

## IBM WEBSPHERE WORKSHOP - LAB EXERCISE

# Build a WebSphere Application

### **What This Exercise is About**

You will build a WebSphere application using the Application Assembly Tool (AAT). The AAT is a WebSphere tool that packages application components into Modules, and combines Modules into an Enterprise Application. The Enterprise Application is contained in a single Enterprise Archive file with a .ear extension. The Enterprise Application contains all the application components, as well as deployment descriptors, which describe how to deploy in a J2EE compliant server like WebSphere Application Server V4.0 Advanced Edition. The Enterprise Application also contains WebSphere specific information, like IBM extensions and bindings.

### **User Requirement**

This lab requires one system with Windows NT Server 4.0, ServicePack 6. IBM DB2 Universal Database Version 7.2a is required. This lab is based on IBM WebSphere Application Server, Release 4.0.1 Advanced Single Server Edition, driver AEs 4.0.1 a0131.07. The MyBank application components must be installed in the C:\MyBank directory.

### **What You Should Be Able to Do**

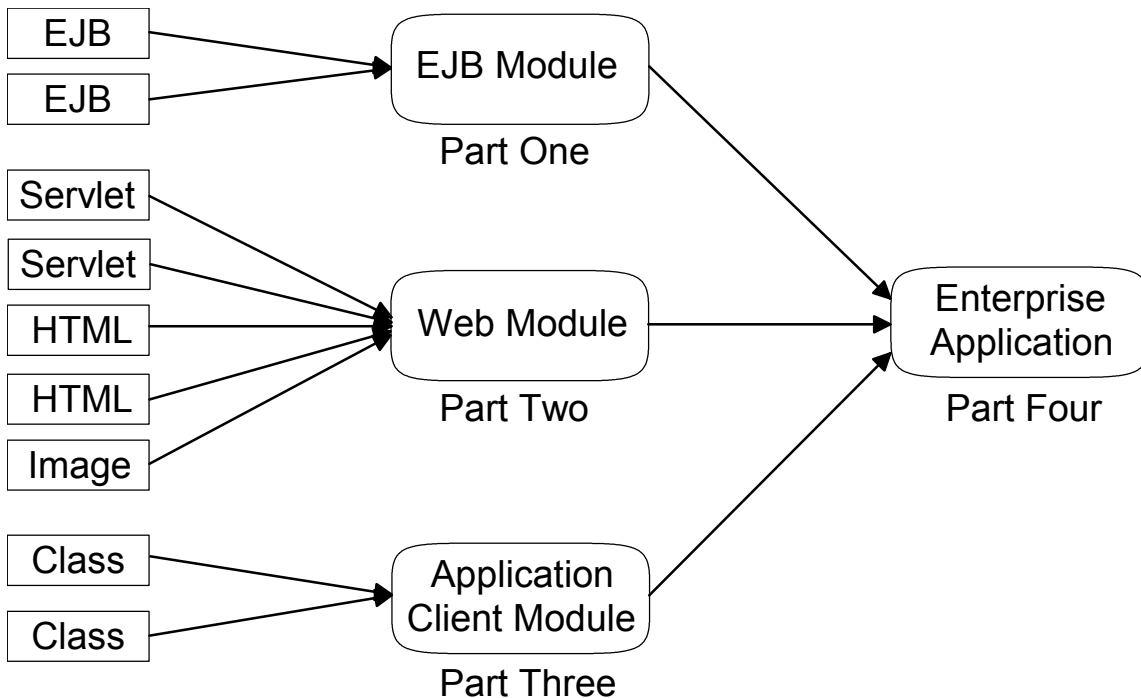
You should be able to use the Application Assembly Tool to package application components into an Enterprise Archive file ready to install on WebSphere Application Server V4.0 Advanced Edition. Application components are provided, including Enterprise JavaBeans (EJBs), Servlets, HTML Web pages, and image files. You will start the Application Assembly Tool, and use wizards to combine components into three Modules. The EJB Module contains the Enterprise JavaBeans. The Web Module contains the Servlets, HTML and image files. The Application Client Module contains classes for a stand alone Java client. You will set properties for the deployment descriptors and IBM extensions and bindings. Finally, you will assemble the Modules into a single Enterprise Archive file, ready to install in WebSphere V4.0.

### **Introduction**

You will build an application called MyBank, which simulates a very simple online bank. The application allows you to create checking and savings accounts, check your account balances, and transfer funds between your accounts. The application has two interfaces. The Web interface runs in a browser. The Java client shows how a stand alone Java application can access the balance and transfer functions in the EJBs without going through an HTTP Server.

This lab consists of four parts:

- Part One: Build an EJB Module from Enterprise JavaBean classes  
 Part Two: Build a Web Module from Servlets, HTML and image files  
 Part Three: Build an Application Client Module from Java classes  
 Part Four: Build an Enterprise Application from the Three Modules



