

# Introduction to Enterprise Java Beans

**Introduction to EJBs**

**EJB Ecosystem**

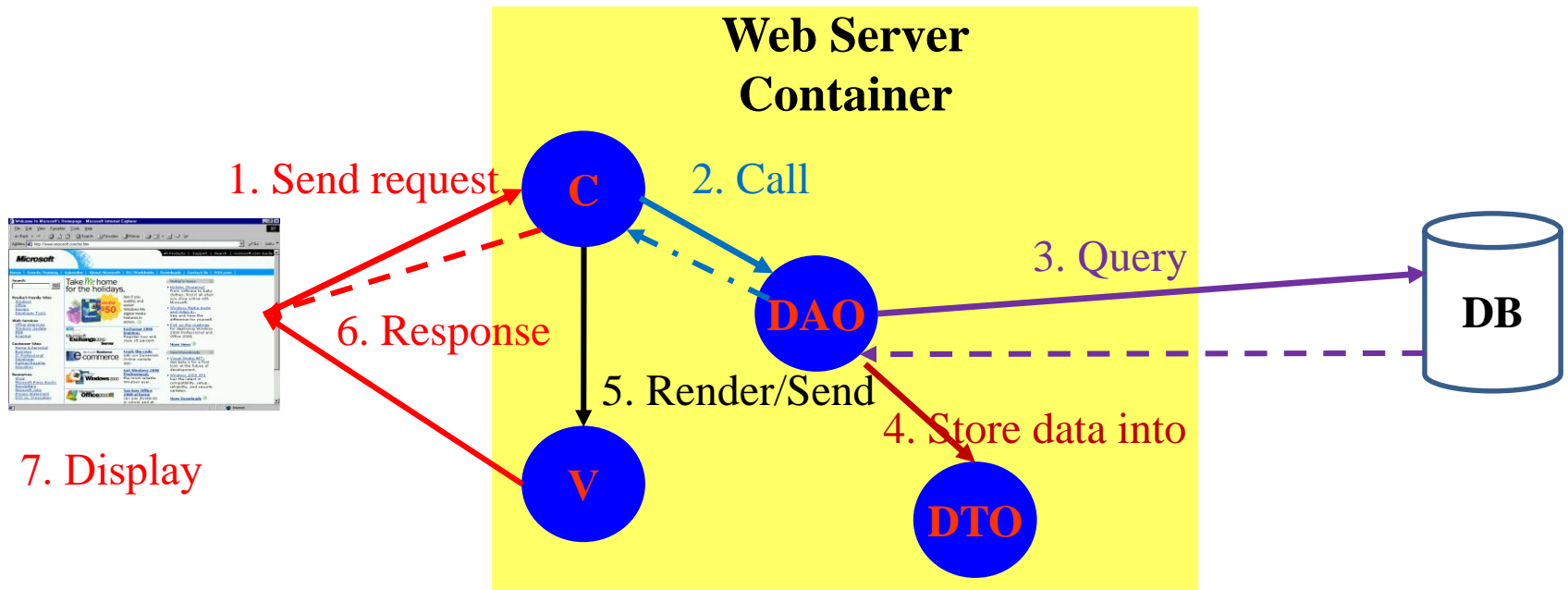
**Enterprise Beans**

**What Constitutes an EJB?**

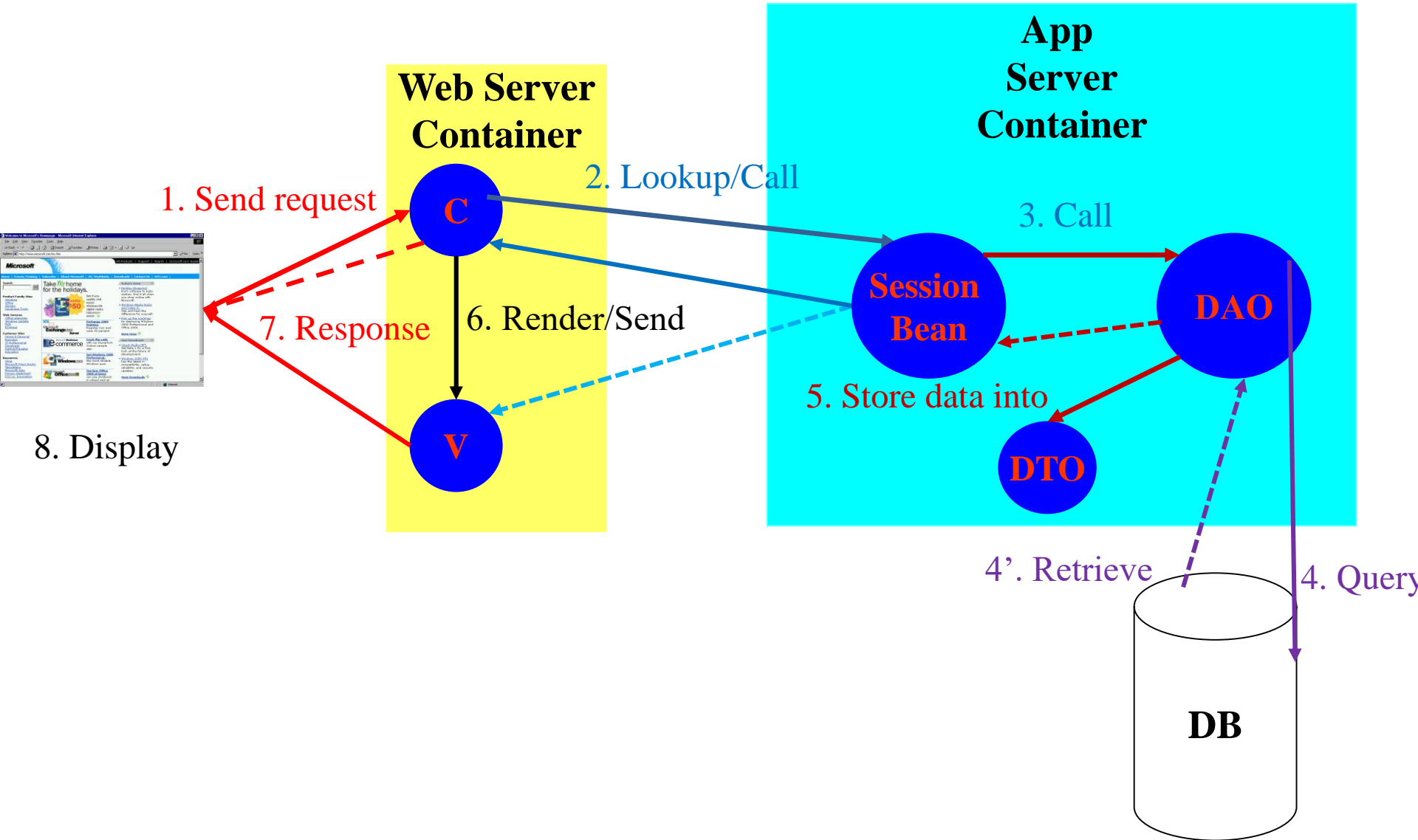
**JNDI**

- **JSTL**
  - Core, Functions, Sql
- **Custom Tag Libraries**
  - Classic Tags
  - Simple Tags
  - Tag Files
  - Tag components
    - Create tld, create tag handler class, import taglib in jsp file
  - Implementation
    - Tag without attributes, tag with attributes, empty tag, tag with body, iterative tag

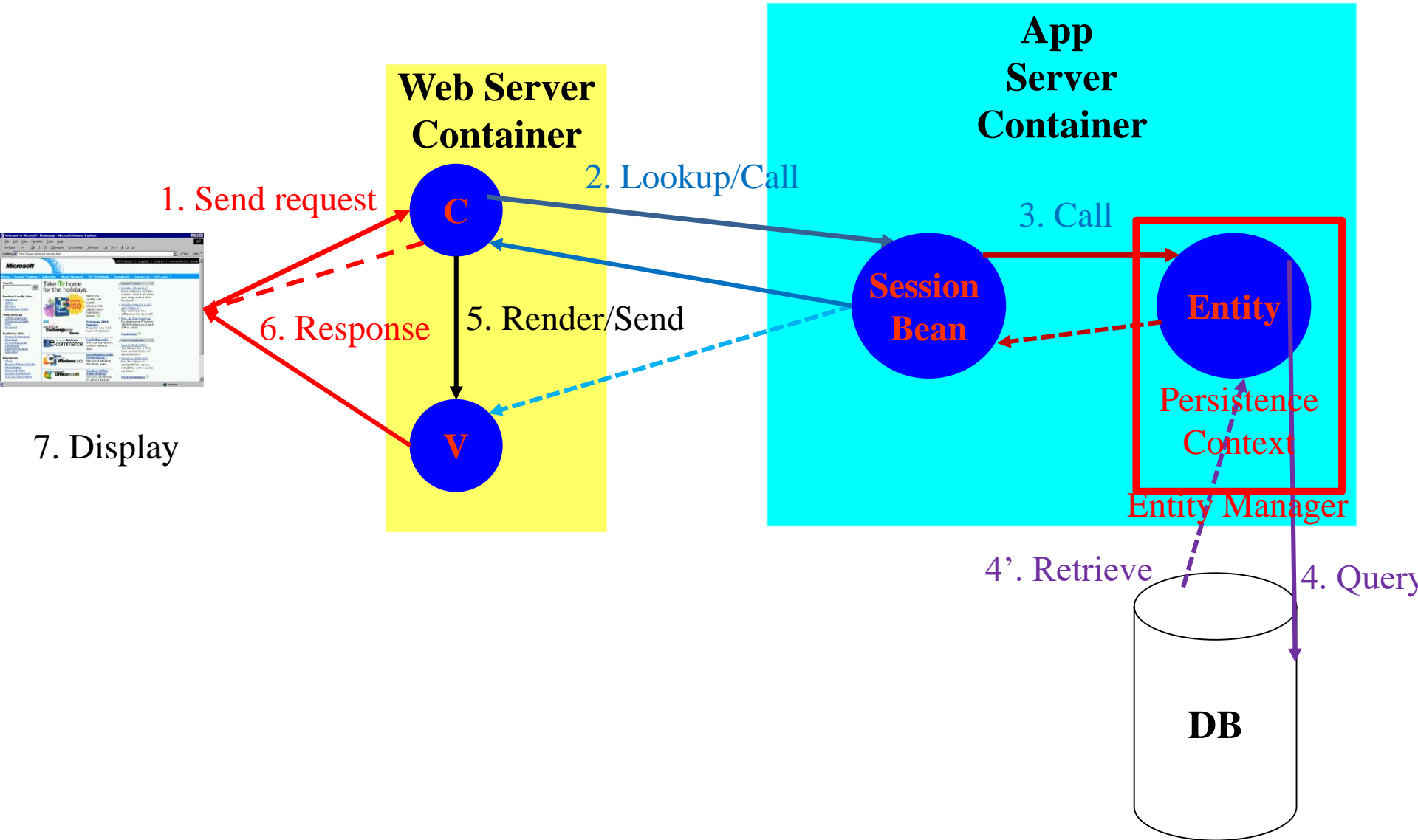
# Review



# Overview



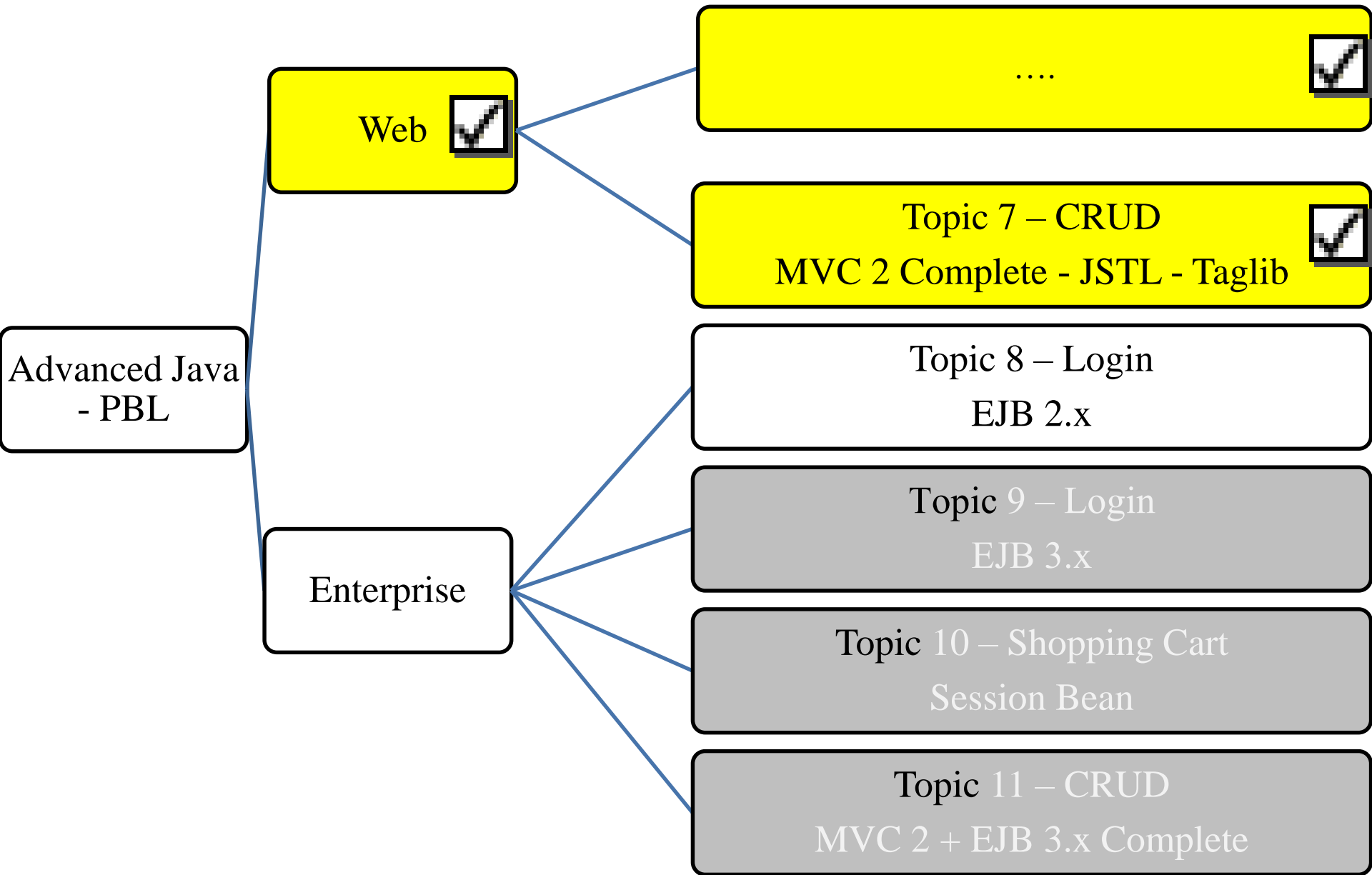
# Overview



# Objectives

- **How to build the simple enterprise application – Login – using EJB 2.0 with GUI as Swing or web?**
  - Logical Architecture of EJB
  - Components of EJB
  - Accessing EJB from the Client/Web Side
  - File and Directory Structure of Enterprise Applications

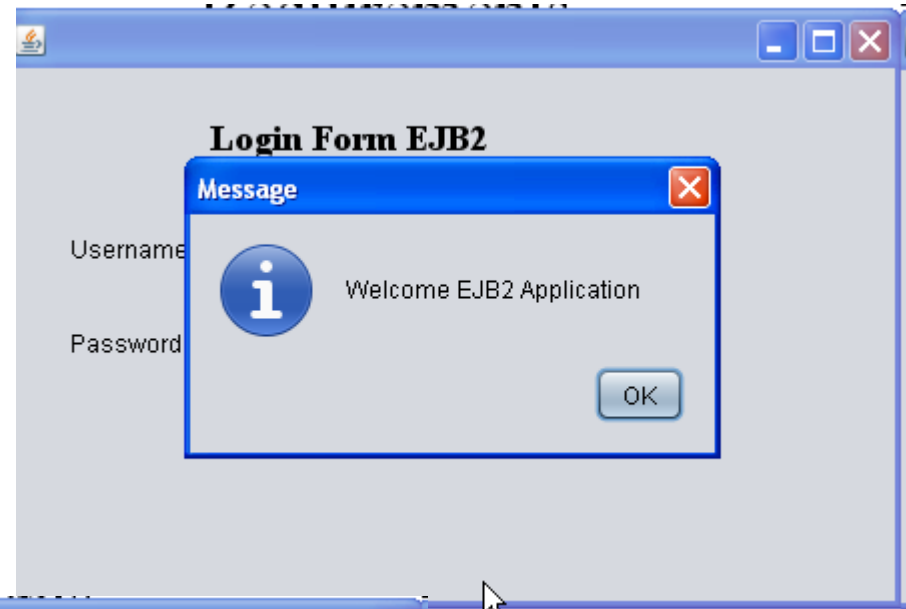
# Objectives



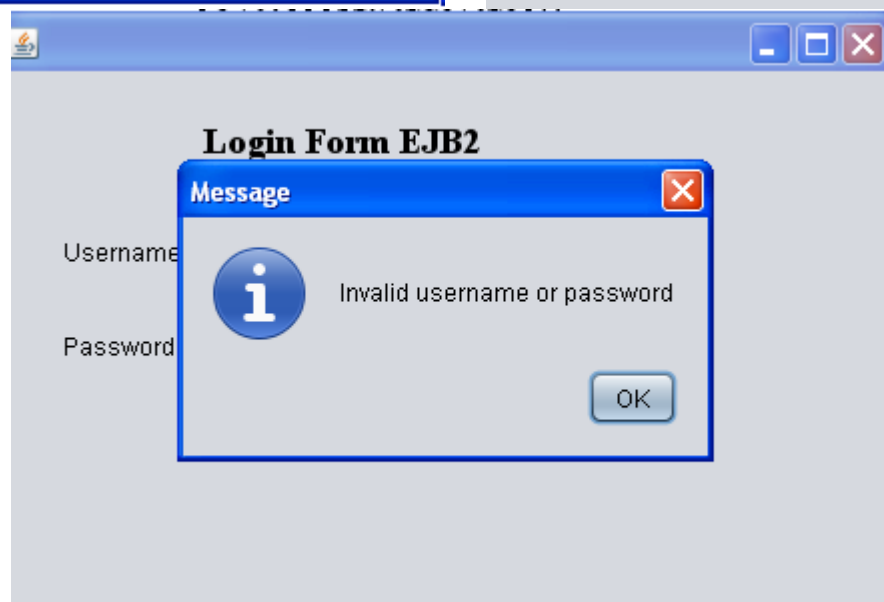
# Build the simple enterprise application Requirements



A screenshot of a Java Swing window titled "Login Form EJB2". It contains two text input fields: "Username" and "Password". The "Password" field is masked with asterisks. Below the fields is a "Login" button.



A screenshot of the "Login Form EJB2" window with a modal "Message" dialog box displayed on top. The dialog box has a blue header with an information icon and the text "Welcome EJB2 Application". It includes an "OK" button.

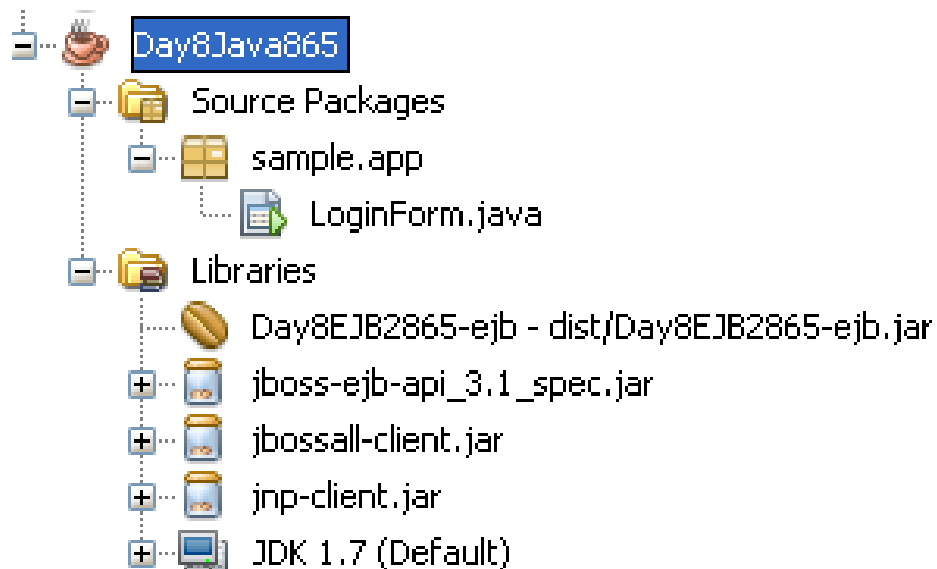
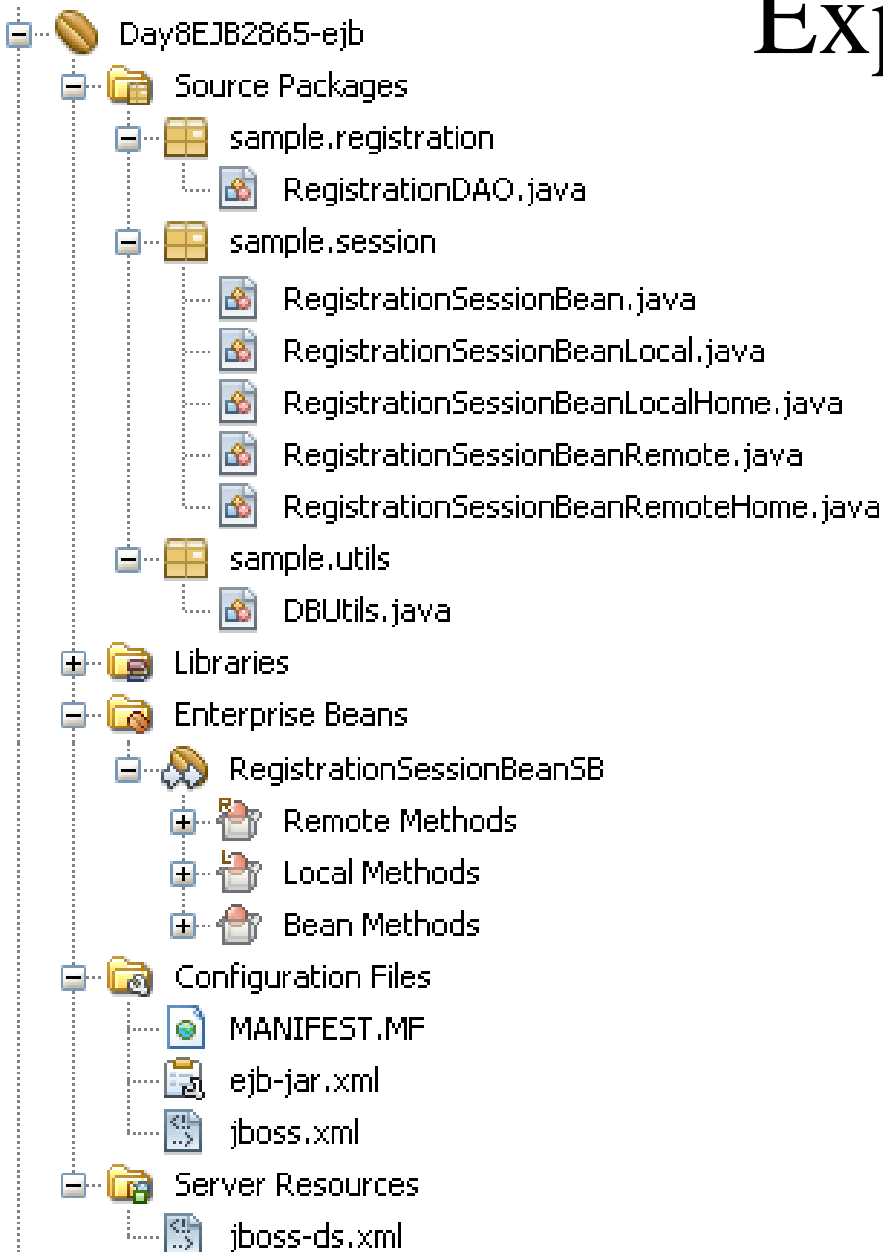


A screenshot of the "Login Form EJB2" window with a modal "Message" dialog box displayed on top. The dialog box has a blue header with an information icon and the text "Invalid username or password". It includes an "OK" button.

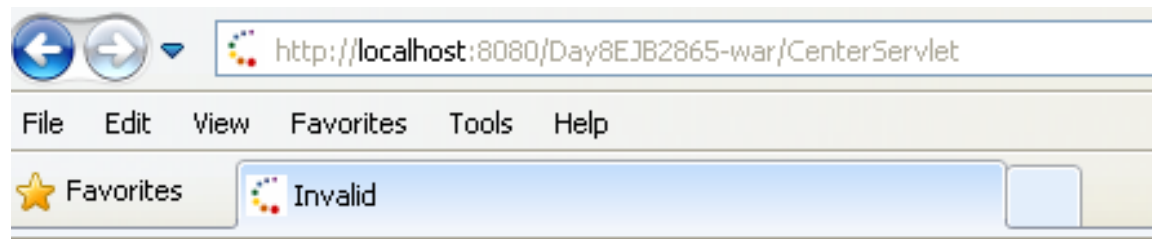
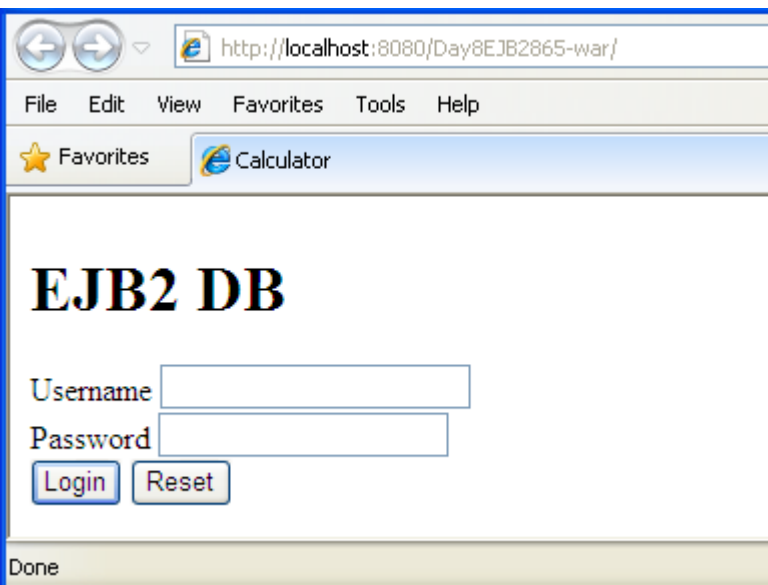


# Build the simple enterprise application

## Expectation



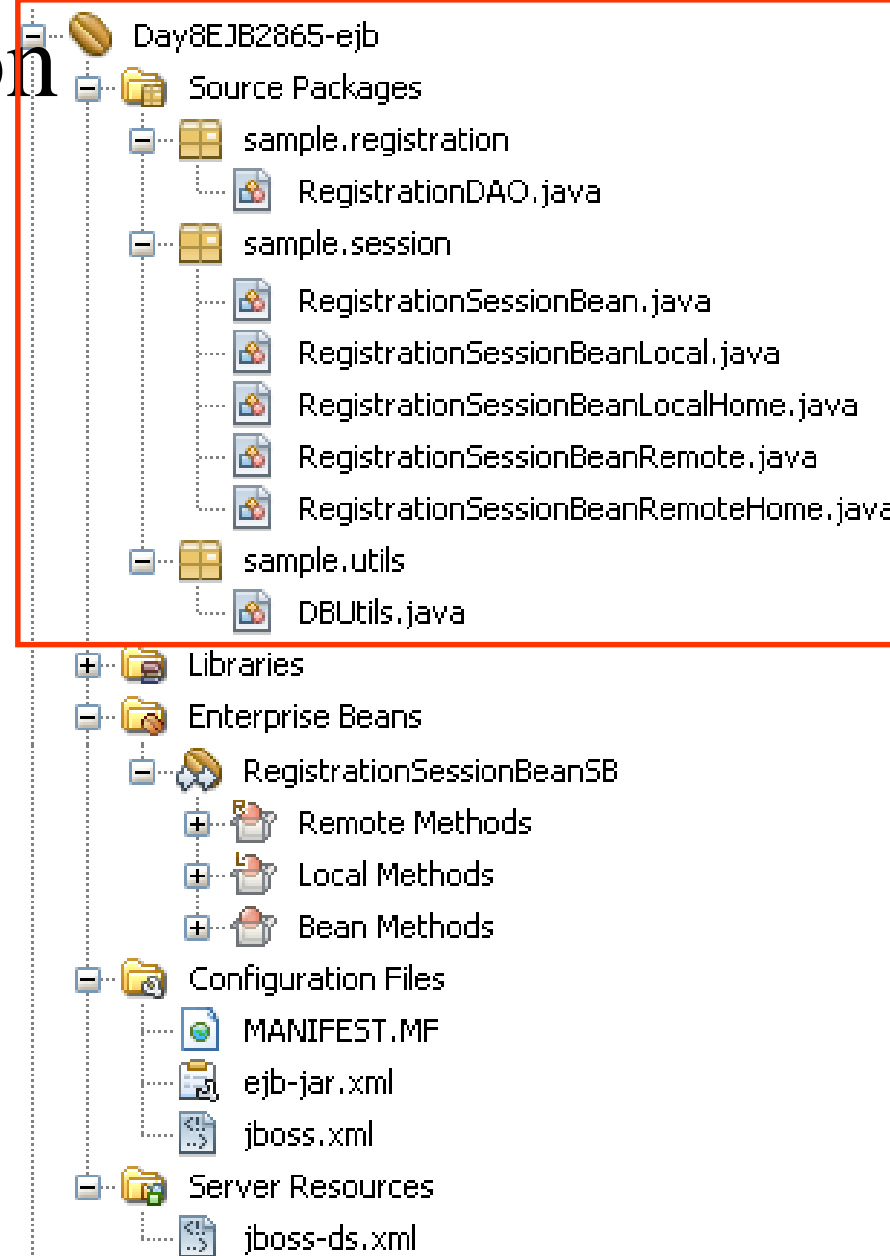
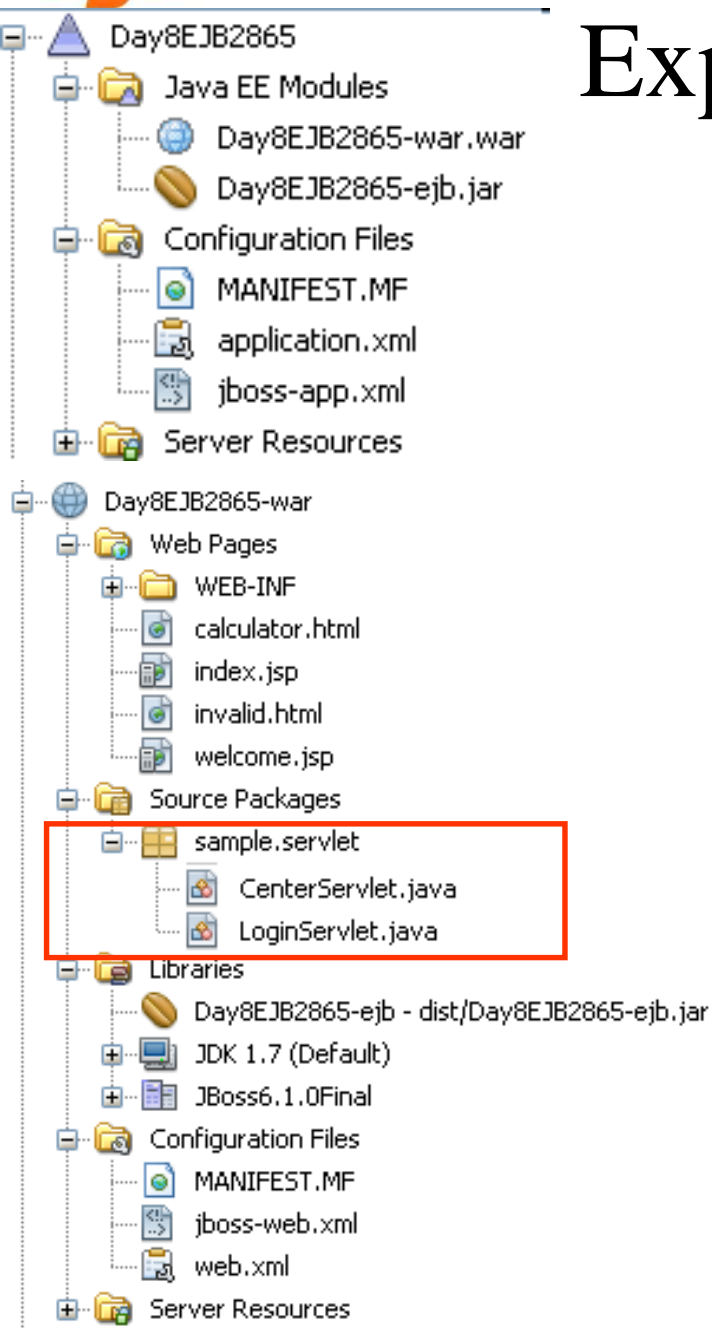
# Build the simple enterprise application Requirements



**Invalid username or password**

# Build the simple enterprise application

## Expectation



# Introduction to EJBs

## Component

- Is a **piece of code** that **exhibits** the **behavior** of a **concept** related to the **real world**
- Can be **reused** in different applications
- **Main requirement** of a component is that it should **encapsulate** the **behavior** of an **application**
  - **Provides a set of services or functions**, such that it can easily interact with other applications or components
  - The **users are not aware** of the **internal processes** of the components in an application **but are aware** of **what they need to pass in as input and what to expect as output**
- **Component framework concept evolved** to **support development and deployment** of enterprise applications
- Components
  - Are **building blocks** of an application
  - Are **distributed over various tiers**

# Introduction to EJBs

## Component Architecture

- Flexible, Portability and Reusable
- Consists mainly of **Web components** (JSP, Servlet, ...), **Business components** (EJB), and **Service components** (Mail, JDBC, JMS ...).
- An enterprise application is usually **composed of a three-layer architecture**
  - **Presentation Layer** (Web Component, GUI Component, Client console)
    - Is responsible for **rendering the graphical user interface and handling user input**
    - **Passes down each request for application functionality** to the business logic layer

# Introduction to EJBs

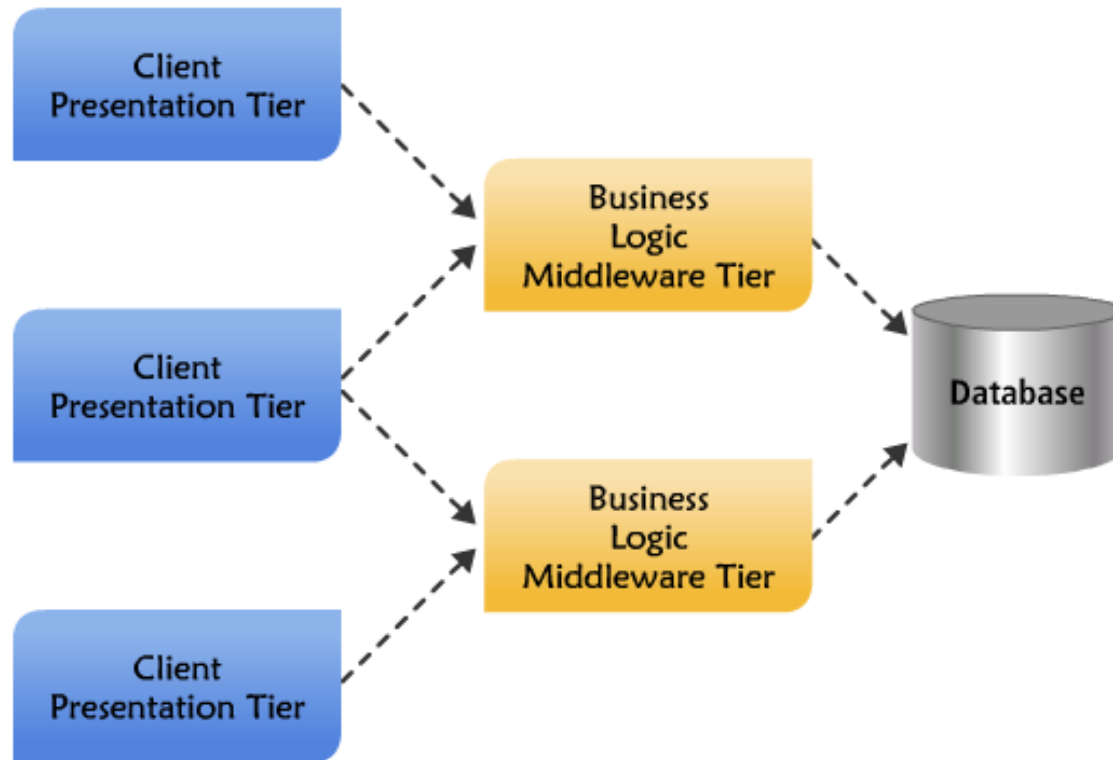
## Component Architecture

- An enterprise application is usually composed of a three-layer architecture
  - **Business Layer** (Business Component)
    - Is the core of the application
    - Comprises business logic and business objects
    - **Business logic**
      - **Comprises business rules or methods** using which **specific business functions** can be managed
      - **Refers to the workflow** or the **ordered task of passing or retrieving data** from one software sub-system to another
  - **Business objects**
    - Are the **set of objects** and the **relationships** between them
    - **Encapsulate** both the data & business behavior associated with the entity that it represents
    - Have the **required features**: reusability, access control, remote access, multi-user, highly available, state maintenance, transactional, and shared data
    - Are **stored to DB or storage** by **using an abstract layer – persistence layer** that lies over the DB layer and interacts with DB

# Introduction to EJBs

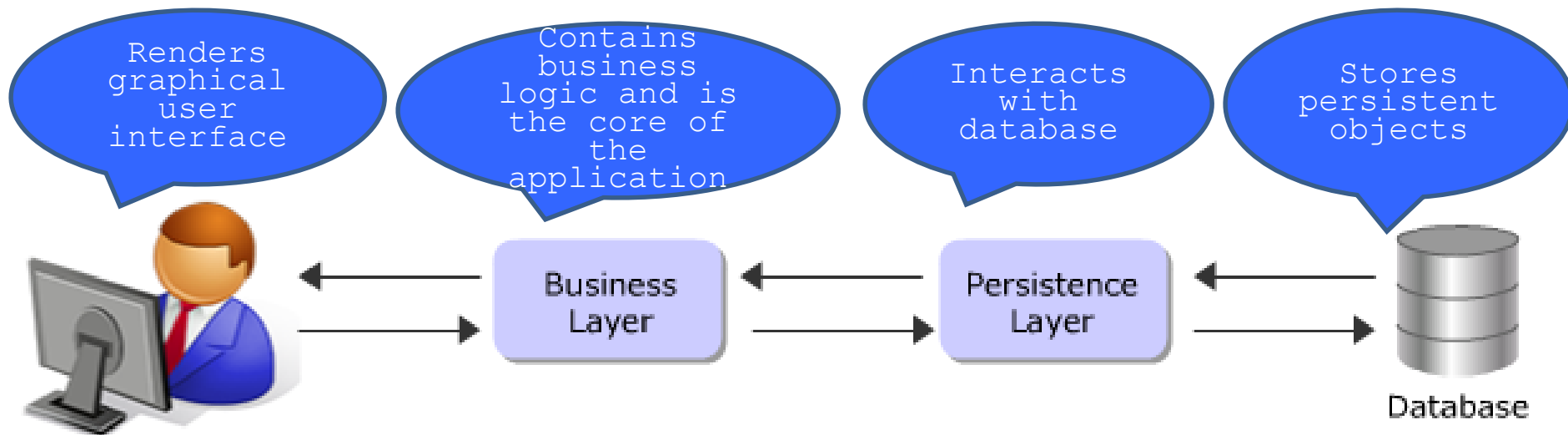
## Component Architecture

- An enterprise application is usually composed of a three-layer architecture
  - **Data Layer**
    - Consists of relational database management systems (**RDBMS**) such as SQL Server, Oracle, DB2, ... **for storing persistent objects**



