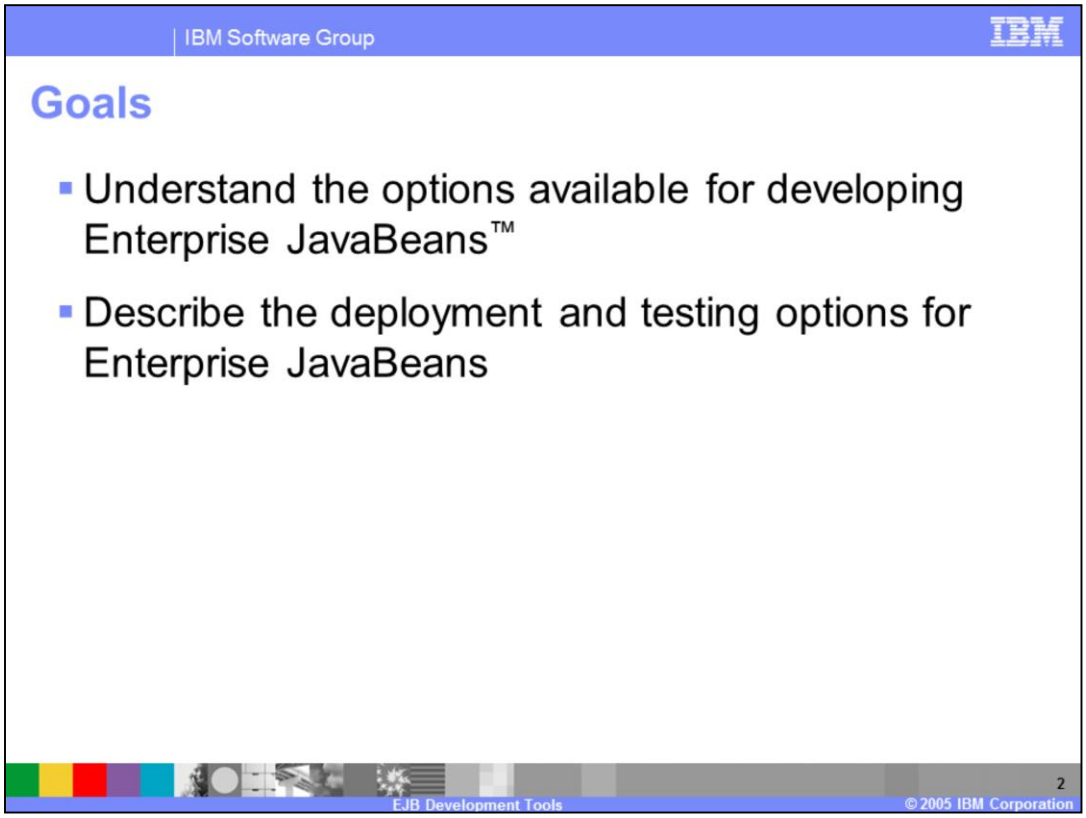
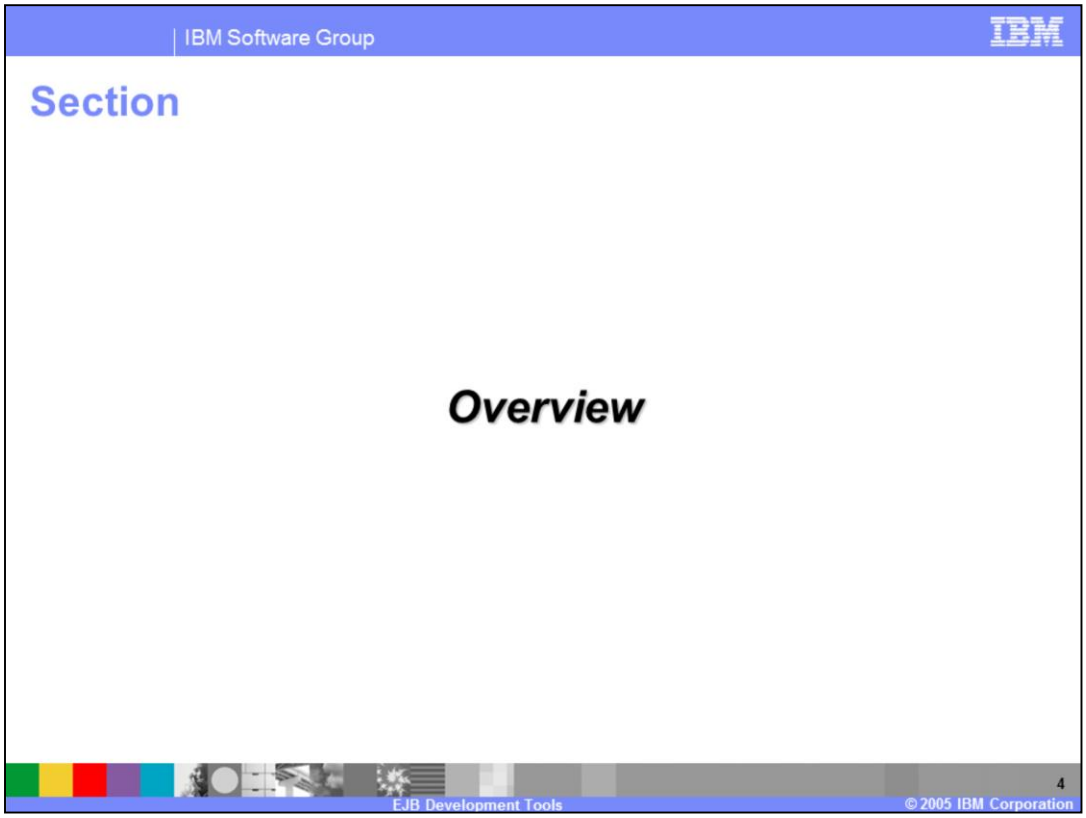
This presentation will focus on IBM Rational® Application Developer V6.0 Enterprise JavaBeans Development Tools.