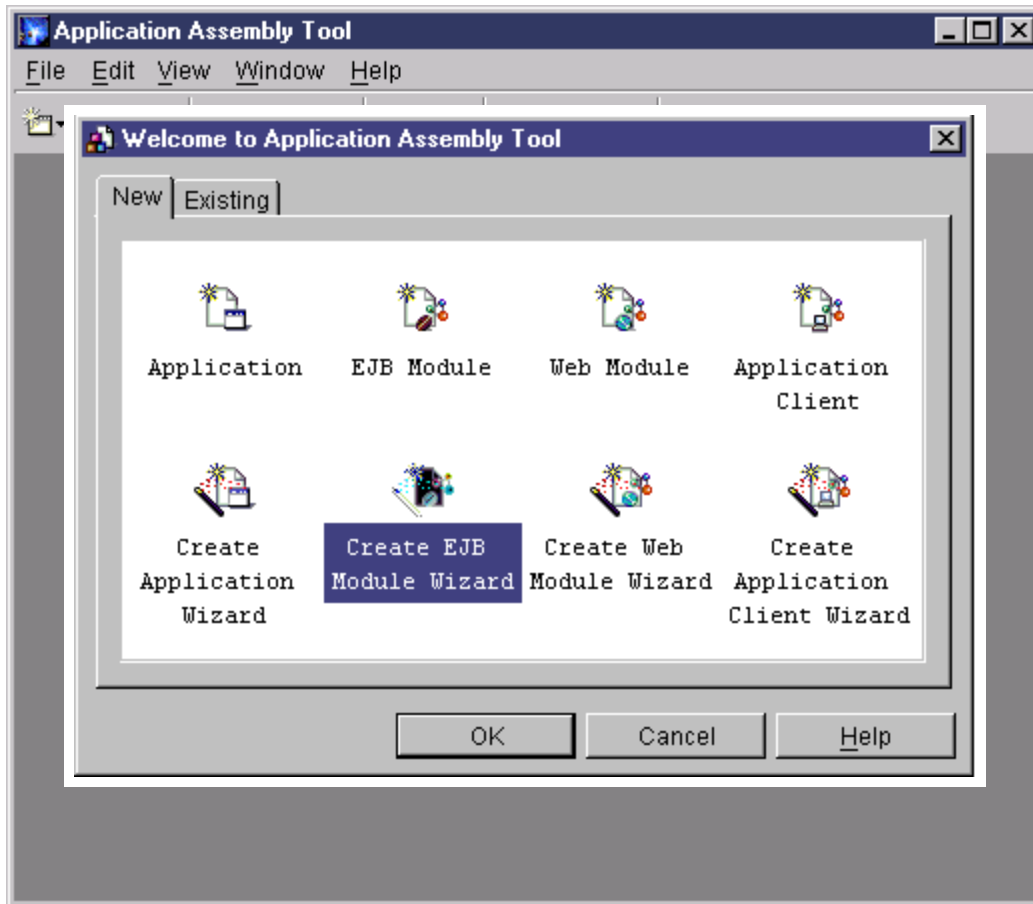
## Exercise Instructions

The Application components are provided under the **C:\MyBank** directory. The **Client** subdirectory contains classes for the stand alone Java application client. The **Ejb** subdirectory contains all the classes for the Enterprise JavaBeans. The classes are in a directory structure corresponding to the package structure. All of the Java components for the MyBank application are in the WebSphereSamples.AccountAndTransfer package. The **Solutions** subdirectory contains completed modules and application files. If you have trouble, copy the file from Solutions to MyBank and proceed. The **Web** subdirectory contains all of the Servlet class files, HTML and image files.

### Part One: Build an EJB Module from Enterprise JavaBean classes

- \_\_\_ 1. Start the Application Assembly Tool (AAT) from the **assembly.bat** batch file in the WebSphere \bin directory.

- \_\_ Click the Windows **Start** button, then select: **Run**
- \_\_ In the **Open** field of the run window, type: **cmd**
- \_\_ Click **OK**.
- \_\_ Type **set | find "WAS"** to view the **WAS\_HOME** environment variable
- \_\_ If **WAS\_HOME** is NOT set to **C:\WebSphere\SingleServer**, type:  
**Set WAS\_HOME=C:\WebSphere\SingleServer** and hit **Enter**
- \_\_ At the prompt in the Command window, type: **cd %WAS\_HOME%\bin**
- \_\_ At the prompt in the Command window, type: **assembly**
- \_\_ Press the **Enter** key.

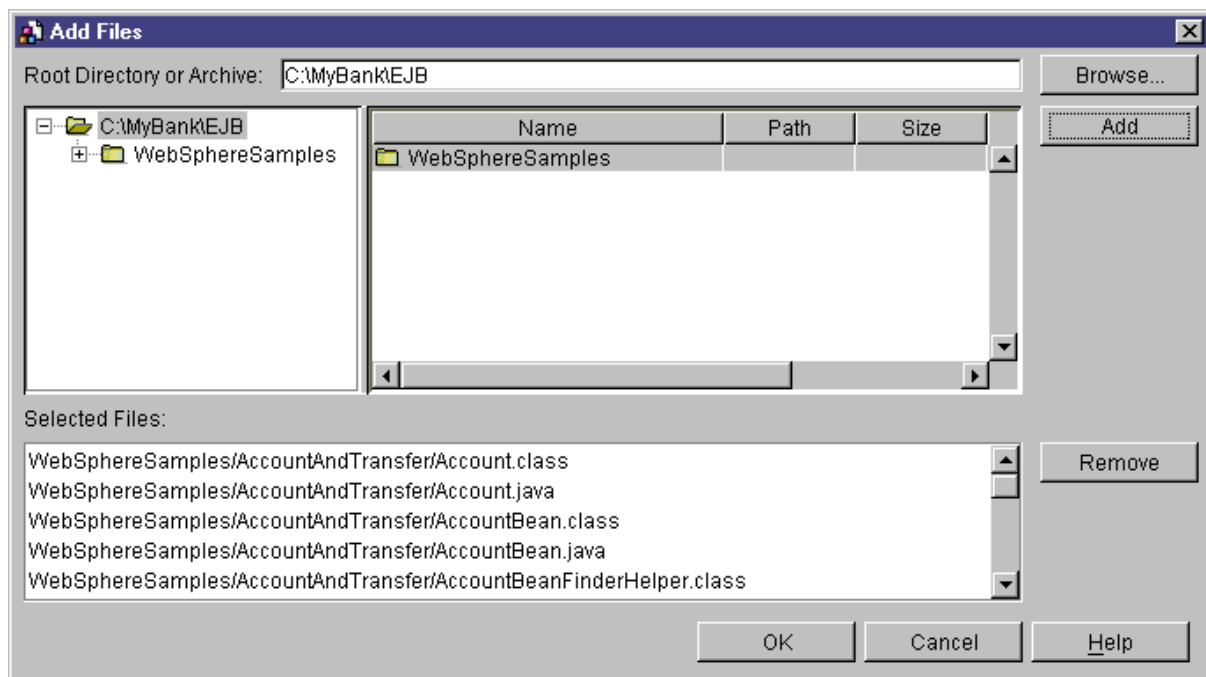


- \_\_2. Select the **Create EJB Module Wizard**, and click **OK**.  
 This wizard goes through the steps needed to assemble EJBs into a Module.
- \_\_3. In the Create EJB Module Wizard: Specifying EJB Module Properties window, required properties are indicated with a red asterisk \*. Other properties are optional. Fill in:

Display Name: **MyBank EJB Module**  
 File Name: **C:\MyBank\MyBankEJB.jar**  
 Description: **Contains Account and Transfer EJBs**

\_\_ Click **Next**.

- \_\_ 4. In the Create EJB Module Wizard: Adding Files window, click the **Add** button.  
You will add all of the EJB class files to the Module.
- \_\_ In the Add Files window, click **Browse**.
  - \_\_ In the Select Root Directory window, navigate to look in **C:\MyBank**.
  - \_\_ Click to highlight the **EJB** directory. (Note: Don't open the EJB directory)
  - \_\_ Click the **Select** button.
- \_\_ 5. In the Add Files window, select all the needed files by directory.
- \_\_ In the upper right portion of the window, click the **WebSphereSamples** directory.
  - \_\_ Click the **Add** button to add the selected Directory.