# Introduction to EJBs

## Distributed Object Architecture





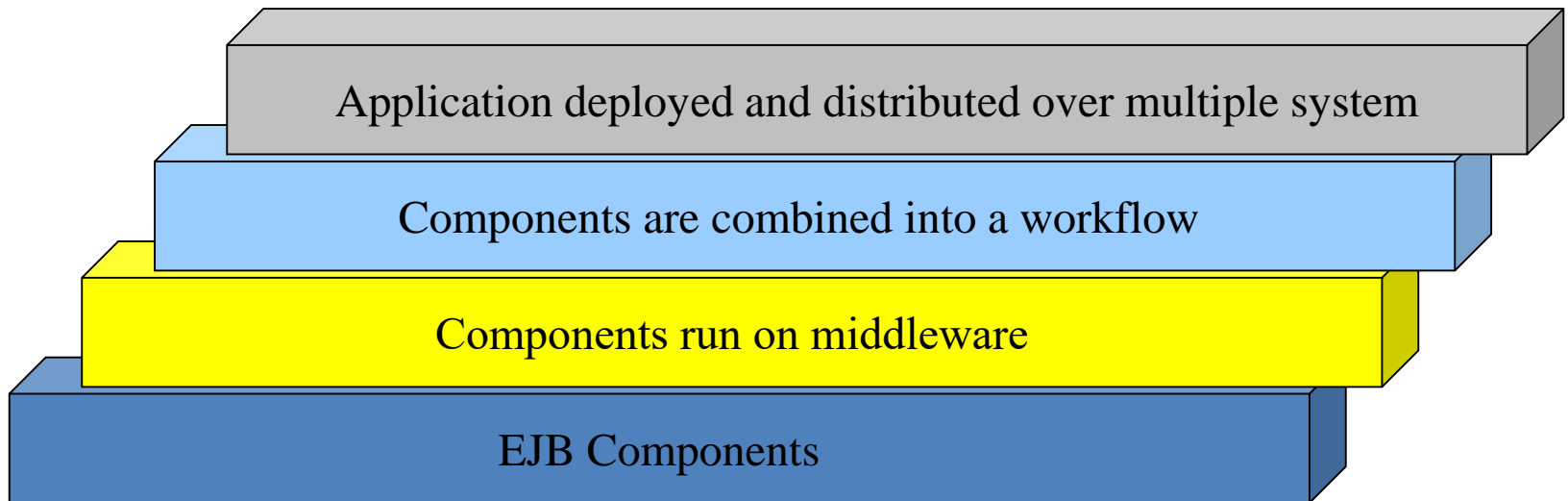
# Introduction to EJBs

## EJB Technology

- Building application software is complexity
  - The software **can process multi-data**
  - The software **is available online.**
  - The developer **worries** about the security, transaction, scalability, concurrency, resource management, persistent, error handling, and **many more system level problems.**
- Software assurance and performance are affected because the developer **can not concentrate fully on the developing the business logic (from implementation logic).**
- EJB was developed so that it would:
  - Specialize in **handling the business logic** of an application
  - Be **robust**
  - **Be secure** so that it cannot be tampered.
  - EJB **provides a component to create middleware** which is deployed on Application Server (3 tiers architecture).
  - EJB Component has been designed to **encapsulate business logic.**

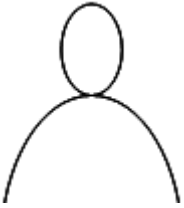
# EJB EcoSystem

## Stages in Developing Business Solution



# EJB EcoSystem

## Parties involved in EJB Development

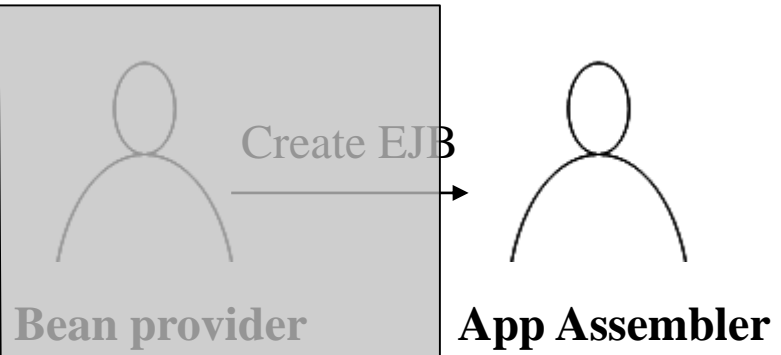


**Bean provider**

- Provide the components to **solve business problem** (that are packaged them to the ejb-jar file)
- **Reusable** components
- **Assemble** other components into application.
- **Distribution**

# EJB EcoSystem

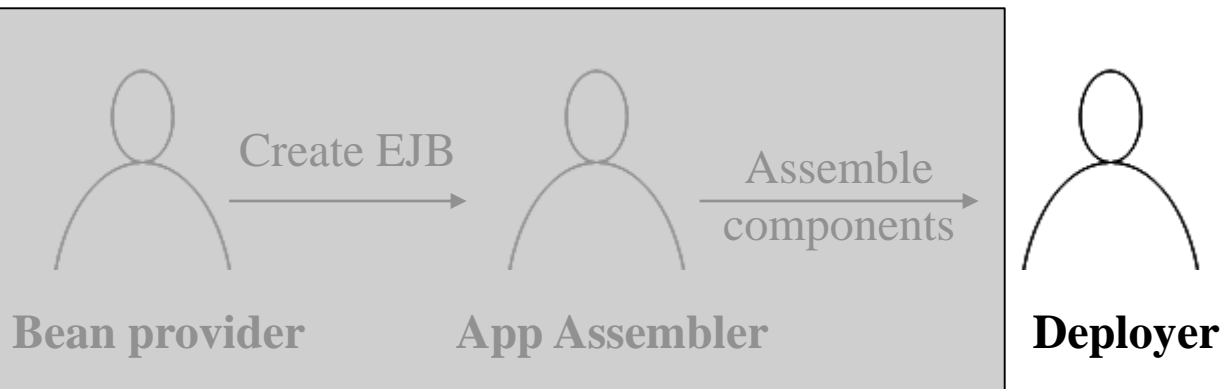
## Parties involved in EJB Development



- For **assembling** different EJB Components in order to build a complete application
- Analyzing** a business problem and **assembling** EJB components **accordingly to solve the problem**
- Building** new EJB components
- Writing the integration code** required to associate the EJB components build by different bean providers

# EJB EcoSystem

## Parties involved in EJB Development



- **Customizing** enterprise bean
- **Accumulate information** about operational requirements such as security, hardware, and transaction before deploying the bean
- For **deploying** an assembled application in an application server

# EJB EcoSystem

## Parties involved in EJB Development



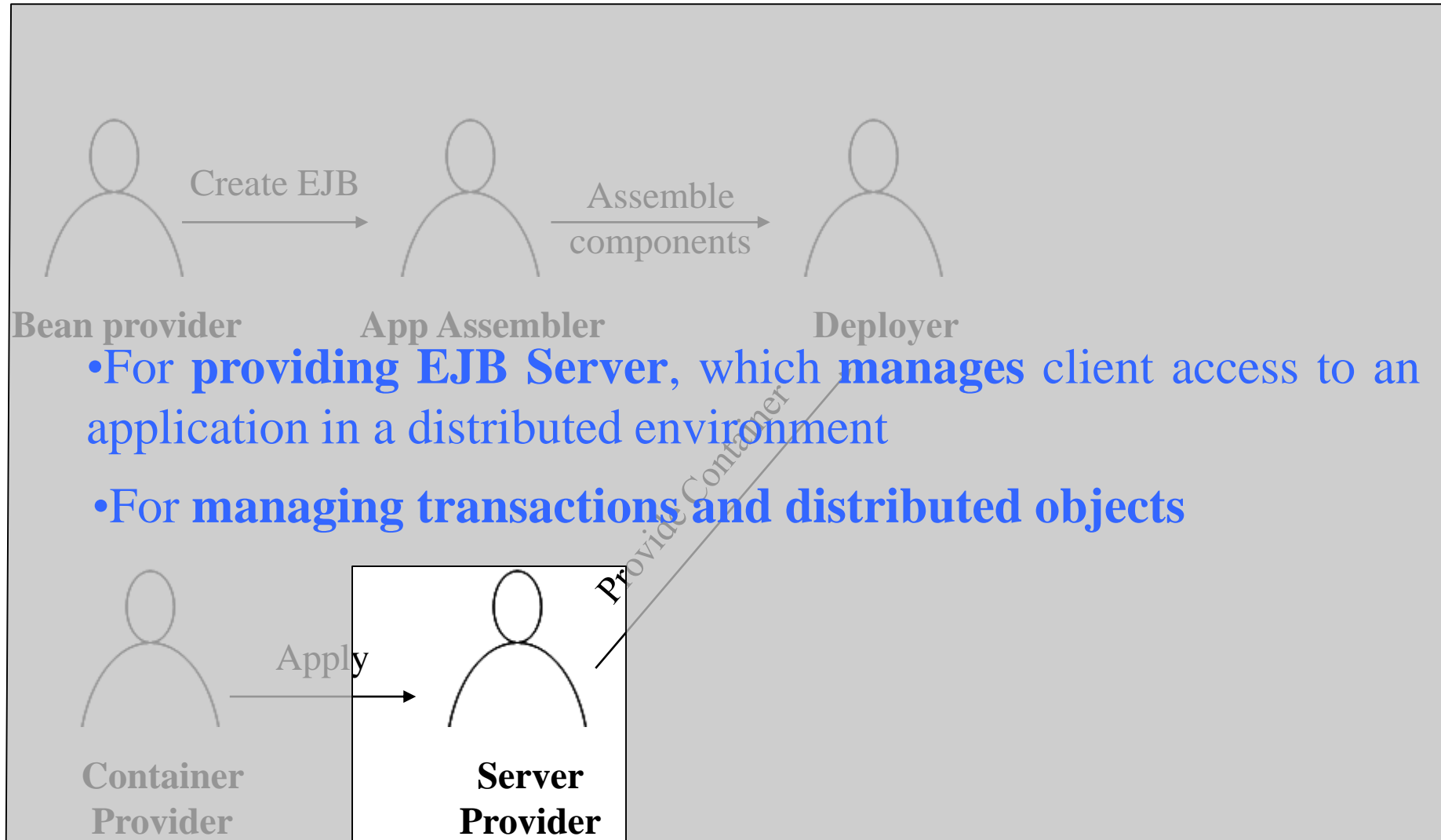
- **Providing the deployment tools** that are required by the deployer for deploying EJB components
- **Providing run-time support for beans** that are deployed on EJB Server



**Container  
Provider**

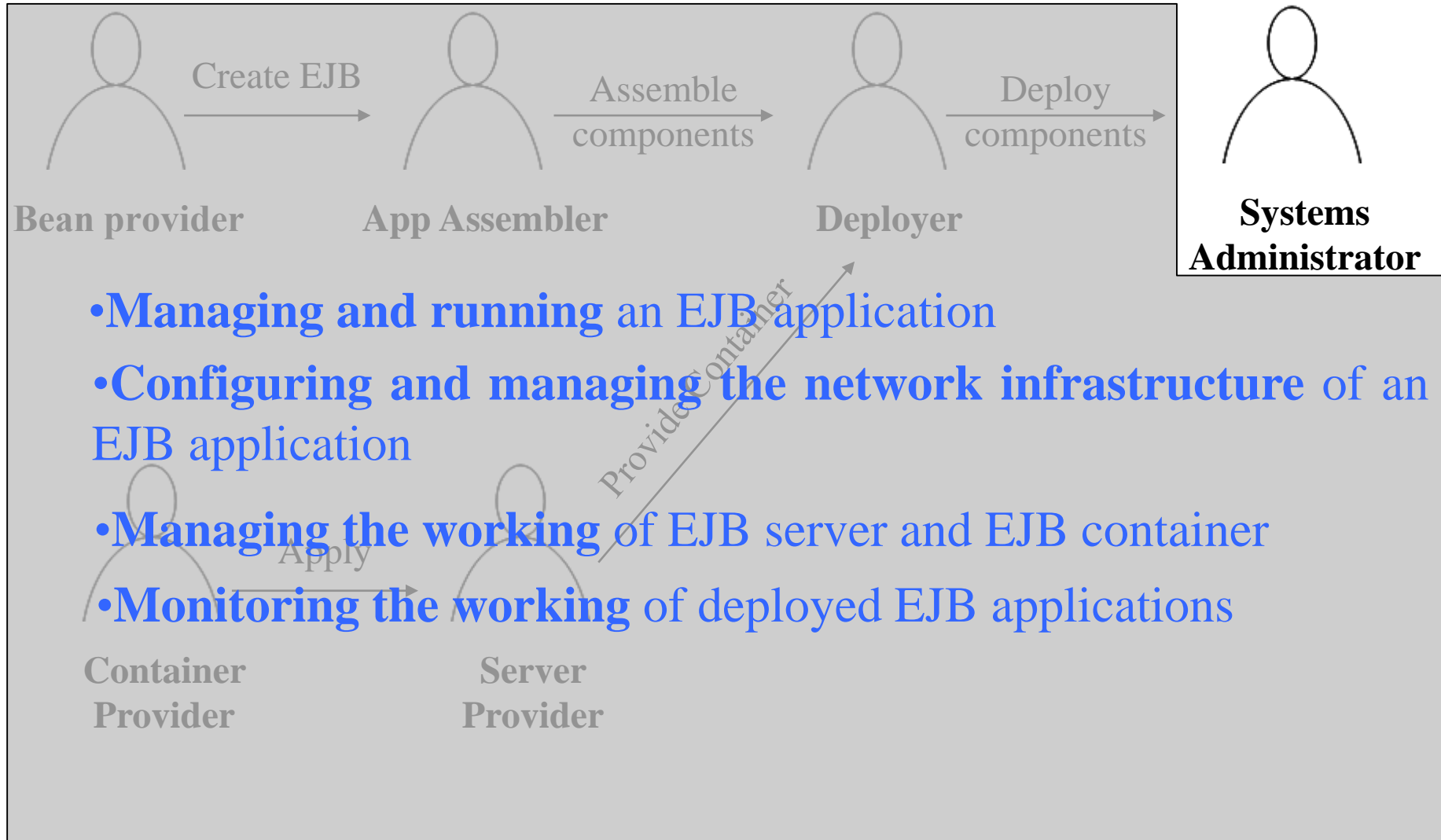
# EJB EcoSystem

## Parties involved in EJB Development



# EJB EcoSystem

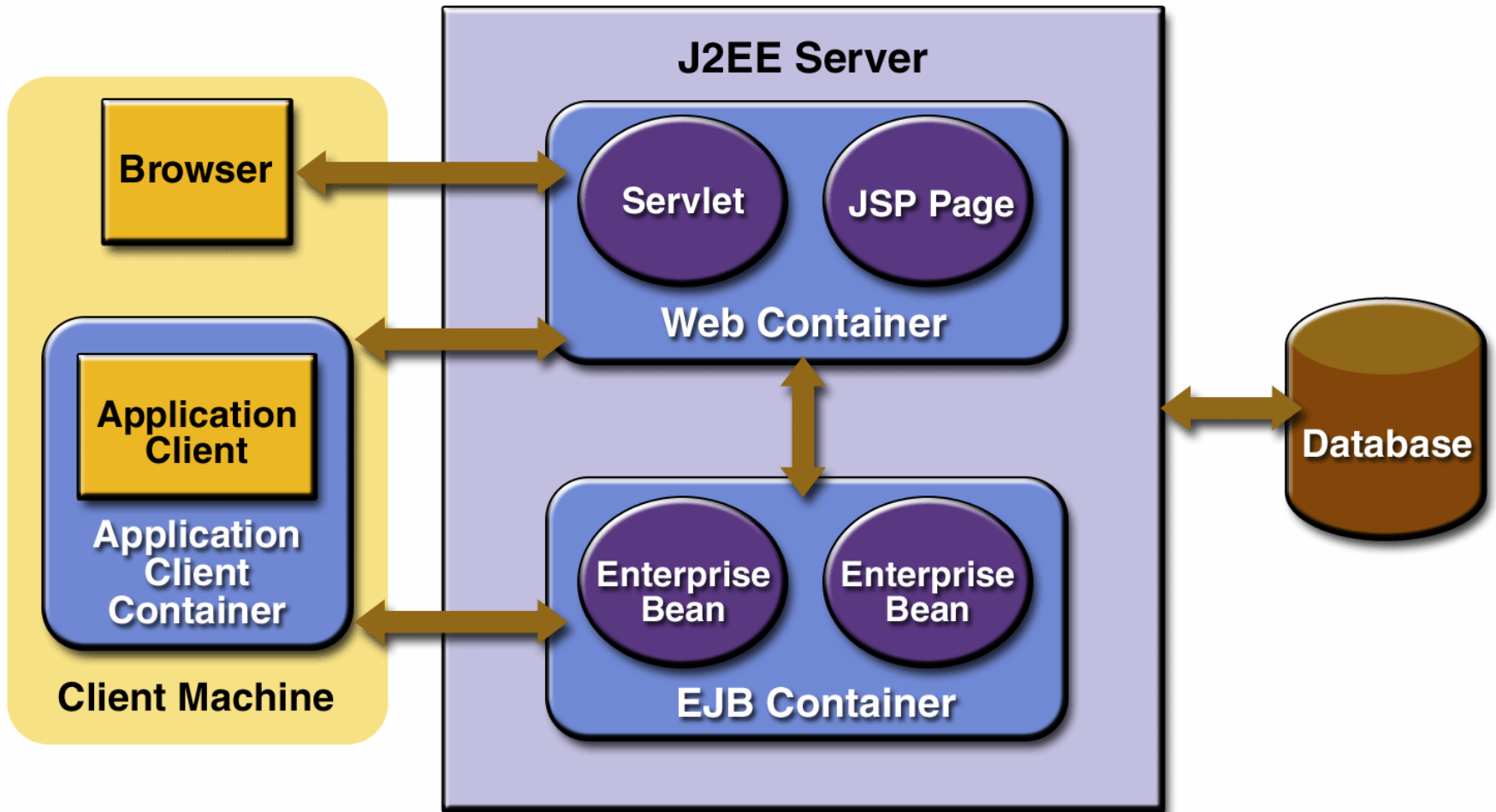
## Parties involved in EJB Development



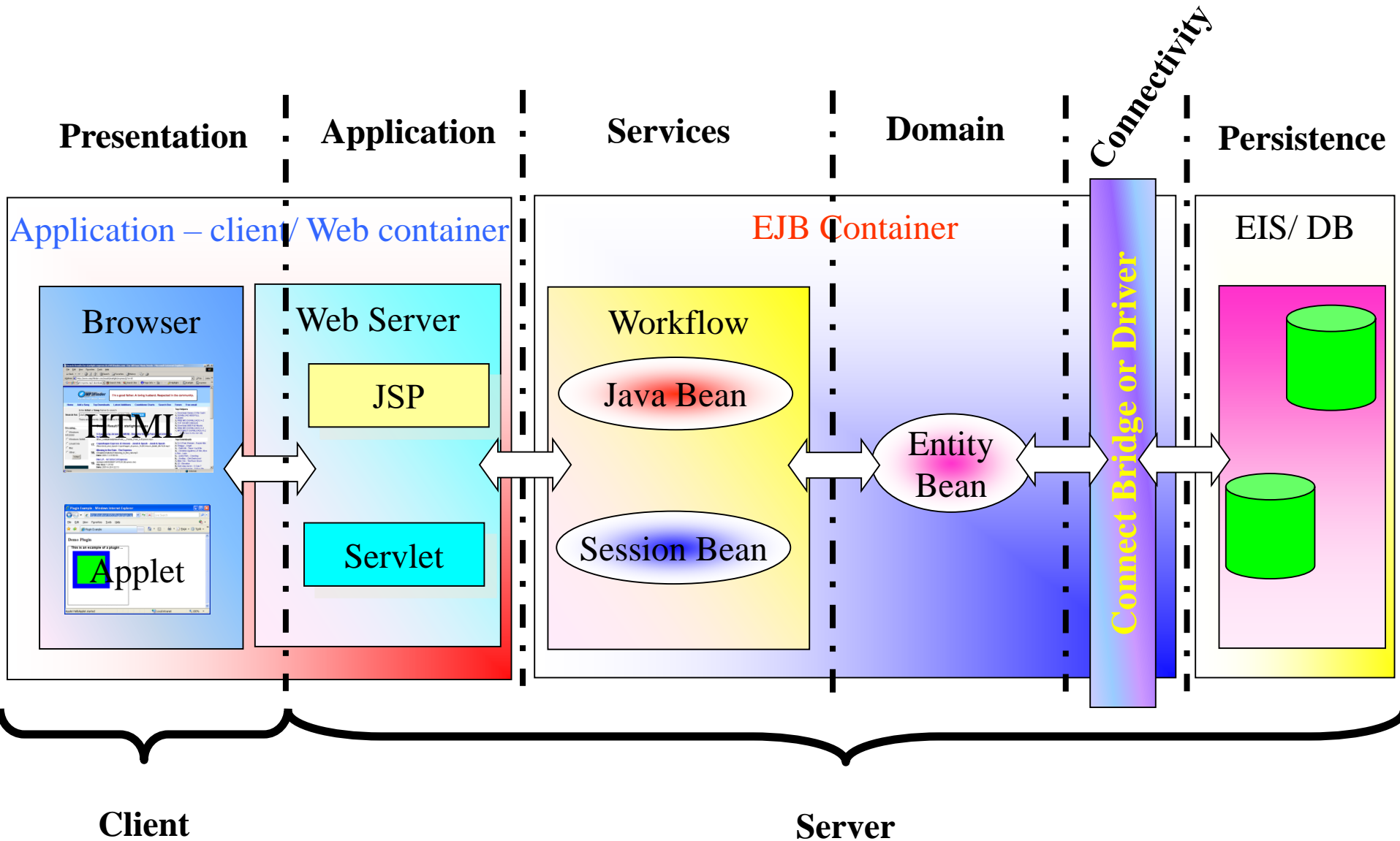


# Enterprise Java Beans

## EJB in J2EE Architecture

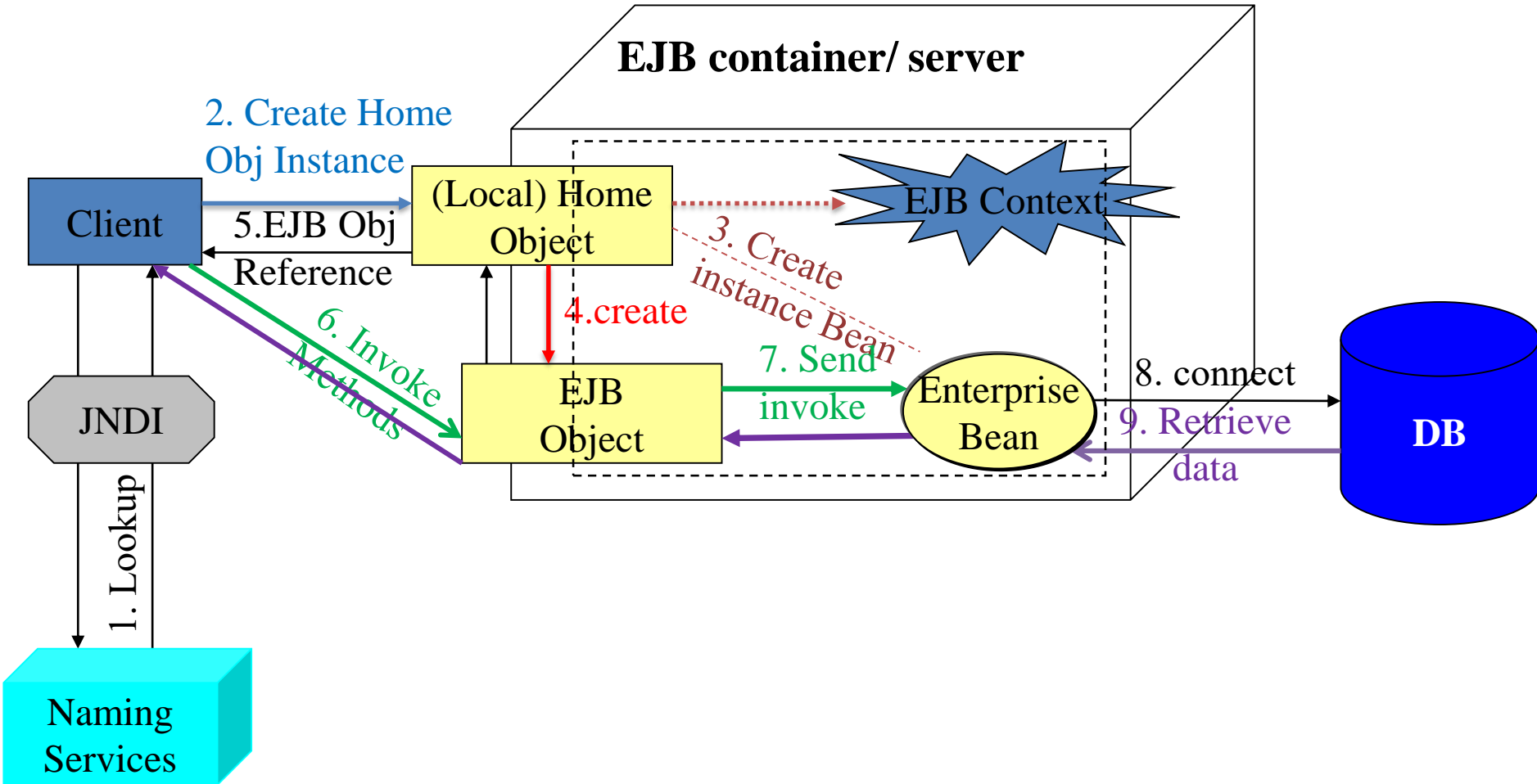


# J2EE Architecture



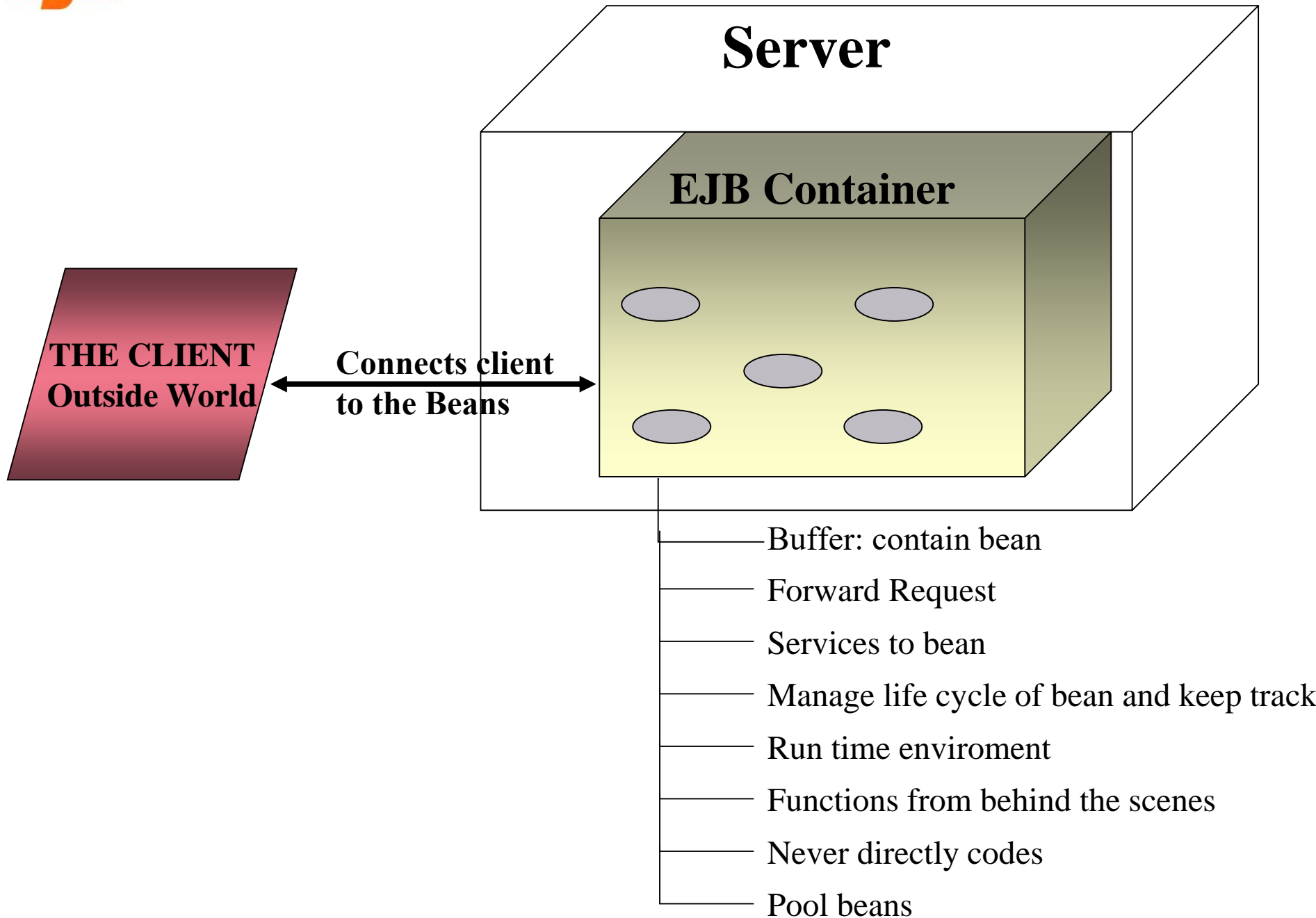
# Enterprise Java Beans

## Logical Architecture of EJB

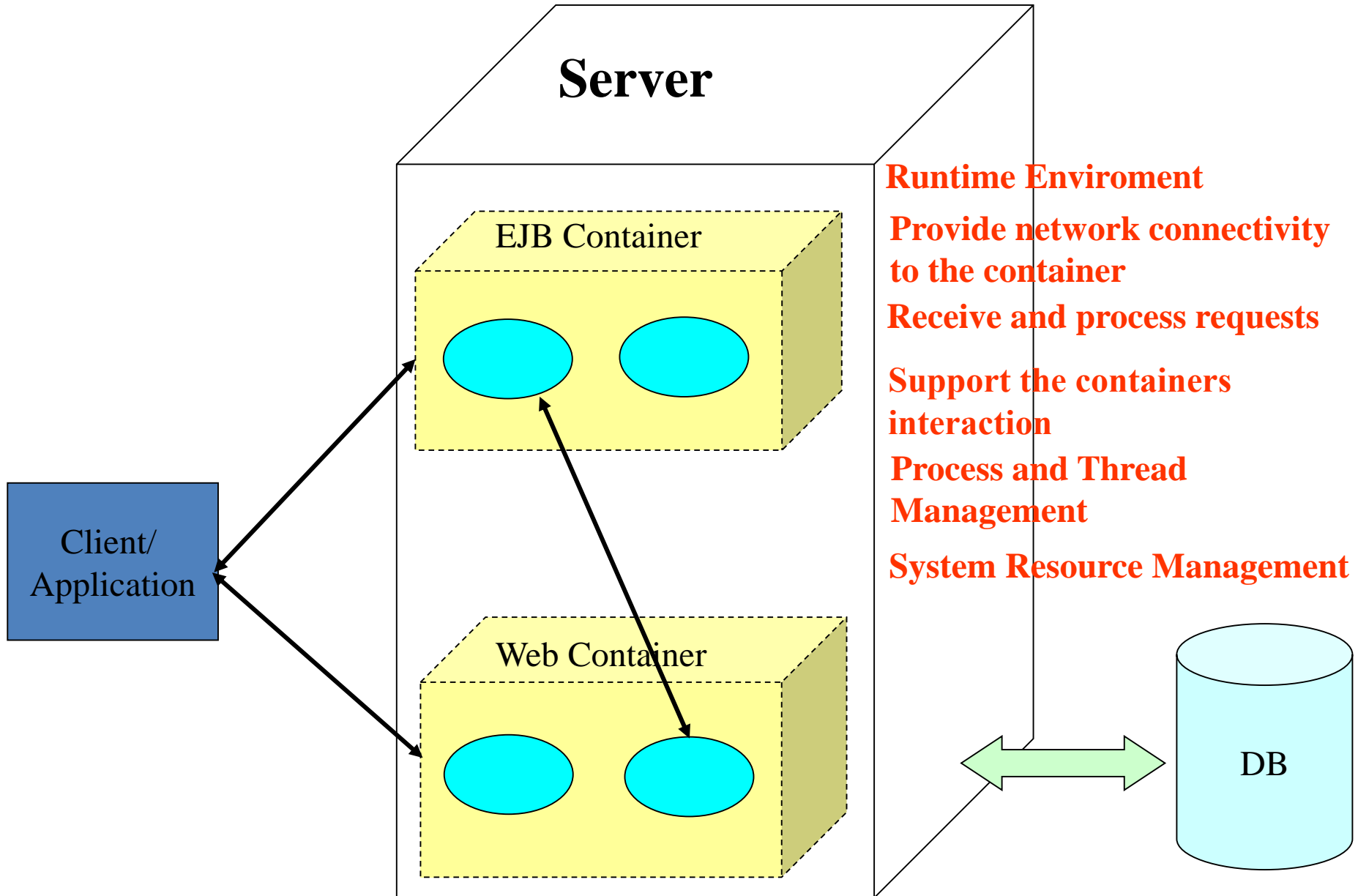


**03 tiers Architecture**

# EJB Container



# EJB Server



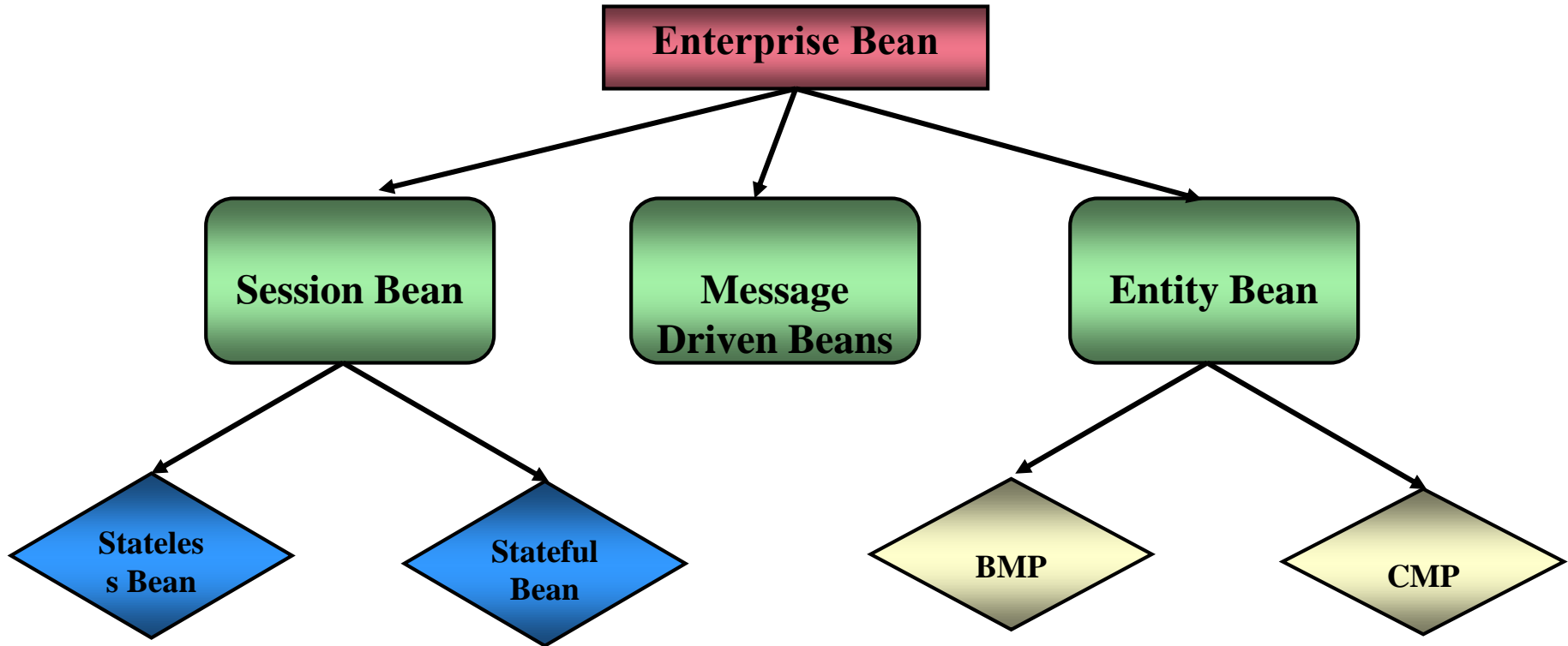
# Enterprise Java Beans

Services provided by the Container and Server

- Transaction
- Security
- Persistence
- Support for Management of multiple instances
- Remote Accessibility
- Location transparency

# Enterprise Java Beans

## Components of EJB



# Enterprise Java Beans

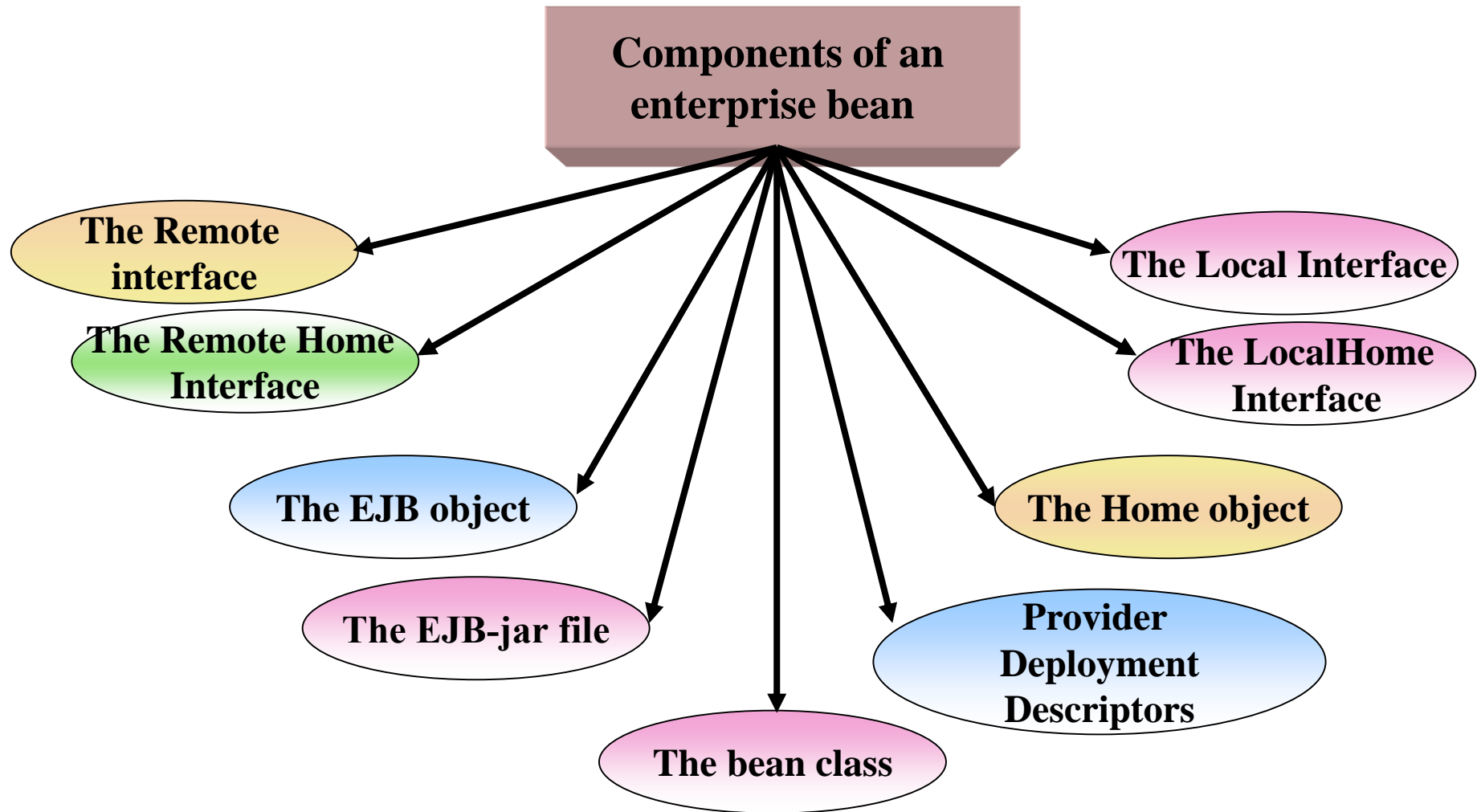
## Components Summary

<b>Feature</b>	<b>Session</b>	<b>Message-Driven</b>	<b>Entity</b>
Process	Business	Communication	DB models
Life Cycle	Short	Short	Longer
Reuseable	Lower	N/A	Higher



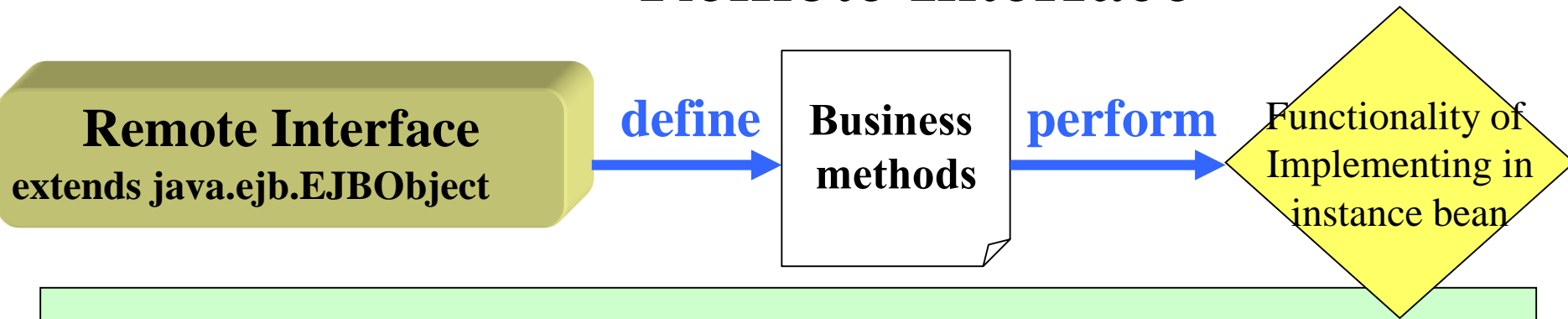
# What Constitutes an EJB?