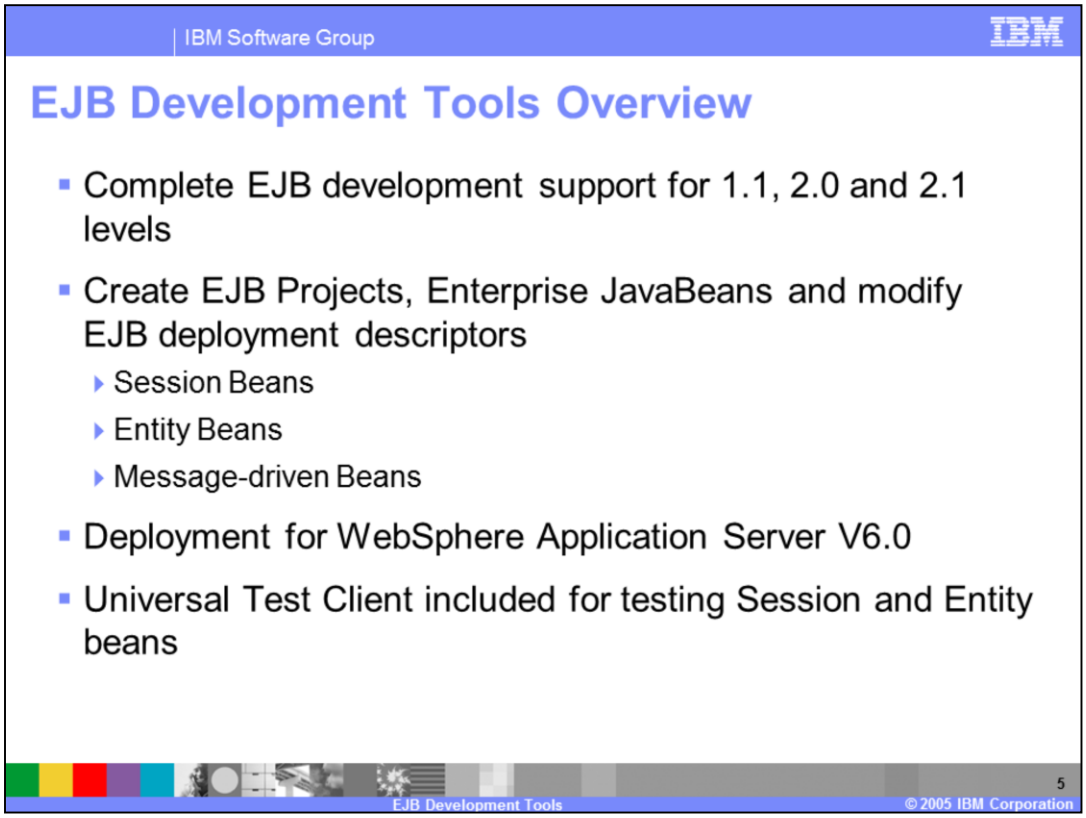


The goals of this presentation are to help you gain a better understanding of the options available for developing Enterprise JavaBeans (EJB) and to describe the complete lifecycle for deployment and testing options.

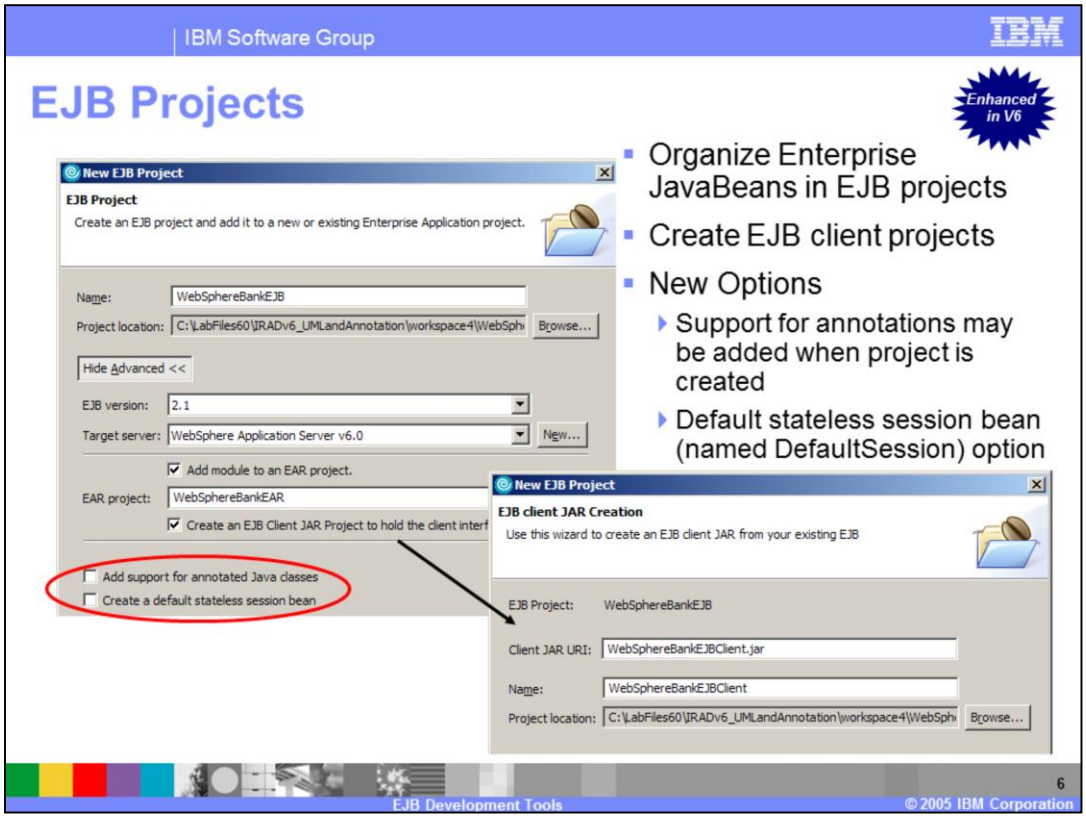
The agenda is to first provide an overview of the EJB Development Tools, and then to discuss Session Beans, Entity Beans, Message-Driven Beans, Deployment, and Testing. Finally, a Summary of topics covered will be provided.