- \_\_ Click **OK**.
- \_\_ 6. In the Create EJB Module Wizard: Adding Files window, click **Next**.
- \_\_ In the Create EJB Module Wizard: Specifying EJB Client Jar and Classpath window, click **Next**.
  - \_\_ In the Create EJB Module Wizard: Choosing EJB Module Icons window, click **Next**.
- \_\_ 7. In the Create EJB Module Wizard: Adding Enterprise Beans window, click **New**.  
The first EJB named Account is an entity bean with container managed persistence.

\_\_\_ In the Please Choose Bean Type window,  
select **Entity bean with container managed persistence**, and click **OK**.

\_\_\_8. In the Create EJB Wizard: Specifying Enterprise Bean Properties window, fill in:

EJB name:               **Account**  
Description:           **Entity bean with CMP to model simple bank accounts**

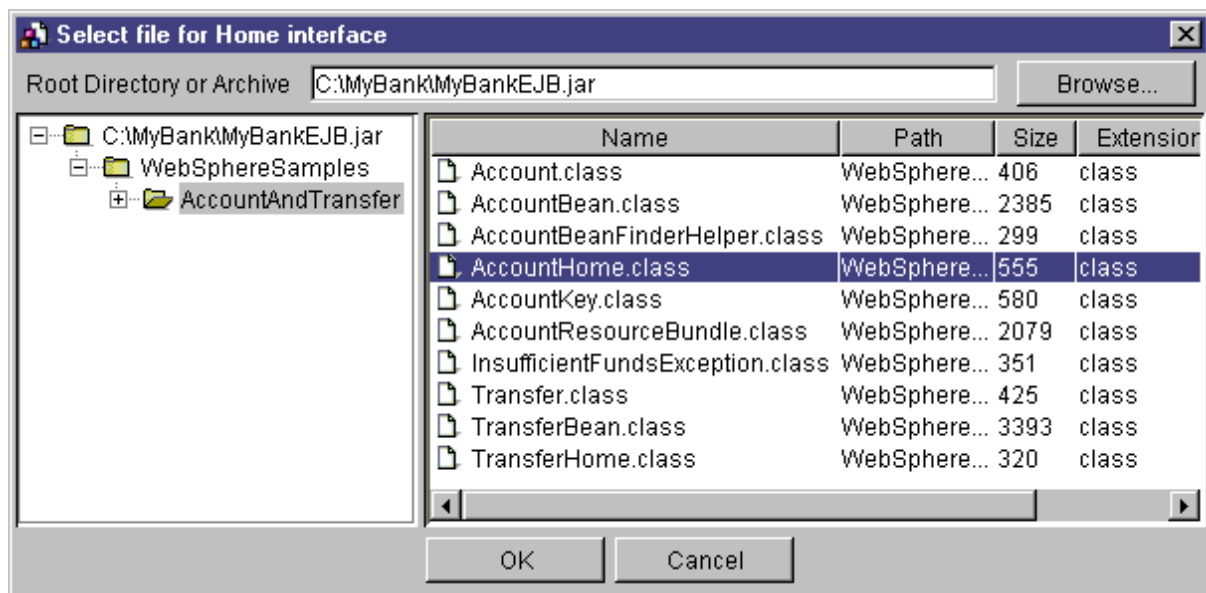
\_\_\_9. Click the **Browse** button by Home interface to locate the appropriate class file.

\_\_\_ The Select File for Home Interface window, should open to  
Root Directory or Archive: **C:\MyBank\MyBankEJB.jar**.

\_\_\_ On the left, expand the **WebSphereSamples** directory.

\_\_\_ On the left, click to select the **AccountAndTransfer** directory. All the files in the  
WebSphereSamples.AccountAndTransfer package appear on the right.

\_\_\_ Select **AccountHome.class**.



\_\_\_ Click **OK**.

Note that *WebSphereSamples.AccountAndTransfer.AccountHome* appears in the Home interface field after selecting *AccountHome.class*. This is the full package class name.

\_\_\_10. Repeat the actions from the previous step if needed to locate the other required classes:

Remote interface:   **WebSphereSamples.AccountAndTransfer.Account**  
EJB Class:           **WebSphereSamples.AccountAndTransfer.AccountBean**

**Create EJB Wizard**

**Specifying Enterprise Bean Properties**

Enter an EJB name, interfaces, and EJB class name for your Enterprise bean. The display name is used to identify your enterprise bean.

Enter a unique EJB name for your enterprise bean. Click Browse to locate files representing your enterprise bean interfaces and EJB classes.

EJB name: \* Account

Display name:

Description: Entity bean with CMP to model simple bank accounts

Home interface: \* phereSamples.AccountAndTransfer.AccountHome

Remote interface: \* WebSphereSamples.AccountAndTransfer.Account

EJB class: \* phereSamples.AccountAndTransfer.AccountBean

\_\_ Click **Next**.

\_\_ 11. In the Create EJB Wizard: Specifying Container-managed persistence (CMP) fields window, click **Add**.

\_\_ In the New CMP field window, select Name: **balance** in the selection box. Click **OK**.

\_\_ Click **Add** two more times to add **type** and **accountId**.

\_\_ Click **Next**.

\_\_ 12. In the Create EJB Wizard: Specifying Specific Enterprise Bean Properties window, select:

Primary key class: **WebSphereSamples.AccountAndTransfer.AccountKey**

Primary key field: **Compound key**

\_\_ This EJB is not Reentrant, so do not check the box. Click **Next**.

\_\_13. In the Create EJB Wizard: Choosing Enterprise Bean Icons window, click **Next**.

- \_\_ In the Create EJB Wizard: Adding Environment Entries window, click **Next**.
- \_\_ In the Create EJB Wizard: Adding Security Role References window, click **Next**.
- \_\_ In the Create EJB Wizard: Adding Resource References window, click **Next**.
- \_\_ In the Create EJB Wizard: Adding EJB References window, click **Finish**.

\_\_14. In the Create EJB Module Wizard: Adding Enterprise Beans window, click **New** again. The second EJB named Transfer is a stateless session bean.

\_\_ In the Please Choose Bean Type window, select **Session Bean**, and click **OK**.

\_\_15. In the Create EJB Wizard: Specifying Enterprise Bean Properties window, fill in:

EJB name:               **Transfer**  
Description:           **Session bean to transfer funds between accounts**

\_\_16. Click the **Browse** button by Home interface to locate the appropriate class files.