## Components of EJB



# What Constitutes an EJB?

## Remote Interface



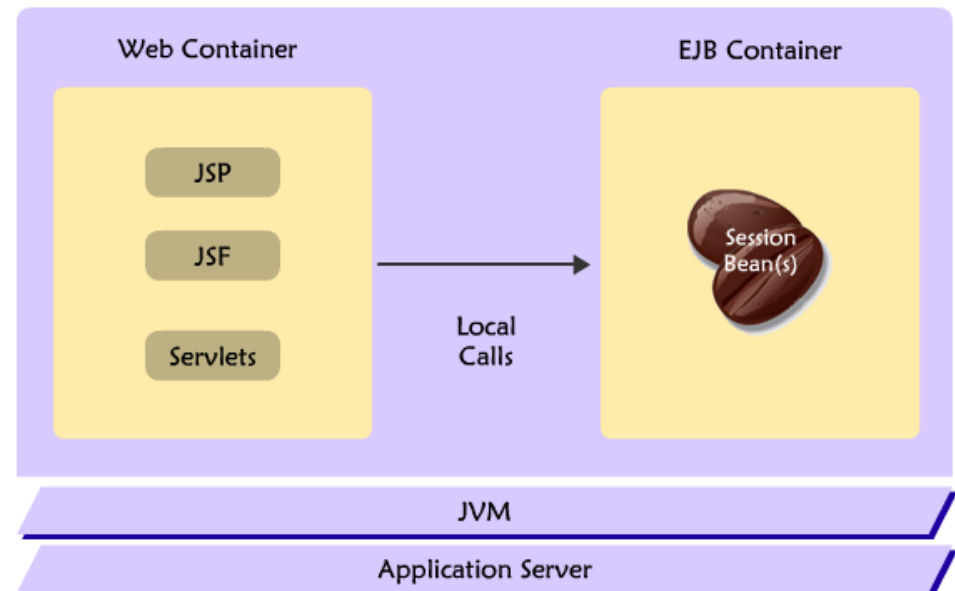
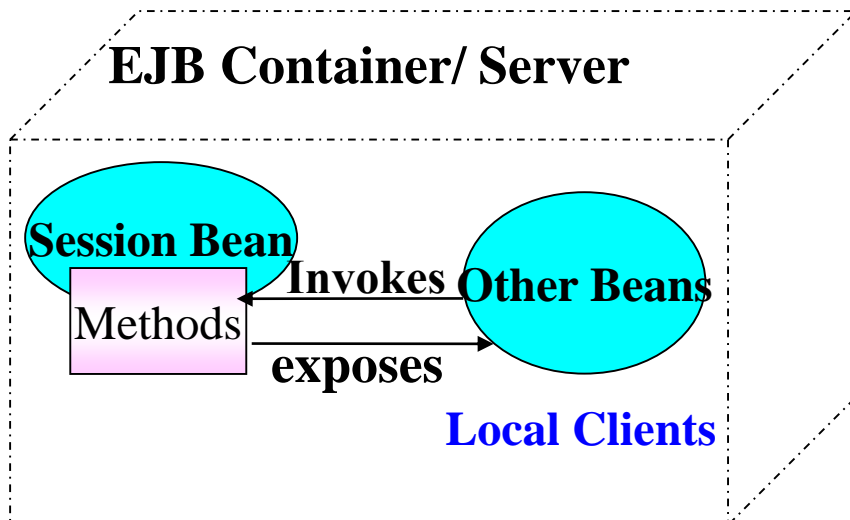
```
public interface WelcomeRemote extends javax.ejb.EJBObject {  
    public String welcome() throws java.rmi.RemoteException;  
    ...  
}
```

•Note: System level operations such as persistence, security and concurrency are not included in remote interface.

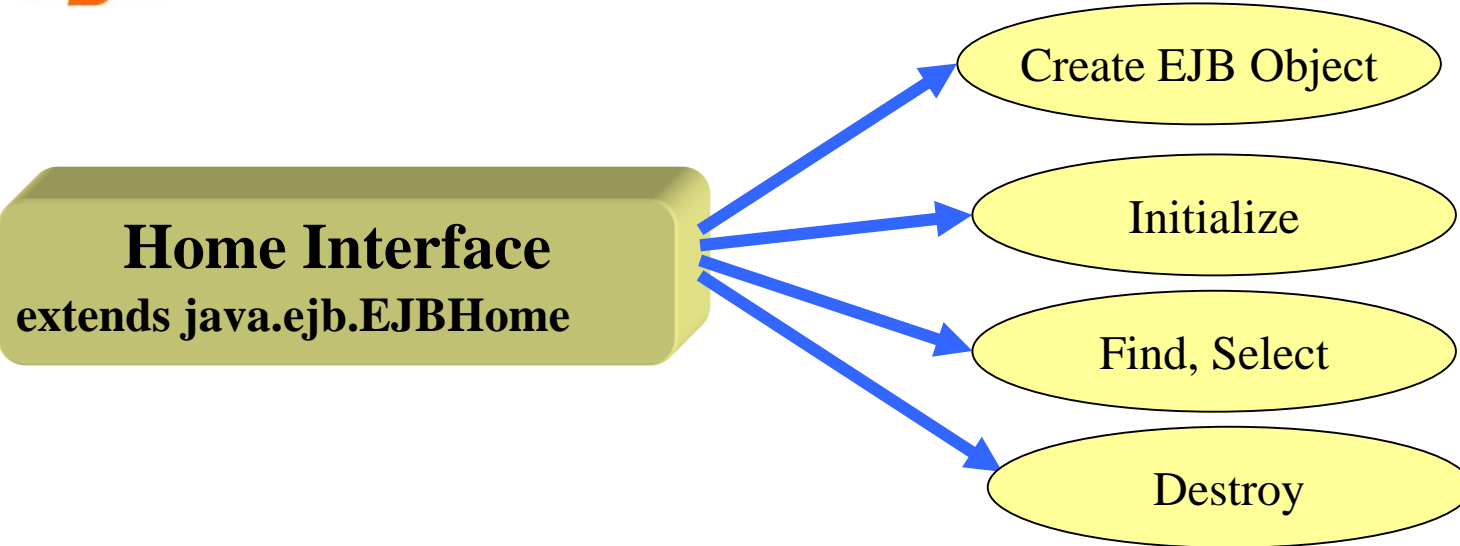
# What Constitutes an EJB?

## Local Interface

```
public interface WelcomeLocal extends javax.ejb.EJBLocalObject {
    public String welcome();
    ...
}
```



# Home Interfaces



```
public interface WelcomeRemoteHome extends java.ejb.EJBHome {  
    public WelcomeRemote create() throws java.rmi.RemoteException;  
    public WelcomeRemote findByPrimaryKey() ...;  
    ...  
}
```

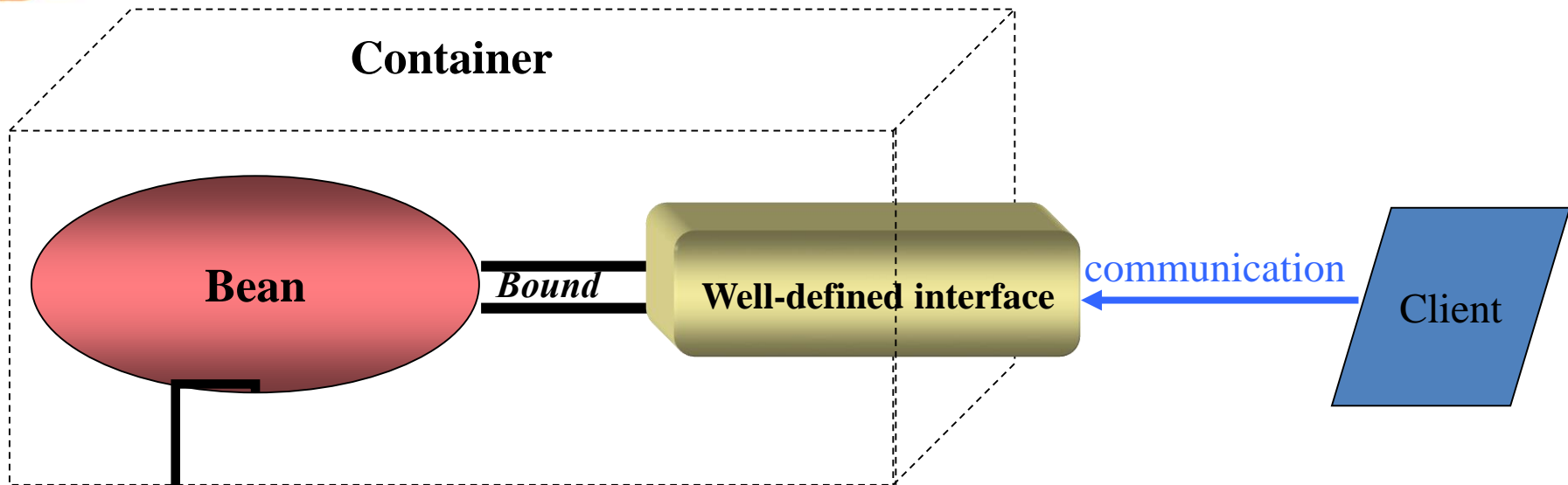
# What Constitutes an EJB?

## Local Home Objects

- Standard Java interface which allows the beans to expose its methods to other bean **reside within the same container** (local clients)
- **Eliminate the overhead** of the remote method call (java.rmi.RemoteException)
- Use pass by reference semantics (speed up in processing and efficiency)
- extends javax.ejb.EJBLocalHome
- **Notes:** LocalObject is used as Return Values

```
public interface WelcomeLocalHome extends javax.ejb.EJBLocalHome {  
    public WelcomeLocal create();  
    public WelcomeLocal findByPrimaryKey();  
    ...  
}
```

# Bean Class



- Implements defined method from Component Interface (Remote and Local)

- Override default Bean class

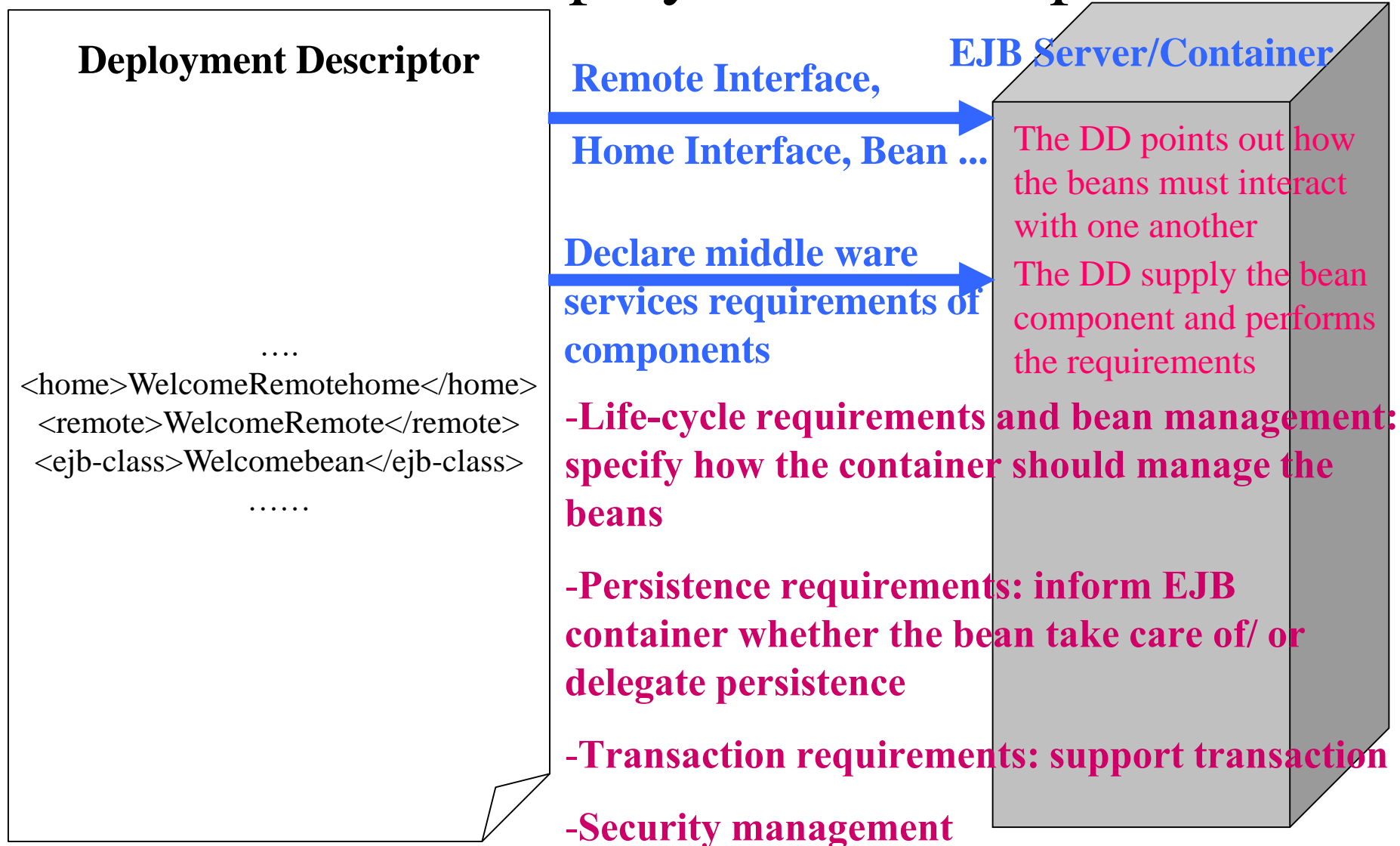
- These methods are then called by container manage bean and keep the bean informed of important events

- EJB can share the properties of the serializable objects because the `javax.ejb.EnterpriseBean` extends `Serializable`

- Once the interface `javax.ejb.EnterpriseBean` is implemented, the bean class is confirmed

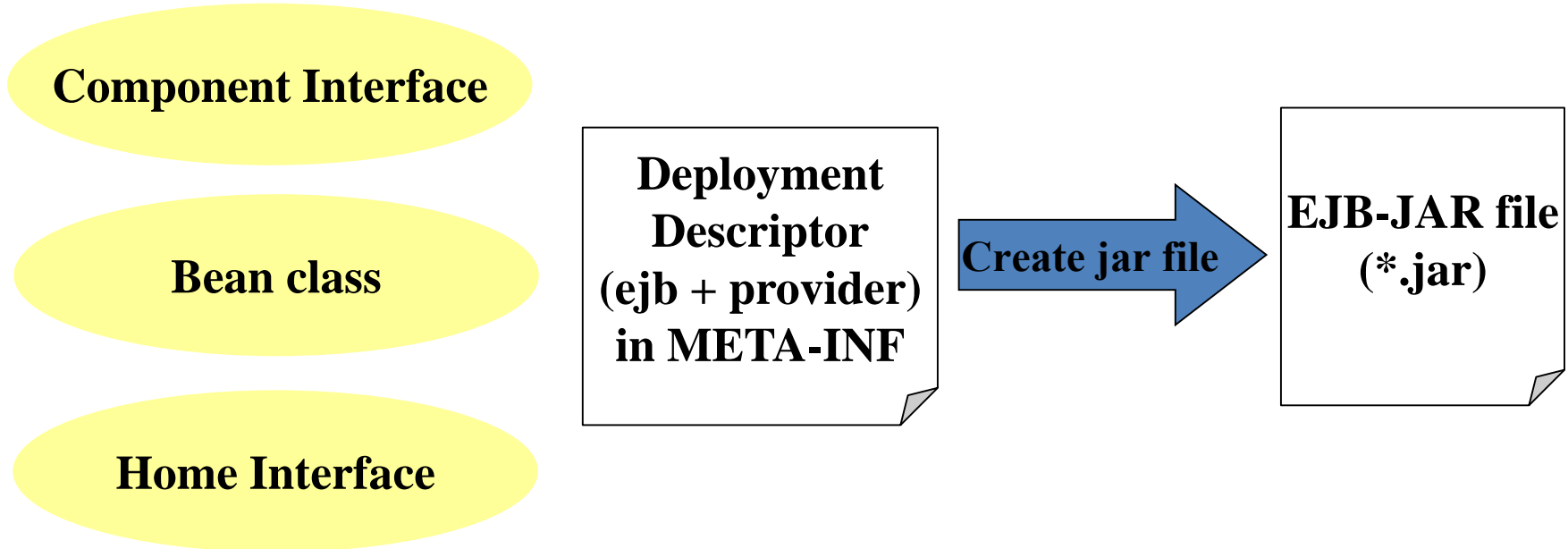
# What Constitutes an EJB?

## Deployment Descriptors



# What Constitutes an EJB?

## EJB-JAR file



- EJB container **decompress, read and extract** information contained in the EJB-JAR file.
- **Generation** of the EJB object and the home objects, and the bean. (**deployer**)



# JNDI

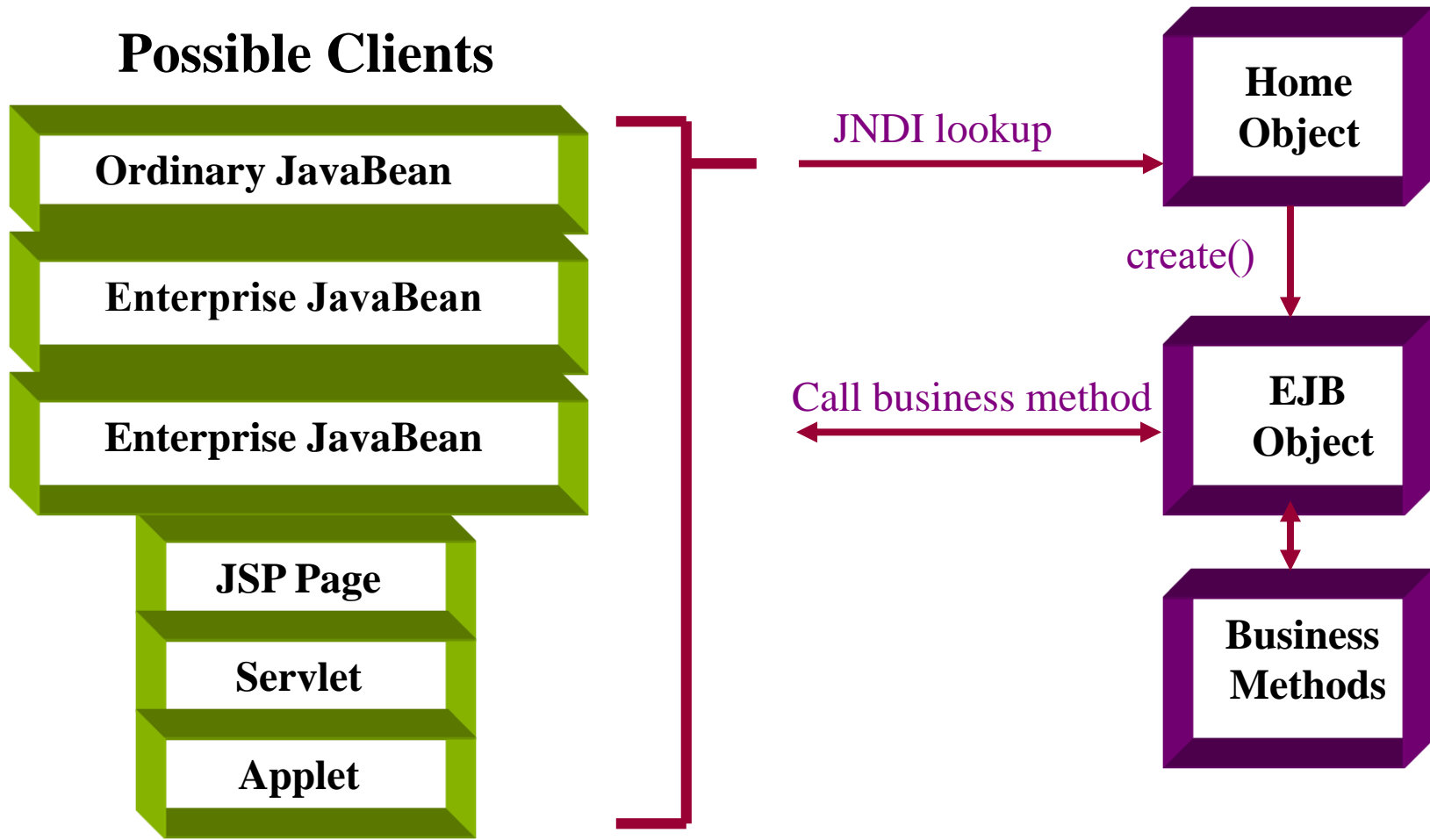
## API and Libraries

- The **Context** is represented by the **javax.naming.Context** interface that has the necessary methods to put objects into the naming service, and also to locate them.
- The starting point is called an **InitialContext**, represented by **javax.naming.InitialContext** interface
- The references in JNDI are represented by **javax.naming.Reference** interface
  - The **lookup()** method retrieves the object bound to the name and throws a **javax.naming.NamingException**, if a naming exception is encountered  
**<context\_variable>.lookup("object\_name")**
- The remote calls in EJB make use of **RMI-IIOP** (Remote Method Invocation-Internet Inter-Orb Protocol) which **does not support explicit casting of the EJB object obtained from the remote object to a local object**. Instead, Java a RMI-IIOP provides a mechanism to narrow the Object you have received from your lookup to the appropriate type by using **the javax.rmi.PortableRemoteObject class & its narrow() method**
  - The method **narrow()** of which parameters **narrowFrom** is the object that has to be narrowed and **narrowTo** is the desired type. It returns the object which is cast to the desired type and **throws ClassCastException**, if **narrowFrom** cannot be cast to **narrowTo**
- The supported files are **jndi.jar, fscontext.jar, providerutil.jar**

# EJB Implementation

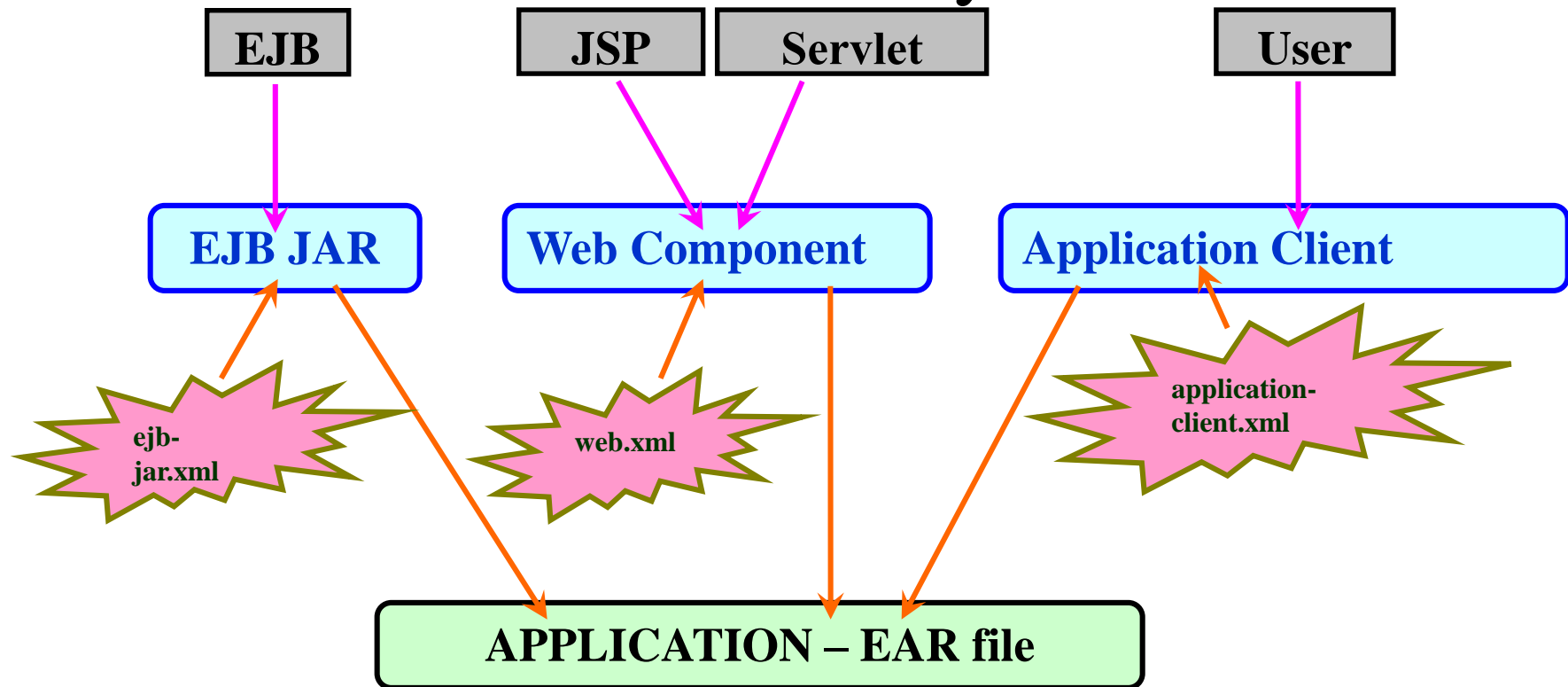
## Accessing EJB from Client Side

### Possible Clients



# EJB Implementation

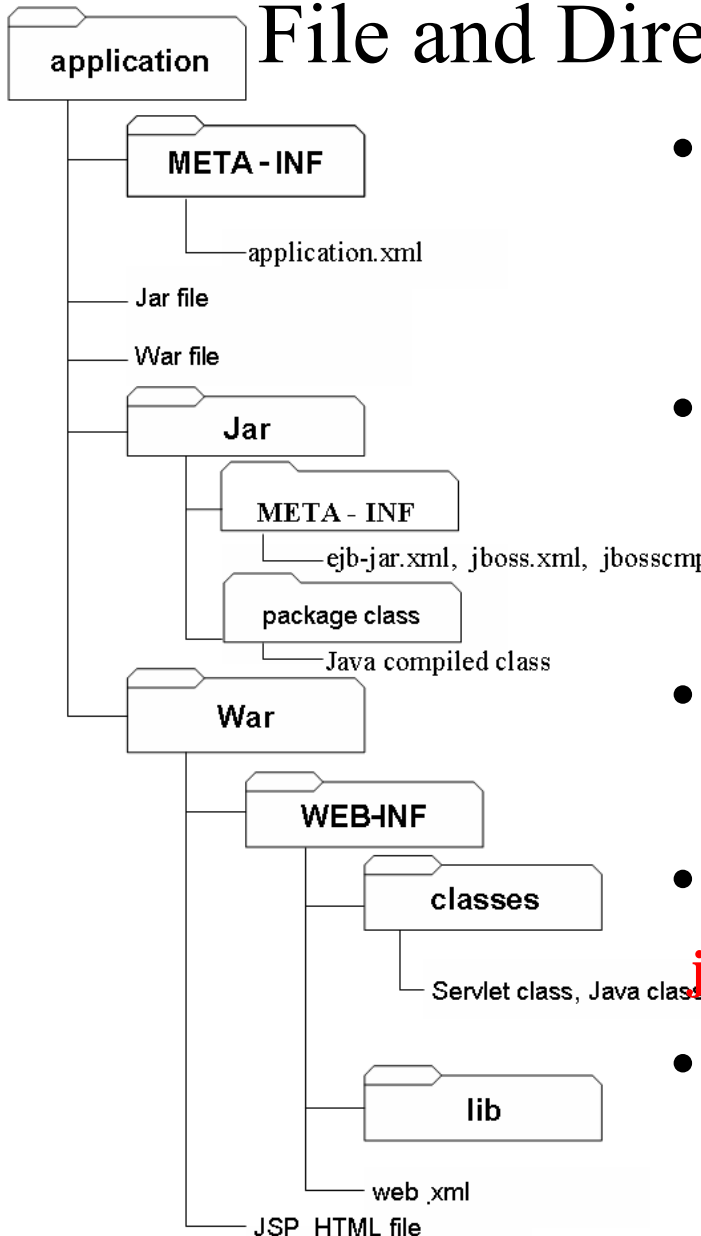
## File and Directory Structure



- **Modules of J2EE Platform (must contain at least one J2EE module)**
  - **EJB modules** cover the data layer and part of the logic layer
  - **Web application** modules cover part of the **logic layer** and the **presentation layer** (web application)
  - **Application client** modules cover part of **logic layer**, and the **presentation layer** (desktop application)

# EJB Implementation

## File and Directory Structure of Enterprise Apps



- This structure is deployed at **JBOSS\_HOME\server\default\deploy** directory
- This structure is name with the extension **.ear** (include jar and war)
- Make deploy ejb file (\*.jar)  
**jar cvf user.jar [package/]\*.class META-INF/\***
- Make deploy web file (\*.war)  
**jar cvf user.war [dir/]\*.jsp [dir/]\*.html WEB-INF/\***
- Make deploy enterprise application file (\*.ear)  
**jar cvf user.ear user.jar user.war META-INF/\***

# EJB Implementation

## Additional – Configure Jboss 6.1.0 Final

- **Go to JBOSS\_HOME/bin**
- **Open run.bat**
- **Edit following content**
  - Search the line “set JAVA\_OPTS=-Dprogram.name=%PROGNAME% -Dlogging.configuration=file:%DIRNAME%\logging.properties %JAVA\_OPTS%” (**line 43**)
  - **Change** the %DIRNAME% to the **absolute path** to the “bin” directory of the installed JBoss
  - **Ex:** set JAVA\_OPTS=-Dprogram.name=%PROGNAME% -Dlogging.configuration=file:“C:\Programming\jboss6.1.0\bin\logging.properties” %JAVA\_OPTS%

# EJB Implementation

## EJB Development Process

- **Requirement: JBoss 6.1.0 Final Application Server & Netbeans 7.2.1**
- **Step 1:** Creating a new EJB Module project
- **Step 2:** Creating the new corresponding bean depending on your purpose.
- **Step 3:** Building/ Modifying the business/callback methods on Beans
- **Step 4:** Mapping the JNDI to beans
- **Step 5:** Building the project to jar file
- **Step 6:** Deploying the project on Application server
- **Step 7:** Creating the client application to consume
- **Step 8:** Running the client to test the EJB

# Summary

- **How to build the simple enterprise application using EJB 2.0 with GUI as Swing or web?**
  - Logical Architecture of EJB
  - Components of EJB
  - Accessing EJB from the Client/Web Side
  - File and Directory Structure of Enterprise Applications

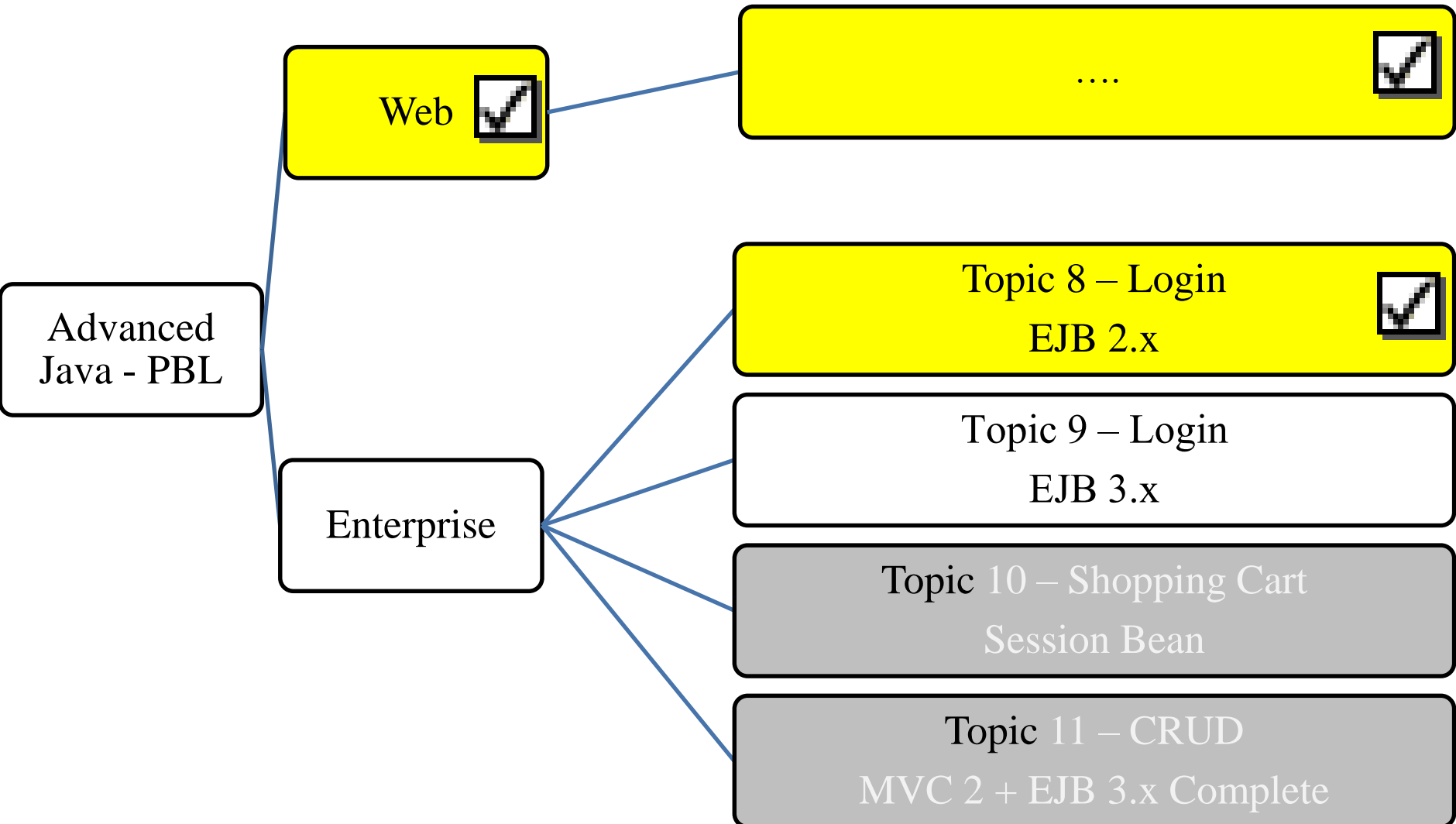
## Q&A

# Next Lecture

- **How to build the application using EJB 3**
  - Need of EJB 3
  - New Features

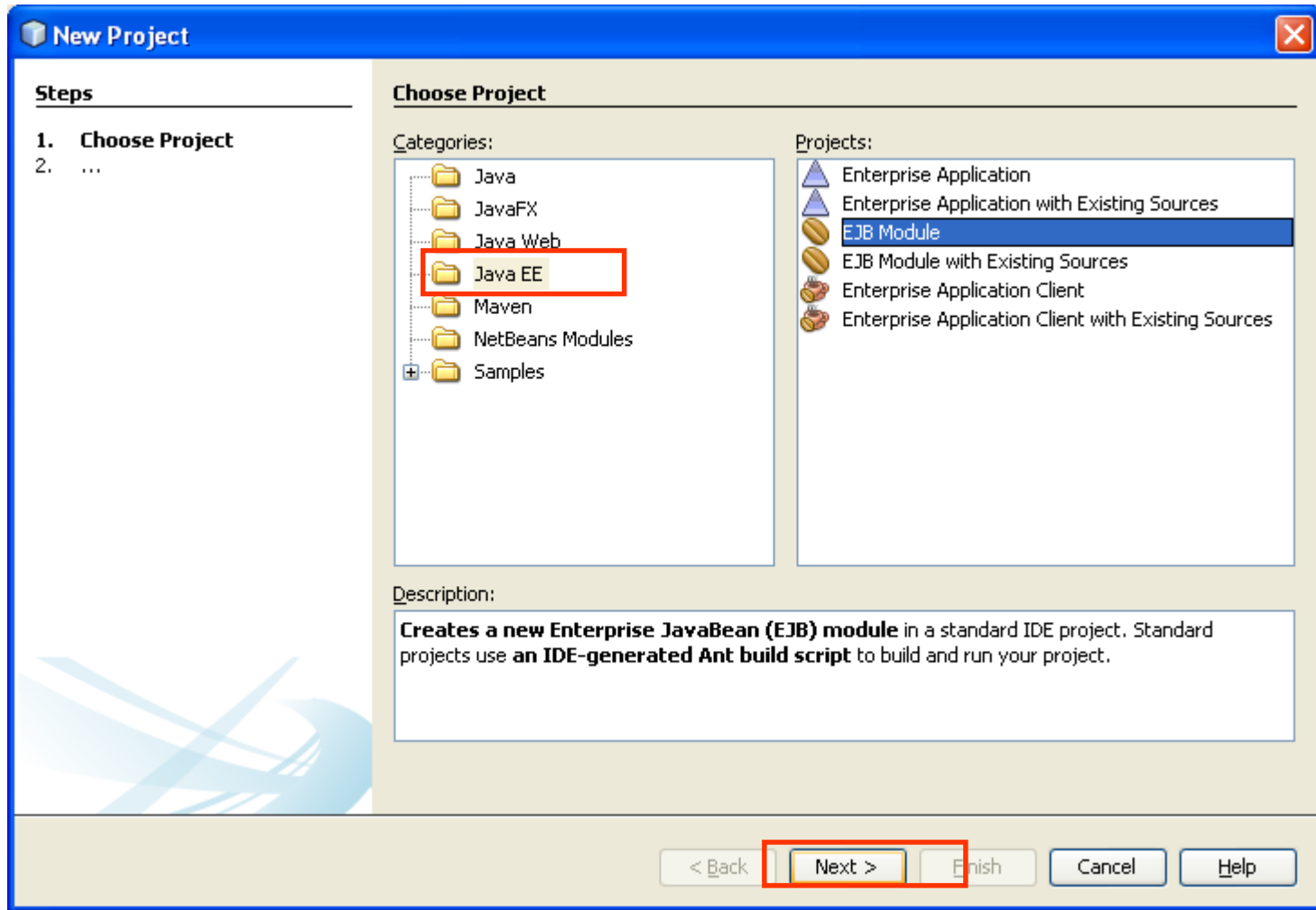


# Next Lecture



# Appendix – EJB Implementation

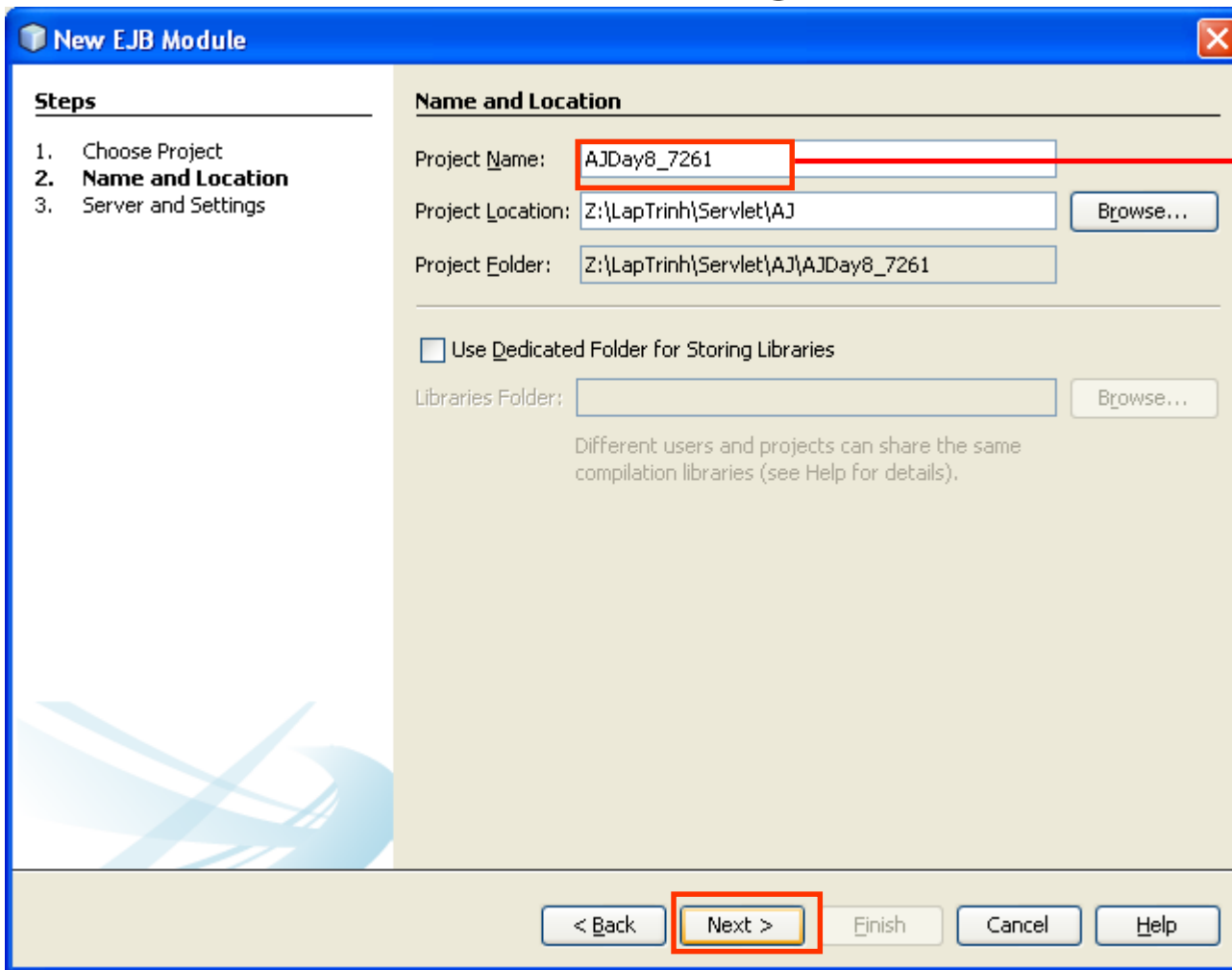
## Step 1: Creating a new EJB Module project



- Choose “**Enterprise Java Beans**” on “**Categories**”
- Then, choose “**EJB Module**” on “**Projects**”. Click **Next** button

# EJB Implementation

## Step 1: Creating a new EJB Module project



The dialog box titled "New EJB Module" has a blue title bar with a close button. On the left, a "Steps" pane lists: 1. Choose Project, 2. **Name and Location**, and 3. Server and Settings. The main area is titled "Name and Location" and contains the following fields:

- Project Name:** A text box containing "AJDay8\_7261", which is highlighted with a red rectangle.
- Project Location:** A text box containing "Z:\LapTrinh\Servlet\AJ" and a "Browse..." button.
- Project Folder:** A text box containing "Z:\LapTrinh\Servlet\AJ\AJDay8\_7261".
- ☐ **Use Dedicated Folder for Storing Libraries**
- Libraries Folder:** An empty text box and a "Browse..." button.

