

This document is to accompany the book IBM WebSphere Application Server Programming by Bassem Jamaledine, 1st edition, ISBN 0072224592.

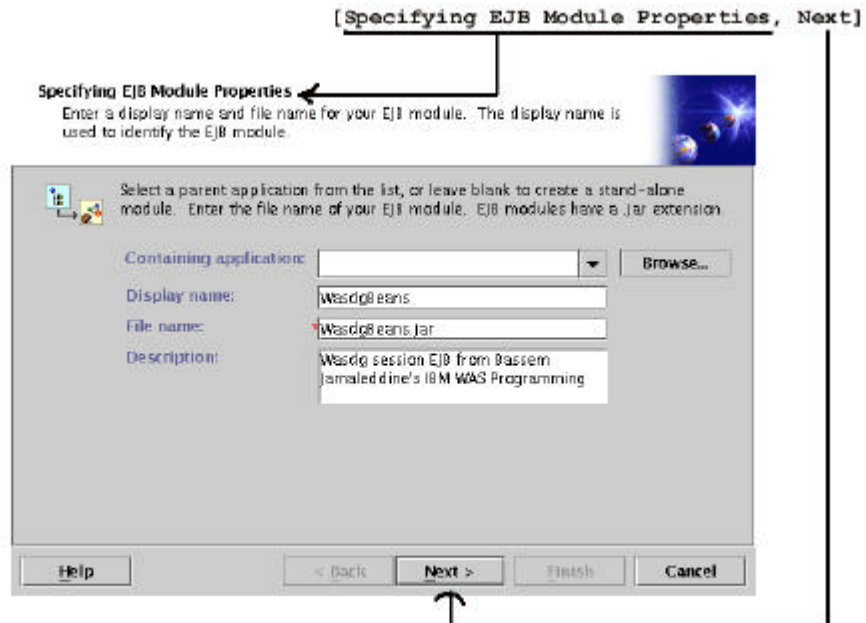
This is ch18_aat.pdf document that is referred to in the book IBM WebSphere Application Server Programming, Chapter 18, Section “Generating the WasdgBeans.jar Using the AAT”

This document shows the reader the steps to follow in using the AAT when generating the WasdgBeans.jar.

Because such a method of assembly is lengthy and tedious it has not been included in the text. However, for completeness, using the AAT for assembly is provided in this document.

(2)Generating the WasdgBeans.jar Using the AAT

Start the AAT and start creating the new EJB module: File | Wizards | Create EJB Module Wizard, and you will be presented with the first window, shown in the following figure to specify the EJB module properties.



As we progress through this section we will adapt the notation shown in the figure above while guiding the reader on how to assemble the WasdgBeans.jar using the AAT. For example, instead of showing the window displayed by AAT, we will refer to it and to the action to be taken by the programmer as follows:

[window title, action]

For example, the following notation shows the first step, and is equivalent to the window shown in Figure 18-16. Such notation is a compact way to describe window-based interaction with a user. We call such notation: HN notation.

Step 1: [Specifying EJB Module Properties, Next]

(Note: this is a simple note to direct the reader in this step)

Display name: WasdgBeans

File name: WasdgBeans.jar

Description: Wasdg session ...