Home interface:       **WebSphereSamples.AccountAndTransfer.TransferHome**  
Remote interface:     **WebSphereSamples.AccountAndTransfer.Transfer**  
EJB Class:             **WebSphereSamples.AccountAndTransfer.TransferBean**

\_\_ Click **Next**.

\_\_17. In the Create EJB Wizard: Specifying Specific Enterprise Bean Type Properties window, pick:

Session type:         **Stateless**  
Transaction type:     **Container**

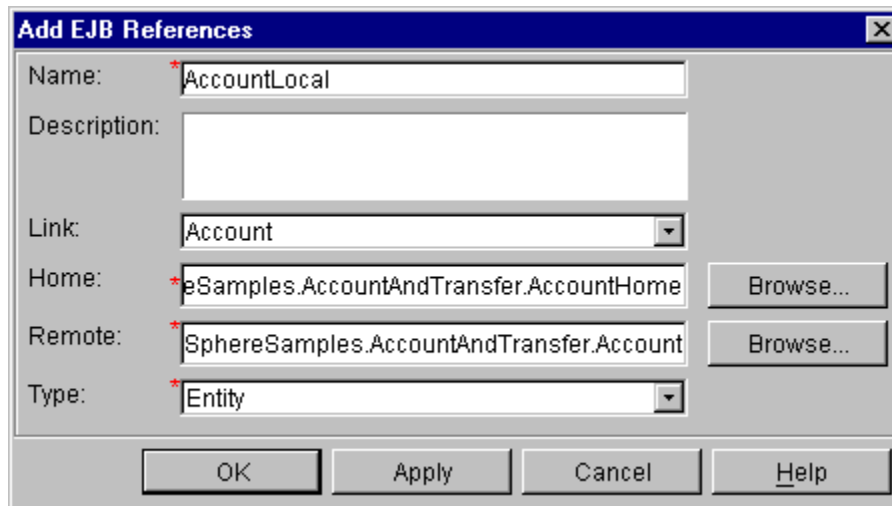
\_\_ Click **Next**.

\_\_18. In the Create EJB Wizard: Choosing Enterprise Bean Icons window, click **Next**.

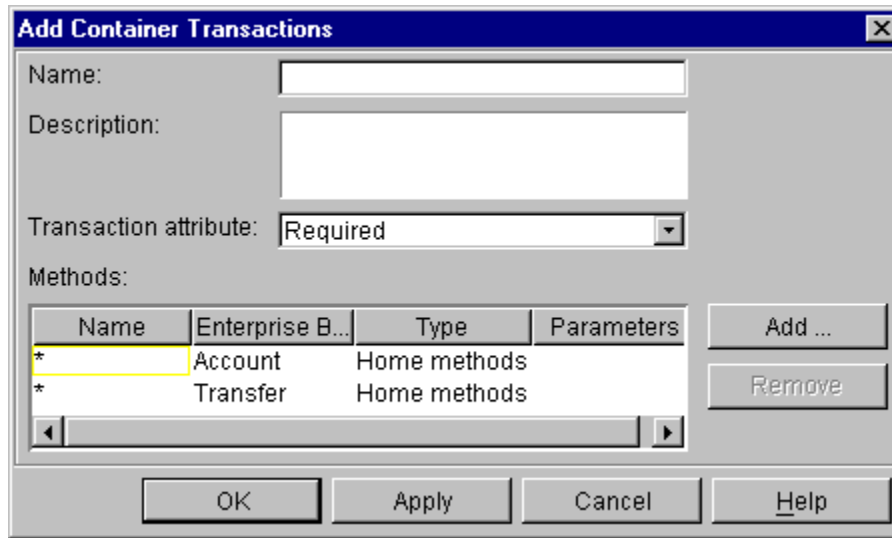
- \_\_ In the Create EJB Wizard: Adding Environment Entries window, click **Next**.
- \_\_ In the Create EJB Wizard: Adding Security Role References window, click **Next**.
- \_\_ In the Create EJB Wizard: Adding Resource References window, click **Next**.

- \_\_19. In the Create EJB Wizard: Adding EJB References window, click **Add**.  
EJB References associate local names for beans with names in the global Java Naming and Directory Interface (JNDI) namespace.

- \_\_ In the Add window, specify Name: **AccountLocal**
- \_\_ In the Link field, select **Account**.
- \_\_ The Home, Remote and Type fields are filled in automatically.



- \_\_ Click **OK**.
  - \_\_ Click **Finish**.
- 
- \_\_20. In the Create EJB Module Wizard: Adding Enterprise Beans window, click **Next**.  
In the Create EJB Module Wizard: Adding Security Roles window, click **Next**.  
In the Create EJB Module Wizard: Adding Method Permissions window, click **Next**.
- 
- \_\_21. In the Create EJB Module Wizard: Adding Container Transactions window, click **Add**.
    - \_\_ In the Add Container Transactions window, select Transaction Attribute: **Required**
    - \_\_ Click **Add**.
    - \_\_ In the Add Methods window, click the + next to **C:\MyBank\MyBankEJB.jar** folder.
    - \_\_ Click the + next to the **Account** folder.
    - \_\_ Click to select the **Home** folder under **Account**.
    - \_\_ Click the + next to the **Transfer** folder.
    - \_\_ Press the **Ctrl** key, and click to select the **Home** folder under **Transfer**.
    - \_\_ Click **OK**.



\_\_ Click **OK**.

\_\_22. In the Create EJB Module Wizard: Adding Container Transactions window, click **Finish**.

\_\_23. The Create EJB Module Wizard created the Module with the EJB class files and deployment descriptor information. A few more properties will complete the Module. Set the default data source that the CMP entity bean will use for storing data.

\_\_ In the Application Assembly Tool window, click the **Bindings** tab.

\_\_ Set the Default Data Source properties:

JNDI name:            **jdbc/MyBank**  
 User ID:              **USERID**  
 Password:            **PASSWORD**

\_\_ Click the **Apply** button to save the changes.

\_\_24. Change the JNDI name for the Transfer session bean from the default value.

\_\_ Click the **+** next to **Session Beans**.

\_\_ Click to select the **Transfer** bean.

\_\_ Click the **Bindings** tab.

\_\_ Change the **JNDI name**

from:            WebSphereSamples/AccountAndTransfer/TransferHome

to:               **TransferHomeGlobal**

\_\_ Click the **Apply** button to save the changes.