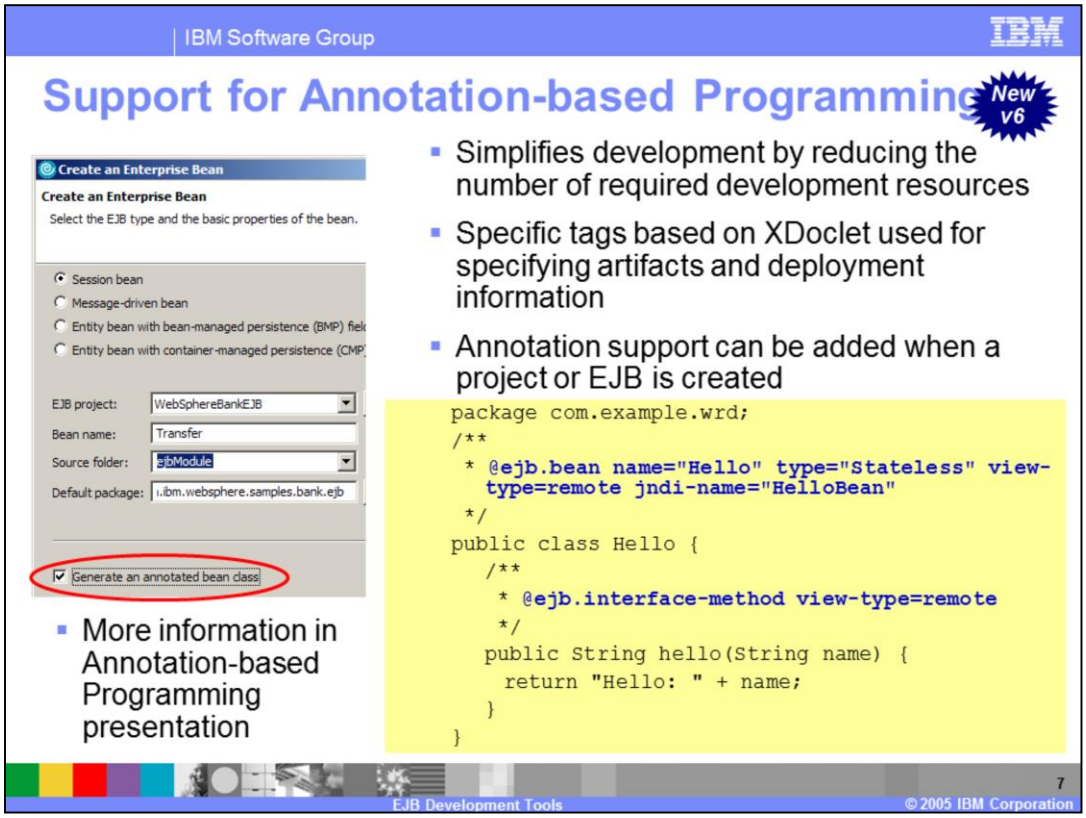
This section will provide an overview of the Enterprise JavaBeans Development Tools available in IBM Rational Application Developer V6.0.



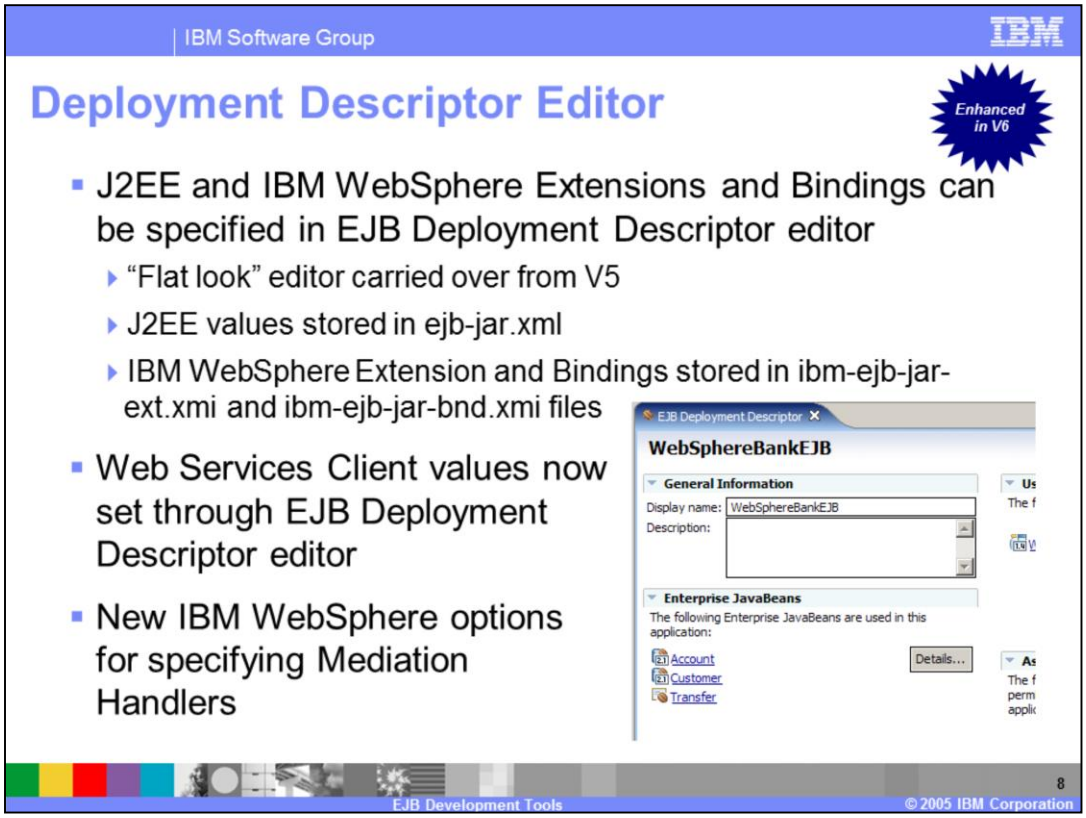
IBM Rational Application Developer provides complete support for developing, testing, and deploying Enterprise JavaBeans compliant with the 1.1, 2.0, and 2.1 specifications, including Session Beans, Entity Beans, and Message-driven Beans. You can create EJB projects containing Enterprise JavaBeans as 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.



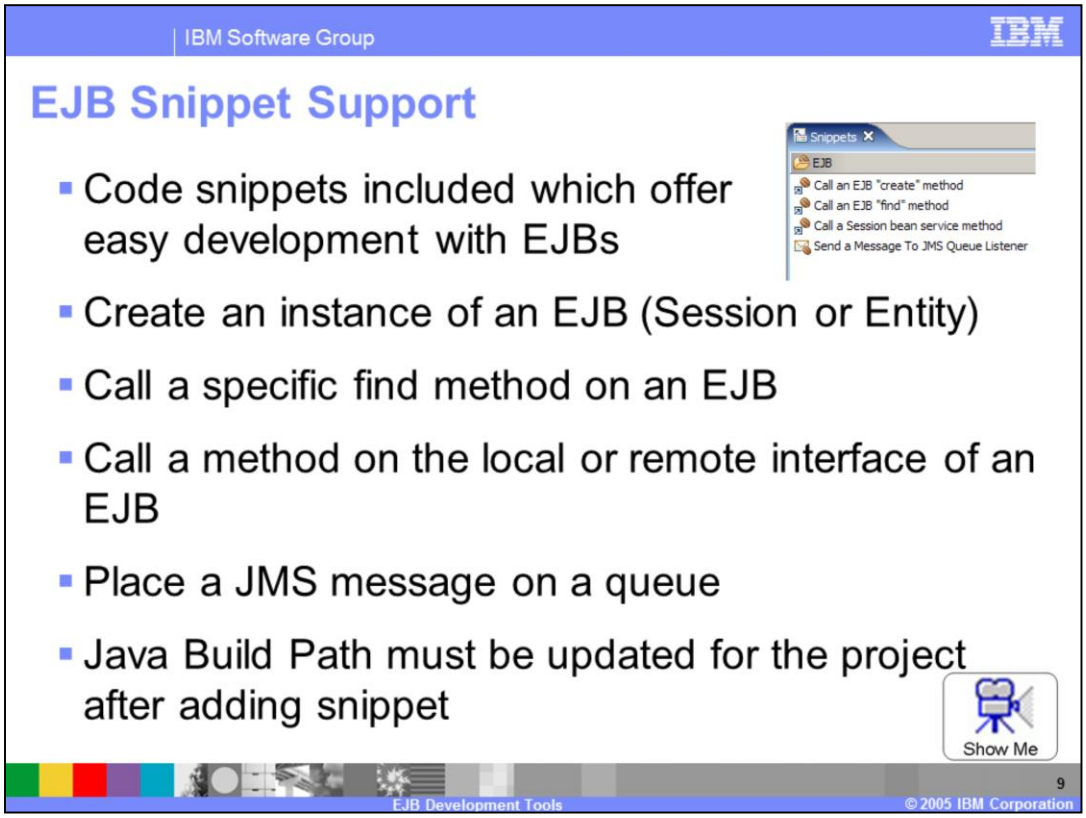
Additional enhancements for V6 include creation of an EJB client project by default when a new EJB project is created. Previous versions required that you specifically choose to have the client project created. Because of this change in default behavior, artifacts such as the stubs used by the client 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.