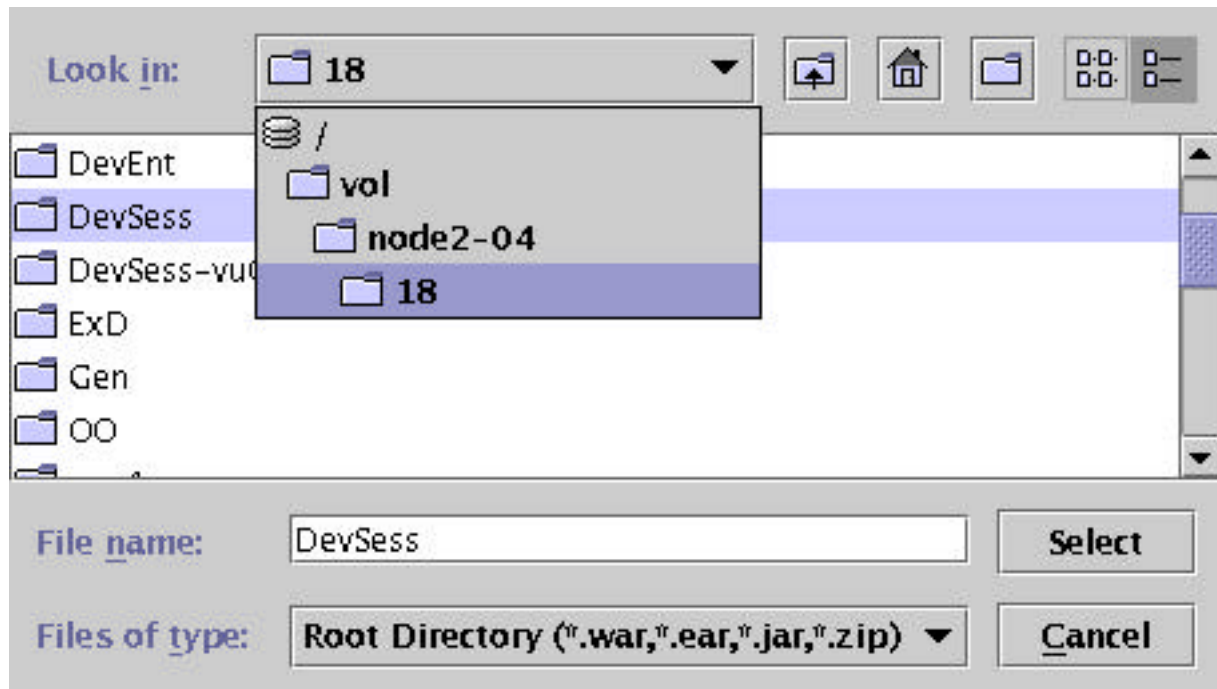
Figure 1: Specifying the EJB Module Properties in AAT

From this point, the following steps use the HN notation to direct the reader in creating the WasdgBeans.jar:

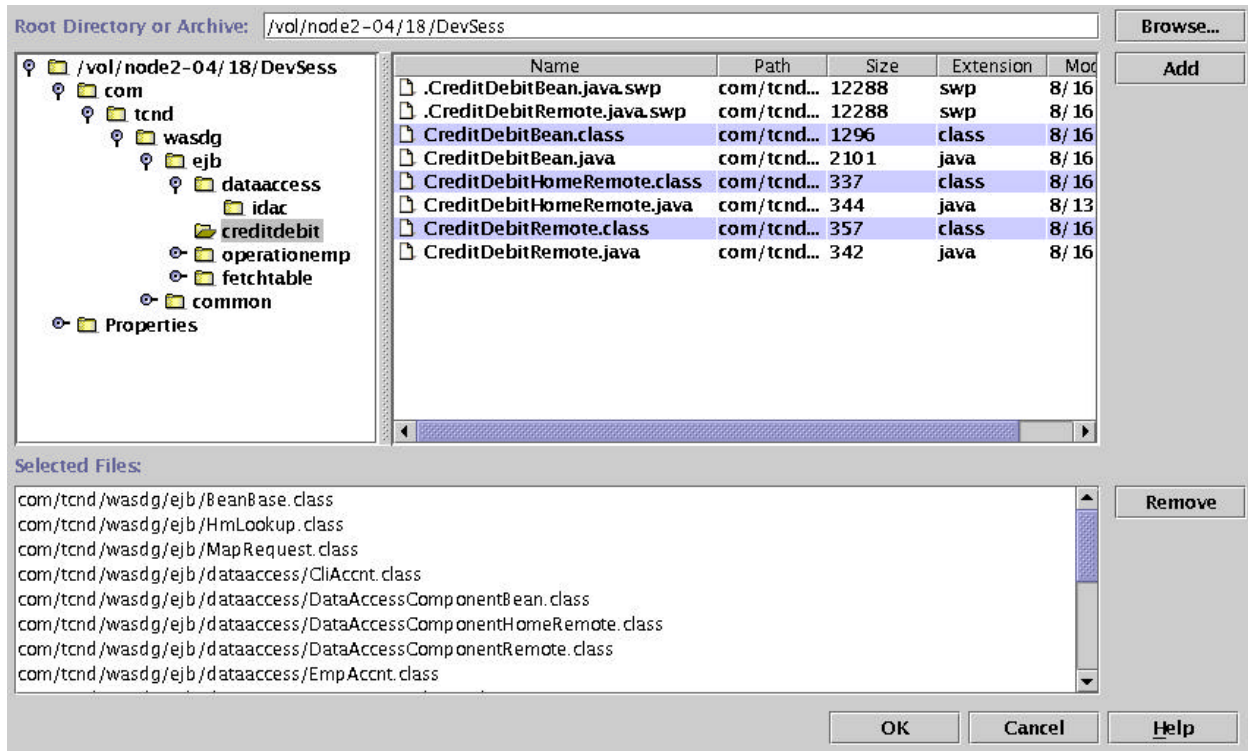
Step 2: This step is to add files. It is to be performed for all .class files in the package directory (and subdirectories), specifically \$BASE_DEV/DevSess/com/tcnd/wasdg. Repeat Step 2a for all these .class files.