\_\_25. Change the JNDI name for the Account entity bean from the default value.

- \_\_ Click the **+** next to **Entity Beans**.
- \_\_ Click to select the **Account** bean.
- \_\_ Click the **Bindings** tab.
- \_\_ Change the **JNDI name**  
from:       WebSphereSamples/AccountAndTransfer/AccountHome  
to:         **AccountHomeGlobal**
- \_\_ Click the **Apply** button to save the changes.

You will not specify the Datasource JNDI name, User ID and Password for the Account bean. This entity bean with container managed persistence will use the default data source for the container that you specified two steps earlier.

\_\_26. Add the JNDI name for the EJB Reference under the Transfer bean.

- \_\_ Click the **+** next to the **Transfer** session bean.
- \_\_ Click to select **EJB References** on the left.
- \_\_ Click the **Bindings** tab for the **AccountLocal** EJB Reference on the right.
- \_\_ Type the JNDI name: **AccountHomeGlobal**
- \_\_ Click the **Apply** button to save the change.

\_\_27. The EJB Module is now complete. Save the EJB Module into the **MyBankEJB.jar** file.

- \_\_ Click the **File** menu, and select **Save**.
- \_\_ A dialog window confirms successful completion. Click **OK**.

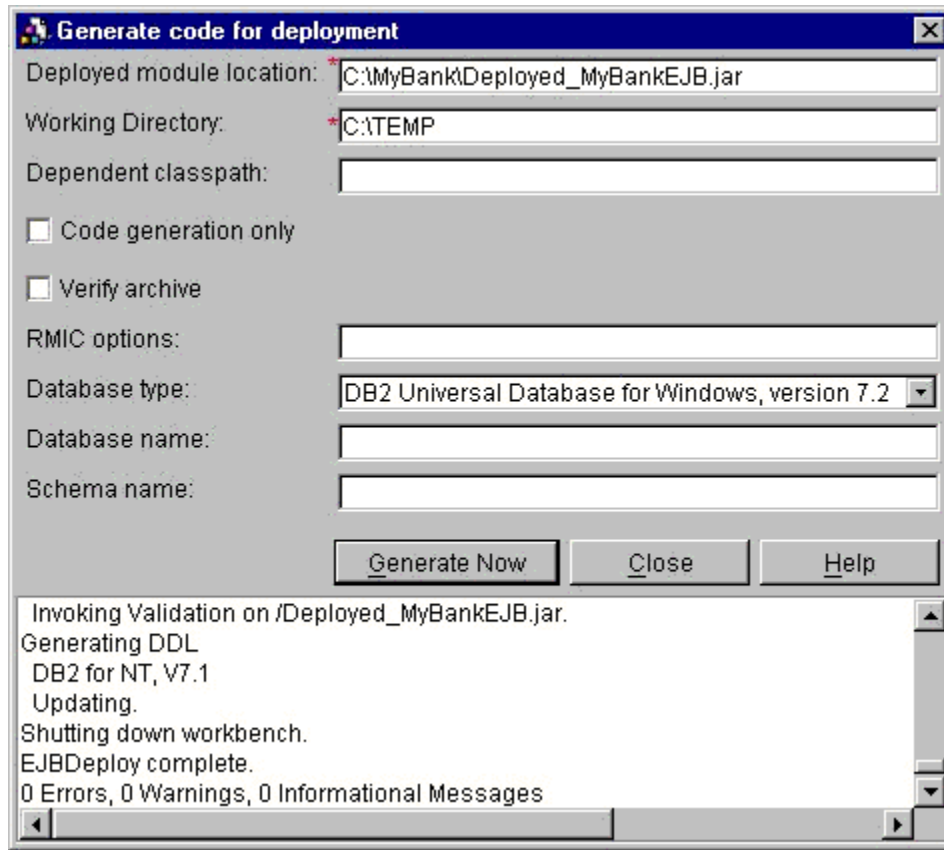
\_\_28. Generate the deployed code for the Enterprise JavaBeans.

- \_\_ Click the **File** menu, and select **Generate code for deployment**, or click the Generate code Button:



Generate code for deployment

- \_\_ In the Generate code for deployment window, click the **Generate Now** Button.



The process for generating code takes a couple of minutes. Scroll down in the message area. After you see the message, “EJBDeploy complete”, **close** the window.

\_\_ Click the **File** menu, and select **Close**.

## **Part Two: Build a Web Module from Servlets, HTML and image files**

- \_\_1. In the Application Assembly Tool window, start the Create Web Module Wizard. This wizard goes through the steps needed to assemble Servlets, HTML and image files into a Web Module.

\_\_ Click the **File** menu, and select **Wizards**, or click the Wizards button:



\_\_ Select the **Create Web Module Wizard** from the menu.

- \_\_2. In the Create Web Module Wizard: Specifying Web Module Properties window, fill in:

Display name: **MyBank Web Module**  
 File name: **C:\MyBank\MyBankWeb.war**  
 Description: **Contains Servlets, HTML and images**

\_\_ Click **Next**.

- \_\_\_ 3. In the Create Web Module Wizard: Adding Files window, click **Add Resource Files**. Resource files are used for the Web interface, such as HTML and image files.

- \_\_\_ In the Add Files window, click **Browse**.
- \_\_\_ In the Select root directory window, look in the **C:\MyBank** directory.
- \_\_\_ Click to highlight the **Web** directory, and click the **Select** button.
- \_\_\_ In the Add Files window, in the right pane click on **create.html**.
- \_\_\_ Press the **Shift** key, and click **transfer.html** at the bottom of the list.
- \_\_\_ Click the **Add** button to add all five of the selected files:  
**create.html**, **index.html**, **javaclient.html**, **mybank.gif**, and **transfer.html**.
- \_\_\_ Click **OK**.

- \_\_\_ 4. In the Create Web Module Wizard: Adding Files window, click **Add Class Files**. Class files are the Servlets and supporting classes that control the application logic.

- \_\_\_ In the Add Files window, click **Browse**.
- \_\_\_ In the Select root directory window, look in the **C:\MyBank** directory.
- \_\_\_ Double click the **Web** directory.
- \_\_\_ Double click the **WEB-INF** directory.
- \_\_\_ Click to highlight the **classes** directory, and click the **Select** button.
- \_\_\_ In the Add Files window, on the left click the **+** to expand **WebSphereSamples**.
- \_\_\_ On the left, click the **AccountAndTransfer** directory.
- \_\_\_ On the right, select all six of the files.
- \_\_\_ Click the **Add** button to add all of the selected files.
- \_\_\_ Click **OK**.
- \_\_\_ Click **Next**.