Below the libraries section, a note states: "Different users and projects can share the same compilation libraries (see Help for details)."

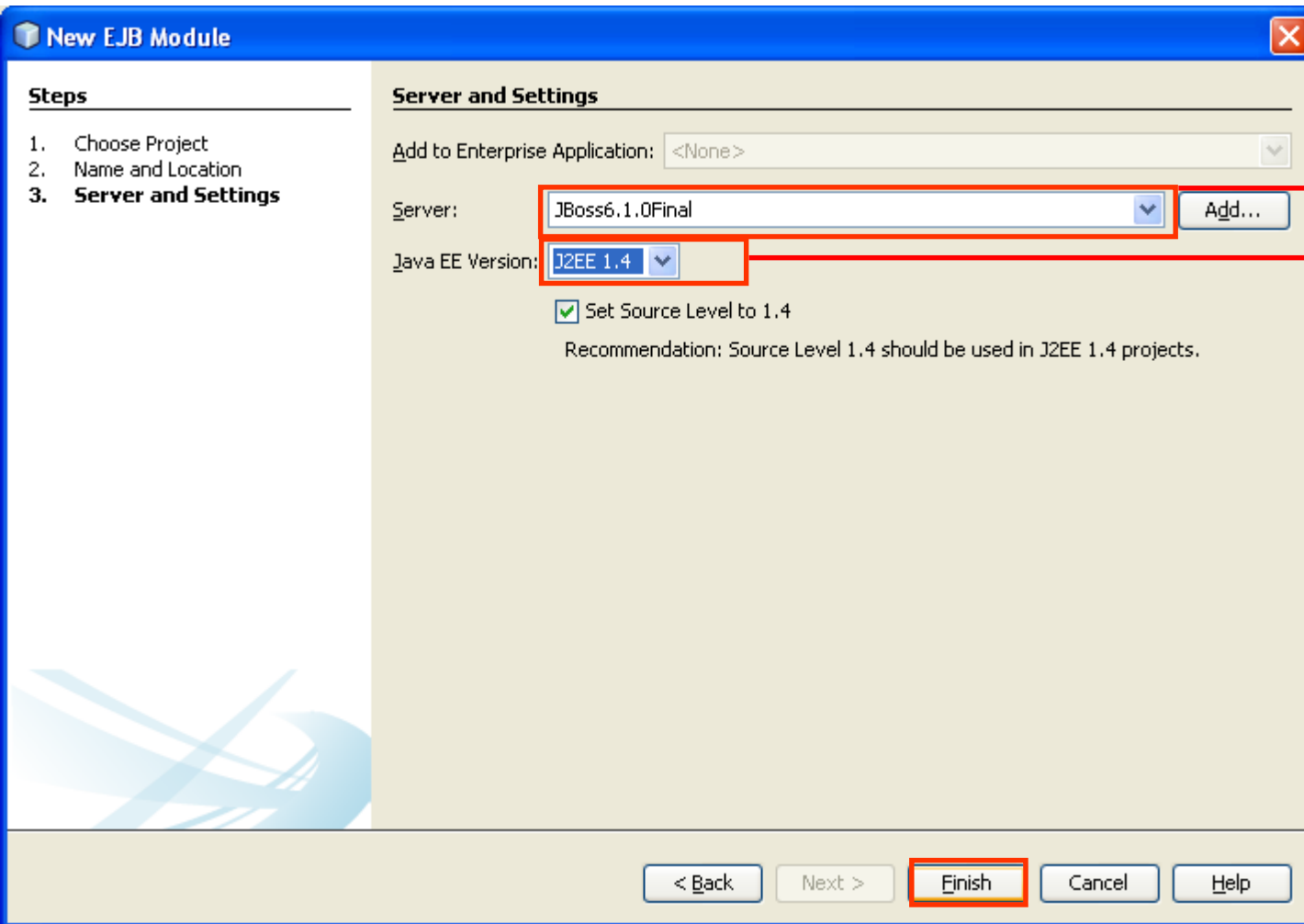
At the bottom, there are five buttons: "< Back", "Next >" (highlighted with a red rectangle), "Finish", "Cancel", and "Help".

Fill your project name

- Click **Next** button

# EJB Implementation

## Step 1: Creating a new EJB Module project



**Steps**

1. Choose Project
2. Name and Location
3. **Server and Settings**

**Server and Settings**

Add to Enterprise Application: <None>

Server: JBoss6.1.0Final Add...

Java EE Version: J2EE 1.4

☒ Set Source Level to 1.4  
Recommendation: Source Level 1.4 should be used in J2EE 1.4 projects.

< Back Next > **Finish** Cancel Help

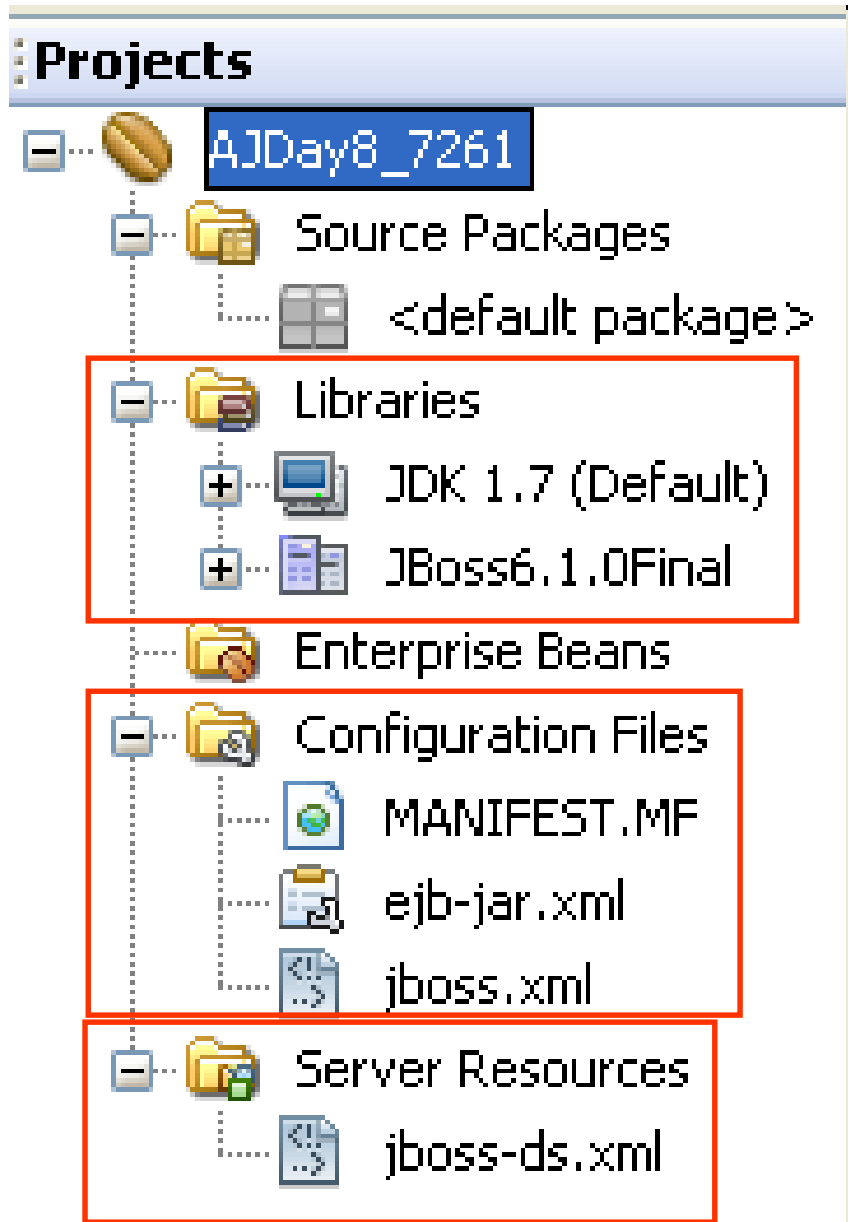
Choose the Jboss  
Server that is  
added

Choose the  
J2EE1.4

- Click **Finish** button

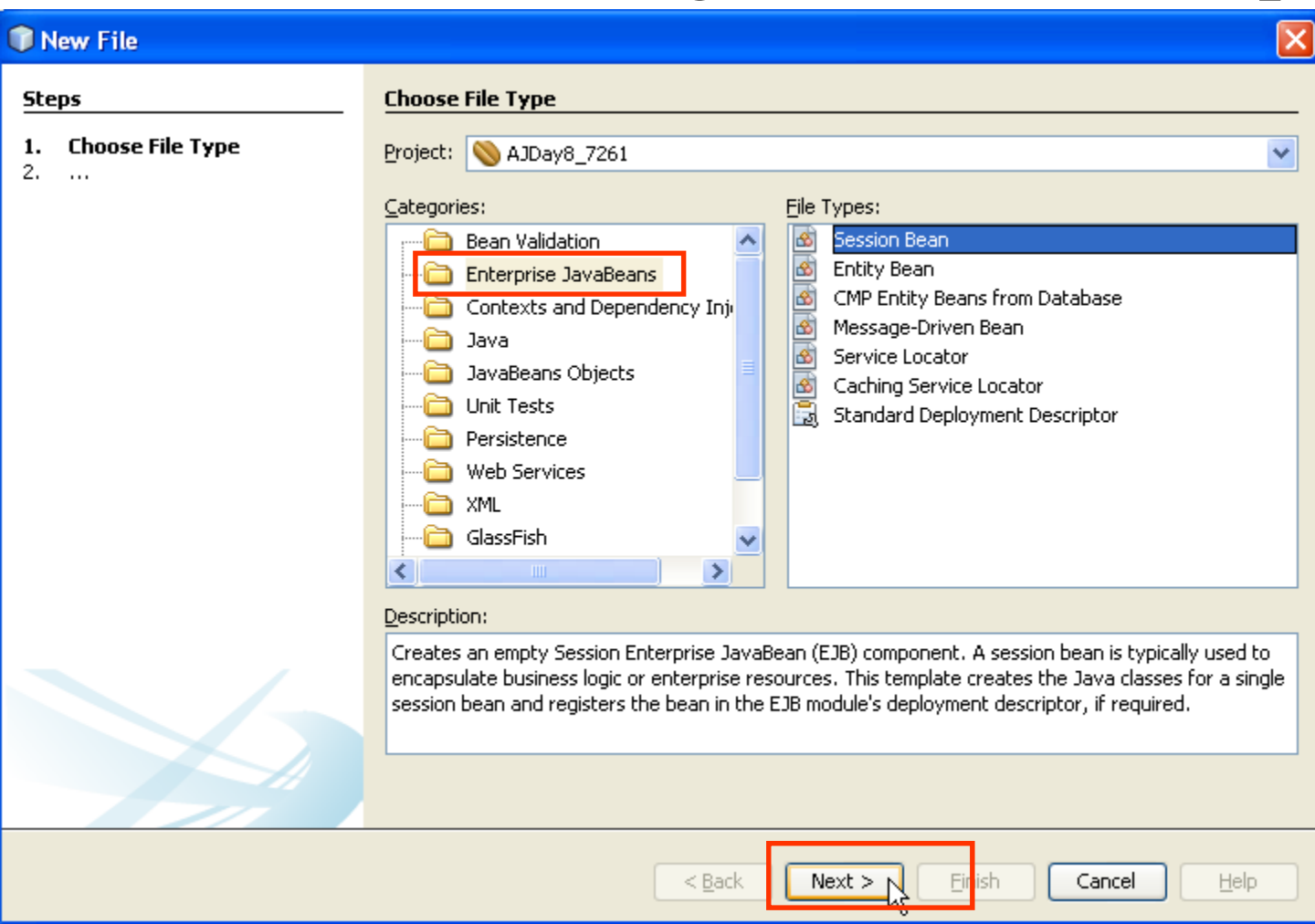
# EJB Implementation

## Step 1: Creating a new EJB Module project



# EJB Implementation

## Step 2: Creating the new corresponding bean



- Choose “**Enterprise JavaBeans**” on “**Categories**”
- Then, choose “**Session Bean**” on “**File Types**”. Click **Next** button

# EJB Implementation

## Step 2: Creating the new corresponding bean

**New Session Bean**

**Steps**

1. Choose File Type
2. **Name and Location**

**Name and Location**

EJB Name:

Project:

Location:

Package:

Session Type:

☒ Stateless

☐ Stateful

Create Interface:

☒ Local

☐ Remote

< Back   Next >   **Finish**   Cancel   Help

Fill your bean name

Fill or choose the package name

Choose stateless or stateful

Choose remote or local or both

- Click **Finish** button

# EJB Implementation

## Step 2: Creating the new corresponding bean

Projects

AJDay8\_7261

Source Packages

sample.session

CalculatorSessionBean.java

CalculatorSessionBeanLocal.java

CalculatorSessionBeanLocalHome.java

Libraries

JDK 1.7 (Default)

JBoss6.1.0Final

Enterprise Beans

CalculatorSessionBeanSB

Local Methods

create

Bean Methods

Configuration Files

MANIFEST.MF

ejb-jar.xml

jboss.xml

Server Resources

jboss-ds.xml

CalculatorSessionBean.java

```

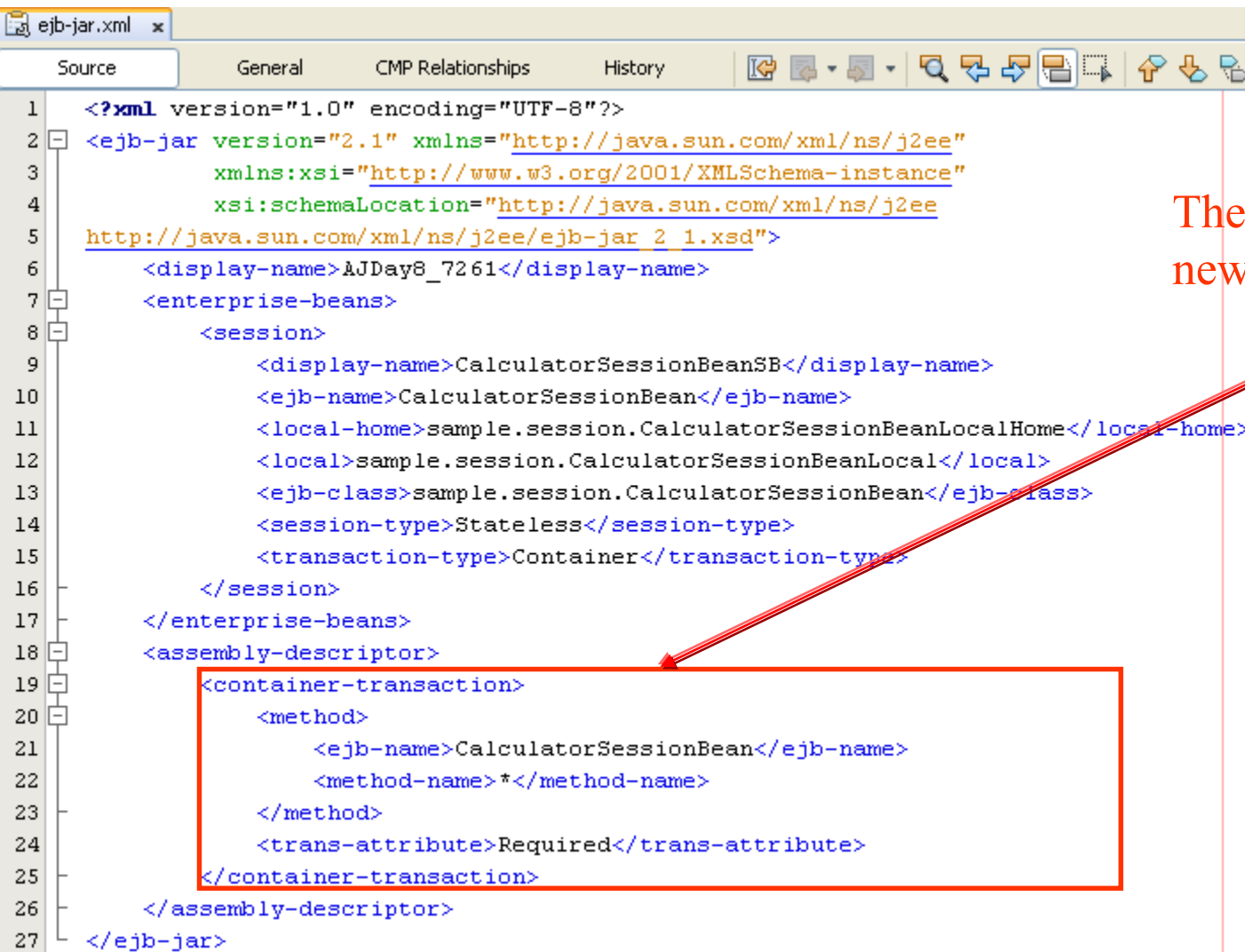
12 * @author Trong Khanh
13 */
14 public class CalculatorSessionBean implements SessionBean {
15
16     private SessionContext context;
17
18     EJB infrastructure methods. Click the + sign on the left to edit the code.;
19
20     /**
21      * See section 7.10.3 of the EJB 2.0 specification See section 7.11.3 of the
22      * EJB 2.1 specification
23      */
24     public void ejbCreate() {
25         // TODO implement ejbCreate if necessary, acquire resources
26         // This method has access to the JNDI context so resource acquisition
27         // spanning all methods can be performed here such as home interfaces
28         // and data sources.
29     }
30
31     // Add business logic below. (Right-click in editor and choose
32     // "Insert Code > Add Business Method" or "Web Service > Add Operation")
33 }
    
```

Generate automatically four callback methods:

- setSessionContext
- ejbActivate
- ejbPassivate
- ejbRemove



## Step 2: Creating the new corresponding bean

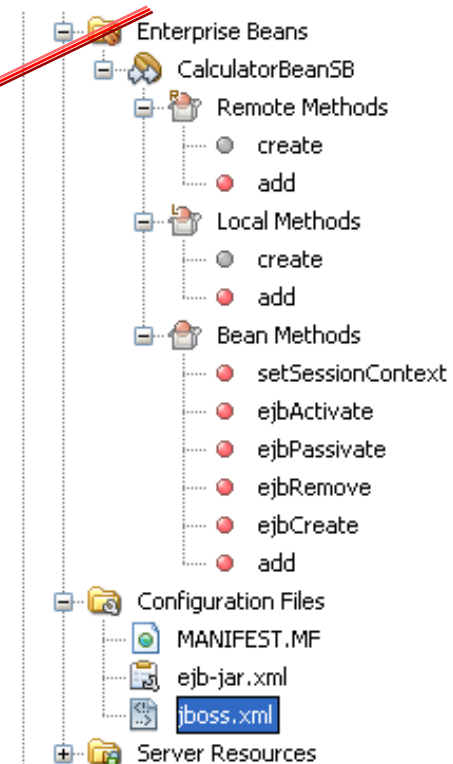


```

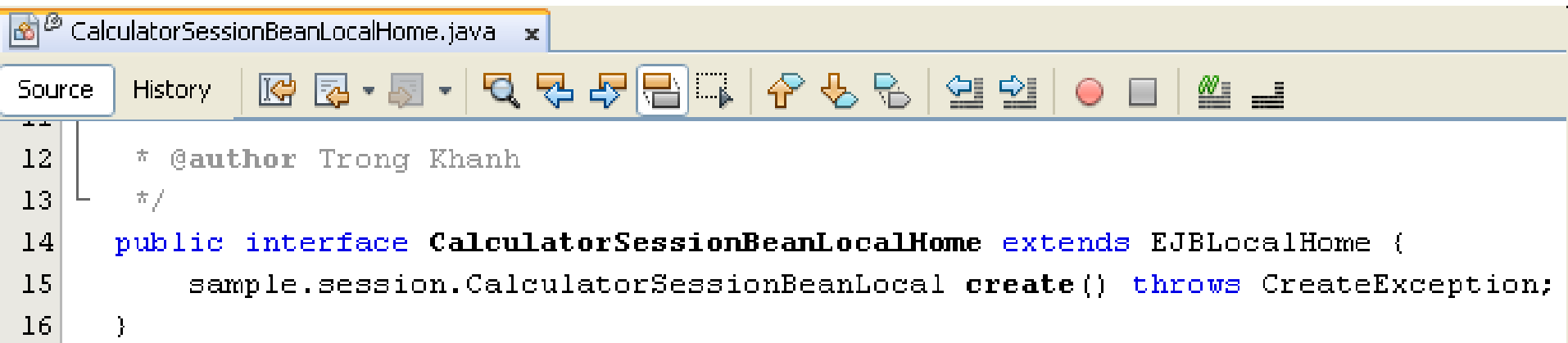
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ejb-jar version="2.1" xmlns="http://java.sun.com/xml/ns/j2ee"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
5      http://java.sun.com/xml/ns/j2ee/ejb-jar 2.1.xsd">
6      <display-name>AJDay8_7261</display-name>
7      <enterprise-beans>
8          <session>
9              <display-name>CalculatorSessionBeanSB</display-name>
10             <ejb-name>CalculatorSessionBean</ejb-name>
11             <local-home>sample.session.CalculatorSessionBeanLocalHome</local-home>
12             <local>sample.session.CalculatorSessionBeanLocal</local>
13             <ejb-class>sample.session.CalculatorSessionBean</ejb-class>
14             <session-type>Stateless</session-type>
15             <transaction-type>Container</transaction-type>
16          </session>
17      </enterprise-beans>
18      <assembly-descriptor>
19          <container-transaction>
20              <method>
21                  <ejb-name>CalculatorSessionBean</ejb-name>
22                  <method-name>*</method-name>
23              </method>
24              <trans-attribute>Required</trans-attribute>
25          </container-transaction>
26      </assembly-descriptor>
27  </ejb-jar>

```

The container creates a new transaction



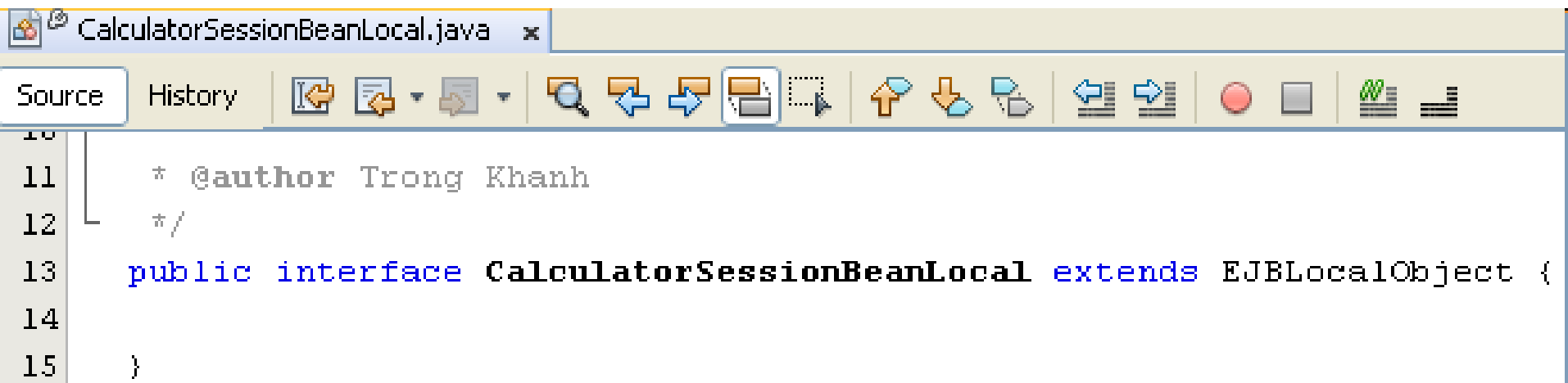
## Step 2: Creating the new corresponding bean



CalculatorSessionBeanLocalHome.java

```

12  * @author Trong Khanh
13  */
14  public interface CalculatorSessionBeanLocalHome extends EJBLocalHome {
15      sample.session.CalculatorSessionBeanLocal create() throws CreateException;
16  }
  
```



CalculatorSessionBeanLocal.java

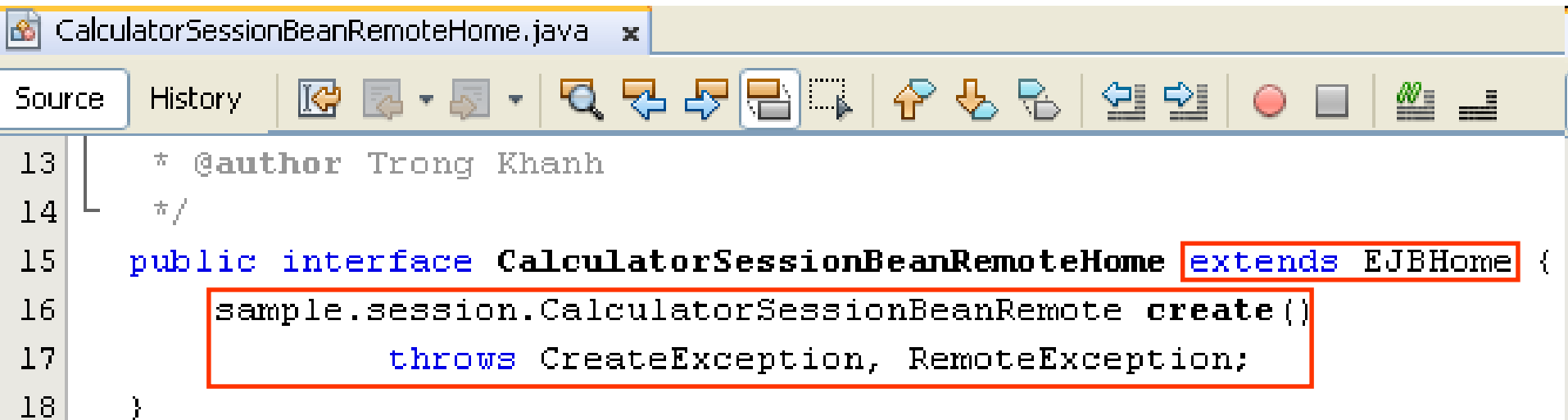
```

11  * @author Trong Khanh
12  */
13  public interface CalculatorSessionBeanLocal extends EJBLocalObject {
14
15  }
  
```

# EJB Implementation

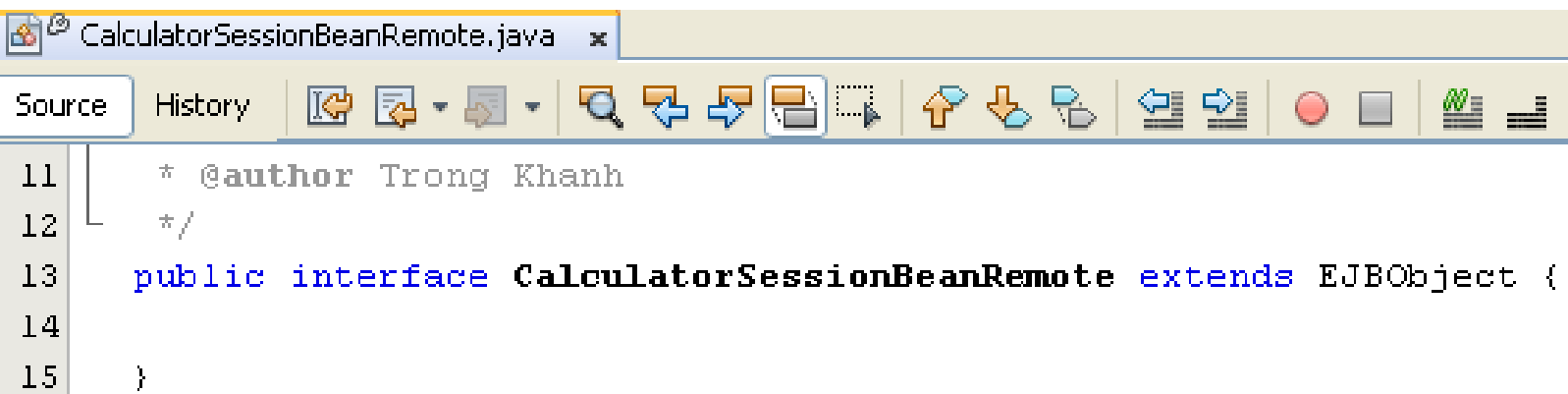
## Step 2: Creating the new corresponding bean

- Create the remote interface and the remote home interface
  - Create two java interface, then extends the EJBHome and EJBObject
  - Then, update the ejb-jar.xml file



```

CalculatorSessionBeanRemoteHome.java
Source History
13  * @author Trong Khanh
14  */
15  public interface CalculatorSessionBeanRemoteHome extends EJBHome {
16      sample.session.CalculatorSessionBeanRemote create()
17          throws CreateException, RemoteException;
18  }
  
```



```

CalculatorSessionBeanRemote.java
Source History
11  * @author Trong Khanh
12  */
13  public interface CalculatorSessionBeanRemote extends EJBObject {
14
15  }
  
```

# EJB Implementation

## Step 2: Creating the new corresponding bean

```

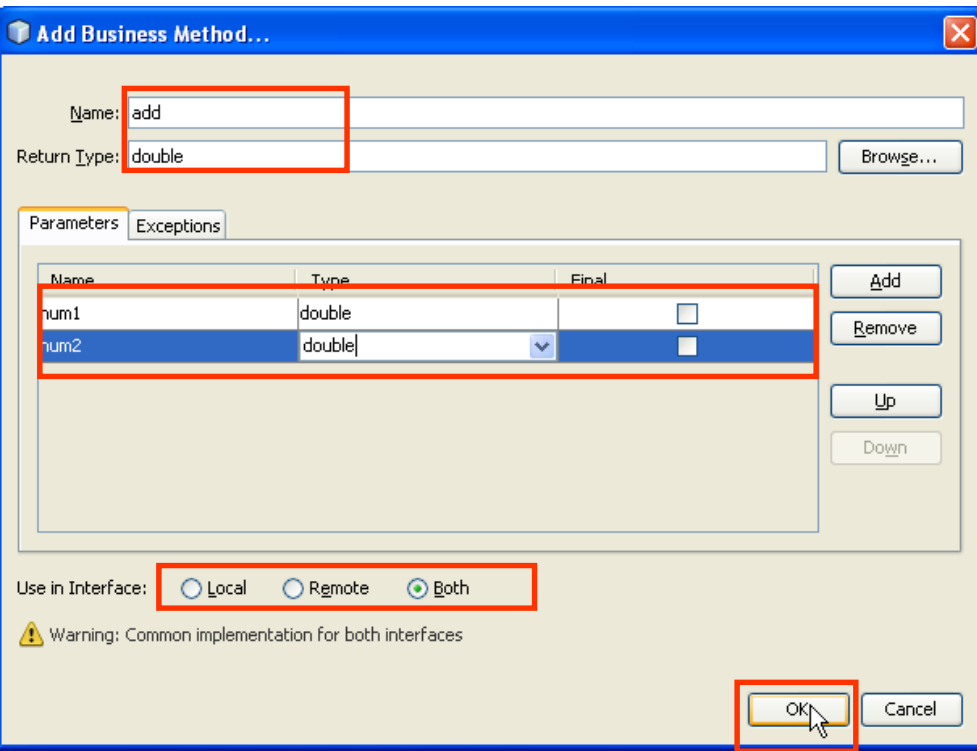
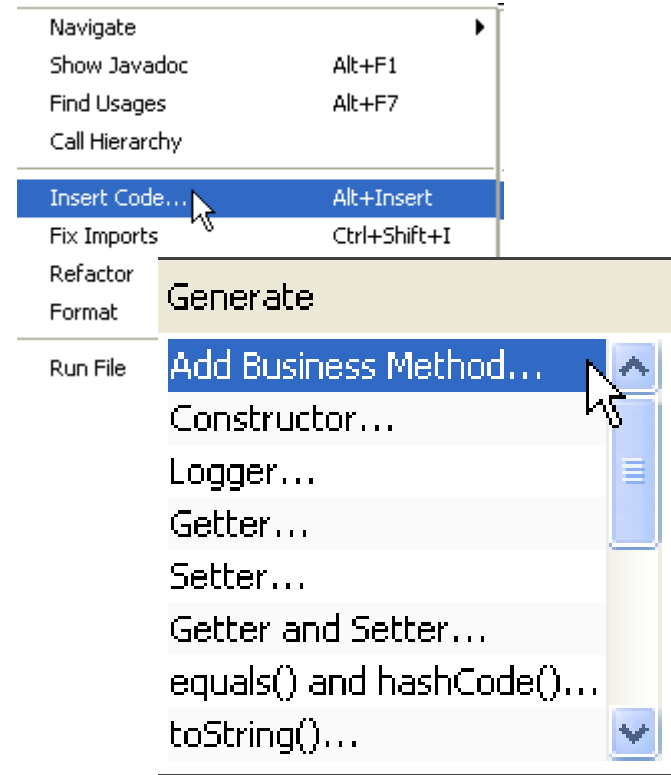
ejb-jar.xml x
Source General CMP Relationships History
1 <?xml version="1.0" encoding="UTF-8"?>
2 <ejb-jar version="2.1" xmlns="http://java.sun.com/xml/ns/j2ee"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
5     http://java.sun.com/xml/ns/j2ee/ejb-jar_2_1.xsd">
6     <display-name>AJDay8_7261</display-name>
7     <enterprise-beans>
8         <session>
9             <display-name>CalculatorSessionBeanSB</display-name>
10            <ejb-name>CalculatorSessionBean</ejb-name>
11            <home>sample.session.CalculatorSessionBeanRemoteHome</home>
12            <remote>sample.session.CalculatorSessionBeanRemote</remote>
13            <local-home>sample.session.CalculatorSessionBeanLocalHome</local-home>
14            <local>sample.session.CalculatorSessionBeanLocal</local>
15            <ejb-class>sample.session.CalculatorSessionBean</ejb-class>
16            <session-type>Stateless</session-type>
17            <transaction-type>Container</transaction-type>
18        </session>
19    </enterprise-beans>
20    <assembly-descriptor>

```

# EJB Implementation

## Building/ Modifying the business/callback methods

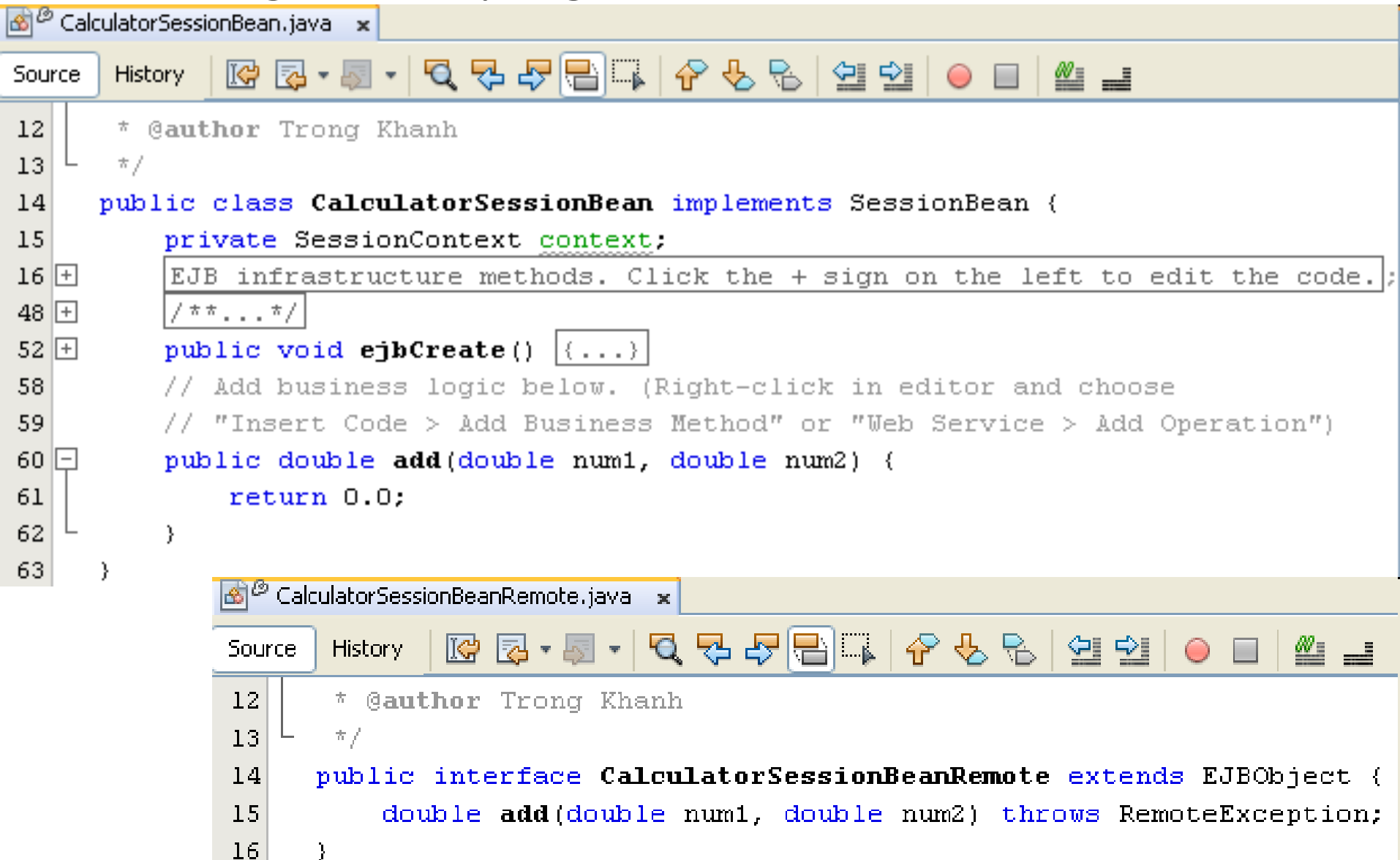
- **Modifying the callback method** if necessary
- **Adding a new business method**
  - Right click on source code of the Bean file (Ex: CalculateBean)
  - Then, choose Insert Code, click Add Business Method...



