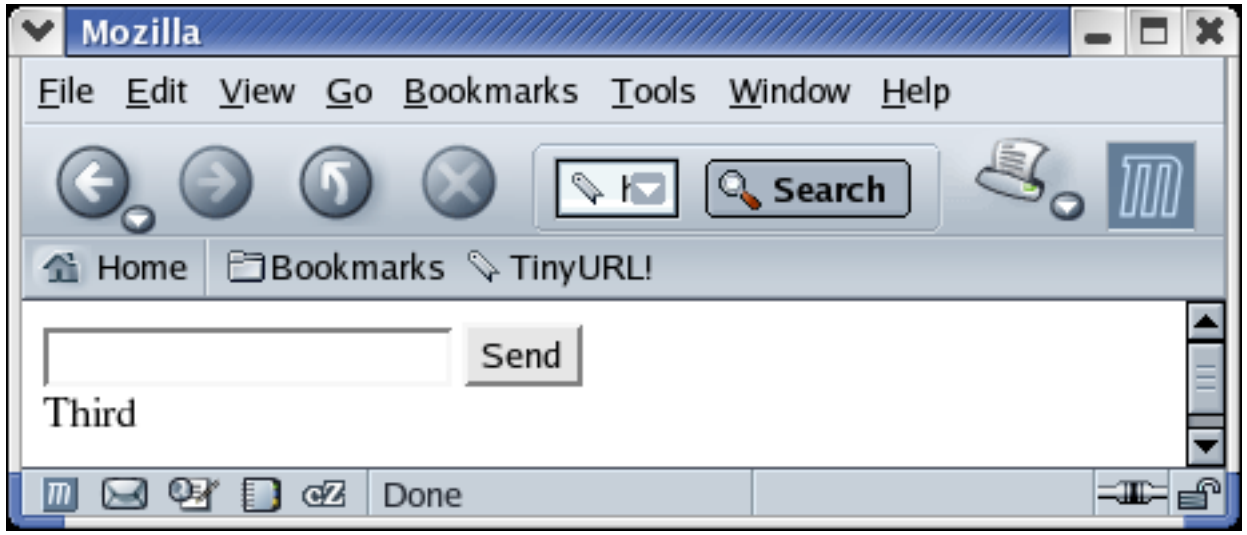
Generated Code (Continued)

```
  <localhomeinterface/>
  <localinterface/>
  <session/>
  <entitycmp/>
  <deploymentdescriptor destdir="gen/ejb/META-INF"/>
  <jboss version="3.2" destdir="gen/ejb/META-INF"/>
</ejbdoclet>
```

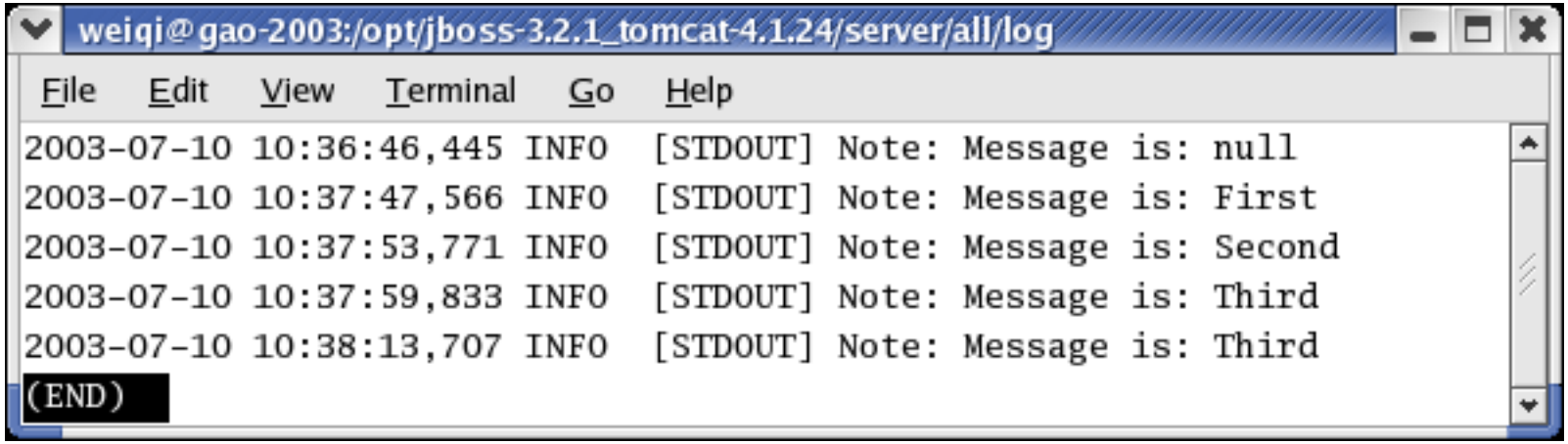
Running the App



Running the App (Continued)



Running the App (Continued)



JBoss Class Loading

JBoss's class loading architecture needs a little getting used to.

- **JBoss uses the concept of class loader repository. For any dynamically deployed file, such as EAR, WAR, EJB jar, RAR, and SAR files each is loaded with a new subordinate class loader. However they also register themselves with a loader repository.**
- **These class loaders will first ask the repository and then load classes from themselves.**
- **They may also decide to become the head of a new loader repository. Classes loaded into child loader repositories are not visible to parent loader repositories.**

JBoss Class Loading (Continued)

- **The order in which the class loaders are added to the repository do matter.**
- **The Russian doll model.**

Add this `jboss-app.xml` along side with `application.xml`:

```
<jboss-app>
  <loader-repository>
    hello:service=LoaderRepository
  </loader-repository>
</jboss-app>
```

Configure Logging

JBoss uses Log4j for its internal logging. It controls Log4j through the `conf/log4j.xml` file. This is a normal Log4j XML config file with one addition. A JBoss specific TRACE level.

To request the trace service add a category section to `log4j.xml`:

```
<category name="org.jboss.mx.loading">
  <priority value="TRACE"
    class="org.jboss.logging.XLevel"/>
</category>
```

You can also configure different File Appenders for different logging categories.

Support Resources

The JBoss Group [http://jboss.org] is founded by Marc Fleury, also the founder of the JBoss project, to provide training and consulting support for JBoss. The forums on JBoss's website is answered by JBoss developers and contains lots of useful information.

The official mailing lists `jboss-development` and `jboss-user` are archived at Source Forge. GMANE [http://gmane.org] offers a bidirectional nntp gateway to many mailing lists, including the JBoss lists.

The Core Developers Network [http://coredevelopers.com] is a newly formed company that provide training and consulting ser-

Support Resources (Continued)

vices for not only JBoss, but also other Open Source Java products.