- \_\_\_ 5. In the Create Web Module Wizard: Specifying Optional Web Module Properties window, add the EJB module that you just created to the classpath of the Web application. This enables the server to find the EJB classes when servlets call EJB methods.

Classpath: **Deployed\_MyBankEJB.jar**

- \_\_\_ Click **Next**.

- \_\_\_ 6. In the Create Web Module Wizard: Choosing Web Module Icons window, click **Next**.

- \_\_\_ 7. In the Create Web Module Wizard: Adding Web Components window, click **New**. The first Web component is a Servlet for creating bank accounts.

- \_\_\_ In the Create Web Component Wizard: Specifying Web Component Properties window, fill in:

Component name: **CreateAccount**  
Display name: **CreateAccount Servlet**  
Description: **Create checking or savings account**

\_\_ Do not check the Load on startup box. Click **Next**.

\_\_8. In the Create Web Module Wizard: Specifying Web Component Type window:

\_\_ Under Component Type, select the **Servlet** radio button.

\_\_ Click **Browse** next to **Class name**.

\_\_ In the Select file for Class name window, look in **C:\MyBank\MyBankWeb.war**.

\_\_ On the left, expand the directories **WEB-INF --> classes --> WebSphereSamples**, and click to select **AccountAndTransfer**.

\_\_ On the right, select **CreateAccount.class**, and click **OK**.

\_\_ Click **Next**.

\_\_9. In the Create Web Component Wizard: Choosing Web Component Icons window, click **Next**.

\_\_ In the Create Web Component Wizard: Adding Security Role References window, click **Next**.

\_\_ In the Create Web Component Wizard: Adding Initialization Parameters window, click **Finish**.

\_\_10. In the Create Web Module Wizard: Adding Web Components window, click **New** again. The second Web component is a Servlet for transferring funds between bank accounts.

\_\_ In the Create Web Component Wizard: Specifying Web Component Properties window, fill in:

Component name:    **TransferFunds**  
Display name:       **TransferFunds Servlet**  
Description:        **Transfer funds between accounts**

\_\_ Do not check the Load on startup box. Click **Next**.

\_\_11. In the Create Web Module Wizard: Specifying Web Component Type window:

Component Type:    **Servlet**  
Class Name:         **WebSphereSamples.AccountAndTransfer.TransferFunds**

\_\_ Click **Next**.

\_\_12. In the Create Web Component Wizard: Choosing Web Component Icons window, click **Next**.

\_\_ In the Create Web Component Wizard: Adding Security Role References window, click **Next**.

\_\_ In the Create Web Component Wizard: Adding Initialization Parameters window, click **Finish**.

\_\_13. In the Create Web Module Wizard: Adding Web Components window, click **Next**.  
In the Create Web Module Wizard: Adding Security Roles window, click **Next**.

\_\_14. In the Create Web Module Wizard: Adding Servlet Mappings window, click **Add**.  
Servlet Mappings associate requests URLs with Servlets that handle the requests.

\_\_ In the Add Servlet Mappings window, fill in:

URL Pattern:            **servlet/Create**  
Servlet:                **CreateAccount**

\_\_ Click **OK**.

\_\_15. In the Create Web Module Wizard: Adding Servlet Mappings window, click **Add**.

\_\_ In the Add Servlet Mappings window, fill in:

URL Pattern:            **servlet/Transfer**  
Servlet:                **TransferFunds**

\_\_ Click **OK**.

\_\_ Click **Next**.

\_\_16. In the Create Web Module Wizard: Adding Resource References window, click **Next**.

In the Create Web Module Wizard: Adding Context Parameters window, click **Next**.

In the Create Web Module Wizard: Adding Error Pages window, click **Next**.

In the Create Web Module Wizard: Adding MIME Mappings window, click **Next**.

In the Create Web Module Wizard: Adding Tag Libraries window, click **Next**.

\_\_17. In the Create Web Module Wizard: Specifying Welcome Files window, click **Add**.  
A Welcome file is a default home page for a Web application.

\_\_ In the Add Welcome Files window, click **Browse** next to Welcome File.

\_\_ In the Select file for Welcome File window, look in **C:\MyBank\MyBankWeb.war**.

\_\_ On the left, click to highlight **C:\MyBank\MyBankWeb.war**.

\_\_ On the right, select **index.html**.

\_\_ Click **OK**. Click **OK**.

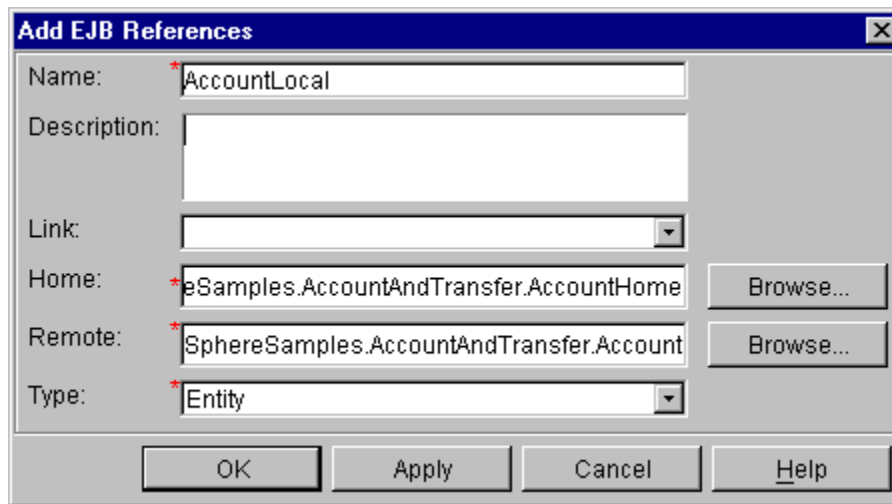
\_\_ Click **Next**.

\_\_18. In the Create Web Module Wizard: Specifying EJB References window, click **Add**.  
EJB References associate local names for beans with names in the global Java Naming and Directory Interface (JNDI) namespace.

\_\_ In the Add EJB References window, specify Name: **AccountLocal**

\_\_ Click the **Browse** button next to **Home**.

- \_\_\_ In the Select file for Home window, click **Browse**.
- \_\_\_ In the Select root directory window, navigate up to look in **C:\MyBank**
- \_\_\_ Click to highlight **Deployed\_MyBankEJB.jar**, and click **Select**.
- \_\_\_ In the Select file for Home window, click **+** to expand **WebSphereSamples**.
- \_\_\_ On the left, click the **AccountAndTransfer** directory.
- \_\_\_ On the right, select **AccountHome.class**.
- \_\_\_ Click **OK**.
  