- Fill or type the method name with return type and add all parameters
- Then, click OK Button

# EJB Implementation

## Building/ Modifying the business/callback methods



The screenshot shows an IDE with two open files. The top file, `CalculatorSessionBean.java`, contains the implementation of the `CalculatorSessionBean` class. The bottom file, `CalculatorSessionBeanRemote.java`, contains the definition of the `CalculatorSessionBeanRemote` interface.

```

12  * @author Trong Khanh
13  */
14  public class CalculatorSessionBean implements SessionBean {
15      private SessionContext context;
16      EJB infrastructure methods. Click the + sign on the left to edit the code.;
48  /**...*/
52  public void ejbCreate() {...}
58  // Add business logic below. (Right-click in editor and choose
59  // "Insert Code > Add Business Method" or "Web Service > Add Operation")
60  public double add(double num1, double num2) {
61      return 0.0;
62  }
63  }

```

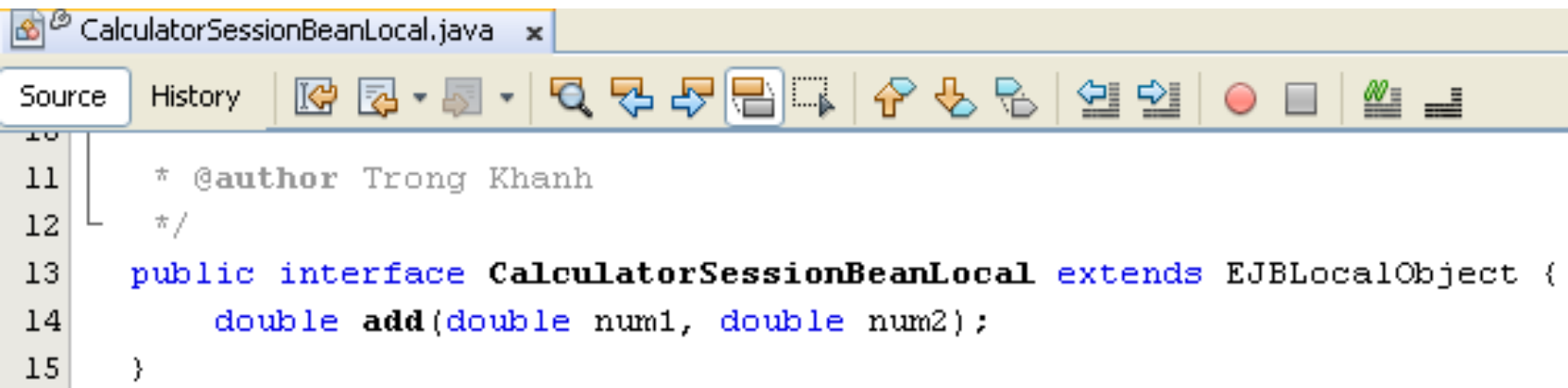
```

12  * @author Trong Khanh
13  */
14  public interface CalculatorSessionBeanRemote extends EJBObject {
15      double add(double num1, double num2) throws RemoteException;
16  }

```

# EJB Implementation

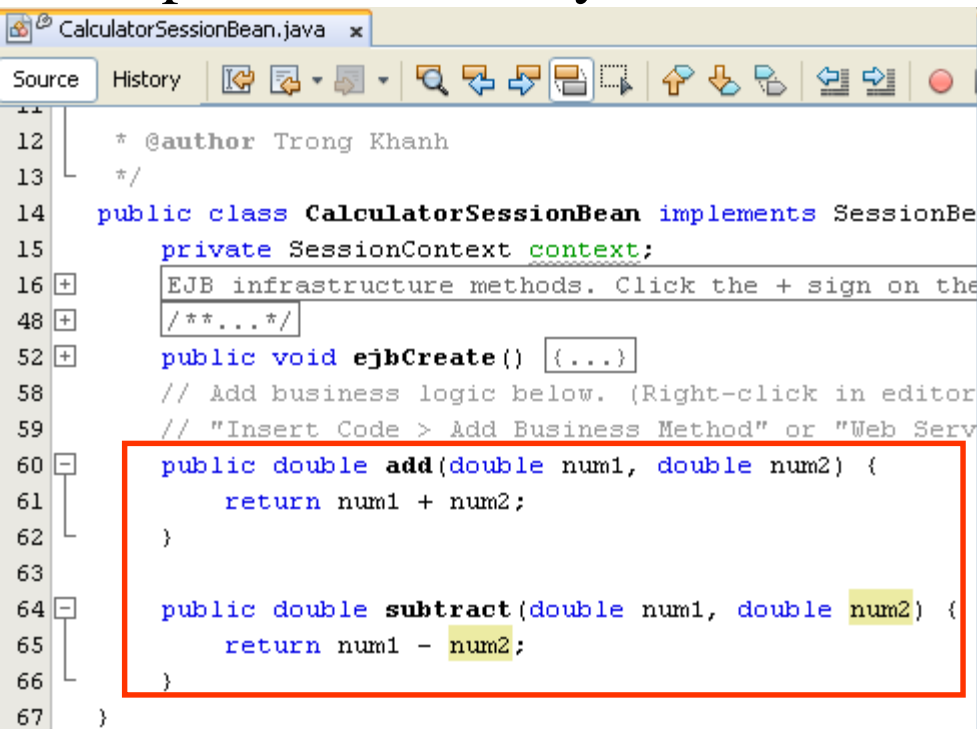
## Building/ Modifying the business/callback methods



```

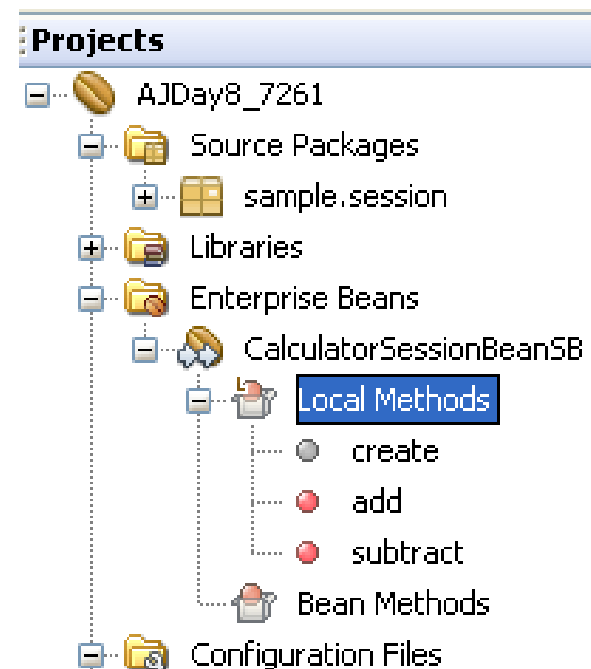
10
11  * @author Trong Khanh
12  */
13  public interface CalculatorSessionBeanLocal extends EJBLocalObject {
14      double add(double num1, double num2);
15  }
    
```

- Implement the body of method corresponding with your purpose



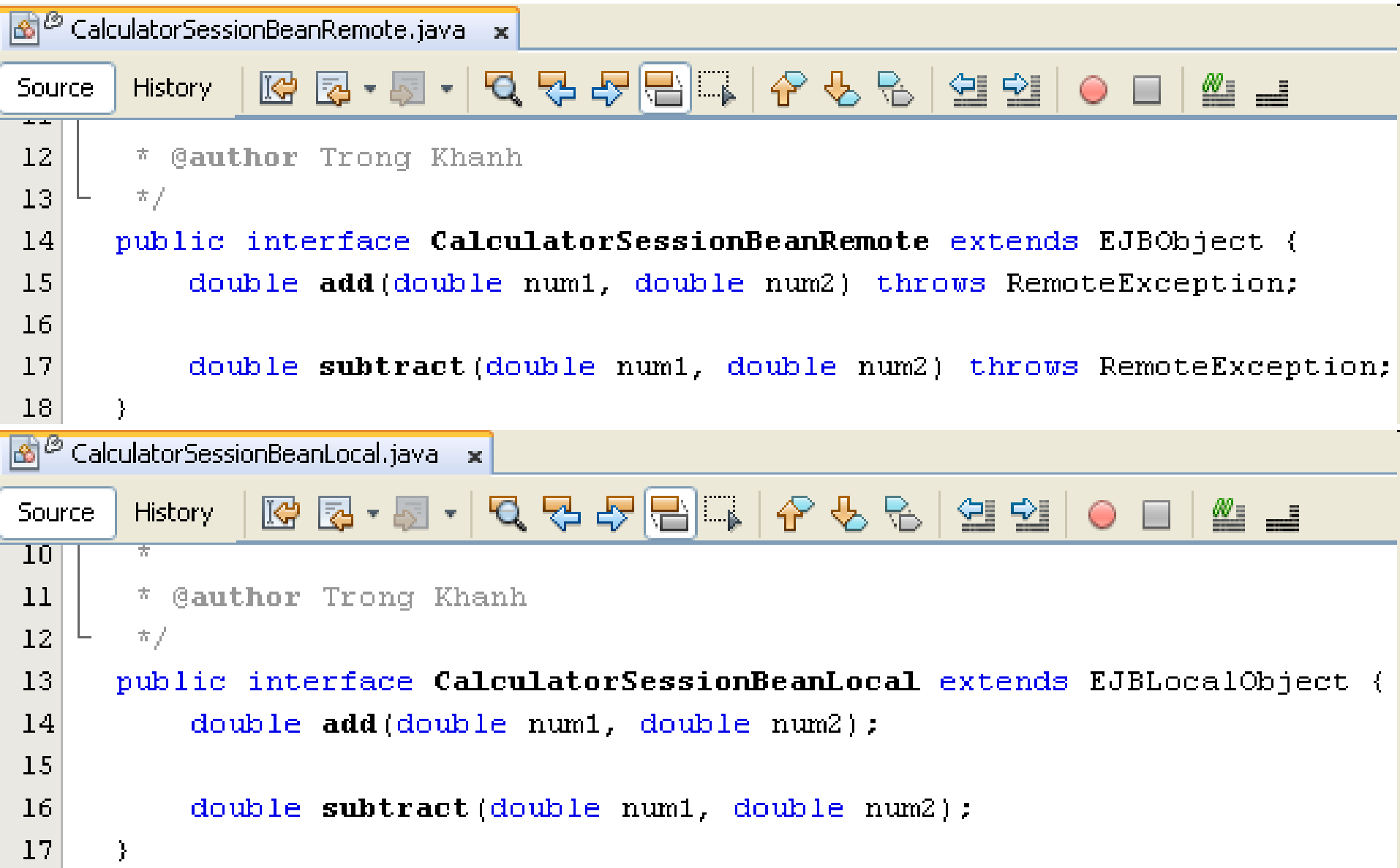
```

11
12  * @author Trong Khanh
13  */
14  public class CalculatorSessionBean implements SessionBe
15      private SessionContext context;
16      EJB infrastructure methods. Click the + sign on the
17      /**...*/
18
19      public void ejbCreate() {...}
20
21      // Add business logic below. (Right-click in editor
22      // "Insert Code > Add Business Method" or "Web Serv
23
24      public double add(double num1, double num2) {
25          return num1 + num2;
26      }
27
28      public double subtract(double num1, double num2) {
29          return num1 - num2;
30      }
31
32  }
    
```



# EJB Implementation

## Building/ Modifying the business/callback methods



The image shows two code snippets from an IDE, illustrating the implementation of EJB business methods.

**CalculatorSessionBeanRemote.java**

```

12  * @author Trong Khanh
13  */
14  public interface CalculatorSessionBeanRemote extends EJBObject {
15      double add(double num1, double num2) throws RemoteException;
16
17      double subtract(double num1, double num2) throws RemoteException;
18  }
  
```

**CalculatorSessionBeanLocal.java**

```

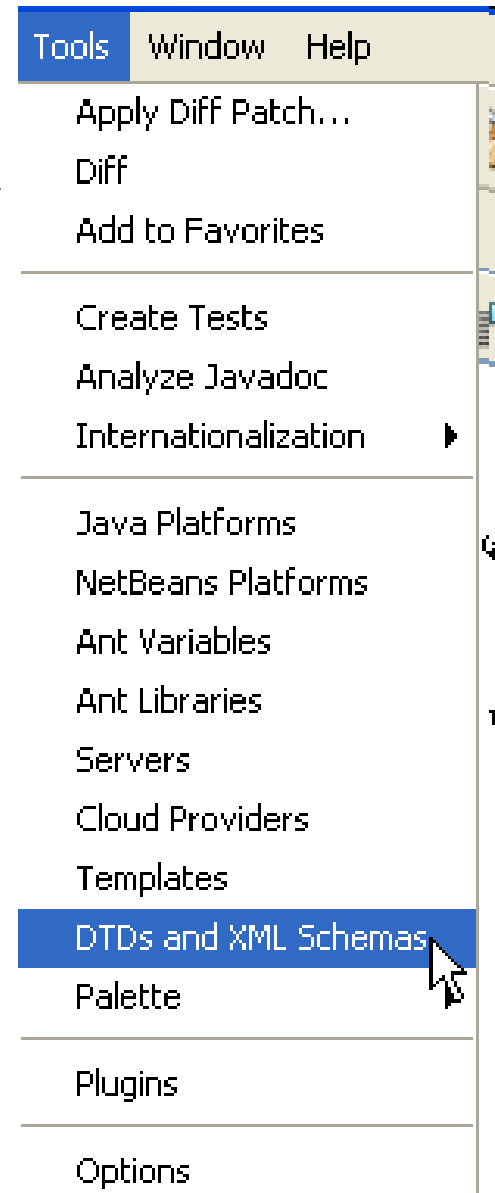
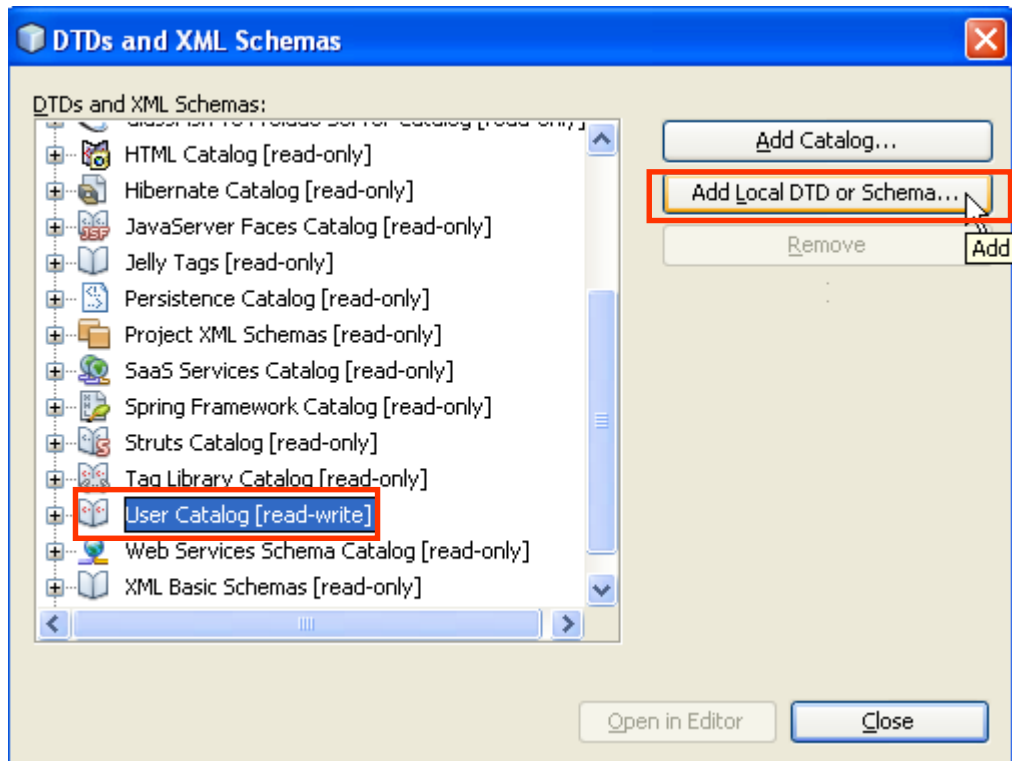
10  *
11  * @author Trong Khanh
12  */
13  public interface CalculatorSessionBeanLocal extends EJBLocalObject {
14      double add(double num1, double num2);
15
16      double subtract(double num1, double num2);
17  }
  
```



# EJB Implementation

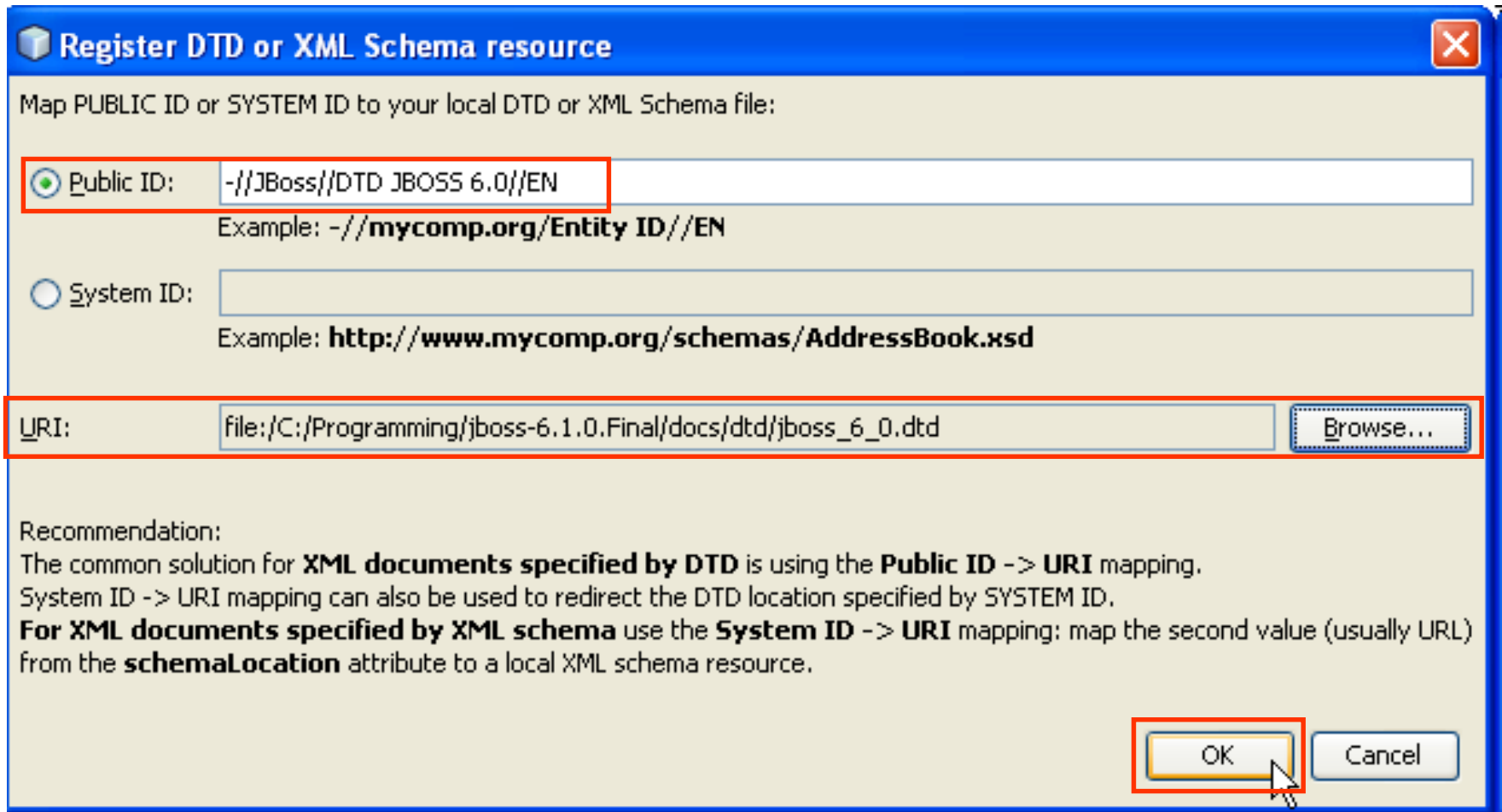
## Additional - Mapping JNDI

- Set up visual typing to **jboss.xml** file
  - Copy the **jboss\_6\_0.dtd** file to your local disk
  - Mapping this file to Netbeans as following steps
    - Click menu Tools, click “**DTDs and XML Schemas**” items



# EJB Implementation

## Additional - Mapping JNDI



**Register DTD or XML Schema resource**

Map PUBLIC ID or SYSTEM ID to your local DTD or XML Schema file:

☒ **Public ID:**   
Example: -//mycomp.org/Entity ID//EN

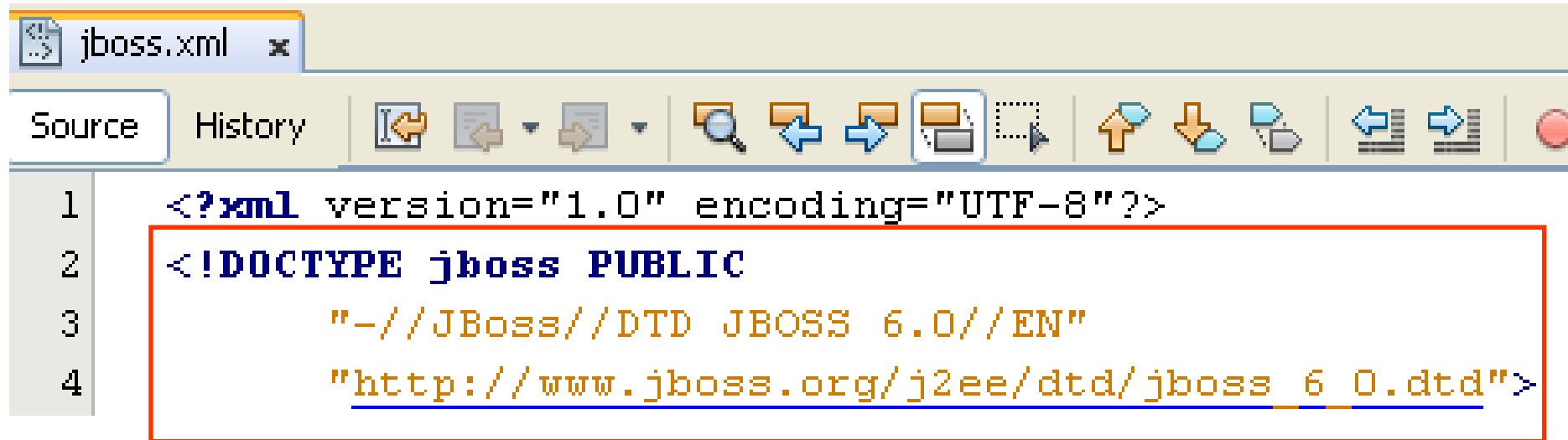
☐ **System ID:**   
Example: <http://www.mycomp.org/schemas/AddressBook.xsd>

**URI:**

Recommendation:  
The common solution for **XML documents specified by DTD** is using the **Public ID -> URI** mapping.  
System ID -> URI mapping can also be used to redirect the DTD location specified by SYSTEM ID.  
**For XML documents specified by XML schema** use the **System ID -> URI** mapping: map the second value (usually URL) from the **schemaLocation** attribute to a local XML schema resource.

# EJB Implementation

## Additional - Mapping JNDI



The screenshot shows an IDE window titled 'jboss.xml'. The 'Source' tab is active, displaying the following XML content:

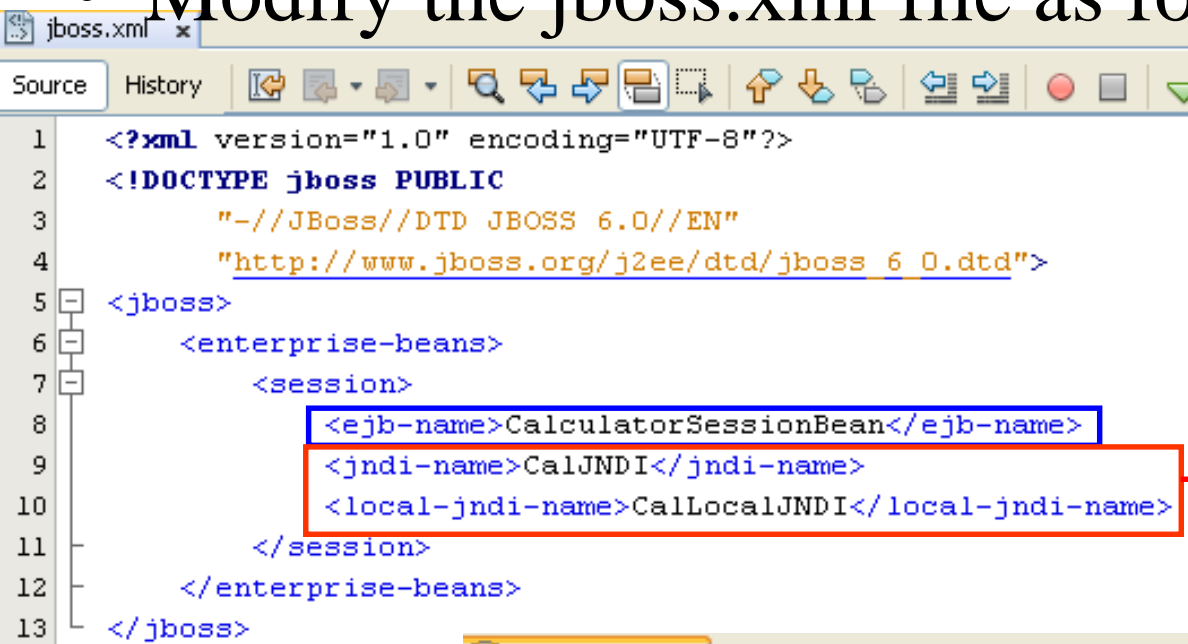
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE jboss PUBLIC
3     "-//JBoss//DTD JBOSS 6.0//EN"
4     "http://www.jboss.org/j2ee/dtd/jboss\_6\_0.dtd>
```

The content from line 2 to line 4 is enclosed in a red rectangular box.

# EJB Implementation

## Step 4: Mapping the JNDI to beans

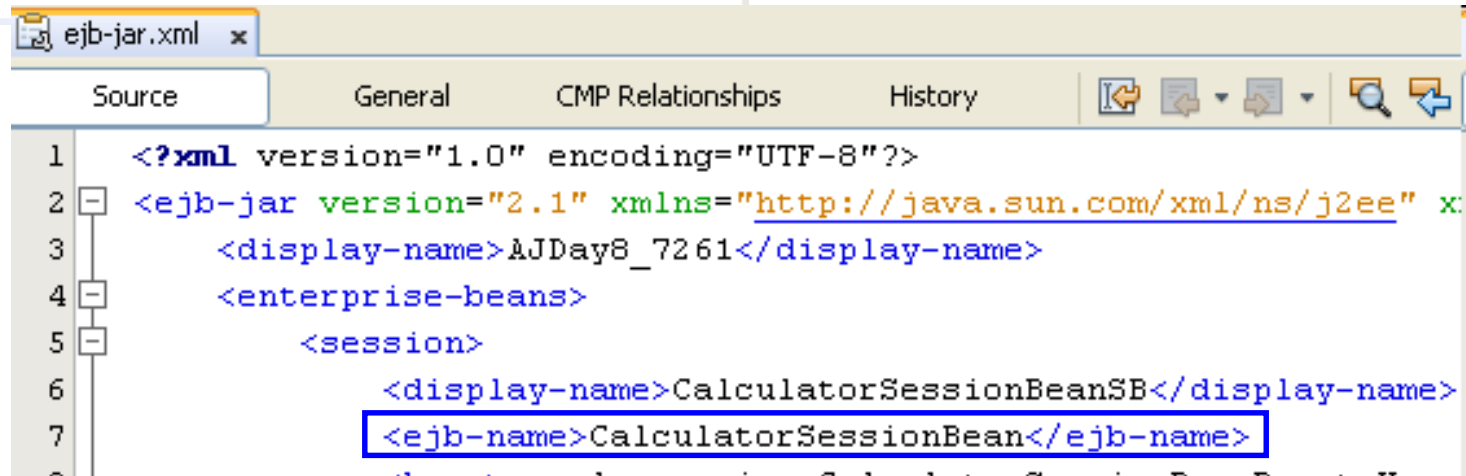
- Modify the jboss.xml file as following



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE jboss PUBLIC
3      "-//JBoss//DTD JBOSS 6.0//EN"
4      "http://www.jboss.org/j2ee/dtd/jboss_6_0.dtd">
5  <jboss>
6      <enterprise-beans>
7          <session>
8              <ejb-name>CalculatorSessionBean</ejb-name>
9              <jndi-name>CalJNDI</jndi-name>
10             <local-jndi-name>CalLocalJNDI</local-jndi-name>
11          </session>
12      </enterprise-beans>
13  </jboss>
    
```

Fill your wanted JNDI that you want to reference



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <ejb-jar version="2.1" xmlns="http://java.sun.com/xml/ns/j2ee" x
3      <display-name>AJDay8_7261</display-name>
4      <enterprise-beans>
5          <session>
6              <display-name>CalculatorSessionBeanSB</display-name>
7              <ejb-name>CalculatorSessionBean</ejb-name>
    
```

# EJB Implementation

## Building & Deploying

z:\LapTrinh\Servlet\AJ\AJDay8\_7261\dist\\*. \*

Name	Ext
[..]	
AJDay8_7261	jar

z:\LapTrinh\Servlet\AJ\AJDay8\_7261\dist\AJDay8\_7261.zip\META-INF\\*. \*

Name	Ext	Size
[..]		<DIR>
ejb-jar	xml	1.317
jboss	xml	410
MANIFEST	MF	103

z:\LapTrinh\Servlet\AJ\AJDay8\_7261\dist\AJDay8\_7261.zip\sample\session\\*. \*

Name	Ext	Size
[..]		<DIR>
CalculatorSessionBean	class	1.027
CalculatorSessionBeanLocal	class	221
CalculatorSessionBeanLocalHome	class	305
CalculatorSessionBeanRemote	class	281
CalculatorSessionBeanRemoteHome	class	334

z:\LapTrinh\Servlet\AJ\AJDay8\_7261\dist\AJDay8\_7261.zip\\*. \*

Name	Ext	Size
[..]		<DIR>
[META-INF]		<DIR>
[sample]		<DIR>

c:\Programming\jboss-6.1.0.Final\server\default\deploy\\*. \*

Name	Ext	Size
[..]		<DIR>
[hornetq]		<DIR>
[http-invoker.sar]		<DIR>
[jbossweb.sar]		<DIR>
[jms-ra.rar]		<DIR>
[mod_cluster.sar]		<DIR>
[ROOT.war]		<DIR>
[security]		<DIR>
[uuid-key-generator.sar]		<DIR>
[xnio-provider.jar]		<DIR>
AJDay8_7261	jar	5.522

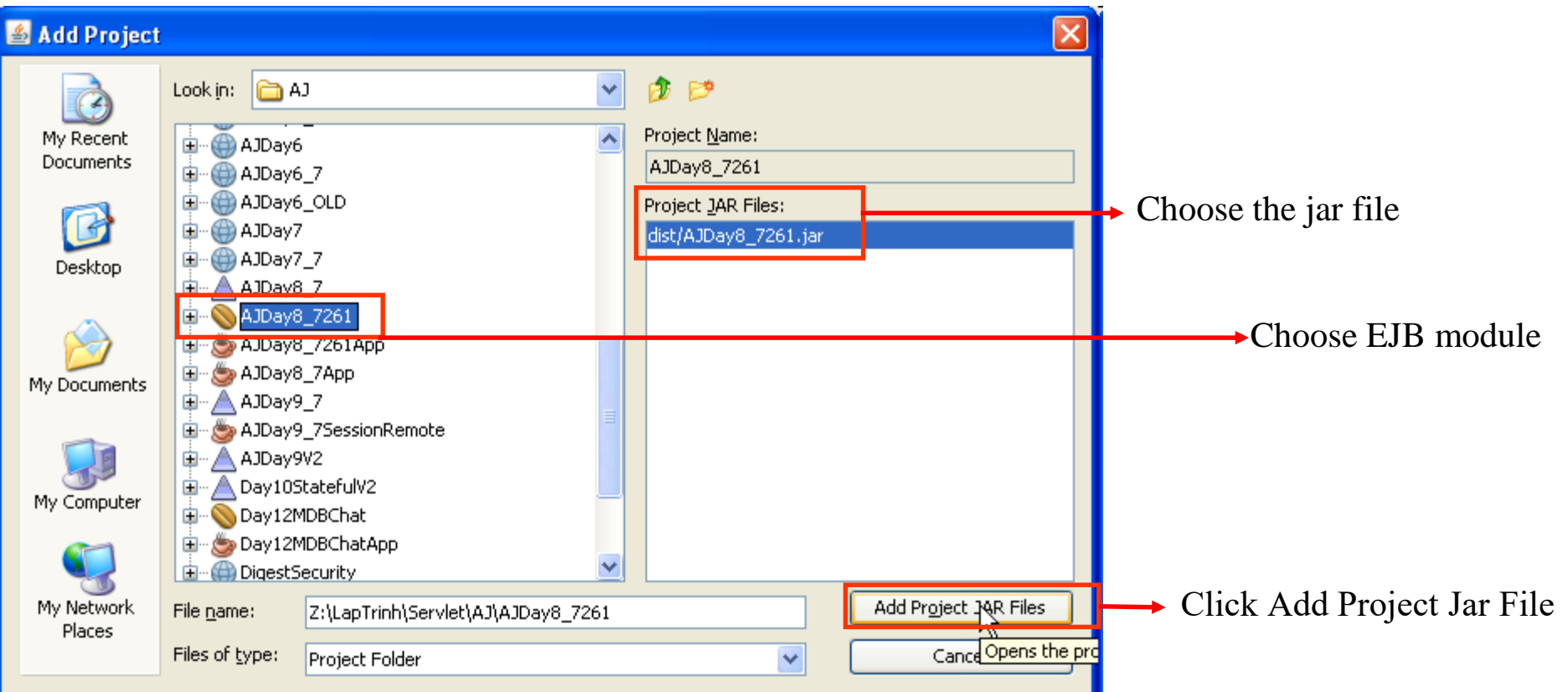
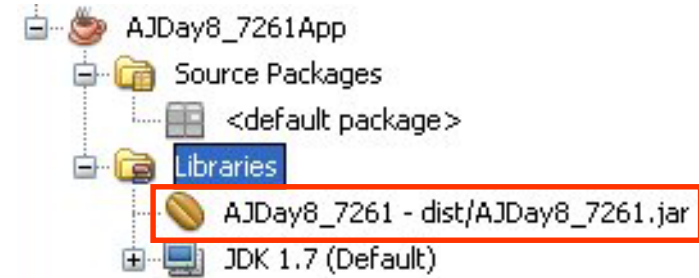
Output

JBoss6.1.0Final	AJDay8_7261 (clean,dist)
22:15:26,046 INFO	[org.jboss.ejb.deployers.EjbDeployer] installing bean: ejb/#CalculatorSessionBean,uid28945344
22:15:26,046 INFO	[org.jboss.ejb.deployers.EjbDeployer] with dependencies:
22:15:26,046 INFO	[org.jboss.ejb.deployers.EjbDeployer] and supplies:
22:15:26,046 INFO	[org.jboss.ejb.deployers.EjbDeployer] jndi:CalculatorSessionBean/sample.session.CalculatorSessionBeanRemote
22:15:26,046 INFO	[org.jboss.ejb.deployers.EjbDeployer] jndi:CalculatorSessionBean/sample.session.CalculatorSessionBeanLocal
22:15:26,046 INFO	[org.jboss.ejb.deployers.EjbDeployer] jndi:CalJNDI
22:15:26,046 INFO	[org.jboss.ejb.deployers.EjbDeployer] jndi:CalLocalJNDI
22:15:26,078 INFO	[org.jboss.ejb.EjbModule] Deploying CalculatorSessionBean
22:15:26,109 INFO	[org.jboss.ejb.plugins.local.BaseLocalProxyFactory] Bound EJB LocalHome 'CalculatorSessionBean' to jndi 'CalLocalJNDI'
22:15:26,109 INFO	[org.jboss.proxy.ejb.ProxyFactory] Bound EJB Home 'CalculatorSessionBean' to jndi 'CalJNDI'

# EJB Implementation

## Creating the client application

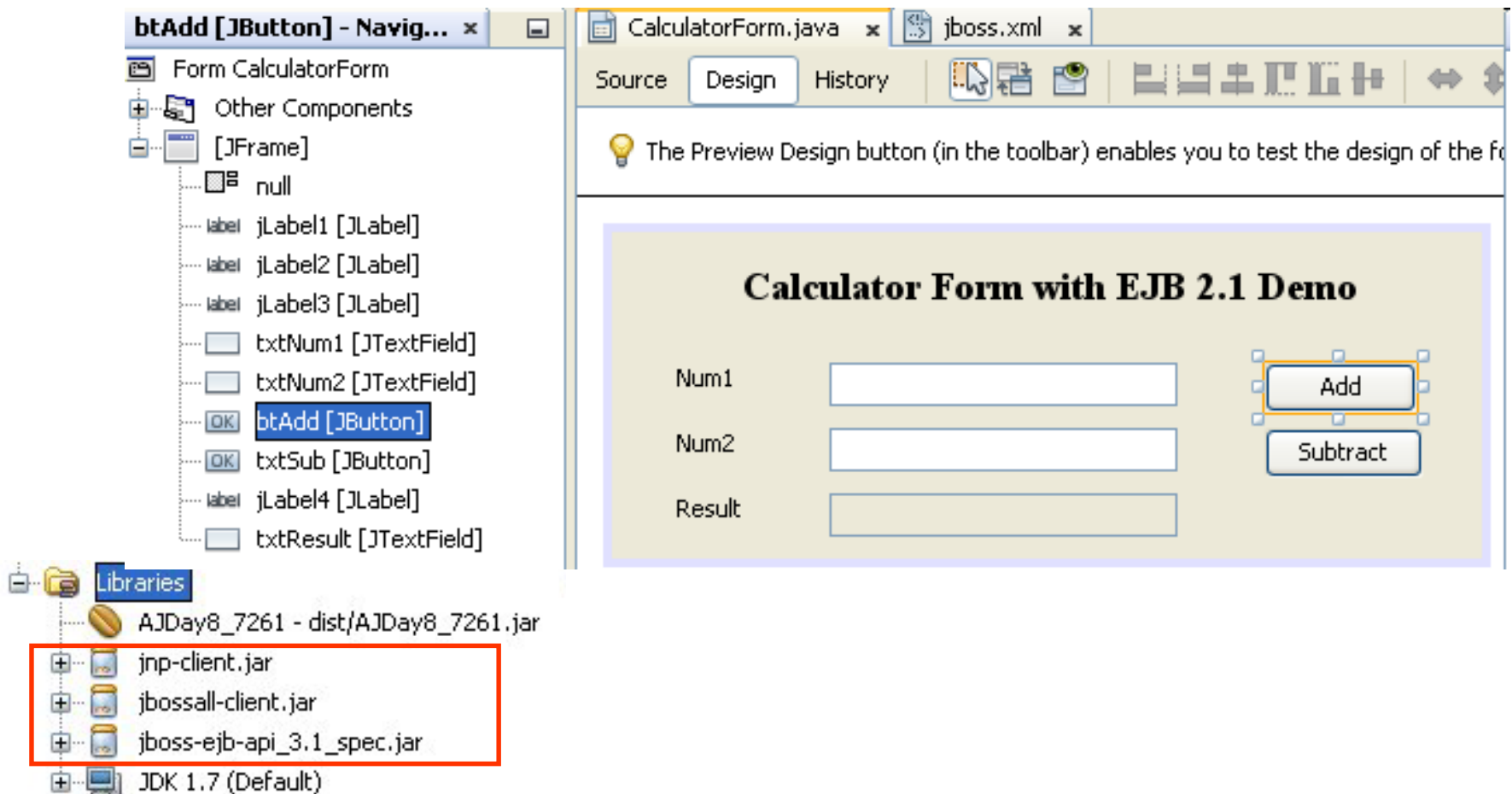
- Create Java console application
- Add reference to EJB project mapping to invoke the remote method on application Server
  - Right click on library of client project/ click “Add Project ...”



# EJB Implementation

## Creating the client application

- Adding the code as following (notes: addition the `jbossall-client.jar`, `jnp-client.jar`, and `jboss-ejb-api_3.1_spec.jar` from `JBOSS_HOME\client` to application project)



The screenshot displays an IDE interface for creating a client application. On the left, the 'btAdd [JButton] - Navig...' window shows a project structure for 'Form CalculatorForm'. The components list includes 'Other Components', '[JFrame]', 'null', 'jLabel1 [JLabel]', 'jLabel2 [JLabel]', 'jLabel3 [JLabel]', 'txtNum1 [JTextField]', 'txtNum2 [JTextField]', 'btAdd [JButton]' (highlighted), 'txtSub [JButton]', 'jLabel4 [JLabel]', and 'txtResult [JTextField]'. Below this, the 'Libraries' section shows the project's classpath, including 'AJDay8\_7261 - dist/AJDay8\_7261.jar', 'jnp-client.jar', 'jbossall-client.jar', 'jboss-ejb-api\_3.1\_spec.jar' (all three highlighted with a red box), and 'JDK 1.7 (Default)'. The main editor area shows the 'CalculatorForm.java' file in 'Design' mode. The design view displays a 'Calculator Form with EJB 2.1 Demo' with two input fields labeled 'Num1' and 'Num2', a 'Result' field, and two buttons labeled 'Add' and 'Subtract'. A toolbar at the top includes buttons for 'Source', 'Design', 'History', and a 'Preview Design' button (highlighted with a lightbulb icon). A tooltip for the 'Preview Design' button states: 'The Preview Design button (in the toolbar) enables you to test the design of the fo'.

# EJB Implementation

## Creating the client application