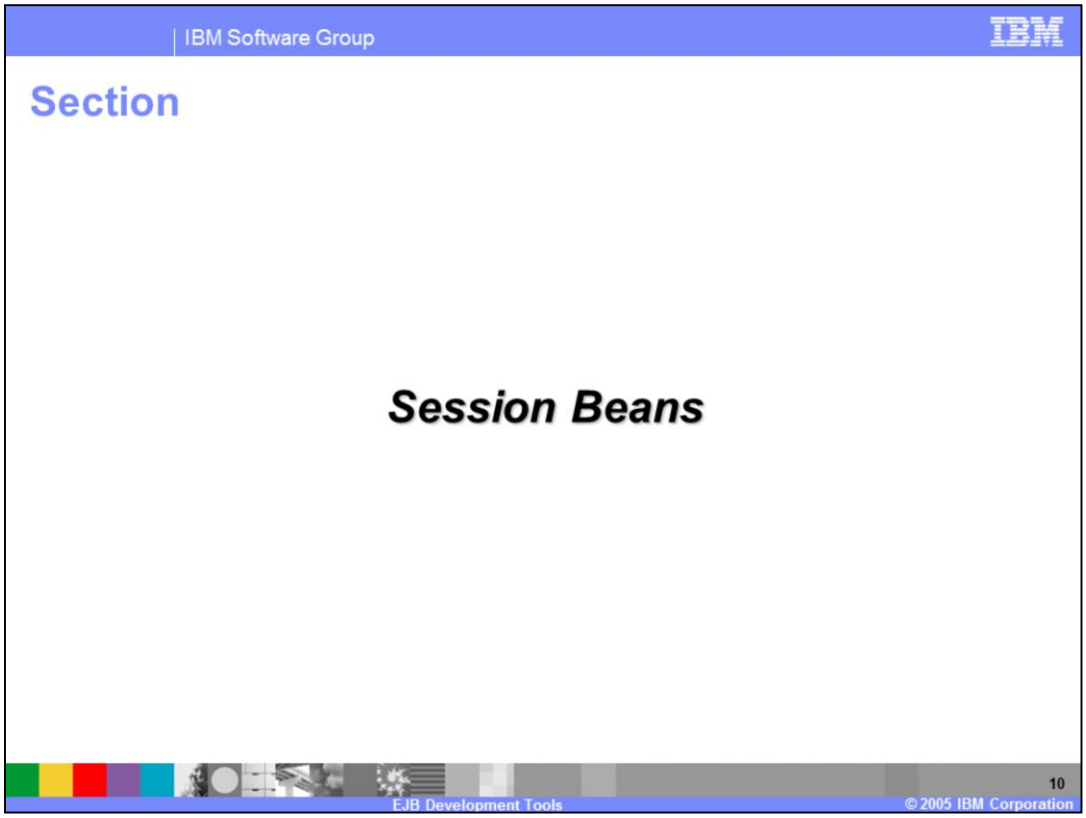
New support is included for annotation-based programming in IBM Rational Application Developer V6.0. Annotation-based programming allows developers to use special tags to define application and deployment information in Java resources. The development and deployment environment is responsible for generating the correct resources and deployment information from the tags before installation into the production environment. Annotation base 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.



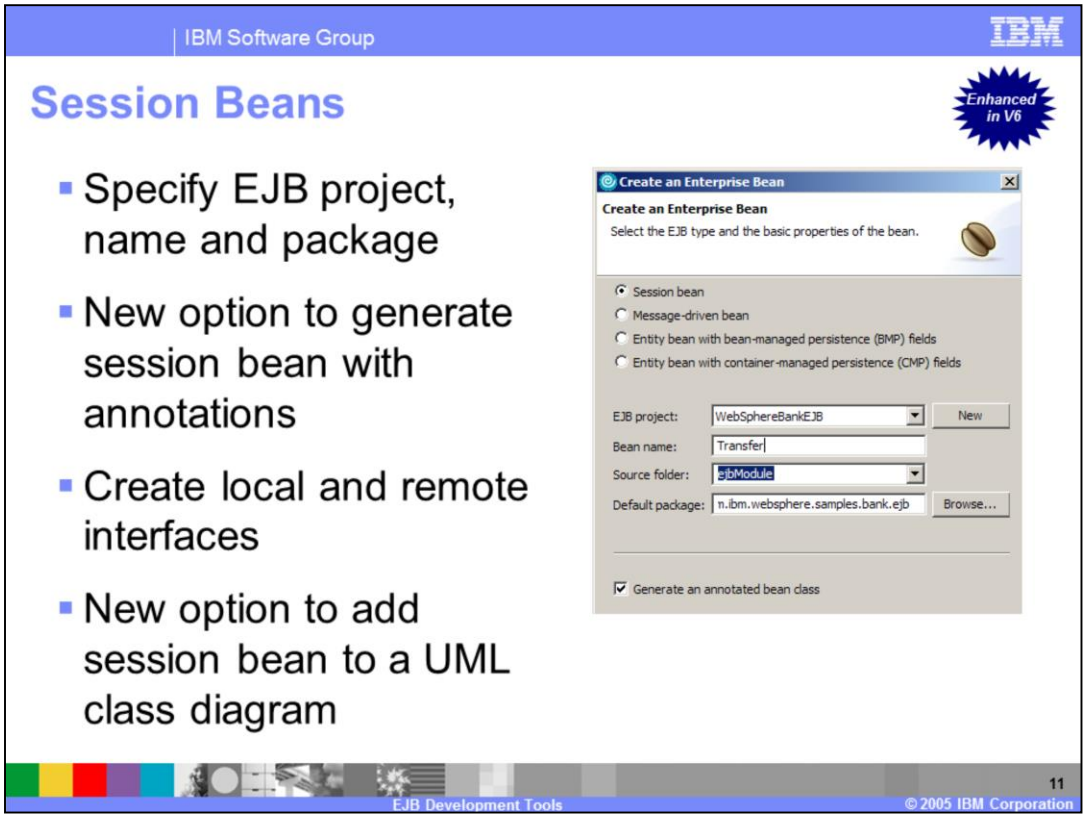
The EJB deployment descriptor editor maintains the "flat look" from version 5. When you make modifications using the editor, you are really updating multiple files. The Web service client values are now specified using the EJB deployment descriptor editor rather than the Web service client XML or Web 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.



