



IBM Software Group

IBM Rational Application Developer V6.0

IBM Rational Web Developer V6.0

Enterprise JavaBeans Development Tools



@business on demand.

© 2005 IBM Corporation
Converted to video July 6, 2015

This presentation will focus on IBM Rational® Application Developer V6.0 Enterprise JavaBeans Development Tools.

Goals

- Understand the options available for developing Enterprise JavaBeans™
- Describe the deployment and testing options for Enterprise JavaBeans



The goals of this presentation are to help you gain a better understanding of the options available for developing Enterprise JavaBeans (EJB) and to describe the complete lifecycle for deployment and testing options.

Agenda

- EJB Development Tools Overview
- Session Beans
- Entity Beans
- Message-driven Beans
- Deployment
- Testing
- Summary



The agenda is to first provide an overview of the EJB Development Tools, and then to discuss Session Beans, Entity Beans, Message-Driven Beans, Deployment, and Testing. Finally, a Summary of topics covered will be provided.

Section

Overview



This section will provide an overview of the Enterprise JavaBeans Development Tools available in IBM Rational Application Developer V6.0.

EJB Development Tools Overview

- Complete EJB development support for 1.1, 2.0 and 2.1 levels
- Create EJB Projects, Enterprise JavaBeans and modify EJB deployment descriptors
 - ▶ Session Beans
 - ▶ Entity Beans
 - ▶ Message-driven Beans
- Deployment for WebSphere Application Server V6.0
- Universal Test Client included for testing Session and Entity beans



IBM Rational Application Developer provides complete support for developing, testing, and deploying Enterprise JavaBeans compliant with the 1.1, 2.0, and 2.1 specifications, including Session Beans, Entity Beans, and Message-driven Beans. You can create EJB projects containing Enterprise JavaBeans as well as Deployment Descriptors specific to Enterprise JavaBeans. Support is included to publish Enterprise JavaBeans to WebSphere Application Server V6 and a universal test client for unit testing is also included.

EJB Projects



The screenshot displays two dialog boxes from the IBM EJB Development Tools. The top dialog, 'New EJB Project', is for creating a new project. It includes fields for Name (WebSphereBankEJB), Project location, EJB version (2.1), and Target server (WebSphere Application Server v6.0). It also has checkboxes for 'Add module to an EAR project', 'Create an EJB Client JAR Project to hold the client interfaces', 'Add support for annotated Java classes', and 'Create a default stateless session bean'. The bottom dialog, 'EJB client JAR Creation', is for creating a client JAR from an existing EJB project. It includes fields for EJB Project (WebSphereBankEJB), Client JAR URI (WebSphereBankEJBClient.jar), Name (WebSphereBankEJBClient), and Project location. A red circle highlights the 'Add support for annotated Java classes' and 'Create a default stateless session bean' options in the top dialog, with an arrow pointing to the 'EJB client JAR Creation' dialog.

- Organize Enterprise JavaBeans in EJB projects
- Create EJB client projects
- New Options
 - Support for annotations may be added when project is created
 - Default stateless session bean (named DefaultSession) option

Additional enhancements for V6 include creation of an EJB client project by default when a new EJB project is created. Previous versions required that you specifically choose to have the client project created. Because of this change in default behavior, artifacts such as the stubs used by the client that appear to be missing from the EJB project could in fact be found in the client project. In addition, support for annotations has been added to EJB development. Annotations can be added when the project is initially created or later, when you create your EJB.

Support for Annotation-based Programming New v6

Create an Enterprise Bean

Create an Enterprise Bean
Select the EJB type and the basic properties of the bean.

Session bean
 Message-driven bean
 Entity bean with bean-managed persistence (BMP) field
 Entity bean with container-managed persistence (CMP)

EJB project: WebSphereBankEJB

Bean name: Transfer

Source folder: ejbModule

Default package: /..ibm.websphere.samples.bank.ejb

Generate an annotated bean class

- More information in Annotation-based Programming presentation

- Simplifies development by reducing the number of required development resources
- Specific tags based on XDoclet used for specifying artifacts and deployment information
- Annotation support can be added when a project or EJB is created

```
package com.example.wrd;
/**
 * @ejb.bean name="Hello" type="Stateless" view-
 * type=remote jndi-name="HelloBean"
 */
public class Hello {
  /**
   * @ejb.interface-method view-type=remote
   */
  public String hello(String name) {
    return "Hello: " + name;
  }
}
```