```

CalculatorForm.java x
Source Design History
20 * @author Trong Khanh
21 */
22 public class CalculatorForm extends javax.swing.JFrame {
23     /**...*/
26     public CalculatorForm() { ... }
29     /**...*/
34 @SuppressWarnings("unchecked")
35     Generated Code
92
93     private void btAddActionPerformed(java.awt.event.ActionEvent evt) {
94         try {
95             System.setProperty("java.naming.factory.initial",
96                 "org.jnp.interfaces.NamingContextFactory");
97             System.setProperty("java.naming.provider.url", "localhost:1099");
98             Context context = new InitialContext();
99             Object obj = context.lookup("CalJNDI");
100             CalculatorSessionBeanRemoteHome ejbHome = (CalculatorSessionBeanRemoteHome)
101                 PortableRemoteObject.narrow(obj, CalculatorSessionBeanRemoteHome.class);
102             CalculatorSessionBeanRemote ejbObj = ejbHome.create();
103             String n1 = txtNum1.getText();
104             String n2 = txtNum2.getText();
105             double num1 = Double.parseDouble(n1);
106             double num2 = Double.parseDouble(n2);
107             double result = ejbObj.add(num1, num2);
108             txtResult.setText(result + "");
109         } catch (CreateException ex) {
110             Logger.getLogger(CalculatorForm.class.getName()).log(Level.SEVERE, null, ex);
111         } catch (RemoteException ex) {
112             Logger.getLogger(CalculatorForm.class.getName()).log(Level.SEVERE, null, ex);
113         } catch (NamingException ex) {
114             Logger.getLogger(CalculatorForm.class.getName()).log(Level.SEVERE, null, ex);

```



# EJB Implementation

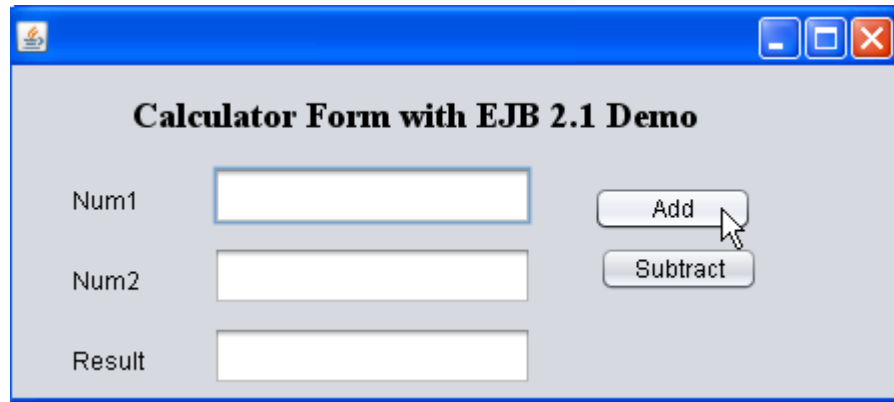
## Creating the client application

```

123 private void txtSubActionPerformed(java.awt.event.ActionEvent evt) {
124     try {
125         System.setProperty("java.naming.factory.initial",
126             "org.jnp.interfaces.NamingContextFactory");
127         System.setProperty("java.naming.provider.url", "localhost:1099");
128         Context context = new InitialContext();
129         Object obj = context.lookup("CalJNDI");
130         CalculatorSessionBeanRemoteHome ejbHome = (CalculatorSessionBeanRemoteHome)
131             PortableRemoteObject.narrow(obj, CalculatorSessionBeanRemoteHome.class);
132         CalculatorSessionBeanRemote ejbObj = ejbHome.create();
133         String n1 = txtNum1.getText();
134         String n2 = txtNum2.getText();
135         double num1 = Double.parseDouble(n1);
136         double num2 = Double.parseDouble(n2);
137         double result = ejbObj.subtract(num1, num2);
138         txtResult.setText(result + "");
139     } catch (CreateException ex) {
140         Logger.getLogger(CalculatorForm.class.getName()).log(Level.SEVERE, null, ex);
141     } catch (RemoteException ex) {
142         Logger.getLogger(CalculatorForm.class.getName()).log(Level.SEVERE, null, ex);
143     } catch (NamingException ex) {
144         Logger.getLogger(CalculatorForm.class.getName()).log(Level.SEVERE, null, ex);
145     }
146 }
  
```

# EJB Implementation

## Creating the client application



**Calculator Form with EJB 2.1 Demo**

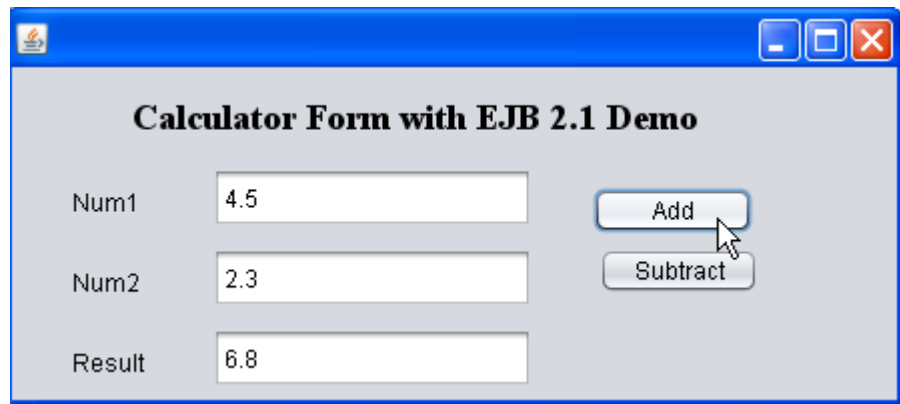
Num1

Num2

Result

Add

Subtract



**Calculator Form with EJB 2.1 Demo**

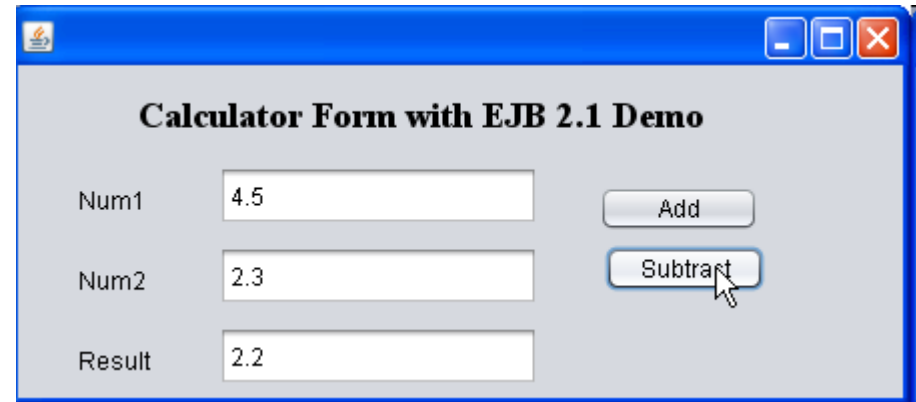
Num1

Num2

Result

Add

Subtract



**Calculator Form with EJB 2.1 Demo**

Num1

Num2

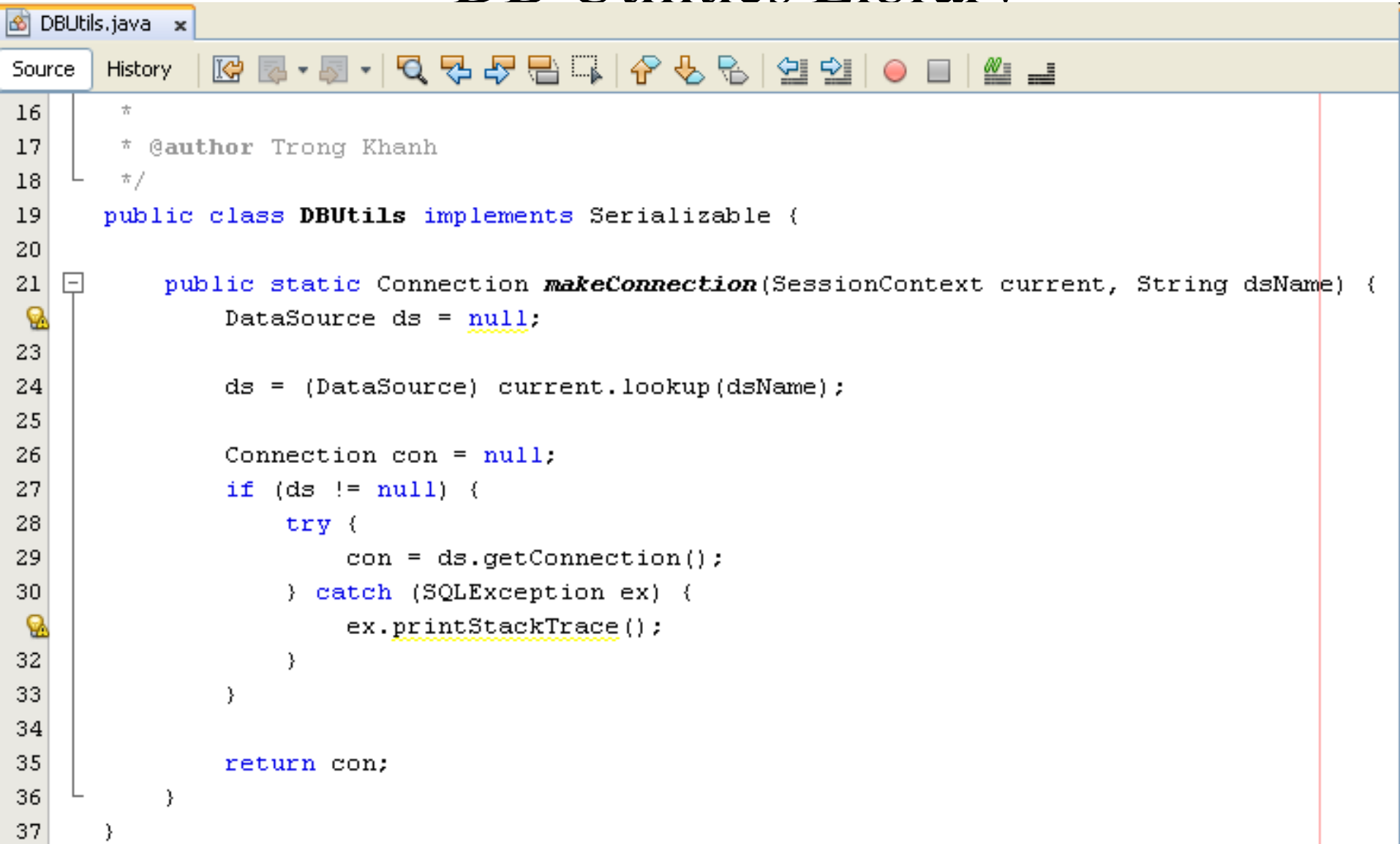
Result

Add

Subtract

# Build the simple enterprise application

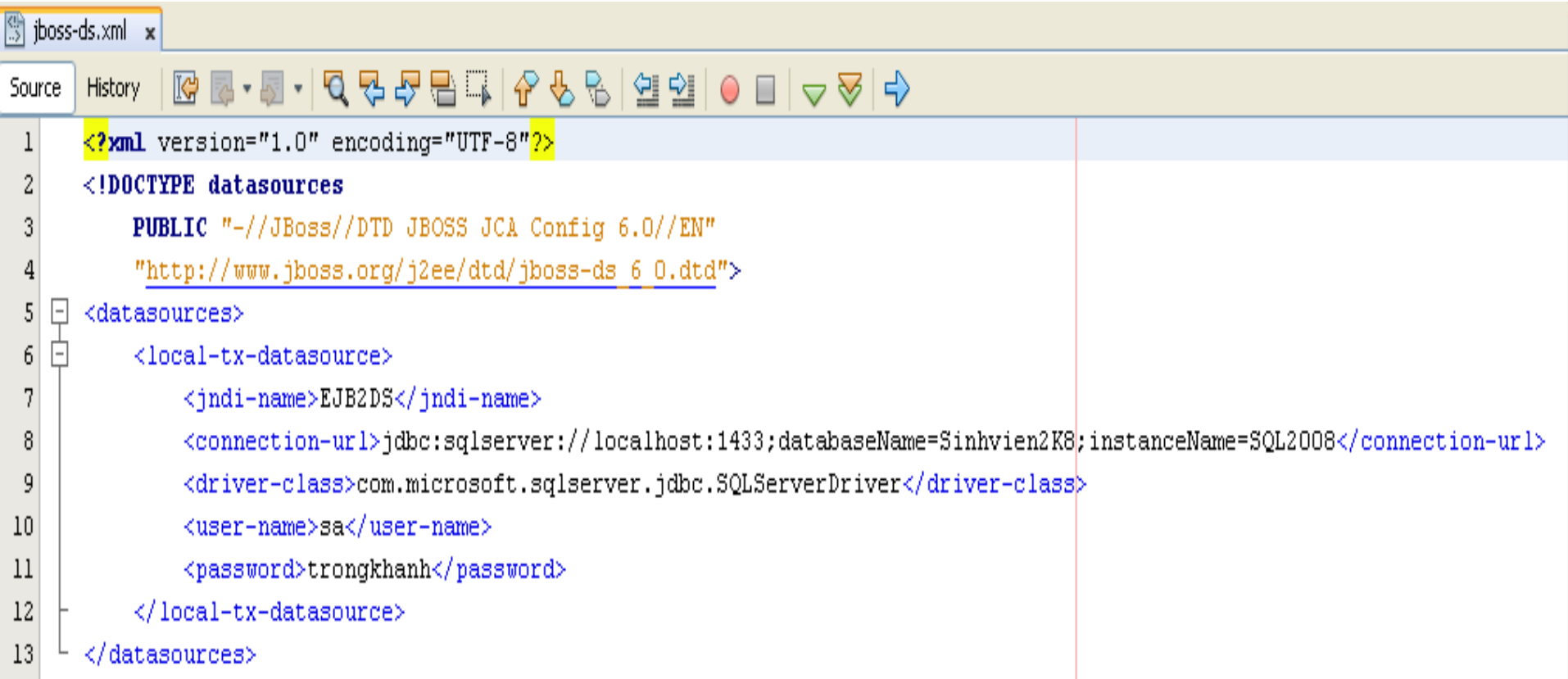
## DB Utilities Library



```
DBUtils.java x
Source History
16 *
17 * @author Trong Khanh
18 */
19 public class DBUtils implements Serializable {
20
21     public static Connection makeConnection(SessionContext current, String dsName) {
22         DataSource ds = null;
23
24         ds = (DataSource) current.lookup(dsName);
25
26         Connection con = null;
27         if (ds != null) {
28             try {
29                 con = ds.getConnection();
30             } catch (SQLException ex) {
31                 ex.printStackTrace();
32             }
33         }
34
35         return con;
36     }
37 }
```

# Build the simple enterprise application

## Data Source Description



The image shows a screenshot of an XML editor window titled 'jboss-ds.xml'. The editor displays the following XML code:

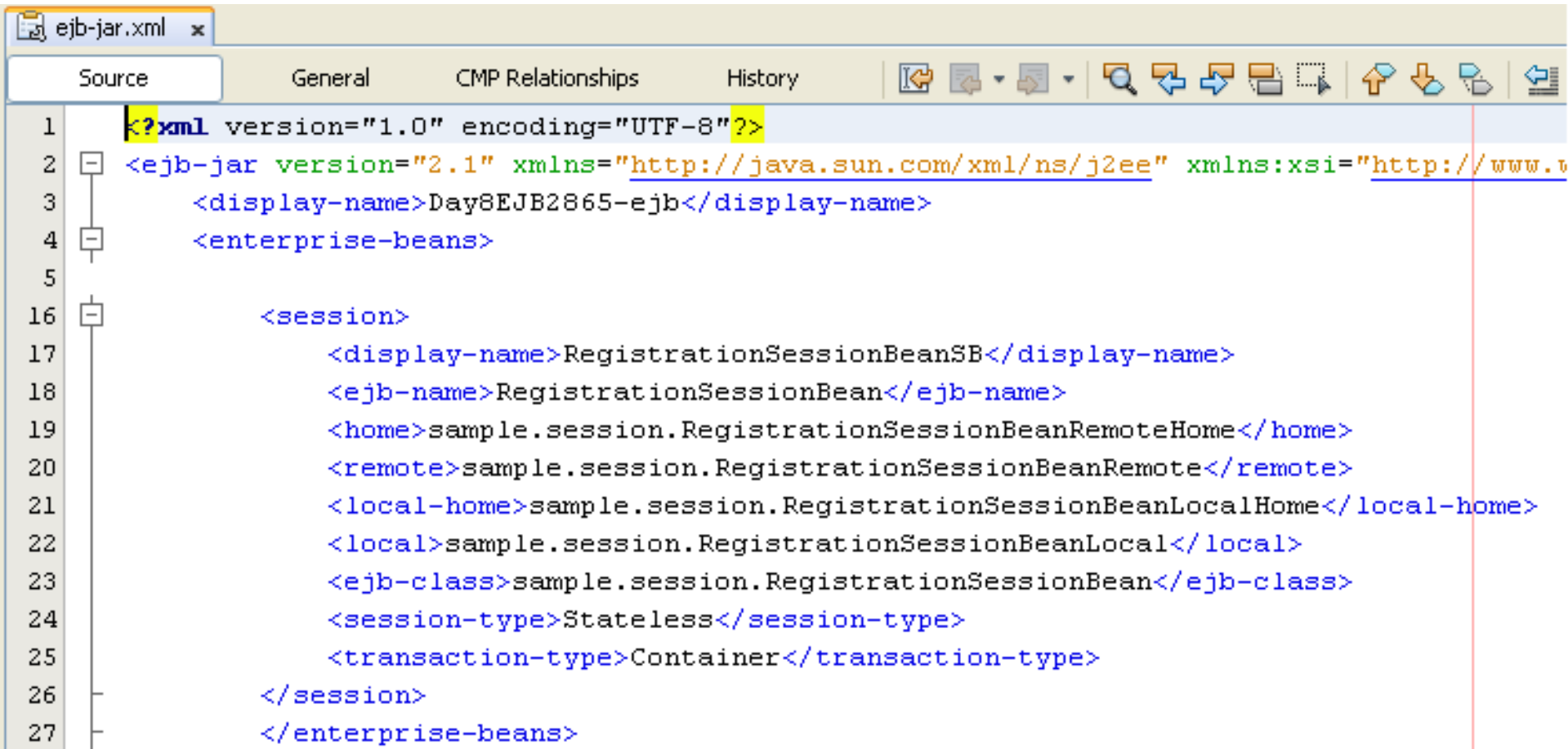
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE datasources
3     PUBLIC "-//JBoss//DTD JBOSS JCA Config 6.0//EN"
4     "http://www.jboss.org/j2ee/dtd/jboss-ds_6_0.dtd">
5 <datasources>
6   <local-tx-datasource>
7     <jndi-name>EJB2DS</jndi-name>
8     <connection-url>jdbc:sqlserver://localhost:1433;databaseName=Sinhvien2K8;instanceName=SQL2008</connection-url>
9     <driver-class>com.microsoft.sqlserver.jdbc.SQLServerDriver</driver-class>
10    <user-name>sa</user-name>
11    <password>trongkhanh</password>
12  </local-tx-datasource>
13 </datasources>
```

The code defines a local transaction data source named 'EJB2DS' using a Microsoft SQL Server driver. The connection URL is 'jdbc:sqlserver://localhost:1433;databaseName=Sinhvien2K8;instanceName=SQL2008', the driver class is 'com.microsoft.sqlserver.jdbc.SQLServerDriver', the user is 'sa', and the password is 'trongkhanh'.



# Build the simple enterprise application

## EJB-JAR

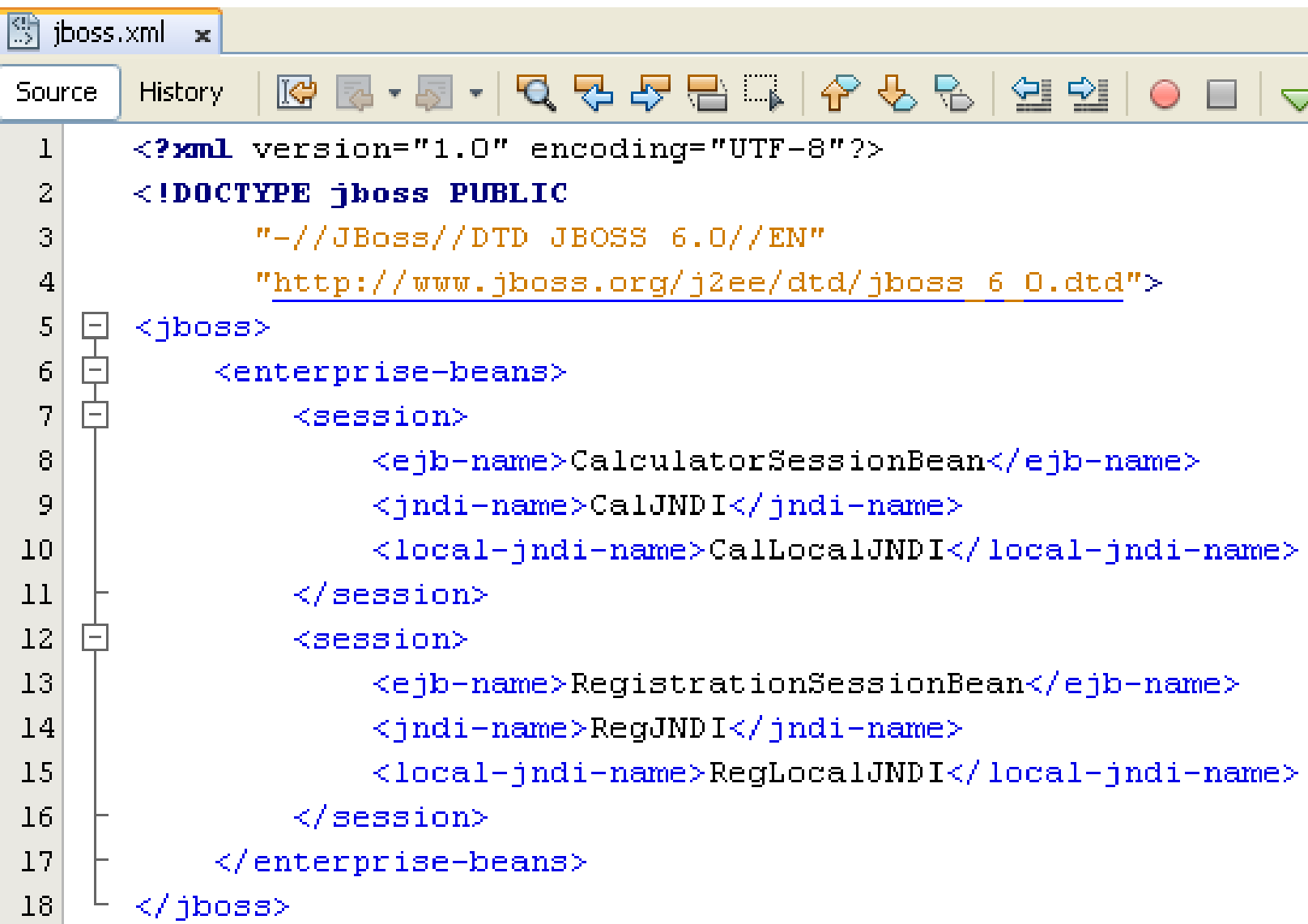


```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <ejb-jar version="2.1" xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3   <display-name>Day8EJB2865-ejb</display-name>
4   <enterprise-beans>
5
16     <session>
17       <display-name>RegistrationSessionBeanSB</display-name>
18       <ejb-name>RegistrationSessionBean</ejb-name>
19       <home>sample.session.RegistrationSessionBeanRemoteHome</home>
20       <remote>sample.session.RegistrationSessionBeanRemote</remote>
21       <local-home>sample.session.RegistrationSessionBeanLocalHome</local-home>
22       <local>sample.session.RegistrationSessionBeanLocal</local>
23       <ejb-class>sample.session.RegistrationSessionBean</ejb-class>
24       <session-type>Stateless</session-type>
25       <transaction-type>Container</transaction-type>
26     </session>
27   </enterprise-beans>
```



# Build the simple enterprise application

## JBoss



The image shows a screenshot of an XML editor window titled 'jboss.xml'. The editor has a toolbar with various icons for file operations and a 'Source' tab selected. The XML content is as follows:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE jboss PUBLIC
3     "-//JBoss//DTD JBOSS 6.0//EN"
4     "http://www.jboss.org/j2ee/dtd/jboss\_6\_0.dtd">
5 <jboss>
6   <enterprise-beans>
7     <session>
8       <ejb-name>CalculatorSessionBean</ejb-name>
9       <jndi-name>CalJNDI</jndi-name>
10      <local-jndi-name>CalLocalJNDI</local-jndi-name>
11    </session>
12    <session>
13      <ejb-name>RegistrationSessionBean</ejb-name>
14      <jndi-name>RegJNDI</jndi-name>
15      <local-jndi-name>RegLocalJNDI</local-jndi-name>
16    </session>
17  </enterprise-beans>
18 </jboss>
```



# Build the simple enterprise application

## Remote

RegistrationSessionBeanRemoteHome.java x

Source History

```
13  * @author Trong Khanh
14  */
15  public interface RegistrationSessionBeanRemoteHome extends EJBHome {
16      RegistrationSessionBeanRemote create() throws CreateException, RemoteException;
17  }
```

RegistrationSessionBeanRemote.java x

Source History

```
12  * @author Trong Khanh
13  */
14  public interface RegistrationSessionBeanRemote extends EJBObject{
15
16      boolean checkLogin(String username, String password) throws RemoteException;
17
18  }
```

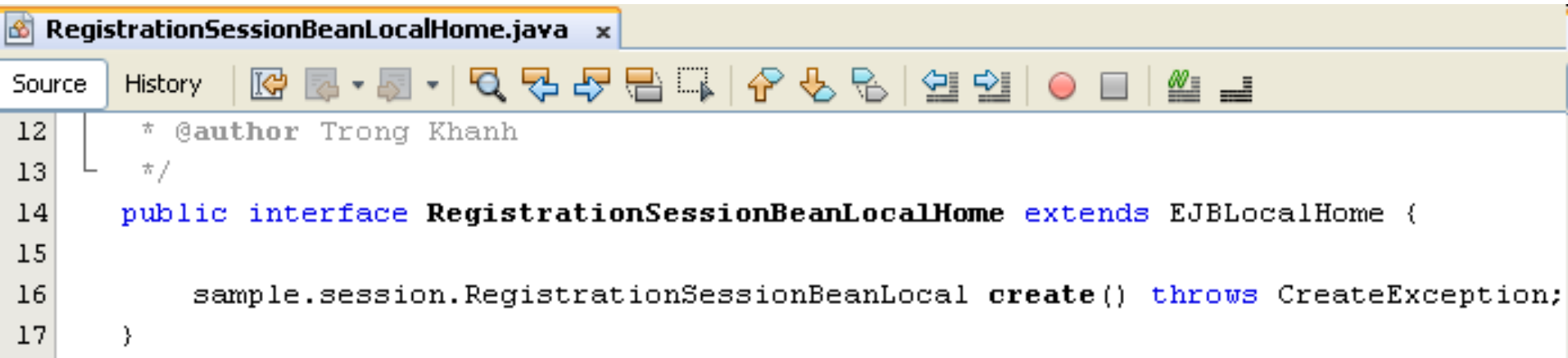
# Build the simple enterprise application

## Local



The screenshot shows an IDE window titled "RegistrationSessionBeanLocal.java". The editor displays the following Java code:

```
11  * @author Trong Khanh
12  */
13  public interface RegistrationSessionBeanLocal extends EJBLocalObject {
14
15      boolean checkLogin(String username, String password);
16
17  }
```

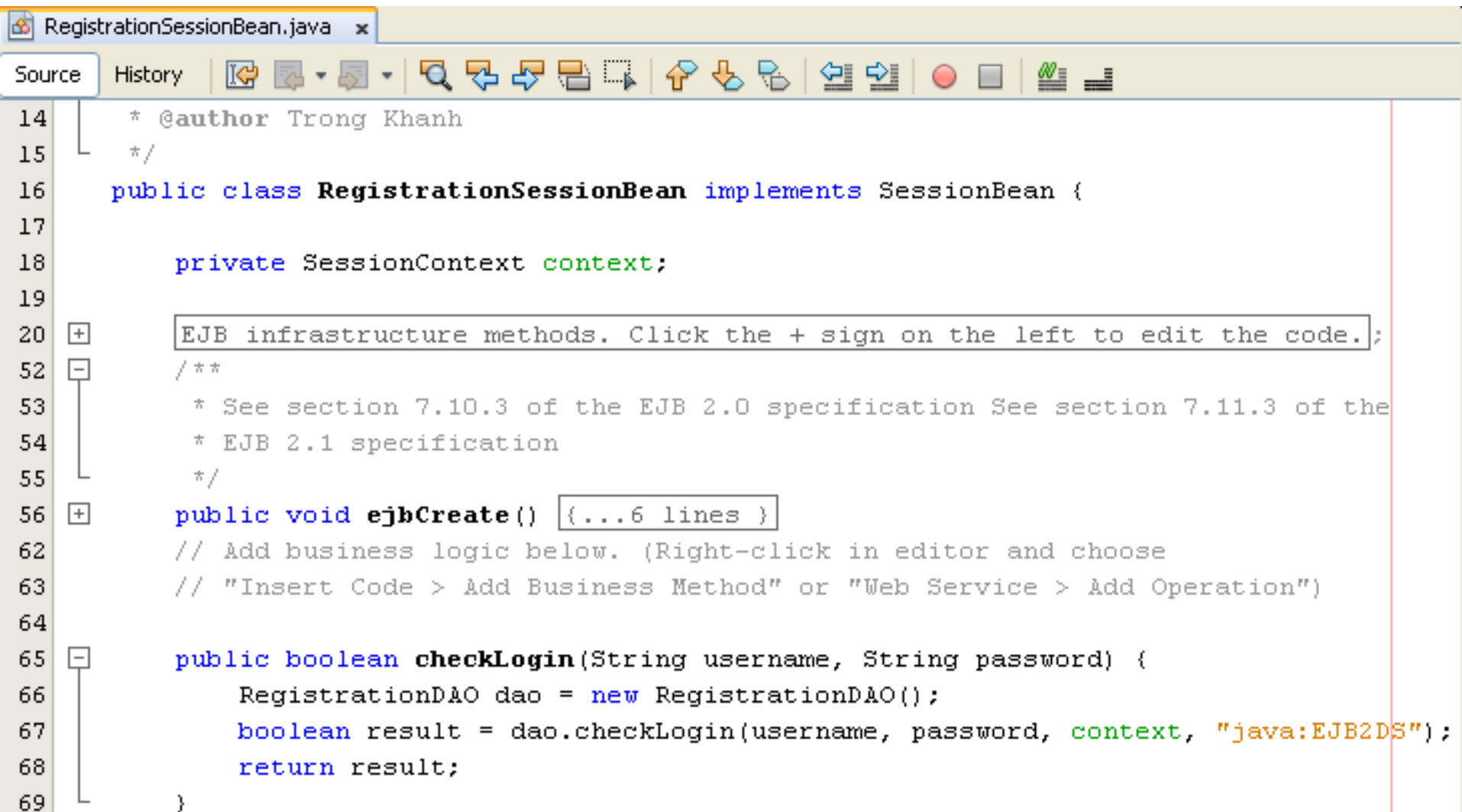


The screenshot shows an IDE window titled "RegistrationSessionBeanLocalHome.java". The editor displays the following Java code:

```
12  * @author Trong Khanh
13  */
14  public interface RegistrationSessionBeanLocalHome extends EJBLocalHome {
15
16      sample.session.RegistrationSessionBeanLocal create() throws CreateException;
17  }
```



# Build the simple enterprise application Bean



```
RegistrationSessionBean.java x
Source History
14 * @author Trong Khanh
15 */
16 public class RegistrationSessionBean implements SessionBean {
17
18     private SessionContext context;
19
20     EJB infrastructure methods. Click the + sign on the left to edit the code.;
21
22     /**
23      * See section 7.10.3 of the EJB 2.0 specification See section 7.11.3 of the
24      * EJB 2.1 specification
25      */
26     public void ejbCreate() { ...6 lines }
27
28     // Add business logic below. (Right-click in editor and choose
29     // "Insert Code > Add Business Method" or "Web Service > Add Operation")
30
31     public boolean checkLogin(String username, String password) {
32         RegistrationDAO dao = new RegistrationDAO();
33         boolean result = dao.checkLogin(username, password, context, "java:EJB2DS");
34         return result;
35     }
36 }
```



# Build the simple enterprise application

## Client Consume

The screenshot displays a Java IDE with a file named `LoginForm.java` open. The interface is in the **Design** view, showing a visual representation of a login form titled **Login Form EJB2**. The form contains the following elements:

- A **Username** label followed by a text input field.
- A **Password** label followed by a password input field (masked with dots).
- A **Login** button.

On the right side, the **[JFrame] - Navigator** panel shows the component hierarchy for the `LoginForm` window:

- Form LoginForm
  - Other Components
    - [JFrame]
      - null
      - label jLabel1 [JLabel]
      - label jLabel2 [JLabel]
      - txtUsername [JTextField]
      - label jLabel3 [JLabel]
      - txtPassword [JPasswordField]
      - OK btLogin [JButton]

A tooltip at the top of the design canvas reads: "To add a component multiple times, select it via click in palette and then Shift-click on design canvas".



# Build the simple enterprise application

## Client Consume

```
LoginForm.java x
Source Design History
19  * @author Trong Khanh
20  */
21  public class LoginForm extends javax.swing.JFrame {
22
23      /** Creates new form LoginForm ...3 lines */
24
25      public LoginForm() { ...3 lines }
26
27
28
29
30      /** This method is called from within the constructor to initialize the form ...5 lines */
31
32      @SuppressWarnings("unchecked")
33
34      Generated Code
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
267
```

# Build the simple enterprise application

## Client Consume