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.



This section will cover Session Beans.

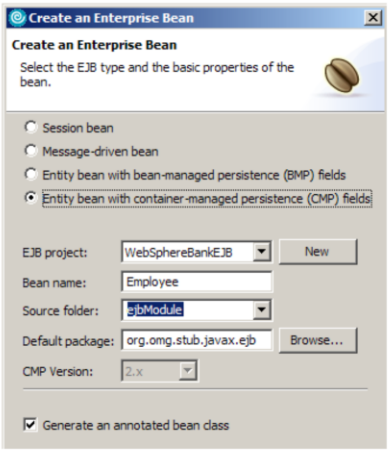


Tools provided in Rational Application Developer simplify the creation of Session Beans. There is a new option to use annotations to create beans or to add the beans to a UML class diagram associated with the project.

This section will cover Entity Beans.

Entity Beans

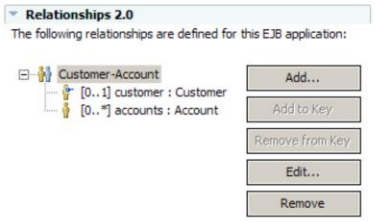
- Top-down, meet-in-the-middle and bottom-up development options
- Bean-managed and Container-managed persistence options
- Full support for specifying container-managed relationships and EJB Query Language (QL) queries
- Different mapping options available for container-managed persistence



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

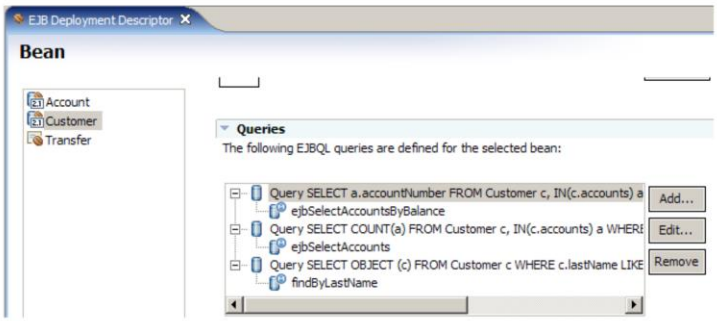
- Relationships can be created between two container-managed entity beans
  - Local interfaces must be created on at least one of the entity beans
- Relationships created through EJB Deployment Descriptor editor, annotations or UML class diagram editor



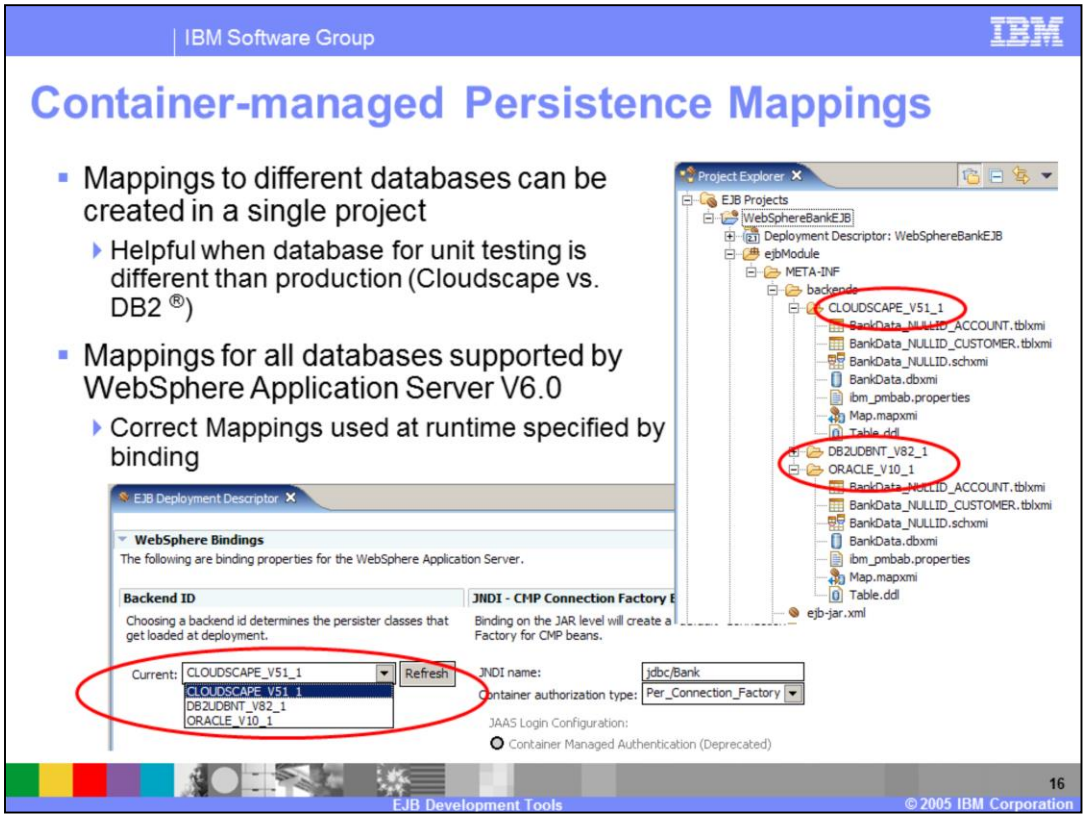
You can create relationships between two CMP entity beans if at least one of the beans contains a local interface. These relationships can be defined through the deployment descriptor editor, annotations, or a UML class diagram.