- \_\_\_ Click the **Browse** button next to **Remote**.
- \_\_\_ In the Select file for Remote window, click **Browse**.
- \_\_\_ In the Select root directory window, look in **C:\MyBank**
- \_\_\_ Click to highlight **Deployed\_MyBankEJB.jar**, and click **Select**.
- \_\_\_ On the right, select **Account.class**.
- \_\_\_ Click **OK**.
- \_\_\_ In the Add window, select Type: **Entity**.



- \_\_\_ Click **OK**.
  
- \_\_\_ 19. In the Create Web Module Wizard: Specifying EJB References window, click **Add** again.

- \_\_\_ In the Add EJB References window, specify Name: **TransferLocal**
- \_\_\_ Click the **Browse** button next to **Home**.
- \_\_\_ In the Select file for Home window, click **Browse**.
- \_\_\_ In the Select root directory window, look in **C:\MyBank**
- \_\_\_ Click to highlight **Deployed\_MyBankEJB.jar**, and click **Select**.
- \_\_\_ On the right, select **TransferHome.class**.
- \_\_\_ Click **OK**.
  
- \_\_\_ Click the **Browse** button next to **Remote**.
- \_\_\_ In the Select file for Remote window, click **Browse**.
- \_\_\_ In the Select root directory window, look in **C:\MyBank**
- \_\_\_ Click to highlight **Deployed\_MyBankEJB.jar**, and click **Select**.

- \_\_\_ On the right, select **Transfer.class**.
- \_\_\_ Click **OK**.

- \_\_\_ In the Add window, select Type: **Session**.
- \_\_\_ Click **OK**.
- \_\_\_ Click **Finish**.

- \_\_\_20. The Create Web Module Wizard created the Module with the HTML and image resource files, the Servlet class files and deployment descriptor information. A few more properties will complete the Module.

- \_\_\_ In the Application Assembly Tool window, click the **Bindings** tab.
- \_\_\_ For Virtual hostname, specify: **default\_host**. This value may already appear here.
- \_\_\_ Click the **Apply** button to save if you made a change.

- \_\_\_21. Add the JNDI names for the EJB References.

- \_\_\_ Click to select **EJB References** on the left.
- \_\_\_ Click the **Bindings** tab for the **AccountLocal** EJB Reference on the right.
- \_\_\_ Type the JNDI name: **AccountHomeGlobal**
- \_\_\_ Click the **Apply** button to save the change.

- \_\_\_ Click to select the **TransferLocal** EJB Reference on the right.
- \_\_\_ Click the **Bindings** tab.
- \_\_\_ Type the JNDI name: **TransferHomeGlobal**
- \_\_\_ Click the **Apply** button to save the change.

- \_\_\_22. You have added two Servlets through the Create Web Module wizard. Let's try another method for adding a Servlet without a wizard. This WebSphere internal Servlet is useful for providing feedback when application configuration errors occur.

- \_\_\_ Right click **Web Components**, and select **New**.
- \_\_\_ In the New Web Component window, fill in:

Component Name:	<b>ErrorReporter</b>
Display Name:	<b>ErrorReporter Servlet</b>
Description:	<b>Shows stack trace for errors</b>
Component Type:	<b>Servlet</b>

- \_\_\_ Click **Browse** next to Class Name.
- \_\_\_ In the Select file for Class name window, click **Browse**.
- \_\_\_ In the Select root directory window, navigate to look in:  
**C:\WebSphere\SingleServer\lib**
- \_\_\_ Scroll down and click to highlight **webcontainer.jar**, and click **Select**.

- \_\_\_ In the Select file for Class name window, expand the directories on the left:  
**com --> ibm --> servlet --> engine.**
- \_\_\_ On the left, click to select the **webapp** directory.
- \_\_\_ On the right, click to select **DefaultErrorReporter.class**.
- \_\_\_ Click **OK**.
- \_\_\_ Click **OK** in the New Web Component window.

\_\_\_23. Add the Servlet mapping for the URL to call the ErrorReporter Servlet.

- \_\_\_ In the Application Assembly Tool window, select **Servlet Mapping** on the left.
- \_\_\_ Right click **Servlet Mapping** and select **New**.
- \_\_\_ In the New Servlet Mapping window, specify Url pattern: **servlet/Error**
- \_\_\_ Select Servlet: **ErrorReporter**
- \_\_\_ Click **OK**.

\_\_\_24. Configure the Web Module to use the ErrorReporter Servlet.

- \_\_\_ In the Application Assembly Tool window, select **MyBank Web Module** on the left.
- \_\_\_ On the right, select the **IBM Extensions** tab.
- \_\_\_ Specify the Default error page: **servlet/Error**
- \_\_\_ Click the **Apply** button to save the change.

\_\_\_25. The Web Module is now complete. Save the Web Module in the **MyBankWeb.war** file.

- \_\_\_ Click the **File** menu, and select **Save**.
- \_\_\_ A dialog window confirms successful completion. Click **OK**.
- \_\_\_ Click the **File** menu, and select **Close**.

### **Part Three: Build an Application Client Module from Java classes**

\_\_\_1. In the Application Assembly Tool window, start the Create Application Client Wizard. This wizard goes through the steps to assemble classes into an Application Client.

- \_\_\_ Click the **File** menu, and select **Wizards**, or click the Wizards button.
- \_\_\_ Select the **Create Application Client Wizard** from the menu.

\_\_\_2. In the Create Application Client Wizard: Specifying Application Client Properties window, fill in:

Display Name:           **MyBank Client**  
File Name:              **C:\MyBank\MyBankClient.jar**  
Description:            **MyBank Application Client**

\_\_ Click **Next**.

\_\_3. In the Create Application Client Wizard: Adding Files window, click **Add**.

\_\_ In the Add Files window, click **Browse**.

\_\_ In the Select Root Directory window, look in: **C:\MyBank**

\_\_ Click to highlight the **Client** directory, and click the **Select** button.

\_\_ In the Add Files window, click the **+** to expand **WebSphereSamples** on the left.

\_\_ On the left, click to highlight the **AccountAndTransfer** directory.

\_\_ On the right, select all five of the files.

\_\_ Click the **Add** button to add the selected files.

\_\_ Click **OK**.

\_\_ Click **Next**.