Step 2a [Adding Files, Add...]

(Note: To add the files, select ``Add...`` then a window is presented that allow you to browse. Click on ``Browse``, then select the root directory (or parent directory) of DevSess as shown in the following illustration. Highlight DevSess and click on Select).



(Note: Browse through com.tcnd.wasdg as shown in the following illustration, and select *all* the .class files that are in the subdirectories of the package com.tcnd.wasdg. Add *all* these classes and make sure they have been added as shown in the illustration below.)



(Note: After adding the files, confirm with OK to go back to [Adding Files] window).

Step 2b [Adding Files, Next]

(Note: You can see all the files that you selected previously. Now is the time to go to the next window).

Step 3: [Specifying EJB Client Jar and Classpath, Next]

(Note: Skip this step)

Step 4: [Choosing EJB Module Icons, Next]

(Note: Skip this step)

Step 5: This step is formed of many sub-steps to be repeated when adding a bean

Step 5a [Adding Enterprise Beans, New...]

(Note: Add a new bean by selecting "New..." then select "Session bean" from the following window shown below).



Click OK to confirm and go to the next window.

Step 5b [Specifying Enterprise Bean Properties, Next]

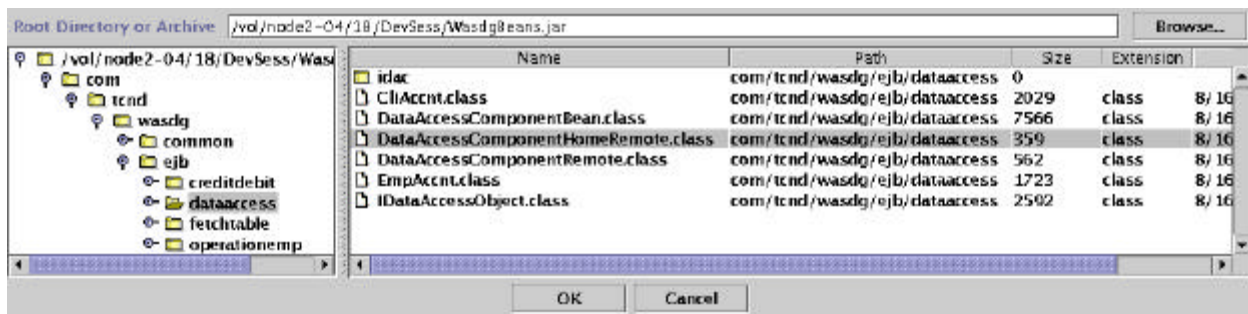
EJB name: DataAccessComponent

Home interface: DataAccessComponentHome

Remote interface: DataAccessComponentHomeRemote

EJB class: DataAccessComponentBean

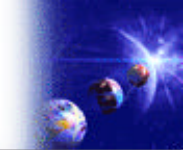
(Note: For each of the three entries specified above, select ``Browse`` then select the corresponding entry from the selected archive (WasdgBeans.jar) that is shown in the following illustration).



(Note: For example, after browsing through the com/tcnd/wasdg/ejb/dataaccess directory and adding the properties for the EJB named DataAccessComponent, the following illustration show the data being filled in. This example addressed the EJB named DataAccessComponent specifically, and will be different for other EJBs as you will go over Step 5 for each of the other beans).