EJB QL Queries

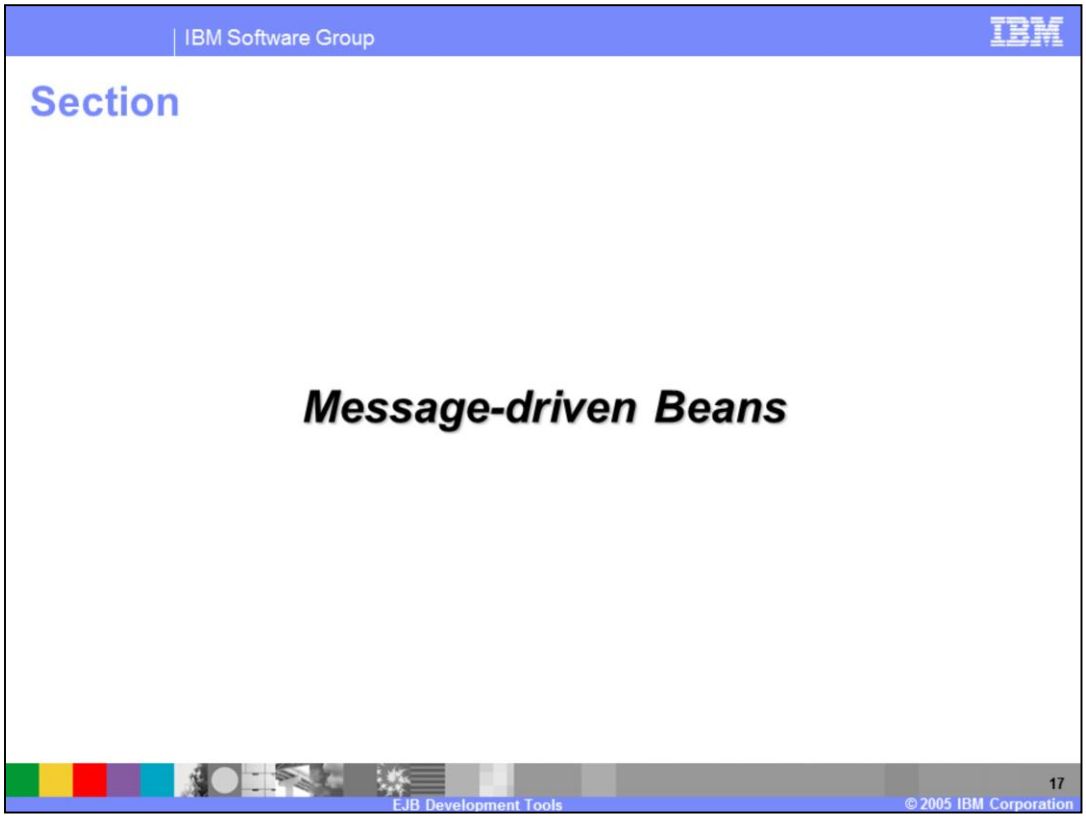
- Find and Select EJB QL queries can be created for finding entity beans or CMP attributes
- Queries created through EJB Deployment Descriptor editor or annotations
- Sample queries available as a starting point for custom queries



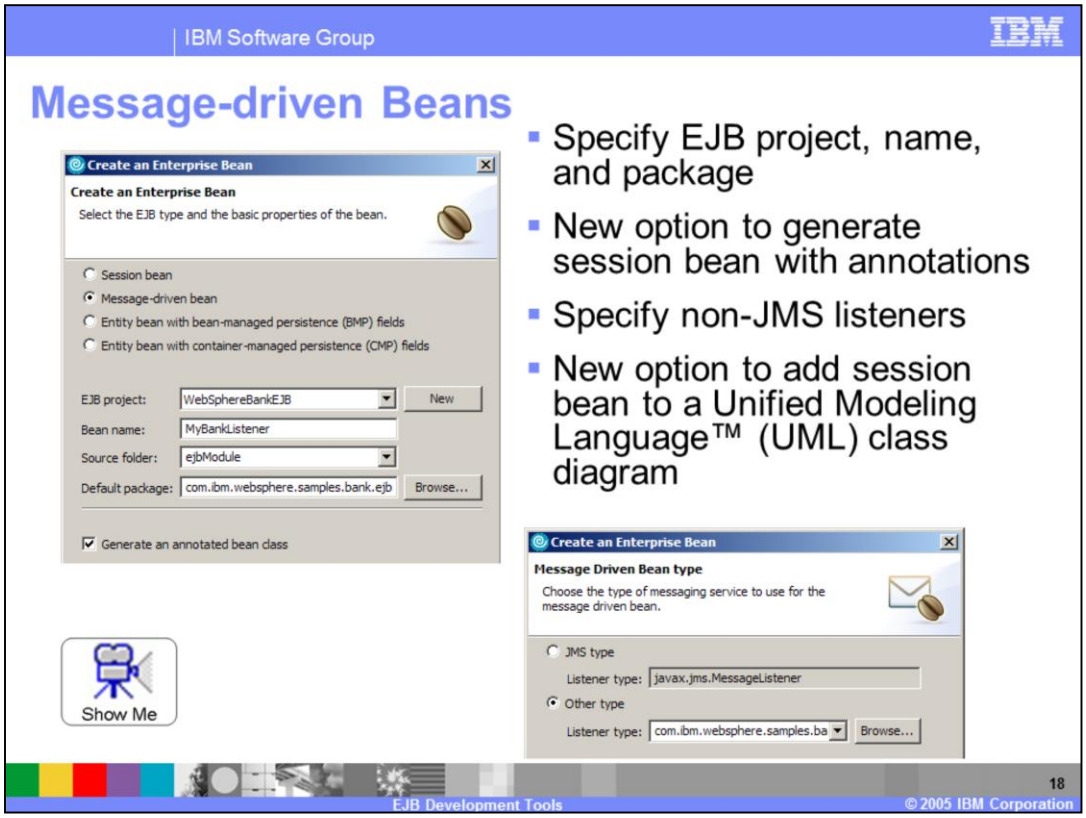
EJB QL queries can be specified for CMP entity beans. The syntax for EJB QL is very similar to SQL. You can start with the sample queries supplied and modify them to suit your needs.