New support is included for annotation-based programming in IBM Rational Application Developer V6.0. Annotation-based programming allows developers to use special tags to define application and deployment information in Java resources. The development and deployment environment is responsible for generating the correct resources and deployment information from the tags before installation into the production environment. Annotation based programming allows developers to focus on development tasks, without having to fully understand or manage all of the artifacts and deployment information. Information can be specified in one location for the resources without searching for the correct location within the deployment package. The tags, which are based on the XDoclet standard, can be added to Enterprise JavaBeans and JavaBeans files using content assist. When the artifact is saved, the appropriate artifacts are generated in the project and the appropriate deployment information is added or updated in the deployment descriptor.

Deployment Descriptor Editor



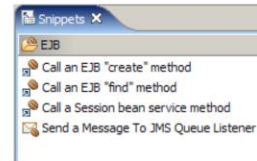
- J2EE and IBM WebSphere Extensions and Bindings can be specified in EJB Deployment Descriptor editor
 - ▶ “Flat look” editor carried over from V5
 - ▶ J2EE values stored in ejb-jar.xml
 - ▶ IBM WebSphere Extension and Bindings stored in ibm-ejb-jar-ext.xmi and ibm-ejb-jar-bnd.xmi files
- Web Services Client values now set through EJB Deployment Descriptor editor
- New IBM WebSphere options for specifying Mediation Handlers



The EJB deployment descriptor editor maintains the “flat look” from version 5. When you make modifications using the editor, you are really updating multiple files. The Web service client values are now specified using the EJB deployment descriptor editor rather than the Web service client XML or Web services client editor as with previous versions. There is also a new option that allows you to specify mediation handlers within a particular EJB module.

EJB Snippet Support

- Code snippets included which offer easy development with EJBs
- Create an instance of an EJB (Session or Entity)
- Call a specific find method on an EJB
- Call a method on the local or remote interface of an EJB
- Place a JMS message on a queue
- Java Build Path must be updated for the project after adding snippet



There is new support for reuse of common code snippets to perform basic operations. This includes snippets to locate a particular EJB using a finder method or call a method on the local or remote interface of the EJB. There is also a new snippet that allows you to create a messaging client and place a message on a JMS queue.

Section

Session Beans

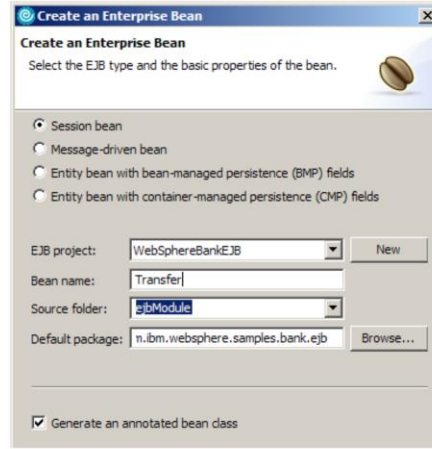


This section will cover Session Beans.

Session Beans



- Specify EJB project, name and package
- New option to generate session bean with annotations
- Create local and remote interfaces
- New option to add session bean to a UML class diagram



Tools provided in Rational Application Developer simplify the creation of Session Beans. There is a new option to use annotations to create beans or to add the beans to a UML class diagram associated with the project.

Section

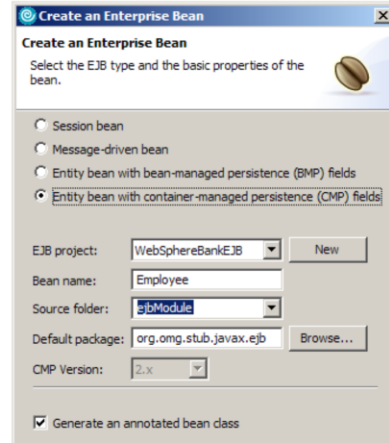
Entity Beans



This section will cover Entity Beans.