```

98  RegistrationSessionBeanRemote ejbObj = null;
99  try {
100      if (homeObj != null) {
101          ejbObj = homeObj.create();
102      }
103      } catch (CreateException ex) {
104          ex.printStackTrace();
105      } catch (RemoteException ex) {
106          ex.printStackTrace();
107      }
108
109  if (ejbObj != null) {
110      try {
111          String username = txtUsername.getText();
112          String password = new String (txtPassword.getPassword());
113          boolean result = ejbObj.checkLogin(username, password);
114
115          if (result) {
116              JOptionPane.showMessageDialog(this, "Welcome EJB2 Application");
117          } else {
118              JOptionPane.showMessageDialog(this, "Invalid username or password");
119          }
120      } catch (RemoteException ex) {
121          ex.printStackTrace();
122      }
123  }
124  }
  
```

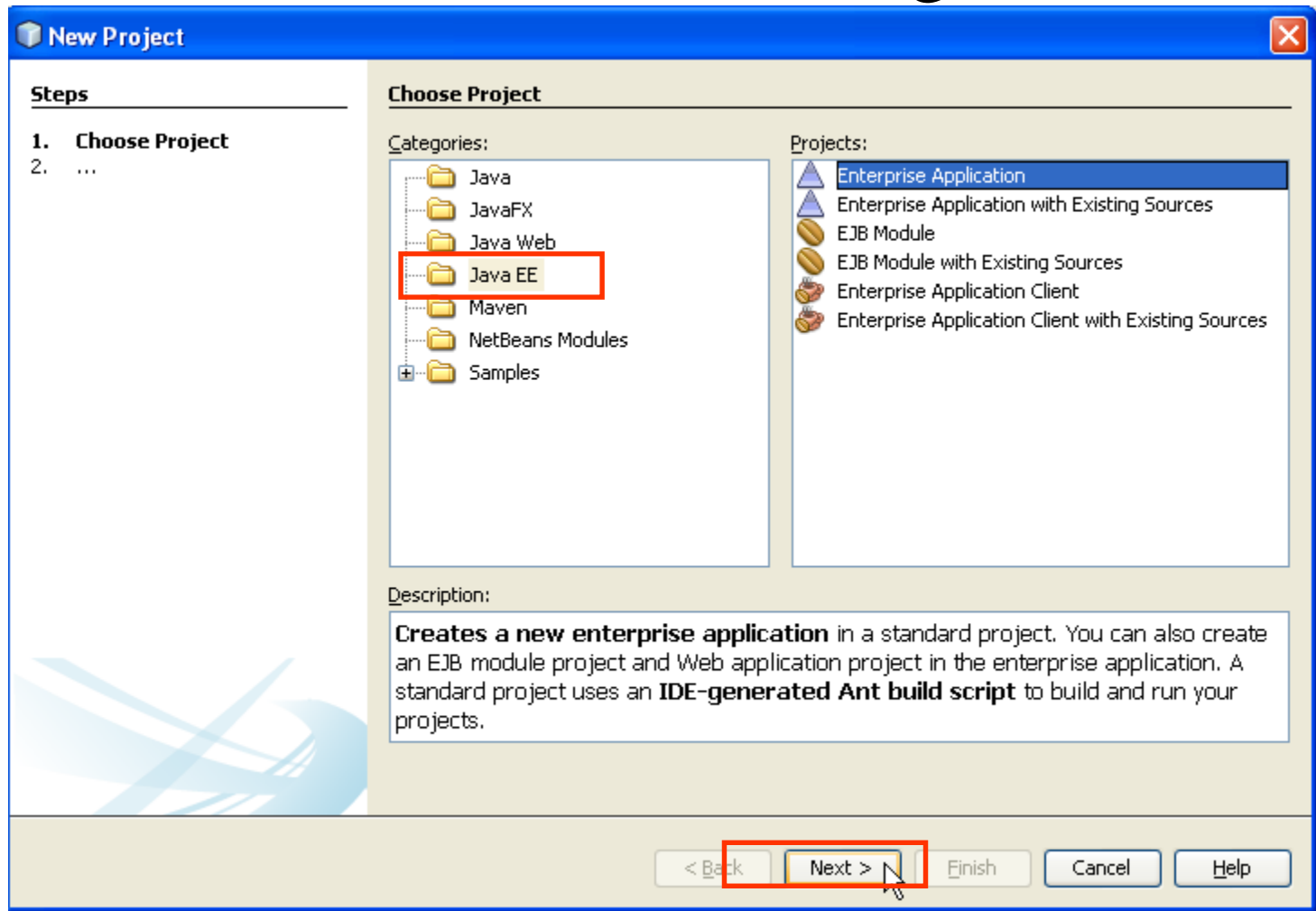
# EJB Implementation

## Enterprise Application Development Process

- **Step 1:** Creating a new Enterprise Application project (EJB and Web Client – ear file)
- **Step 2:** Creating the new corresponding bean depending on your purpose.
- **Step 3:** Building/ Modifying the business/callback methods on Beans
- **Step 4:** Mapping the JNDI to beans
- **Step 5:** Creating the GUI to consumes EJB on web modules
- **Step 6:** Building and Deploying Enterprise application on Application Server
- **Step 7:** Executing the Enterprise Application

# EJB Implementation

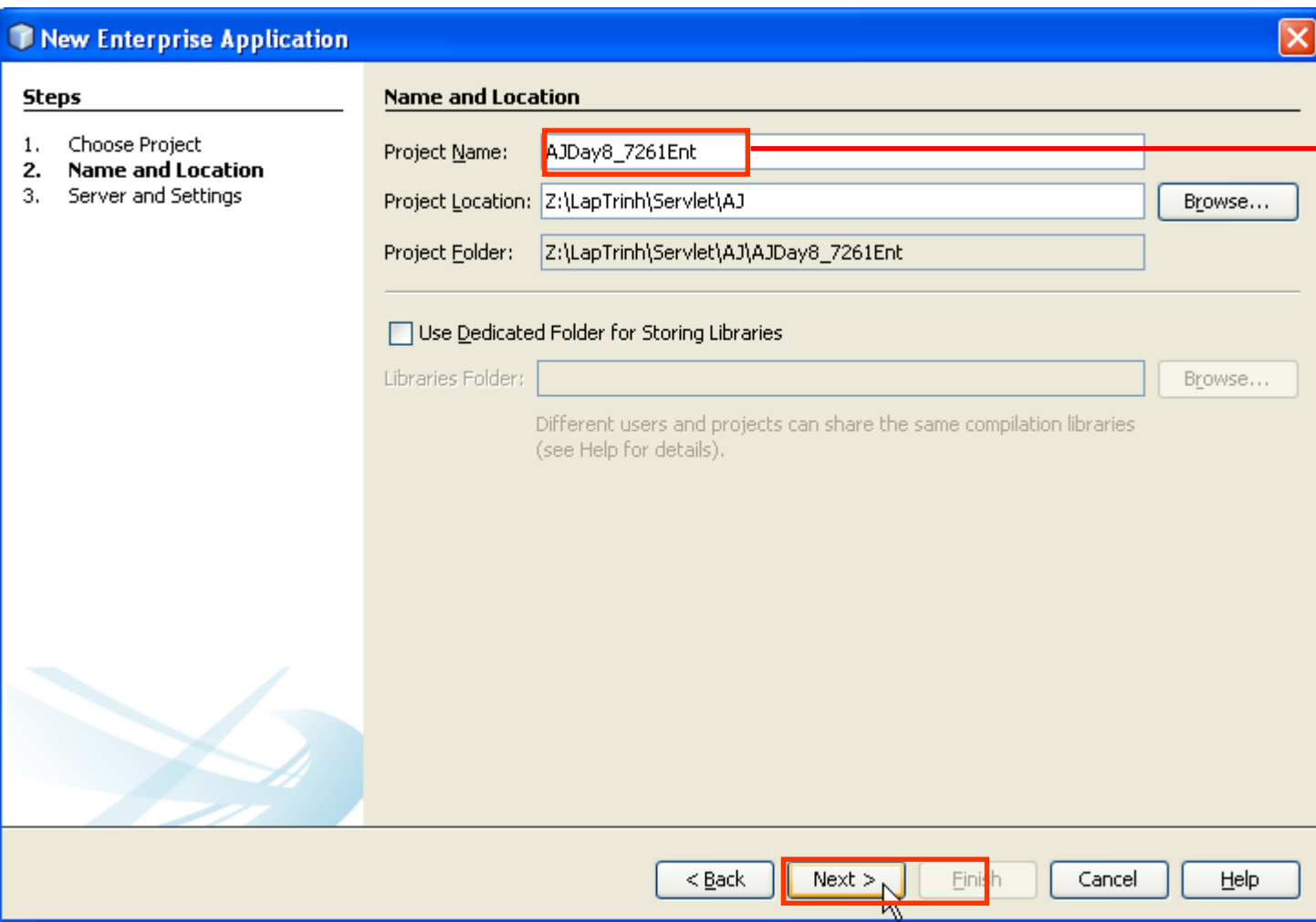
## Creating



- Click Next Button.
- Then type the Project Name, then click Next button

# EJB Implementation

## Creating



**New Enterprise Application**

**Steps**

1. Choose Project
2. **Name and Location**
3. Server and Settings

**Name and Location**

Project Name:

Project Location:

Project Folder:

☐ Use Dedicated Folder for Storing Libraries

Libraries Folder:

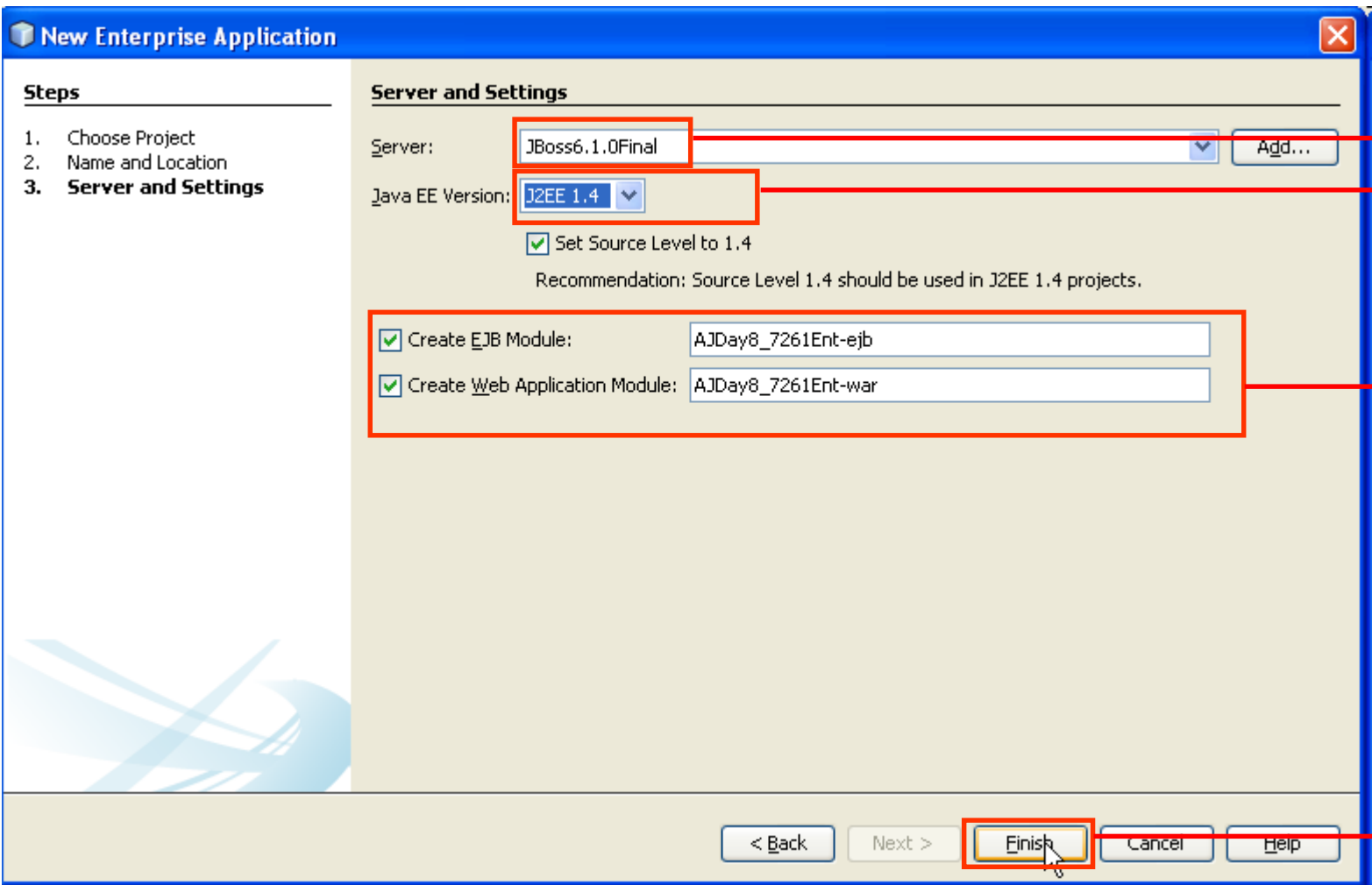
Different users and projects can share the same compilation libraries (see Help for details).

Fill your project name

- Click Next Button.

# EJB Implementation

## Creating



**New Enterprise Application**

**Steps**

1. Choose Project
2. Name and Location
3. **Server and Settings**

**Server and Settings**

Server: JBoss6.1.0.Final Add...

Java EE Version: J2EE 1.4

☒ Set Source Level to 1.4  
Recommendation: Source Level 1.4 should be used in J2EE 1.4 projects.

☒ Create EJB Module: AJDay8\_7261Ent-ejb

☒ Create Web Application Module: AJDay8\_7261Ent-war

< Back Next > **Finish** Cancel Help

Choose the Jboss 4.2.3

Choose J2EE 1.4

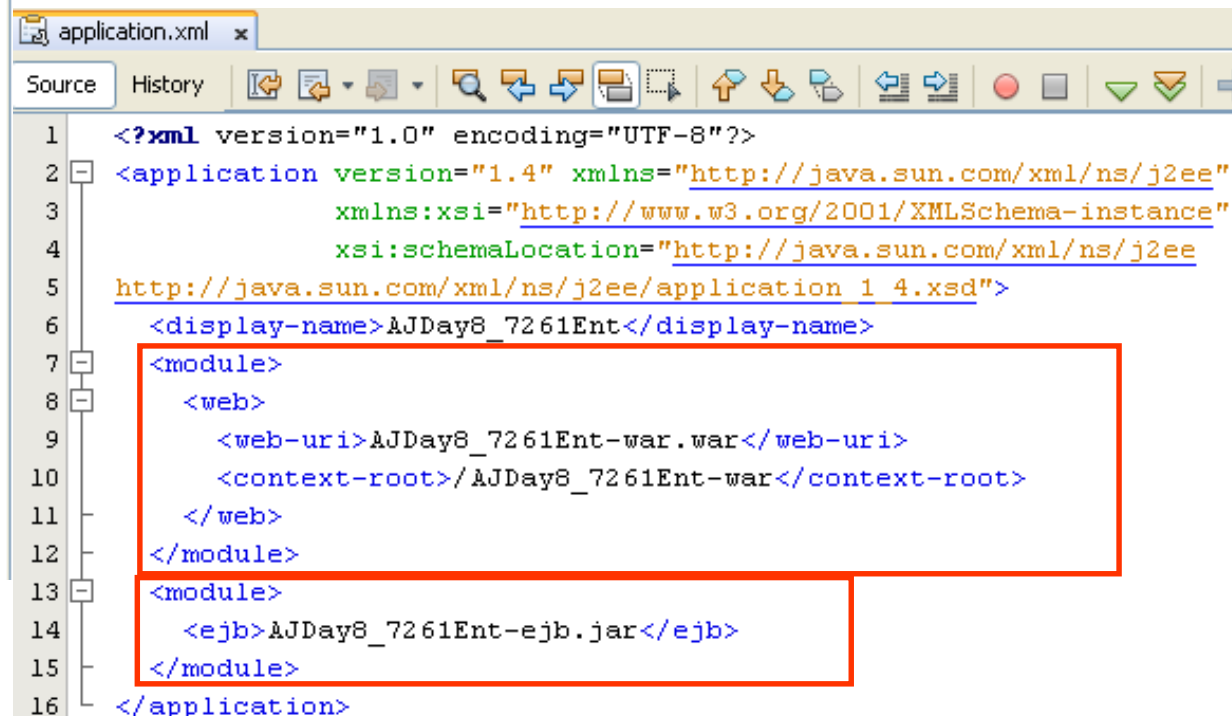
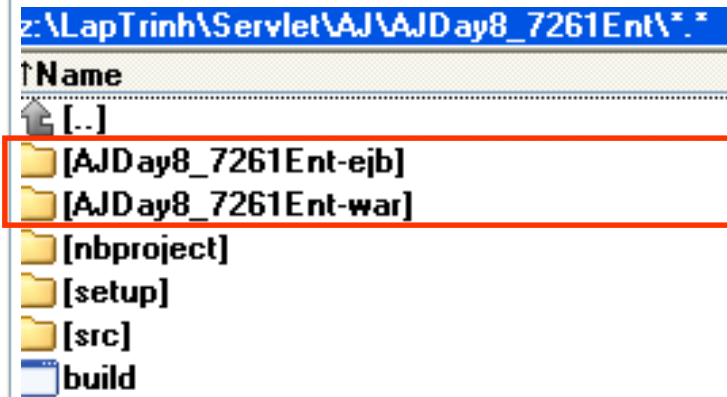
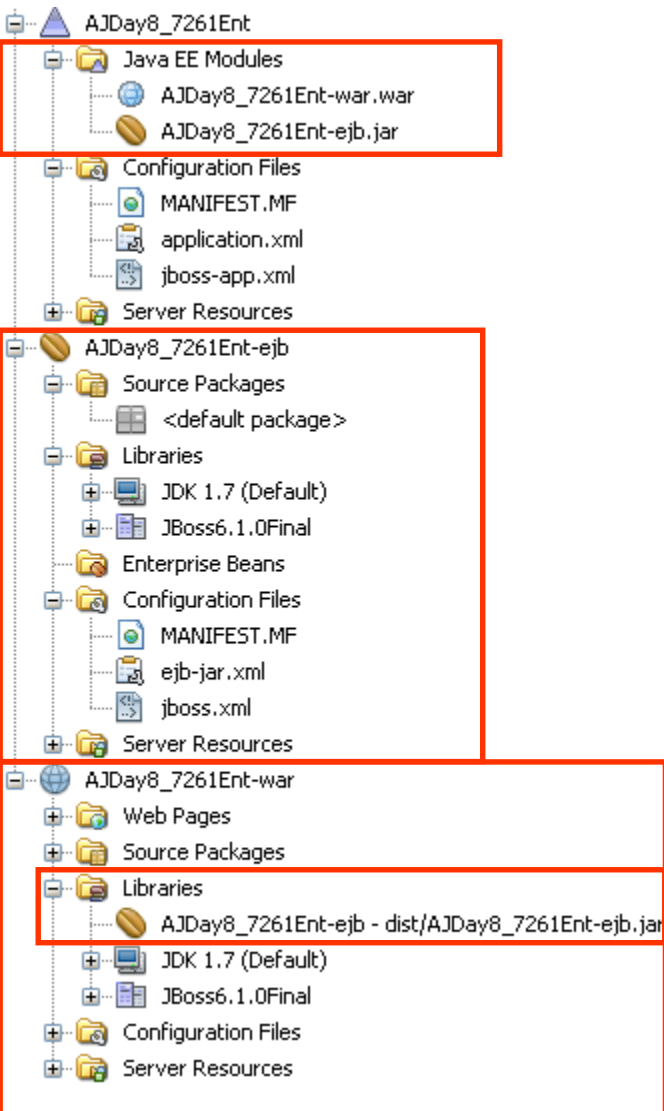
Don't change anything that is default by tools

Click Finish Button



# EJB Implementation

## Creating



# EJB Implementation

## Next Steps

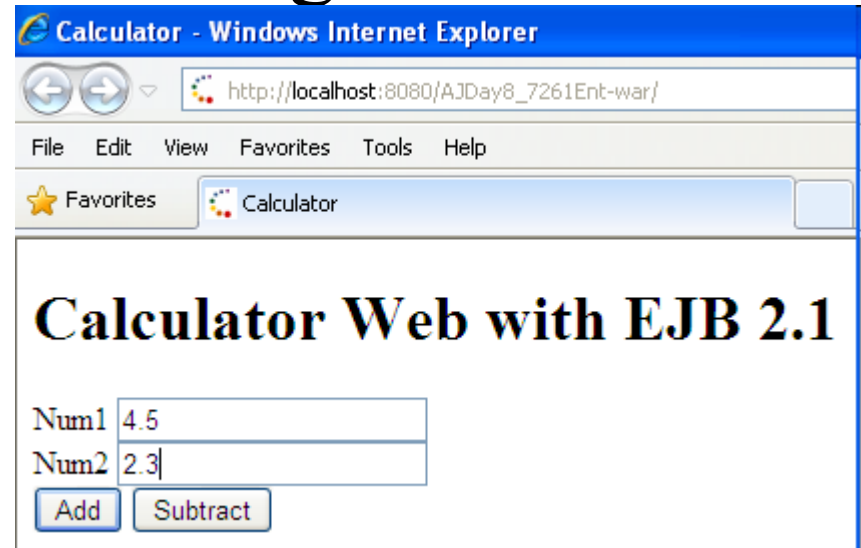
- **Step 2:** Creating the new corresponding bean
- **Step 3:** Building/ Modifying the business/callback methods on Beans
- **Step 4:** Mapping the JNDI to beans

**Creating stateless bean as whole steps in previous tutorials in EJB Development process on the Xxx-ejb module**

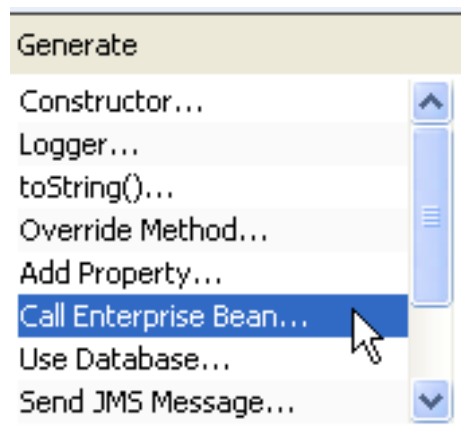
# EJB Implementation

## Creating GUI with Web Page and consumes

- Creating the GUI application

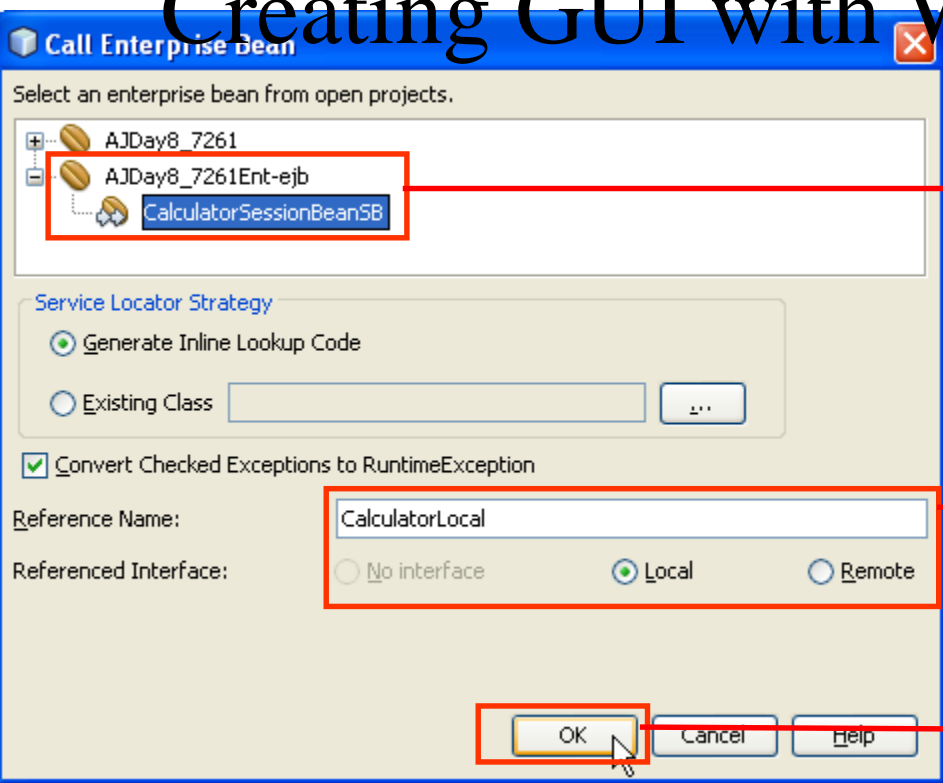


- Creating the Servlet to process and consume the EJB
  - Creating the reference to the EJB on the coding by right click on code
  - Then choose Insert Code, click Call Enterprise Bean ...



# EJB Implementation

## Creating GUI with Web Page and consumes



Choose the appropriate bean

Modify the Reference Name, then choose the scope reference of the Bean

Click Ok Button

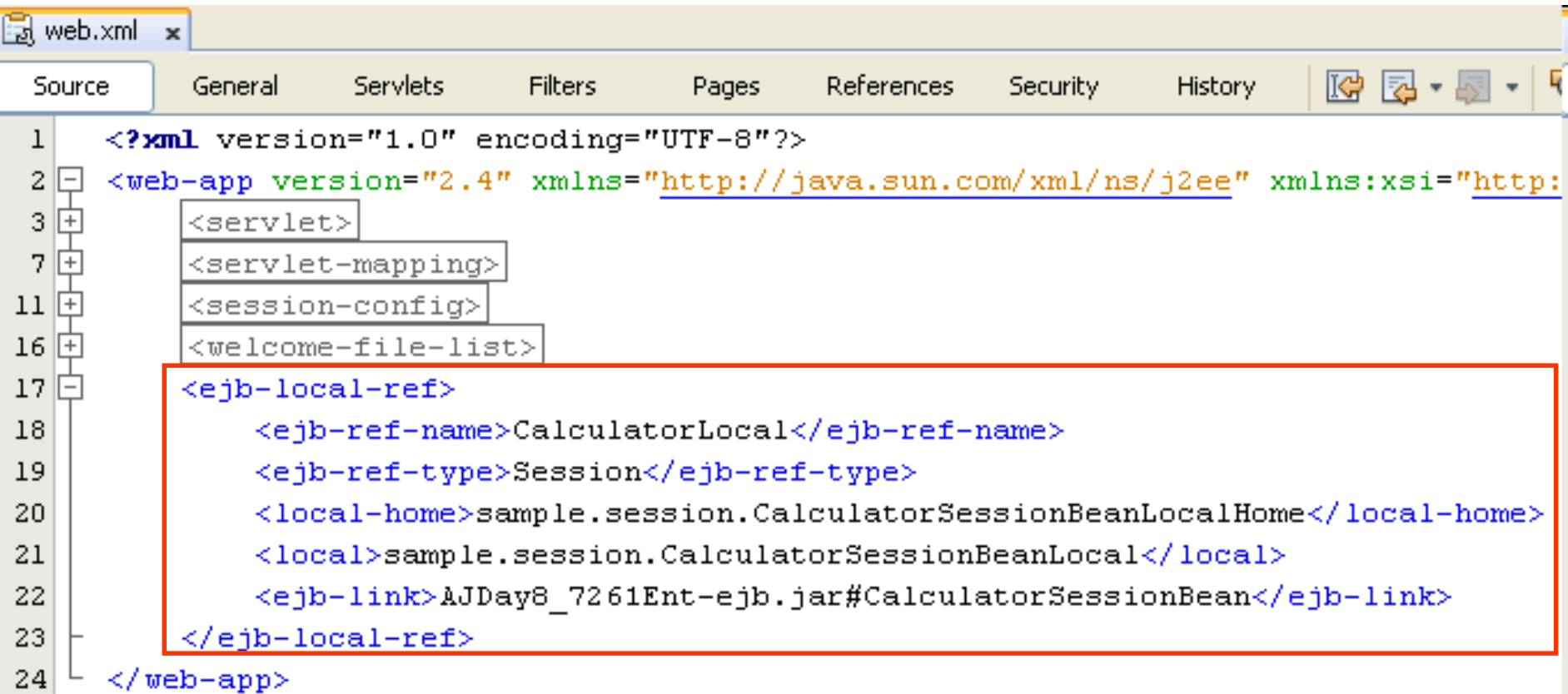
```

151 private CalculatorSessionBeanLocal lookupCalculatorSessionBeanLocal() {
152     try {
153         Context c = new InitialContext();
154         CalculatorSessionBeanLocalHome rv = (CalculatorSessionBeanLocalHome)
155             c.lookup("CalLocalJNDI");
156         return rv.create();
157     } catch (NamingException ne) {
158         Logger.getLogger(getClass().getName()).log(Level.SEVERE, "exception caught", ne);
159         throw new RuntimeException(ne);
160     } catch (CreateException ce) {
161         Logger.getLogger(getClass().getName()).log(Level.SEVERE, "exception caught", ce);
162         throw new RuntimeException(ce);
163     }
164 }
    
```

Modify the Reference Name that is named in jboss.xml at <[local-]jndi-name> tag

# EJB Implementation

## Creating GUI with Web Page and consumes



```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://
3      <servlet>
7      <servlet-mapping>
11     <session-config>
16     <welcome-file-list>
17     <ejb-local-ref>
18         <ejb-ref-name>CalculatorLocal</ejb-ref-name>
19         <ejb-ref-type>Session</ejb-ref-type>
20         <local-home>sample.session.CalculatorSessionBeanLocalHome</local-home>
21         <local>sample.session.CalculatorSessionBeanLocal</local>
22         <ejb-link>AJDay8_7261Ent-ejb.jar#CalculatorSessionBean</ejb-link>
23     </ejb-local-ref>
24 </web-app>
  
```

# EJB Implementation

## Creating GUI with Web Page and consumes

Modifying the code in servlet as following

```

26  * @author Trong Khanh
27  */
28  public class ProcessServlet extends HttpServlet {
29      /**...*/
36  protected void processRequest(HttpServletRequest request, HttpServletResponse response)
37      throws ServletException, IOException {
38      response.setContentType("text/html;charset=UTF-8");
39      PrintWriter out = response.getWriter();
40      try {
41          String button = request.getParameter("btAction");
42          if (button.equals("Add")) {
43              String n1 = request.getParameter("txtNum1");
44              String n2 = request.getParameter("txtNum2");
45              double num1 = Double.parseDouble(n1);
46              double num2 = Double.parseDouble(n2);
47              //Su dung JNDI de tim Initial Context Factory
48              Context context = new InitialContext();
49              //Tim Home Object
50              Object obj = context.lookup("CalJNDI");
51              //Xac dinh kieu cua Home Obj
52              CalculatorSessionBeanRemoteHome home =
53                  (CalculatorSessionBeanRemoteHome) PortableRemoteObject.narrow(obj,
54                      CalculatorSessionBeanRemoteHome.class);
55              //tao EJB obj tu home obj
56              CalculatorSessionBeanRemote ejbObj = home.create();
57              //goi business method tren ejb obj
58              double result = ejbObj.add(num1, num2);
59              out.println("Add " + n1 + " + " + n2 + " = " + result);
60          }
        }
    }

```

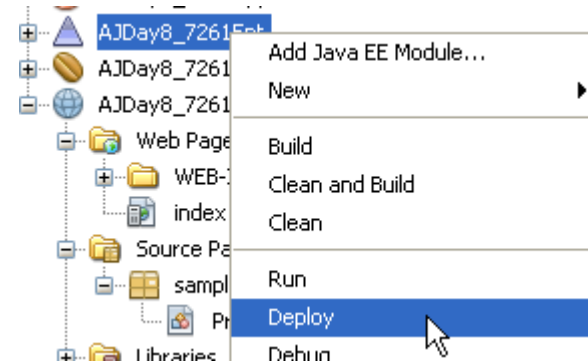
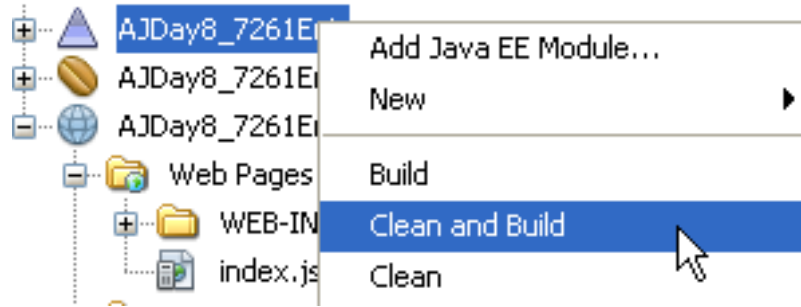
# EJB Implementation

## Creating GUI with Web Page and consumes

```
60     } else if (button.equals("Subtract")) {
61         String n1 = request.getParameter("txtNum1");
62         String n2 = request.getParameter("txtNum2");
63
64         double num1 = Double.parseDouble(n1);
65         double num2 = Double.parseDouble(n2);
66
67         Context context = new InitialContext();
68         Object obj = context.lookup("CalLocalJNDI");
69         CalculatorSessionBeanLocalHome home =
70             (CalculatorSessionBeanLocalHome) obj;
71         CalculatorSessionBeanLocal local = home.create();
72         double result = local.subtract(num1, num2);
73         out.println("Sub " + n1 + " - " + n2 + " = " + result);
74     }
```

# EJB Implementation

## Building, Deploying, and Executing



Output

JBoss6.1.0Final x AJDay8\_7261Ent (clean,dist) x

```
21:54:42,593 INFO [org.jboss.ejb.deployers.EjbDeployer] installing bean: ejb/AJDay8_7261Ent-ejb.jar#CalculatorSessionBean,uid20721493
21:54:42,593 INFO [org.jboss.ejb.deployers.EjbDeployer] with dependencies:
21:54:42,593 INFO [org.jboss.ejb.deployers.EjbDeployer] and supplies:
21:54:42,593 INFO [org.jboss.ejb.deployers.EjbDeployer] jndi:AJDay8_7261Ent/CalculatorSessionBean/sample.session.CalculatorSessionBeanLocal
21:54:42,593 INFO [org.jboss.ejb.deployers.EjbDeployer] jndi:CalJNDI
21:54:42,593 INFO [org.jboss.ejb.deployers.EjbDeployer] jndi:AJDay8_7261Ent/CalculatorSessionBean/sample.session.CalculatorSessionBeanRemote
21:54:42,593 INFO [org.jboss.ejb.deployers.EjbDeployer] jndi:CalLocalJNDI
21:54:43,468 INFO [org.jboss.ejb.EjbModule] Deploying CalculatorSessionBean
21:54:44,156 INFO [org.jboss.ejb.plugins.local.BaseLocalProxyFactory] Bound EJB LocalHome 'CalculatorSessionBean' to jndi 'CalLocalJNDI'
21:54:44,203 INFO [org.jboss.proxy.ejb.ProxyFactory] Bound EJB Home 'CalculatorSessionBean' to jndi 'CalJNDI'
21:54:44,531 INFO [org.jboss.web.tomcat.service.deployers.TomcatDeployment] deploy, ctxPath=/AJDay8_7261Ent-war
```



# EJB Implementation

## Building, Deploying, and Executing

c:\Programming\jboss-6.1.0.Final\server\default\deploy\\*

↑Name	Ext
↑ [..]	
[hornetq]	
[http-invoker.sar]	
[jbossweb.sar]	
[jms-ra.rar]	
[mod_cluster.sar]	
[ROOT.war]	
[security]	
[uuid-key-generator.sar]	
[xnio-provider.jar]	
admin-console-activator-jboss-beans	xml
AJDay8_7261Ent	ear

c:\Programming\jboss-6.1.0.Final\server\default\work\jboss.web\localhost\\*

↑Name	Ext	Size
↑ [..]		<DIR>
[..]		<DIR>
[AJDay8_7261Ent-war]		<DIR>
[invoker]		<DIR>

# EJB Implementation

## Building, Deploying, and Executing

The image illustrates the EJB implementation process. On the left, the JBoss IDE shows the project structure: JBoss6.1.0Final > Applications > Enterprise Applications > AJDay8\_7261Ent > AJDay8\_7261Ent-war. A right-click context menu is open over the 'AJDay8\_7261Ent-war' folder, with the 'Open in Browser' option highlighted.

In the center, a snippet of Java code is visible, showing a request object and a double value:

```
request.  
Double  
Double  
I de ti  
xt = ne  
context  
eu cua  
...
```

On the right, two browser windows are shown. The top window, titled 'Calculator - Windows Internet Explorer', displays the 'Calculator Web with EJB 2.1' application. It features two input fields: 'Num1' with the value '4.5' and 'Num2' with the value '2.3'. Below these are 'Add' and 'Subtract' buttons.

The bottom browser window shows the result of the addition operation. The address bar contains the URL: `http://localhost:8080/AJDay8_7261Ent-war/ProcessServlet?txtNum1=4.5&txtNum2=2.3&btAction=Add`. The page content displays the calculation: **Add 4.5 + 2.3 = 6.8**.

The bottom-most browser window shows the result of the subtraction operation. The address bar contains the URL: `http://localhost:8080/AJDay8_7261Ent-war/ProcessServlet?txtNum1=4.5&txtNum2=2.3&btAction=Subtract`. The page content displays the calculation: **Sub 4.5 - 2.3 = 2.2**.

# **Build the simple enterprise application**

## **Web Requirements**

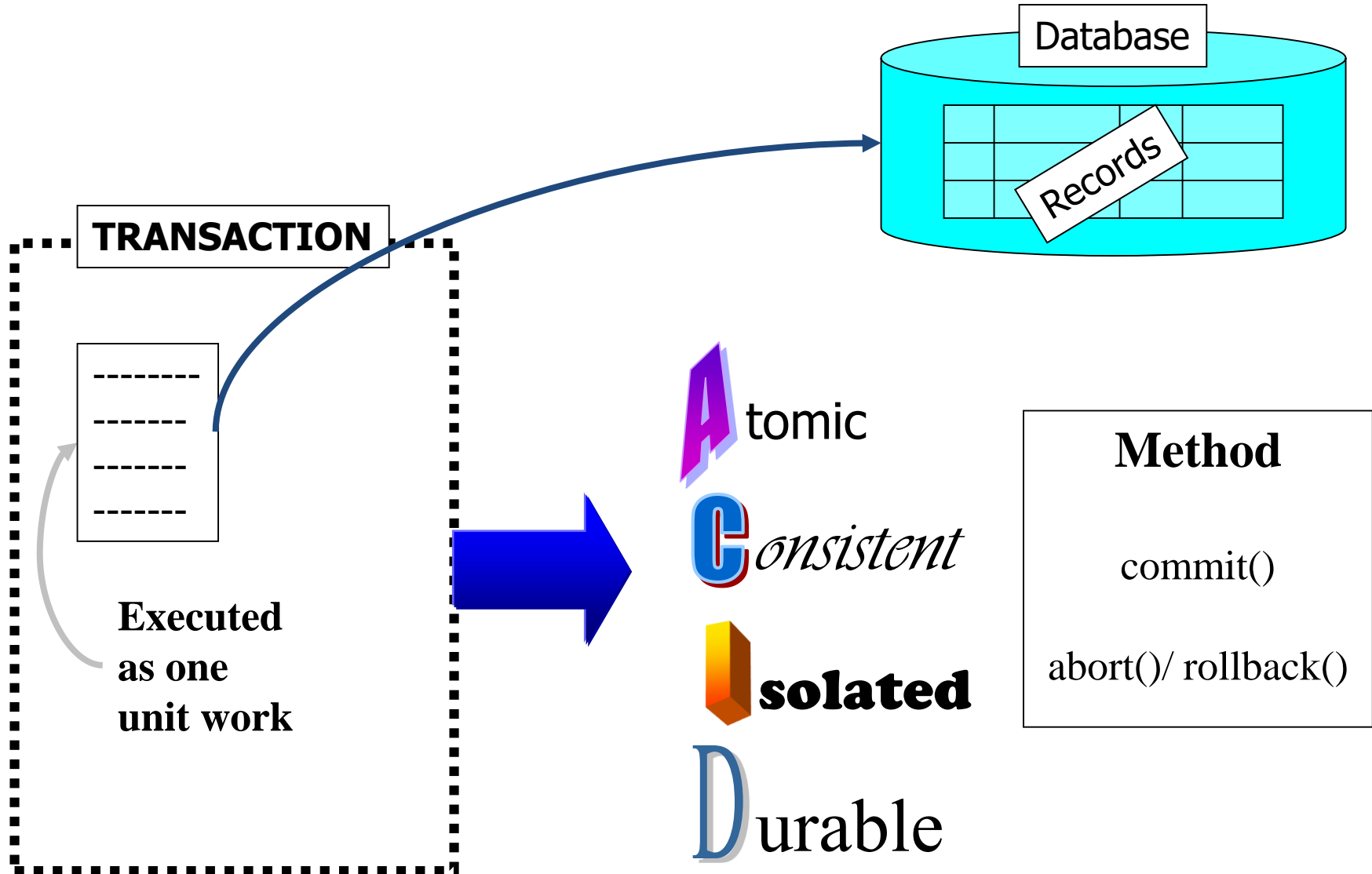
- **Take your self. It is easy**

# Appendix – EJB Container

- **Acts as an interface between an enterprise bean and client**
- **Provides following services**
  - Security
  - Transaction Management
  - Persistence
  - Life Cycle management
  - Remote Client Connectivity
- **Responsible for providing several APIs**
  - J2SE API
  - EJB Standard Extension
  - JDBC Standard Extension
  - JNDI Standard Extension
  - JMS Standard Extension
  - JavaMail Standard Extension (for Sending mail only)
  - JAXP (Java API for XML Processing)
  - JTA Standard Extension (Only UserTransaction interface)

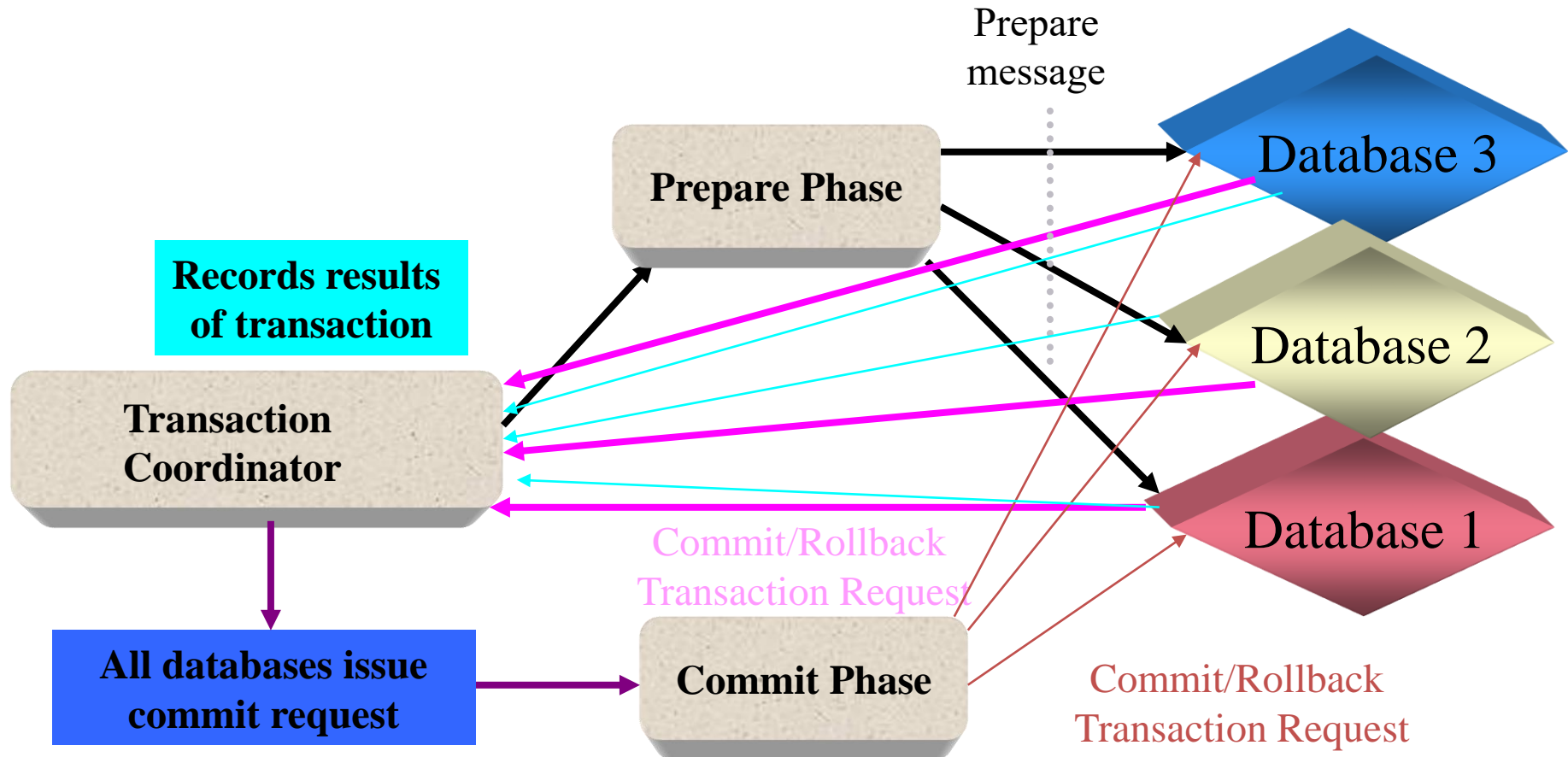
# Enterprise Java Beans

## Transactions



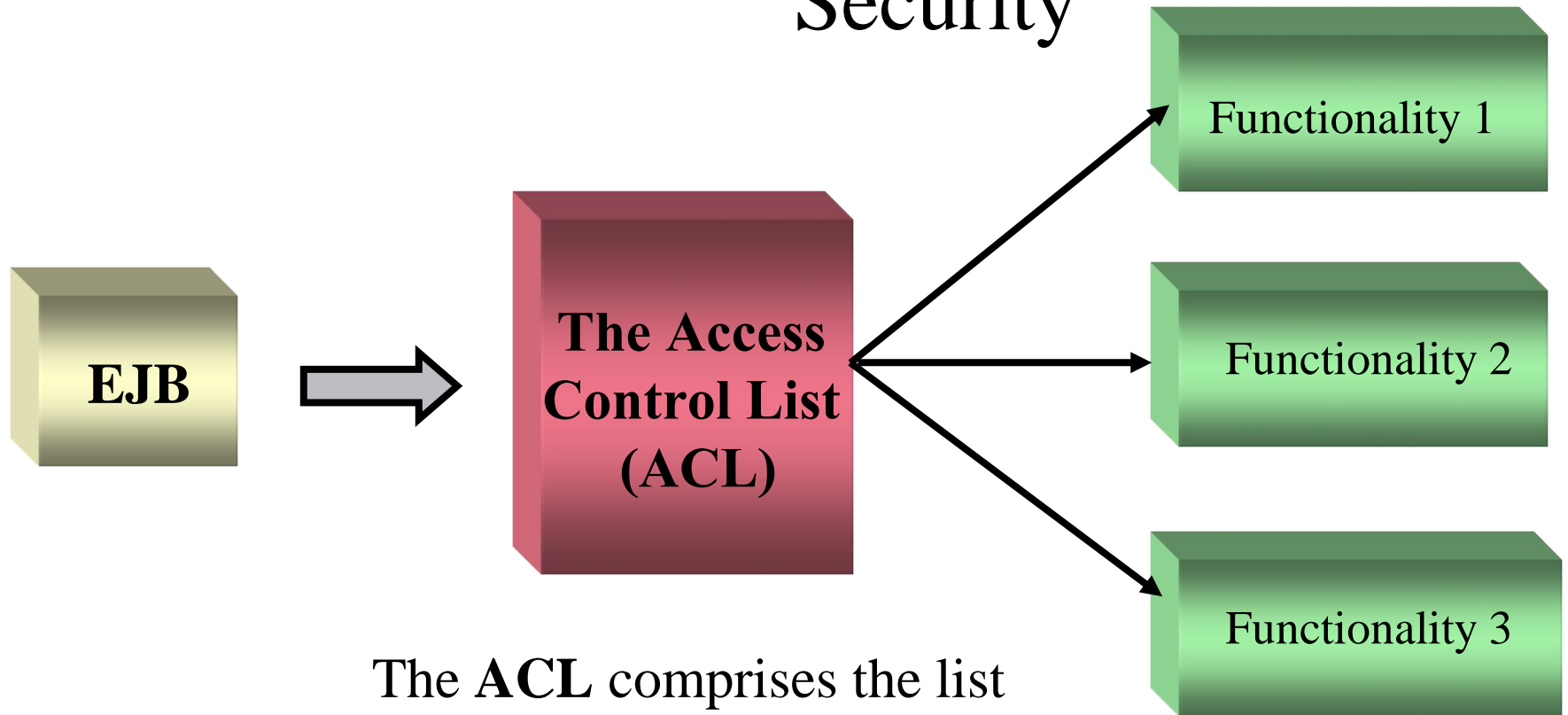
# Enterprise Java Beans

## Two phase Commit Protocol



# Enterprise Java Beans

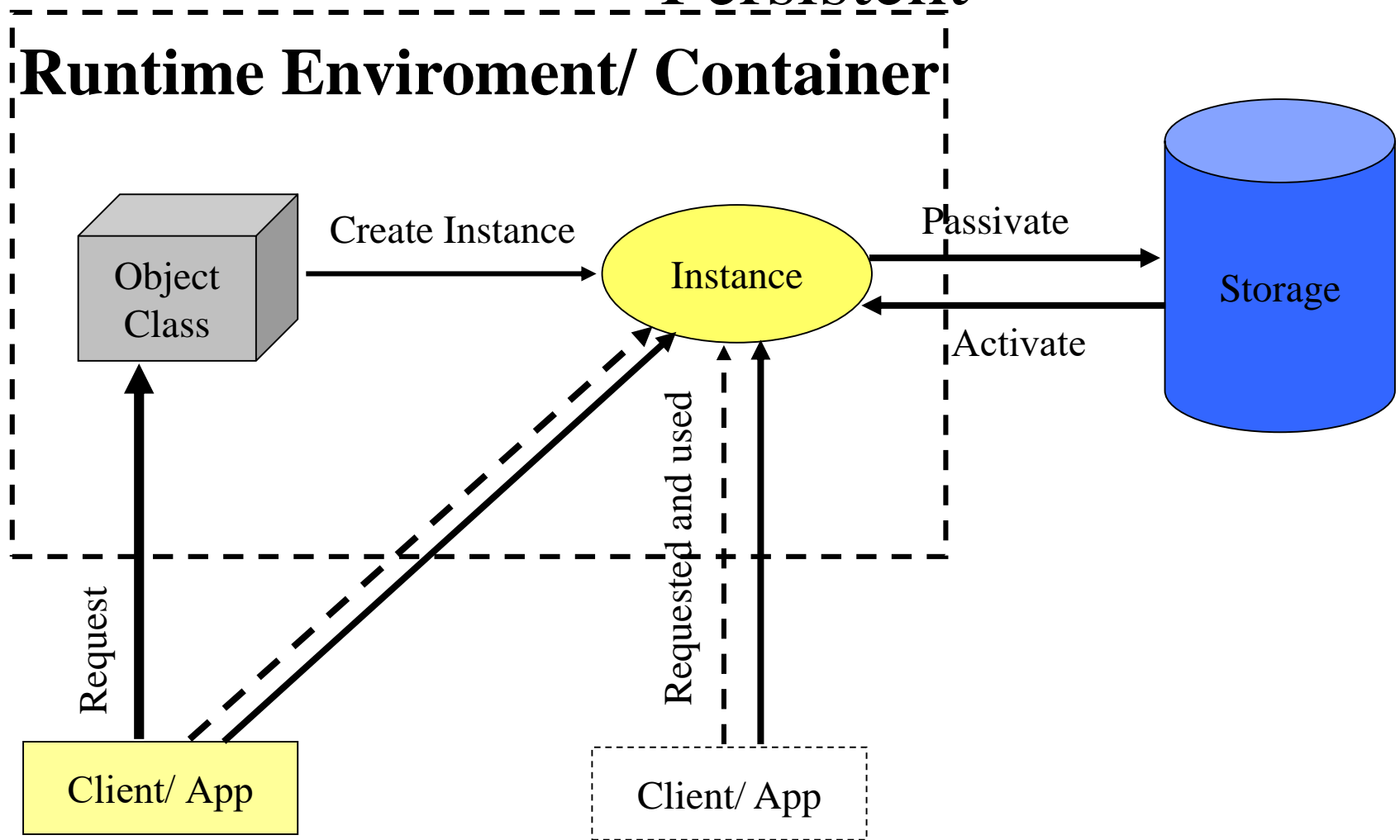
## Security



The **ACL** comprises the list of persons who are allowed to access particular sections of functionality.

# Enterprise Java Beans

## Persistent



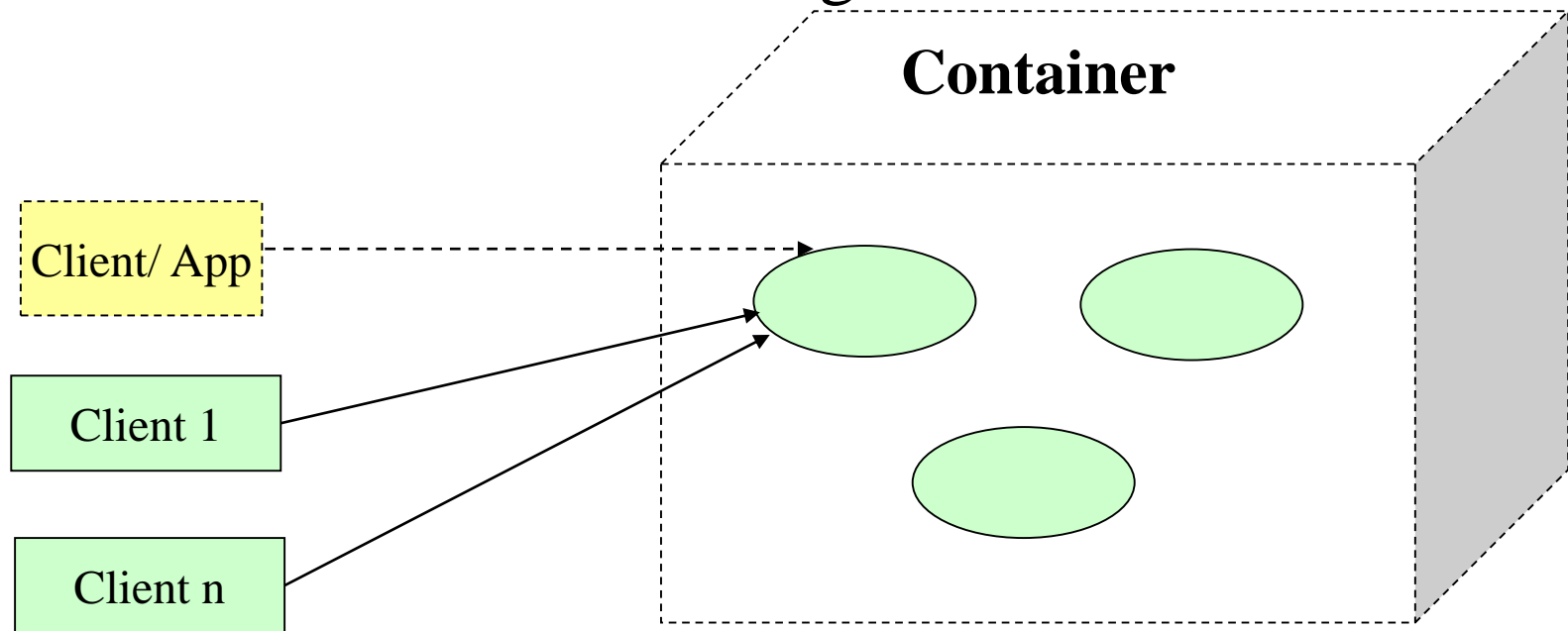
Persistence can be defined as saving the state of an object to a constant storage.



# Enterprise Java Beans

## Management of Multiple Instance

- Instance Passivation
- Instance Pooling
  - **Advantages:** reduces the memory allocation and garbage-collection cycles
- Database Connection Pooling



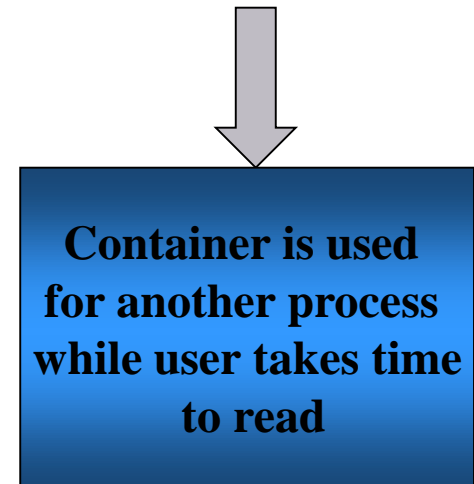
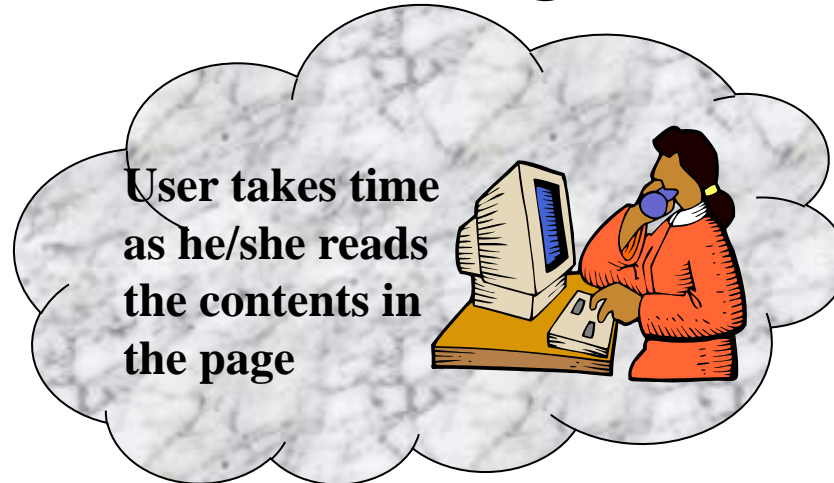
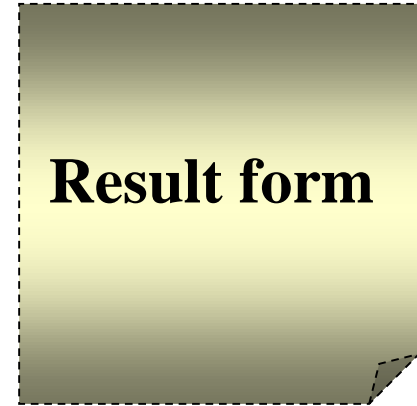
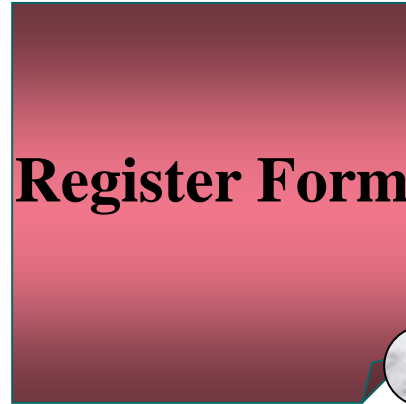
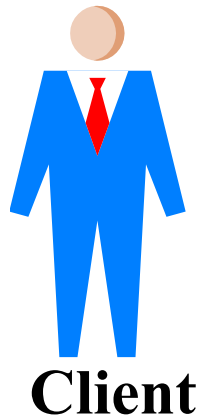
# Enterprise Java Beans

## Resource and lifecycle Management

- Management of resources **enhances the scalability** of a multi-tier architecture.
- The container **provides resource-management services** for resources such as:
  - Threads
  - Socket Connections
  - Database Connections
- EJB Container **responsible the life cycle** of the bean (control the life of the bean).
  - **Notes:** The life time of the bean is managed by EJB server
- EJB Container **instantiates, destroys and reuses** the beans required.
- EJB Container **supports instance pooling**.

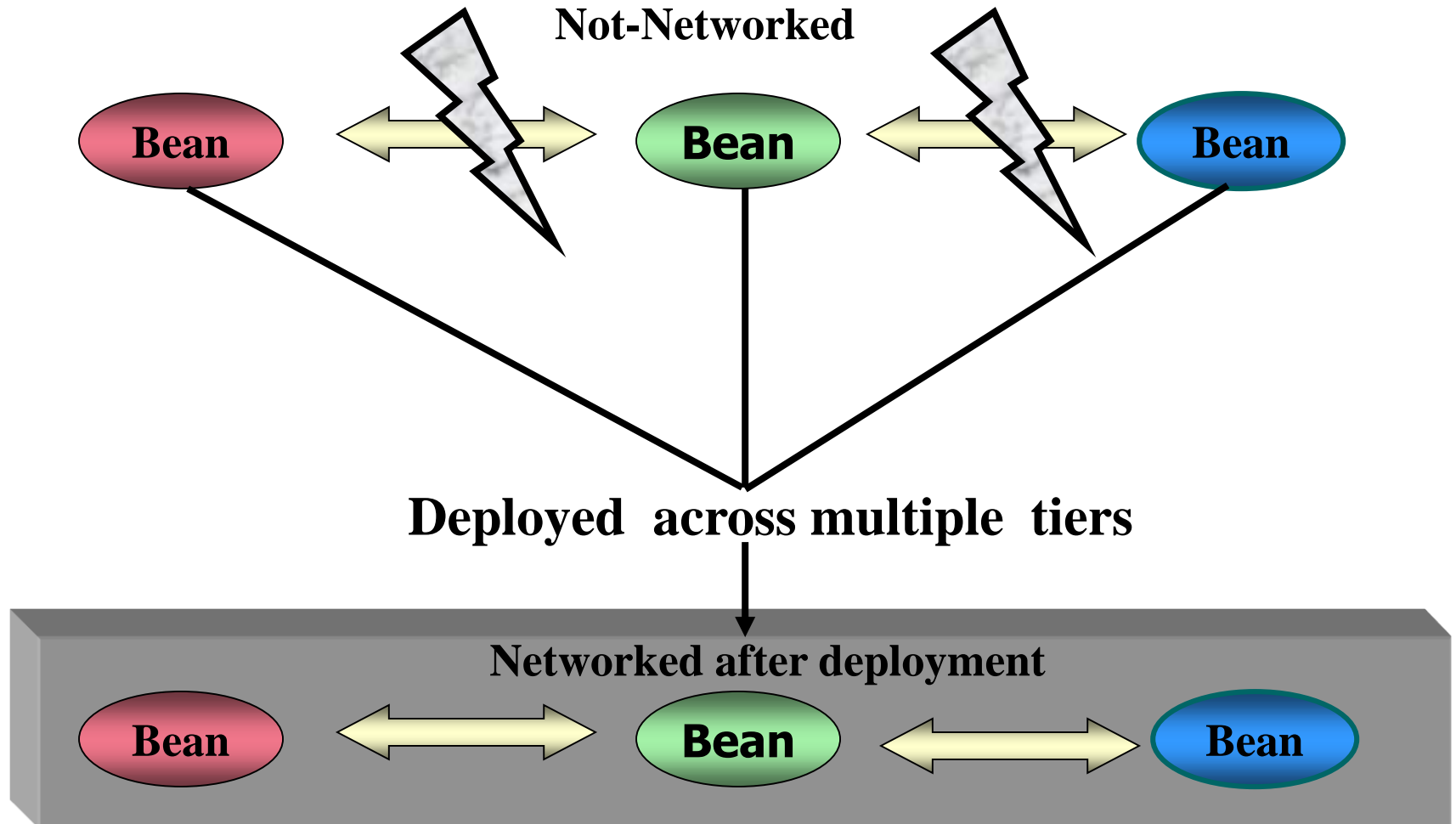
# Enterprise Java Beans

## State Management



# Enterprise Java Beans

## Remote Accessibility



# Enterprise Java Beans

## Location Transparency

- Clients **do not know where** the components are, and whether these components are local or remote
- **Advantages**
  - **Reusable**
  - Vendors can provide value additions in terms of
    - Ability to perform maintenance on a system connected to a network because location transparency **allows a different system provide components for a particular client**
    - **Install new software**
    - **Upgrade the components on a system**
  - When a **system crashes**, the **requests are redirected to another system without the client getting to know about the crash.**

# Enterprise Java Beans

## Components of EJB

- The Enterprise Java Bean is a **server-side component** that is **employed on a distributed multi-tier environment**.
- EJB does **not allow multithreading** (single thread)
- Important Object of EJB is Bean
- **Types**
  - **Session Bean** – **Represents business process without having persistent storage mechanism**
    - Stateless Session Bean
    - Stateful Session Bean
  - **Entity Bean** – **Persists across multiple sessions and multiple clients & Having persistence storage mechanism**
    - Bean-managed Persistence [BMP]
    - Container-managed Persistence [CMP]
  - **Message-driven Bean** – **Asynchronous messaging between components of EJB.**

# Appendix

## J2EE Terminologies in J2EE design patterns

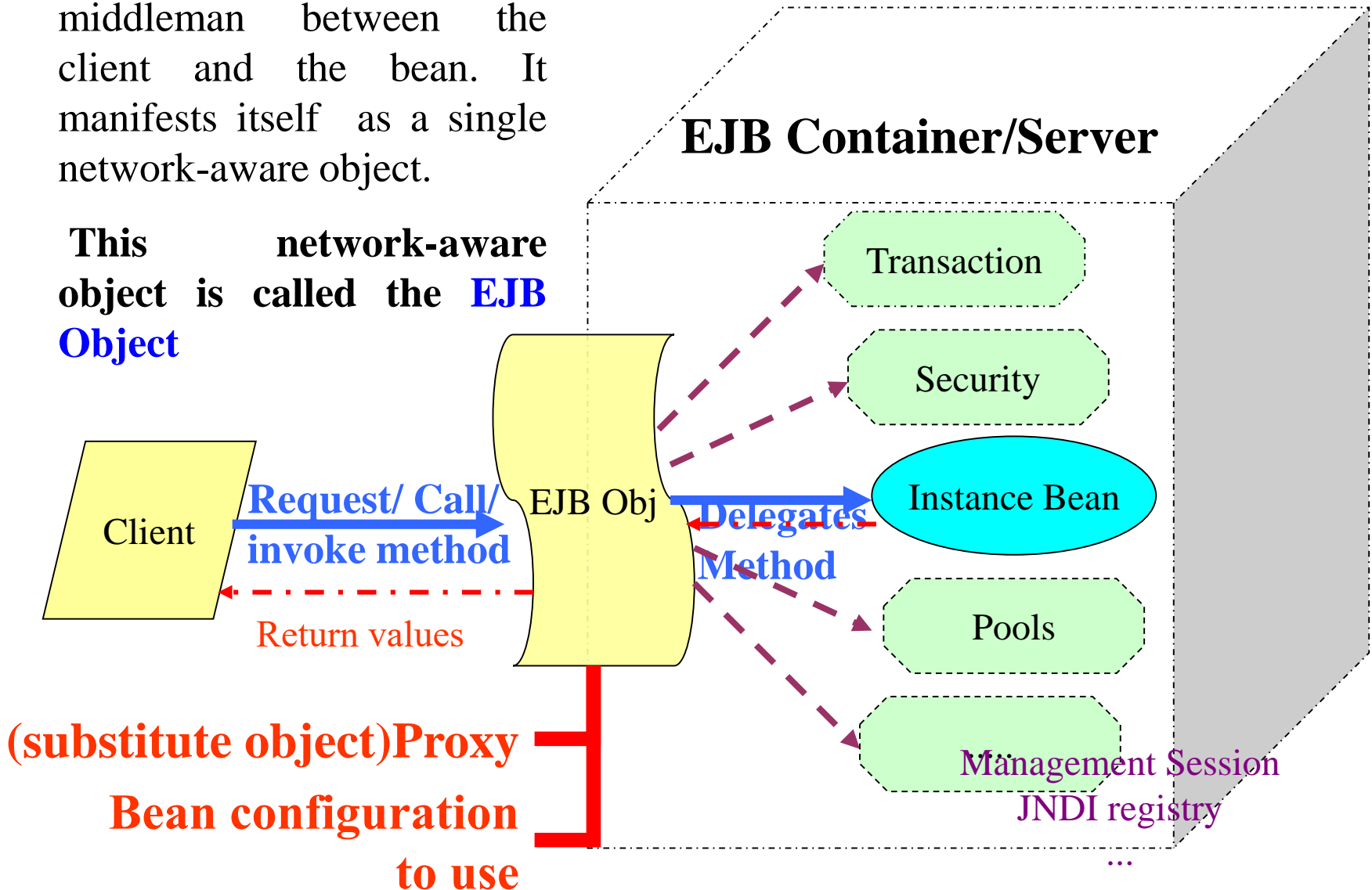
- **Service Locator**
  - **Implement** and **encapsulate** service and component lookup
  - **Hides** the implementation details of the lookup mechanism and encapsulates related dependencies
  - **Transparently locate business components and services in a uniform manner (ex: EJB Home Interface)**
- **Business Delegate**
  - **Encapsulate** access to a business service
  - **Hides** the implementation details of the business service, such as lookup and access mechanisms
  - **Hide clients from the complexity of remote communication with business service components**
- **Abstract Factory**
  - Provides a way to **encapsulate a group of individual factories** that have a common theme
  - **Separates the details of implementation of a set of objects from their general usage (ex: EJBHome interface)**

# What Constitutes an EJB?

## EJB Objects

The container is the middleman between the client and the bean. It manifests itself as a single network-aware object.

This network-aware object is called the **EJB Object**





# What Constitutes an EJB?

## EJB Objects

- Interface **javax.ejb.EJBObject**

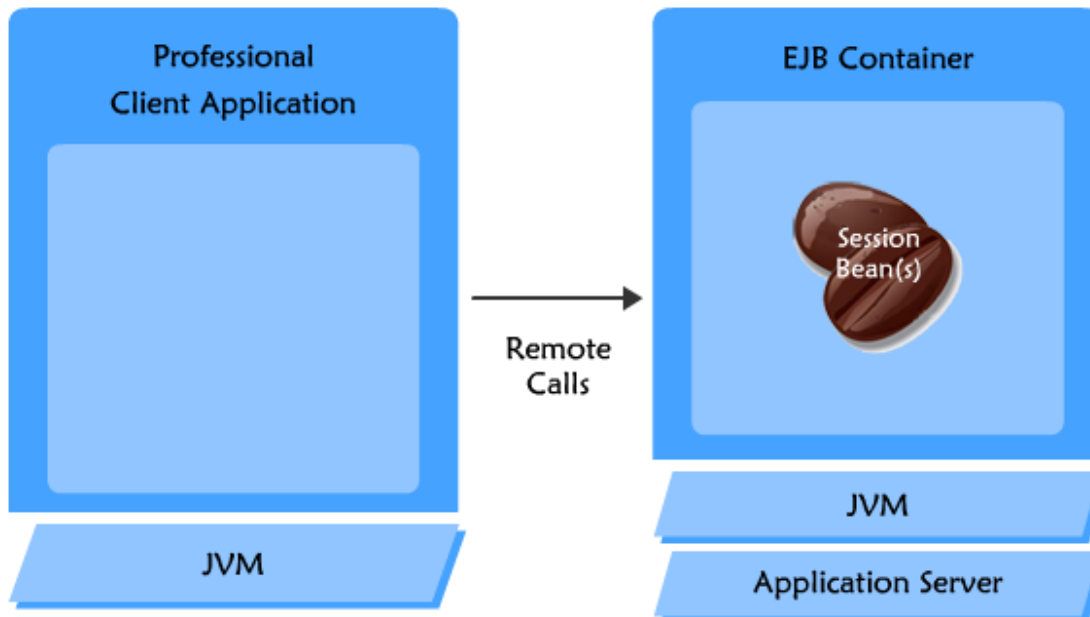
Methods	Descriptions
getEJBHome()	<b>Retrieves</b> the <b>reference</b> to the corresponding Home Object
getPrimaryKey()	<b>Return</b> the <b>Primary Key</b> for EJB Object (Entity Bean)
remove()	<b>Destroy</b> EJB Object ( <b>delete</b> the bean from the underlying persistent store, means delete a record on DB – Entity Bean)
getHandle()	Obtain the <b>handle</b> (is a persistent reference to the EJB Object) for the EJB Object
isIdentical()	Checks whether two EJB Objects are similar

- Relationship between Java RMI and EJB Objects
  - public interface javax.ejb.EJBObject **extends** java.rmi.Remote (*The physical location of remote object is hidden from the Client RMI*)
  - Can be called from a different JVM
  - **Offers Location Transparency** (Portability of Client Code)

# What Constitutes an EJB?

## Remote Interface

- **Is used when the client application runs on a separate JVM than the one that is used to run the Session beans in an EJB Container**
  - **The method invocation in remote business interfaces are received from networked clients**
  - **The method parameters and the return values are copied and is known as call-by-value**

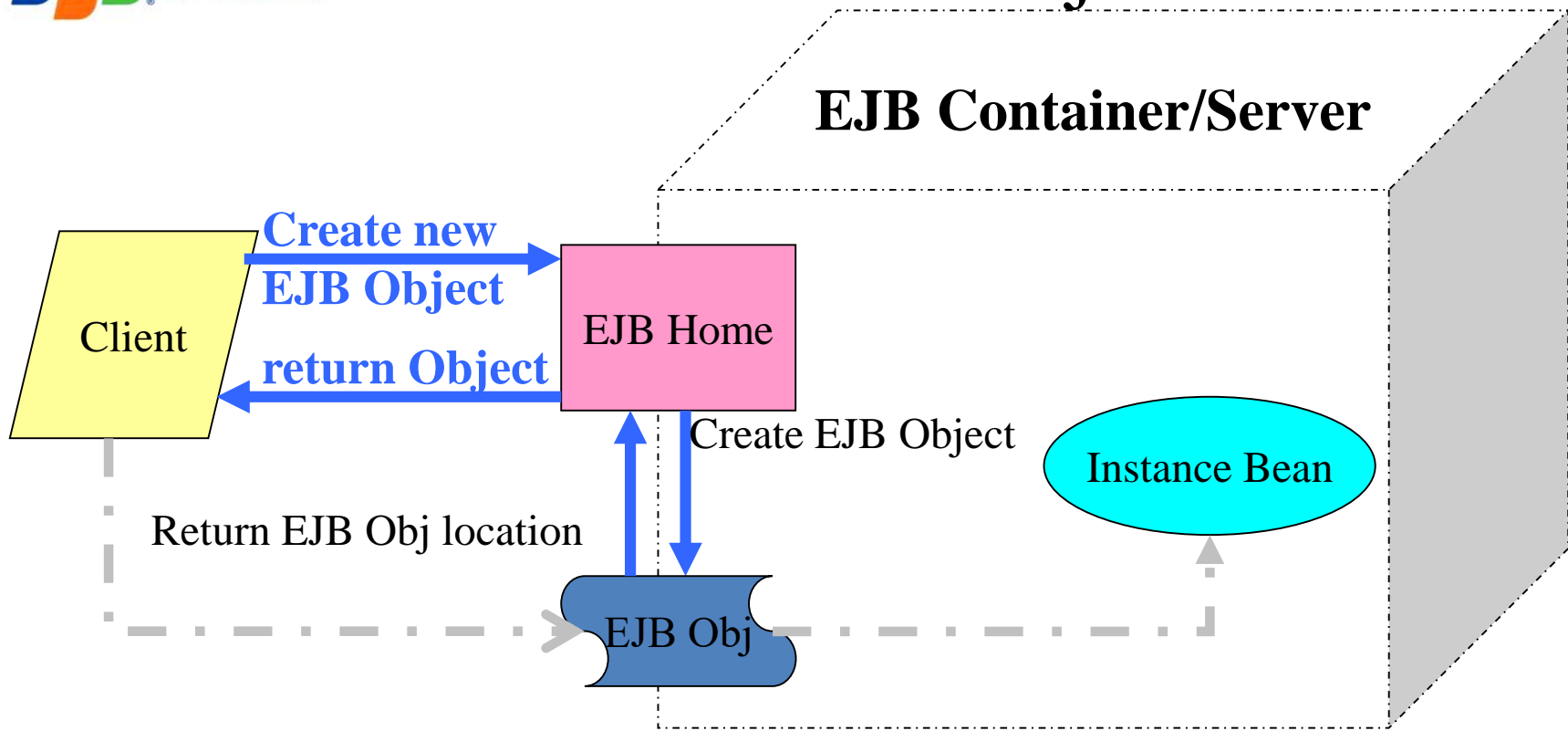


# What Constitutes an EJB?

## Local Interface

- EJB 2.0 can **expose** their **methods** to **clients** through new **Local Interface**
- Standard Java interface which **allows** the **beans** to **expose** its **methods** to other bean **reside** within the **same container** (local clients)
- **Eliminate** the **overhead** of the **remote method call** (java.rmi.RemoteException)
- Is used when the application **uses** the **same JVM** to run both the **client application** and the **Session beans**
  - The method **parameters** and the **return values** are **not copied** and **hence**, it is known as **call-by-reference**
  - Speed up in processing and efficiency
- **Not inherit from RMI** (extends javax.ejb.EJBLocalObject)

# Home Objects



- Client code will request for an object from **the EJB Object Factory**, which know as **the home object** (instantiates EJB Object)
- **Responsibilities**
  - Create (Instantiate) EJB Objects
  - Initial information for EJB Object s
  - Find or search for existing EJB Objects(Entity Bean)
  - Remove EJB Objects (deletes the bean from the underlying persistent store)
  - Select EJB Objects (Entity Bean)

# What Constitutes an EJB?

## Home Objects

- Interface `javax.ejb.EJBHome` (extends `java.rmi.Remote`)

Methods	Descriptions
<code>getEJBMetaData()</code>	<b>Retrieve information</b> about <b>EJB</b> (Beans' information) that are <b>being worked on</b> . The information received is encapsulated in the <code>EJBMetaData</code> object, which returns the method.
<code>remove()</code>	<b>Destroy EJB Object</b> following <ul style="list-style-type: none"><li>- <b>Passing the <code>javax.ejb.Handle</code> object</b>, which <b>remove EJB Object</b> that is based on the already retrieved EJB Handle</li><li>- <b>Passing a primary key to remove beans (one record)</b> from the <b>underlying persistent store</b>.</li></ul>

# Appendix

## J2EE Terminologies in J2EE design patterns

- **Transfer Object**

- A serializable class that **groups related attributes, forming a composite value**
- A class is used as **the return type of a remote business method**
- Clients receive instances of this class by calling coarse-grained business methods, and then locally access the fine-grained values within the transfer object. Fetching multiple values in one server roundtrip decreases network traffic and minimizes latency and server resource usage.

- **Session Façade**

- A higher-level business component contains and centralizes complex interactions between lower-level business components
- Is implemented as a session enterprise bean.
  - It **provides** clients with a **single interface** for the functionality of an application or application subset.
  - It also **decouples lower-level business components** from one another, making designs more flexible and comprehensible

# ***APPLICATION/EJB SERVER***

- **Provides many services**
  - **Network connectivity** to the container
  - **Instance Passivation** – Temporarily swap out a bean from memory storage
  - **Instance Pooling** – Multiple clients share same instance
  - **Database Connection Pooling** – Contains a set of database connection
  - **Precached Instances** – Maintains cache, which contains information about the state of the EJB
- **Other services**
  - Runtime Environment
  - Support the containers interaction
  - Process and Thread Management
  - Receive and process requests
  - System Resource Management

# Enterprise Java Beans

## Session Beans

- **Survive only as long as the client exists.**
- **Are created solely in response to a call made by the client.**
- **Are used to implement business logic, business rules and the workflow**
  - **Ex:** Check login, computation, Document process, book tickets
- **Are not shareable between clients** (only one client can deal with that particular session bean)
- **Stateless Session Bean**
  - **Single Request**
  - **Stateless**
  - **Redirect the others bean** when the errors occur.
  - **Ex:** check Login
- **Stateful Session Bean**
  - **Multiple requests** (The life cycle is very complex)
  - **Keep track**
  - **Persistence**
  - **Ex:** Shopping Cart



# Enterprise Java Beans

## Entity Beans

- DB Model: Entity beans are the **object representations of the underlying data and provide access to data.**
- The components **are persistent.**
- **Have a long life** because they can be reconstructed by reading the data back from the permanent DB.
- Data in Relational DB can be treated as **real objects** and an entire chunk of data from DB can be read at once into an entity bean component. (Allow the transformation of the data in the DB into Java Objects)
- EJB Container **synchronous** between EJB and DB
- **BMP** – Bean managed Persistent Entity Bean
  - **The developer has to write** the code (**CRUD**) to interpret the fields stored in the memory to an underlying DB
- **CMP** – Container managed Persistent Entity Bean
  - The **container perform** all the operations
  - The **developer** has to **describe** that needs to persist and inform the container
  - The developer **concentrates** the **business processes.**

# Enterprise Java Beans

## Message Driven Beans

- New type in EJB Version 2.0
- Process **messages asynchronously** (The bean acts as a message listener)
- **Communication** between software components/application (onMessage(Message msg) method)
- **Similar** to Stateless session bean
- Created and Controlled by Container.
- Do **not** have a **home and remote interface**.
- Support both container managed (which may deliver message within a transaction context) and bean managed transactions.

# JNDI

## Overview

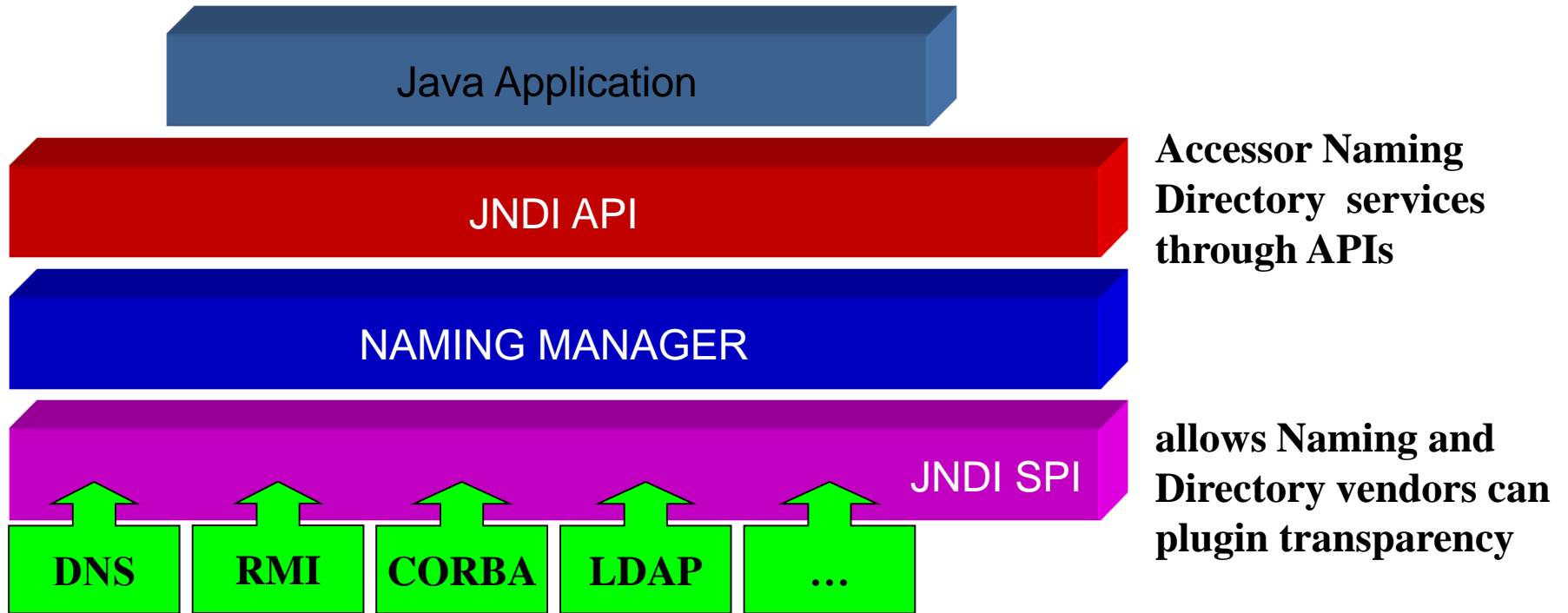
- A **naming service** (which has its own set of rules for creating valid names) allows you **finding an object** in a system based on the **name associated with the object** which is called “**binding**”.
- A **directory service** is an **extension of a naming service** (an object is also associated with a name, which can be look up, and allowed to have attributes)
- Java Naming and Directory Interface (**JNDI**) is a **specification** for accessing naming and directory services
- Java Naming and Directory Interface **provides the naming and directory functionality** to Java applications.
- Provides a **standard interface to locate** the components, users, networks, and services placed **across the network**.
- **Bridges the gap between directory services** and makes it possible for the developer to write portable naming and directory services
- **JNDI abstracts the code from a directory service and allows the user to plug in a different directory services.** (without changing the service code)

# JNDI

## Overview

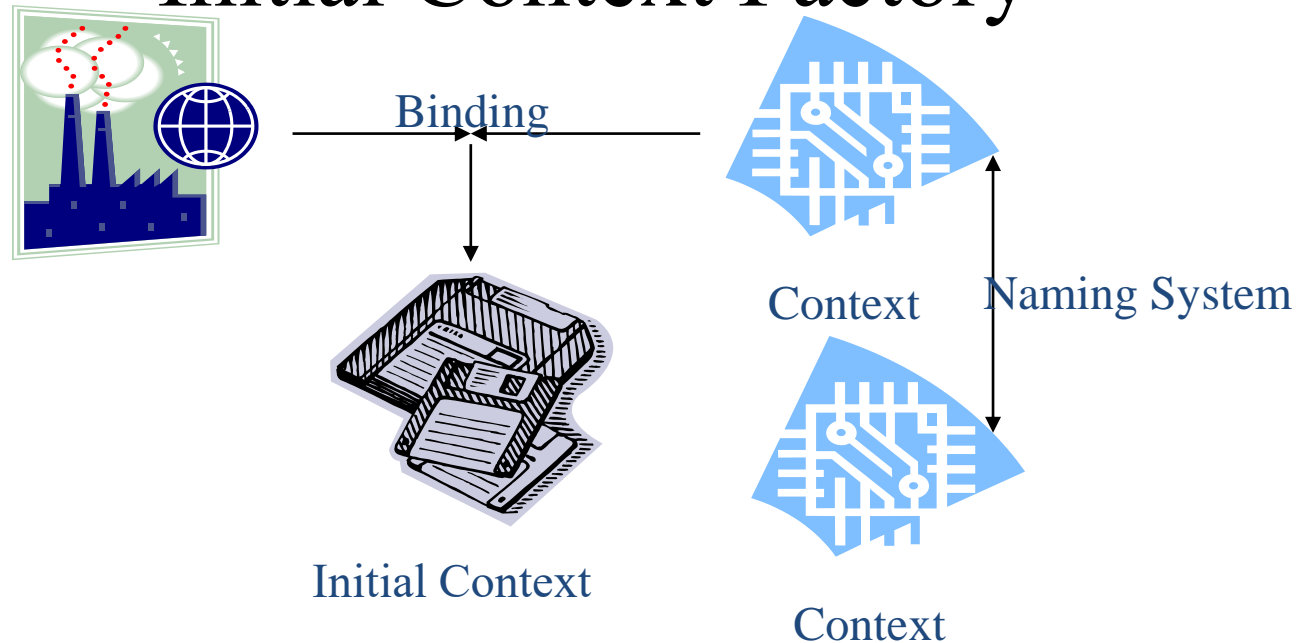
- JNDI provides **javax.naming.\*** interface
- JNDI separate two parts
  - **JNDI API**
  - **JNDI SPI**
- Naming Concepts of JNDI
  - **Atomic**: It's a simple and basic name. Ex: Windows
  - **Compound**: the collection of one or more atomic names.
    - Ex: C:\Windows\System32
  - **Composite**: A name has multiple naming system.
    - Ex: <http://localhost:8080/JSP/index.html>

# JNDI Architecture



# JNDI

## Initial Context Factory



- Initial Context Factory is the **point** where **all naming and directory operations** are **first performed**.
- When the initial context **is acquired**, all information **pertaining** to this must be provided to JNDI.
- The **internal storage** of JNDI **emulates tree data structure**. Each InitialContext **acts like an internal node** and **each reference to the resources acts like the leaves**
- The directory context or directory object is another type of context. It is used to **define methods for inspecting and modifying attributes** associated with a *directory object*.