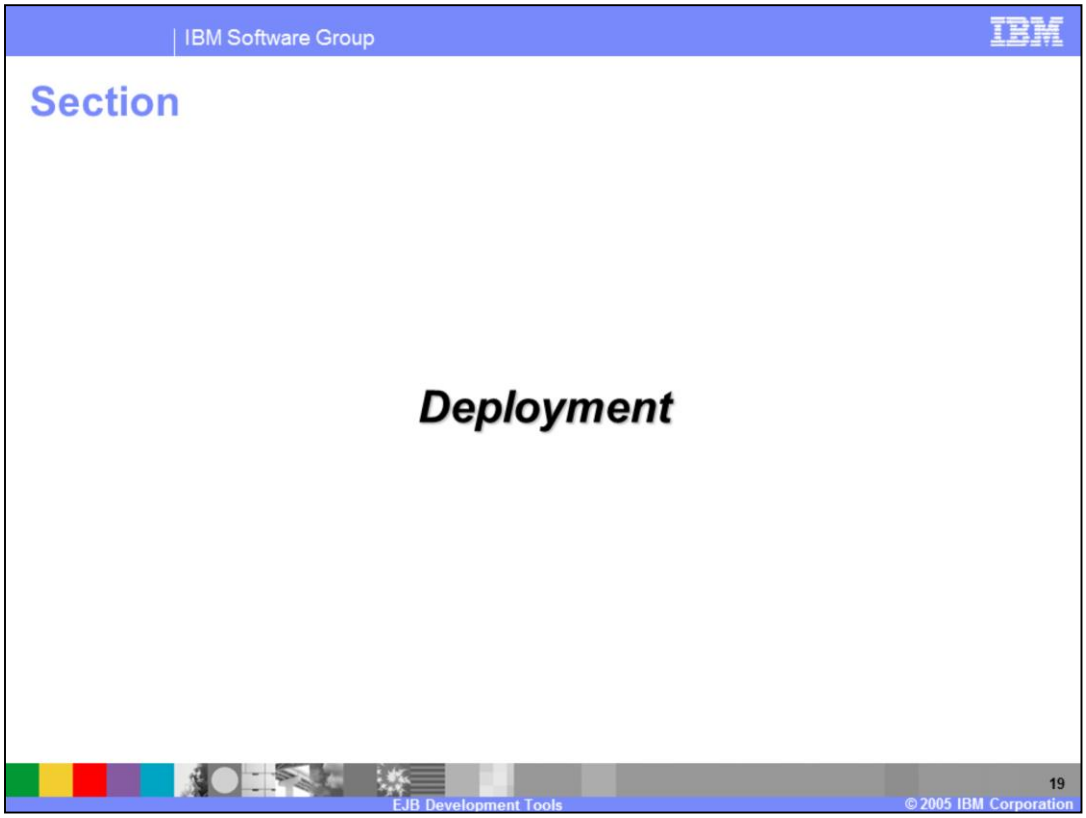
CMP Entity Beans can be mapped to various database products. You can also create mappings for multiple databases in a single project, which can be helpful for unit testing. The backend ID setting is used to determine which database mappings should be used at deploy time.



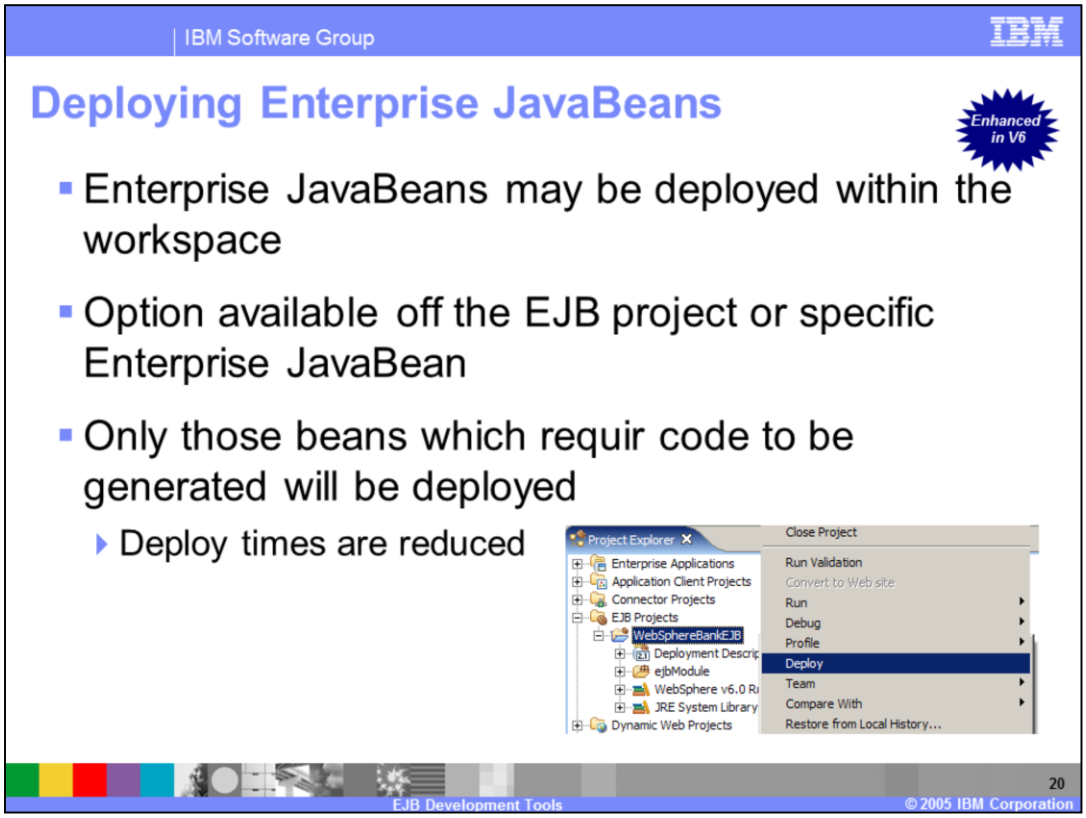
This section will cover Message-driven Beans.



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.

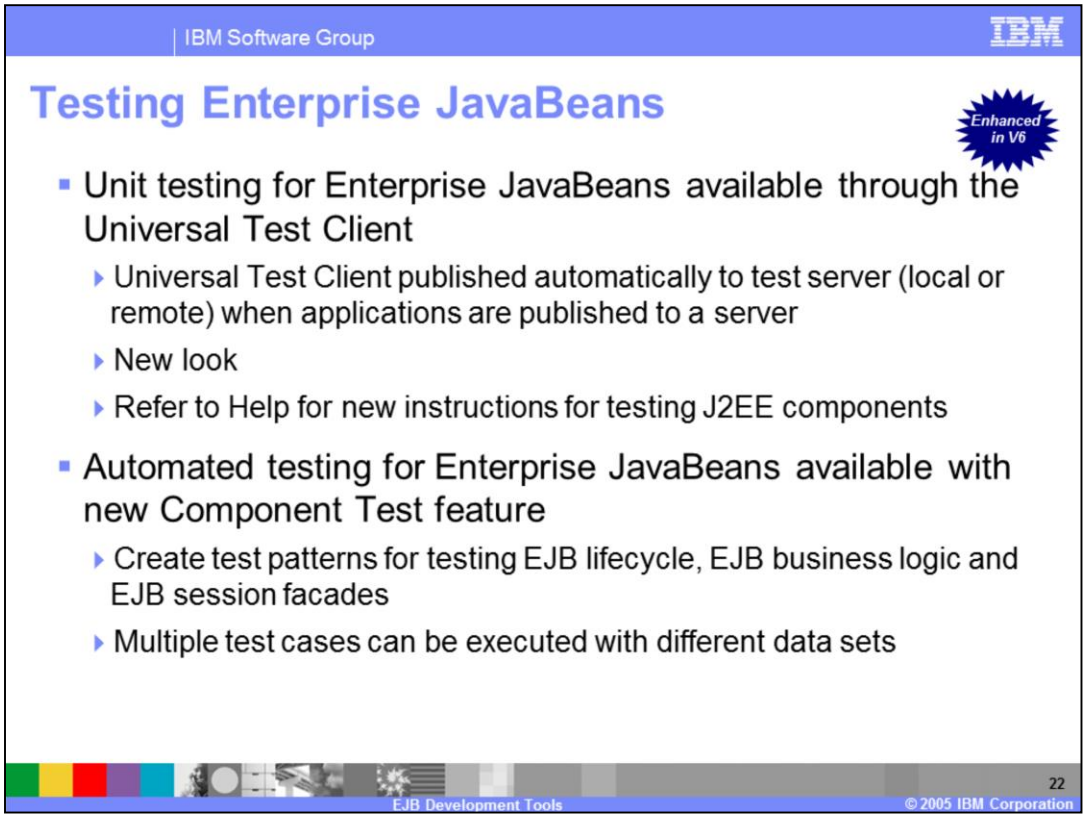


This section will cover deployment of Enterprise JavaBeans.