Entity Beans

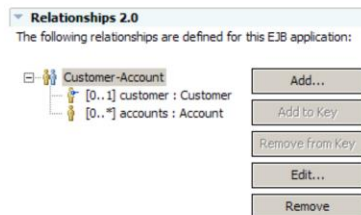
- Top-down, meet-in-the-middle and bottom-up development options
- Bean-managed and Container-managed persistence options
- Full support for specifying container-managed relationships and EJB Query Language (QL) queries
- Different mapping options available for container-managed persistence



Enterprise JavaBeans can be created using top down, bottom up, or meet in the middle options. You can also mix and match these options. Entity beans can use container managed (CMP) or bean managed (BMP) persistence. If you choose to use container managed persistence, you can create container relationships with other CMP beans. You can also use EJB QL queries to search for specific instances of an entity bean. There are also different options available for mapping CMP entity beans to different databases in addition to customized mappings.

Container-managed Relationships

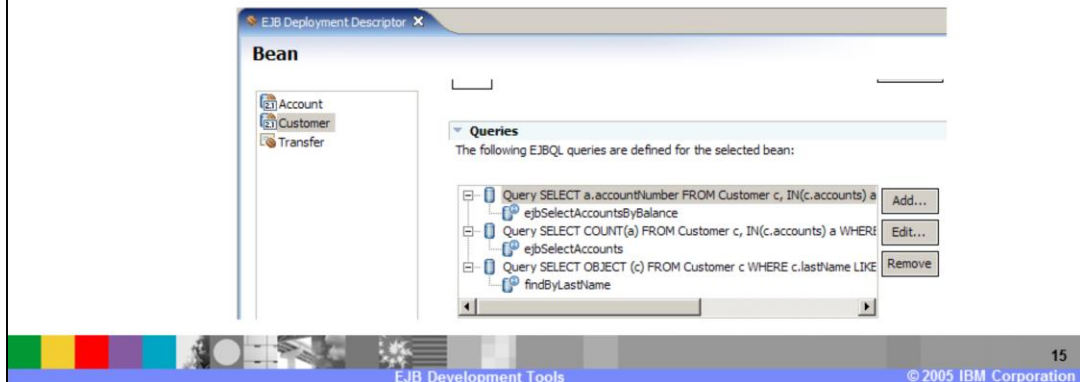
- Relationships can be created between two container-managed entity beans
 - ▶ Local interfaces must be created on at least one of the entity beans
- Relationships created through EJB Deployment Descriptor editor, annotations or UML class diagram editor



You can create relationships between two CMP entity beans if at least one of the beans contains a local interface. These relationships can be defined through the deployment descriptor editor, annotations, or a UML class diagram.

EJB QL Queries

- Find and Select EJB QL queries can be created for finding entity beans or CMP attributes
- Queries created through EJB Deployment Descriptor editor or annotations
- Sample queries available as a starting point for custom queries



EJB QL queries can be specified for CMP entity beans. The syntax for EJB QL is very similar to SQL. You can start with the sample queries supplied and modify them to suit your needs.

Container-managed Persistence Mappings

- Mappings to different databases can be created in a single project
 - ▶ Helpful when database for unit testing is different than production (Cloudscape vs. DB2[®])
- Mappings for all databases supported by WebSphere Application Server V6.0
 - ▶ Correct Mappings used at runtime specified by binding

The screenshot displays the EJB Development Tools interface. On the right, the Project Explorer shows a project named 'WebSphereBankEJB' with a 'backends' folder containing three database mapping entries: 'CLOUDSCAPE_V51_1', 'DB2UDBT_V82_1', and 'ORACLE_V10_1'. Each entry is circled in red. The 'CLOUDSCAPE_V51_1' entry is selected. On the left, the 'WebSphere Bindings' dialog is open, showing the 'Backend ID' dropdown menu with the same three options, also circled in red. The 'JNDI - CMP Connection Factory' section shows the 'JNDI name' set to 'jdbc/Bank' and the 'Container authorization type' set to 'Per_Connection_Factory'. The 'JAAS Login Configuration' section has 'Container Managed Authentication (Deprecated)' selected. The bottom of the window shows 'EJB Development Tools' and '© 2005 IBM Corporation'.

CMP Entity Beans can be mapped to various database products. You can also create mappings for multiple databases in a single project, which can be helpful for unit testing. The backend ID setting is used to determine which database mappings should be used at deploy time.