\_\_4. In the Create Application Client Wizard: Specifying Additional Application Client Module Properties window, fill in:

Classpath:                   **Deployed\_MyBankEJB.jar**

\_\_ Click the **Browse** button next to **Main class**.

\_\_ In the Select File for Main class window, click the **+** to expand **WebSphereSamples**.

\_\_ On the left, click to highlight **AccountAndTransfer**.

\_\_ On the right, select **TransferApplication.class**.

\_\_ Click **OK**.

\_\_ Click **Next**.

\_\_5. In the Create Application Client Wizard: Choosing Application Client Icons window, click **Next**.

\_\_6. In the Create Application Client Wizard: Specifying EJB References window, click **Add**.

\_\_ In the Add EJB References window, specify Name: **TransferLocal**

\_\_ Click the **Browse** button next to **Home**.

\_\_ In the Select file for Home window, click **Browse**.

\_\_ In the Select root directory window, look in **C:\MyBank**

\_\_ Click to highlight **Deployed\_MyBankEJB.jar**, and click **Select**.

\_\_ On the right, select **TransferHome.class**.

\_\_ Click **OK**.

\_\_ Click the **Browse** button next to **Remote**.

\_\_ In the Select file for Remote window, click **Browse**.

\_\_ In the Select root directory window, look in **C:\MyBank**

\_\_ Click to highlight **Deployed\_MyBankEJB.jar**, and click **Select**.

- \_\_\_ On the right, select **Transfer.class**.
  - \_\_\_ Click **OK**.
  - \_\_\_ In the Add EJB References window, select Type: **Session**.
  - \_\_\_ Click **OK**.
  - \_\_\_ Click **Next**.
- \_\_\_7. In the Create Application Client Wizard: Adding Resource References window, click **Next**.
  - \_\_\_ In the Create Application Client Wizard: Adding Environment Entries window, click **Finish**.
- \_\_\_8. The Create Application Client Wizard created the Module with the class files and deployment descriptor information. One more property will complete the Module.
  - \_\_\_ In the Application Assembly Tool window, click **EJB References** on the left.
  - \_\_\_ On the right, click the **Bindings** tab for the **TransferLocal** EJB Reference.
  - \_\_\_ Type the JNDI name: **TransferHomeGlobal**
  - \_\_\_ Click the **Apply** button to save the change.
- \_\_\_9. The Application Client is now complete. Save the Client into the **MyBankClient.jar** file.
  - \_\_\_ Click the **File** menu, and select **Save**.
  - \_\_\_ A dialog window confirms successful completion. Click **OK**.
  - \_\_\_ Click the **File** menu, and select **Close**.

#### **Part Four: Build an Enterprise Application from the Three Modules**

- \_\_\_1. You have built three Modules: MyBank EJB Module, MyBank Web Module, and MyBank Client. In this part, you will assemble the three Modules into a single application, ready to install in WebSphere. The application is stored in an Enterprise Archive .ear file. In the Application Assembly Tool, start the Create Application Wizard.
  - \_\_\_ Click the **File** menu, and select **Wizards**, or click the Wizards button.
  - \_\_\_ Select the **Create Application Wizard** from the menu.
- \_\_\_2. In the Create Application Wizard: Specifying Application Properties window, fill in:

Display name: **MyBank Application**  
File name: **C:\MyBank\MyBankApp.ear**  
Description: **MyBank Enterprise Application**

\_\_ Click **Next**.

\_\_3. In the Create Application Wizard: Adding Supplementary Files window, click **Next**.  
In the Create Application Wizard: Choosing Application Icons window, click **Next**.

\_\_4. In the Create Application Wizard: Adding EJB Modules window, click **Add**.

\_\_ In the Open window, look in: **C:\MyBank**.

\_\_ Click to highlight **Deployed\_MyBankEJB.jar**, and click **Open**.

\_\_ In the Confirm values window, leave **Alternate DD** blank.

Alternate DD represents an alternate Deployment Descriptor.



\_\_ Click **OK**.

\_\_ Click **Next**.

\_\_5. In the Create Application Wizard: Adding Web Modules window, click **Add**.

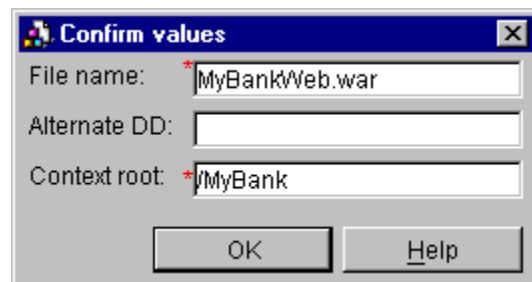
\_\_ In the Open window, look in: **C:\MyBank**.

\_\_ Click to highlight **MyBankWeb.war**, and click **Open**.

\_\_ In the Confirm values window, leave Alternate DD blank.

\_\_ In the Confirm values window, fill in Context root: **/MyBank**

The context root is the beginning of all URI's to resources in the application.



- \_\_ Click **OK**.
  - \_\_ Click **Next**.
- \_\_6. In the Create Application Wizard: Adding Application Clients window, click **Add**.
- \_\_ In the Open window, look in: **C:\MyBank**.
  - \_\_ Click to highlight **MyBankClient.jar**, and click **Open**.
  - \_\_ In the Confirm values window, leave Alternate DD blank, and click **OK**.
  - \_\_ Click **Next**.
- \_\_7. In the Create Application Wizard: Adding Security Roles window, click **Finish**.
- \_\_8. The Create Application Wizard combined the three Modules you created earlier into an enterprise archive file. One more property will complete the application.
- \_\_ In the Application Assembly Tool window, click the **Bindings** tab.
  - \_\_ Set the Enterprise application name: **MyBankApp**
  - \_\_ Click the **Apply** button to save the change.
- \_\_9. The MyBank Enterprise Application is now complete. Save the application into the **MyBankApp.ear** file.
- \_\_ Click the **File** menu, and select **Save**.
  - \_\_ A dialog window confirms successful completion. Click **OK**.
  - \_\_ Click the **File** menu, and select **Close**.
- \_\_10. Close the Application Assembly Tool window.
- \_\_ Click the **File** menu and select **Exit**, or click the  in the upper right corner.

### **What you did in this exercise**

You used the Application Assembly Tool to combine application components into Modules, and to combine Modules into an Enterprise Application. The EJB Module contains the Enterprise JavaBeans. The Web Module contains the Servlets, HTML and image files. The Application Client contains class files. Each Module contains deployment descriptor information, as well as IBM extensions and bindings.