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:

Display name:

Description:

Home interface:

Remote interface:

EJB class:

Step 5c [Specifying Specific Enterprise Bean Type Properties, Next]

Session type: Stateless

Transaction type: Bean

Step 5d [Choosing Enterprise Bean icons, Next]

Step 5e [Adding Environment Entries, Next]

Step 5f [Adding Security Role References, Next]

Step 5g [Adding Resource References, Next]

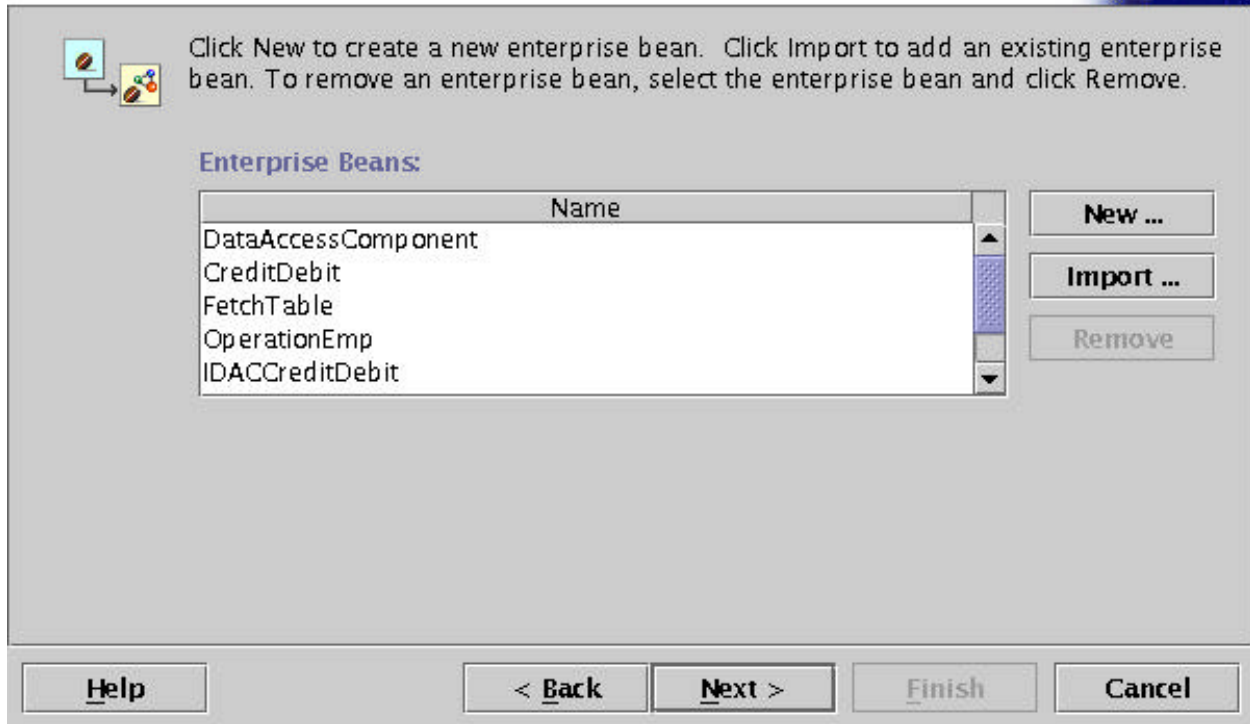
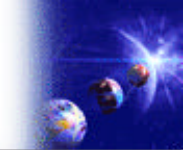
Step 5h [Adding EJB References, Finish]

Go back to the beginning of Step 5 and repeat it for all seven beans.

The following illustration shows the seven beans after they have been added.

Adding Enterprise Beans

An enterprise bean is a Java component that can be combined with other enterprise beans and Java components to create a distributed three-tiered application.