Section

Message-driven Beans



This section will cover Message-driven Beans.

Message-driven Beans

Create an Enterprise Bean

Select the EJB type and the basic properties of the bean.

Session bean
 Message-driven bean
 Entity bean with bean-managed persistence (BMP) fields
 Entity bean with container-managed persistence (CMP) fields

EJB project:
 Bean name:
 Source folder:
 Default package:

Generate an annotated bean class



- Specify EJB project, name, and package
- New option to generate session bean with annotations
- Specify non-JMS listeners
- New option to add session bean to a Unified Modeling Language™ (UML) class diagram

Create an Enterprise Bean

Message Driven Bean type

Choose the type of messaging service to use for the message driven bean.

JMS type
 Listener type:

Other type
 Listener type:



The wizard for creating Message-driven Beans (MDB) has not changed significantly from previous versions. There is a new option for generating the Message-driven Bean with annotations. Another new feature is the ability to specify a non-JMS listener as required by the J2EE specification or EJB 2.1 specification. As long as the listener type is set up, you can specify it when you create the MDB. This is useful if your MDB will handle non-JMS compliant messages.

Section

Deployment

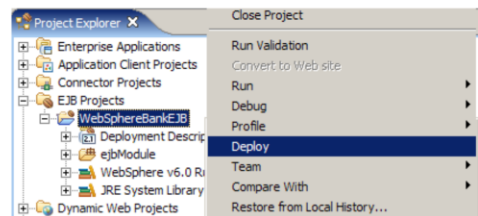


This section will cover deployment of Enterprise JavaBeans.

Deploying Enterprise JavaBeans



- Enterprise JavaBeans may be deployed within the workspace
- Option available off the EJB project or specific Enterprise JavaBean
- Only those beans which require code to be generated will be deployed
 - ▶ Deploy times are reduced



Once you have finished developing your EJB, including mappings, relationships and EJB QL queries, the next step is to prepare the EJB for deployment in a production environment. This can be accomplished within Rational Application Developer using the deploy operation, which has been improved in V6.0 to only deploy items that have been updated. This can also be done when the application is installed, but it is often convenient to take care of it from within the development tooling.

Section

Testing



This section will cover testing.

Testing Enterprise JavaBeans



- Unit testing for Enterprise JavaBeans available through the Universal Test Client
 - ▶ Universal Test Client published automatically to test server (local or remote) when applications are published to a server
 - ▶ New look
 - ▶ Refer to Help for new instructions for testing J2EE components
- Automated testing for Enterprise JavaBeans available with new Component Test feature
 - ▶ Create test patterns for testing EJB lifecycle, EJB business logic and EJB session facades
 - ▶ Multiple test cases can be executed with different data sets



The universal test client allows you to unit test Session and Entity Beans by automatically publishing the beans to any test server in the environment. This feature has a different look and feel when compared with previous versions. It is recommended that you consult the help available for this tool for details on finding, instantiating, and calling methods on an EJB. In addition to the universal test client, which is designed for fine-grain testing, there is the component test feature. This feature allows you to do more automated types of testing based on the JUnit framework. This is more useful when considering such things as the EJB lifecycle, specific business logic, and EJB facades.

Section

Summary

This section will provide a summary of the presentation.

Summary

- Complete Enterprise JavaBean development environment
- Session, Entity, and Message-driven Bean creation with additional details specified through deployment descriptor editor or annotations
- Full deployment and testing options allow for complete testing of Session and Entity beans



Rational Application Developer V6.0 provides a complete enterprise Java development environment for developing Session Beans, Entity Beans, and Message-driven Beans. It also provides full support for generating all the necessary components as well as editors for specifying deployment information. There is also full support for preparing and deploying applications to WebSphere Application Server, publishing and unit testing.

Trademarks, Copyrights, and Disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

| | | | | |
|-------------------|------------------------|----------|----------|-----------|
| IBM | CICS | IMS | MQSeries | Tivoli |
| IBM (logo) | Cloudscape | Informix | OS/390 | WebSphere |
| e (logo)/business | DB2 | Series | OS/400 | xSeries |
| AIX | DB2 Universal Database | Lotus | pSeries | zSeries |

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2004. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.



This completes this presentation covering IBM Rational Application Developer V6.0 Enterprise JavaBeans development tools.