Step 6: [Adding Enterprise Beans, Next]

Step 7: [Adding Security Roles, Add]

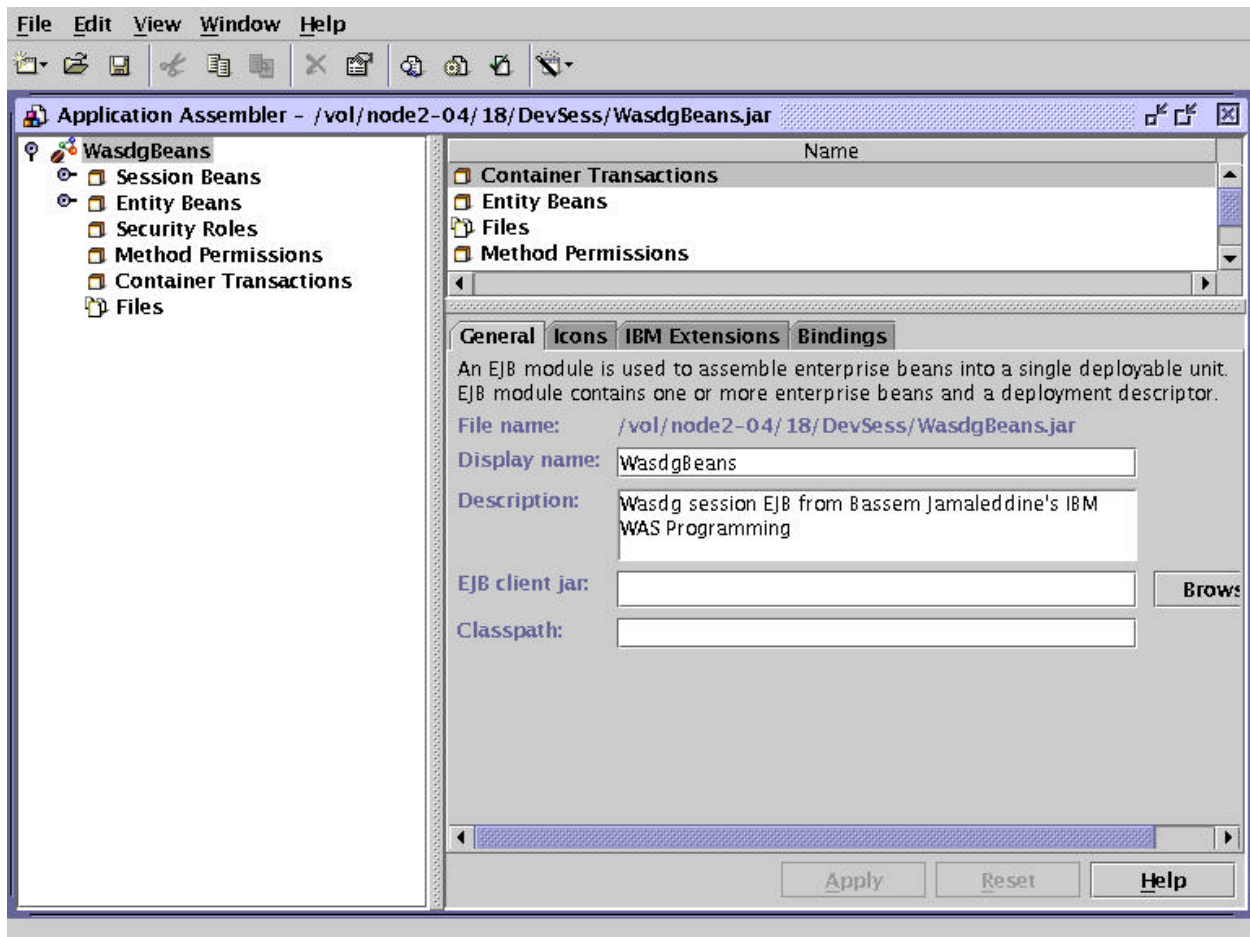
(Note: “General” property screen popped up, and you will enter after "Name:" everyone)

Step 8: [Adding Security Roles, Next]

Step 9: [Adding Method Permissions, Next]

Step 10: [Adding Container Transactions, Finish]

Step 11: Now you are back to the application assembly console shown in the following illustration:



Save the WasdGBeans.jar in the parent directory of DevSess by selecting: ``File | Save As...``, and make sure that the archive is saved successfully.

This will complete the creation of WasdGBeans.jar using the AAT. Yet, you need to verify the module by selecting: ``File | Verify...``

After AAT verifies the WasdGBeans.jar, make sure that you have no error messages. You can ignore warning messages. The following illustration shows the result returned by the verification process:



Using the AAT console, you can view the beans defined in WasdgBeans.jar as shown in the illustration below.