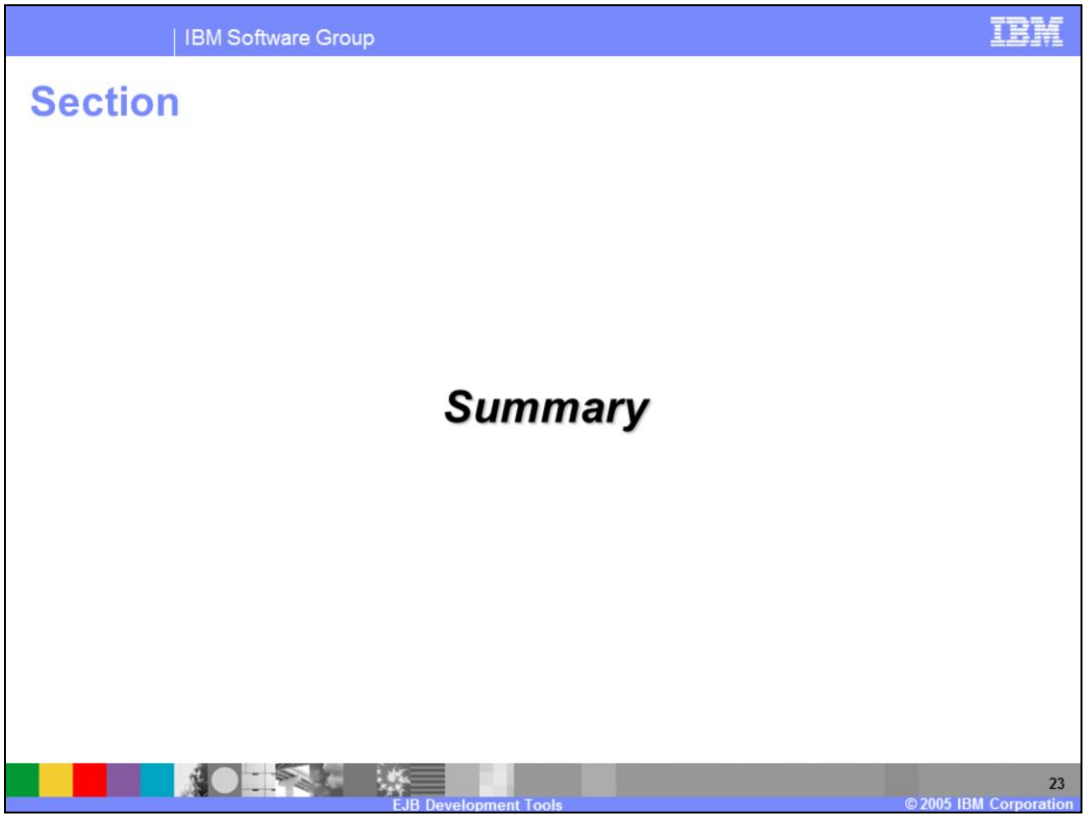


Once you have finished developing your EJB, including mappings, relationships and EJB QL queries, the next step is to prepare the EJB for deployment in a production environment. This can be accomplished within Rational Application Developer using the deploy operation, which has been improved in V6.0 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.

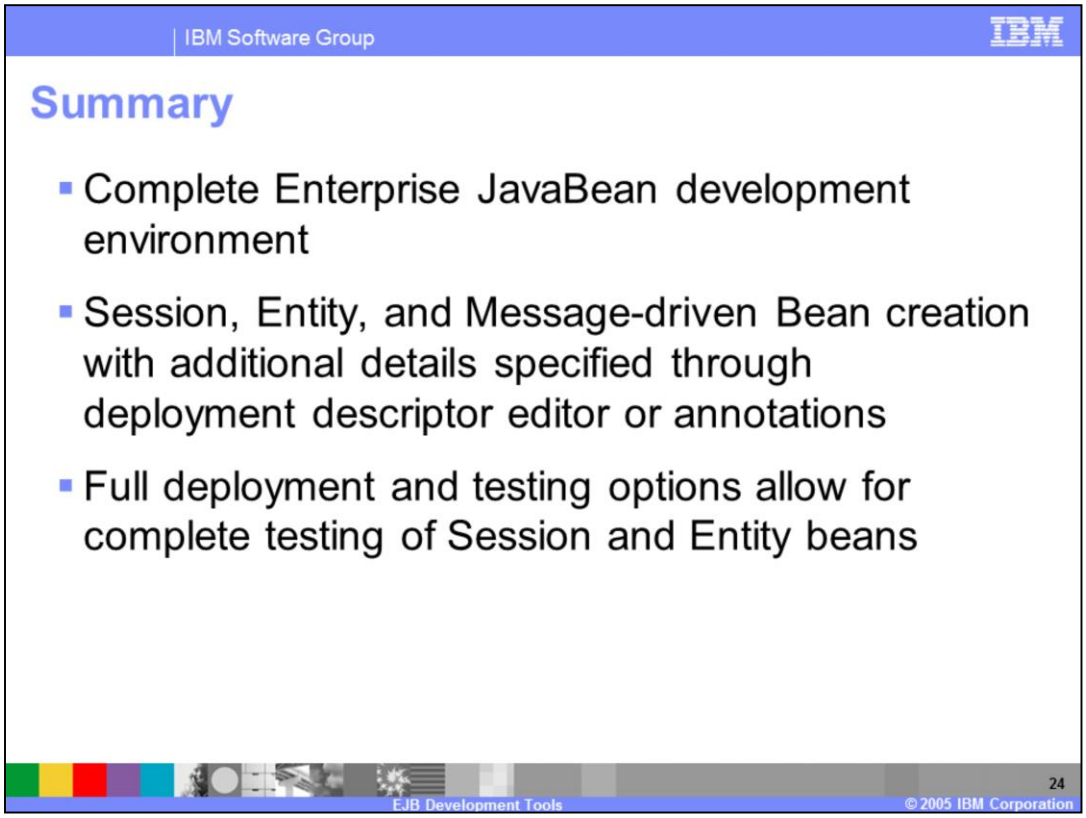
This section will cover testing.



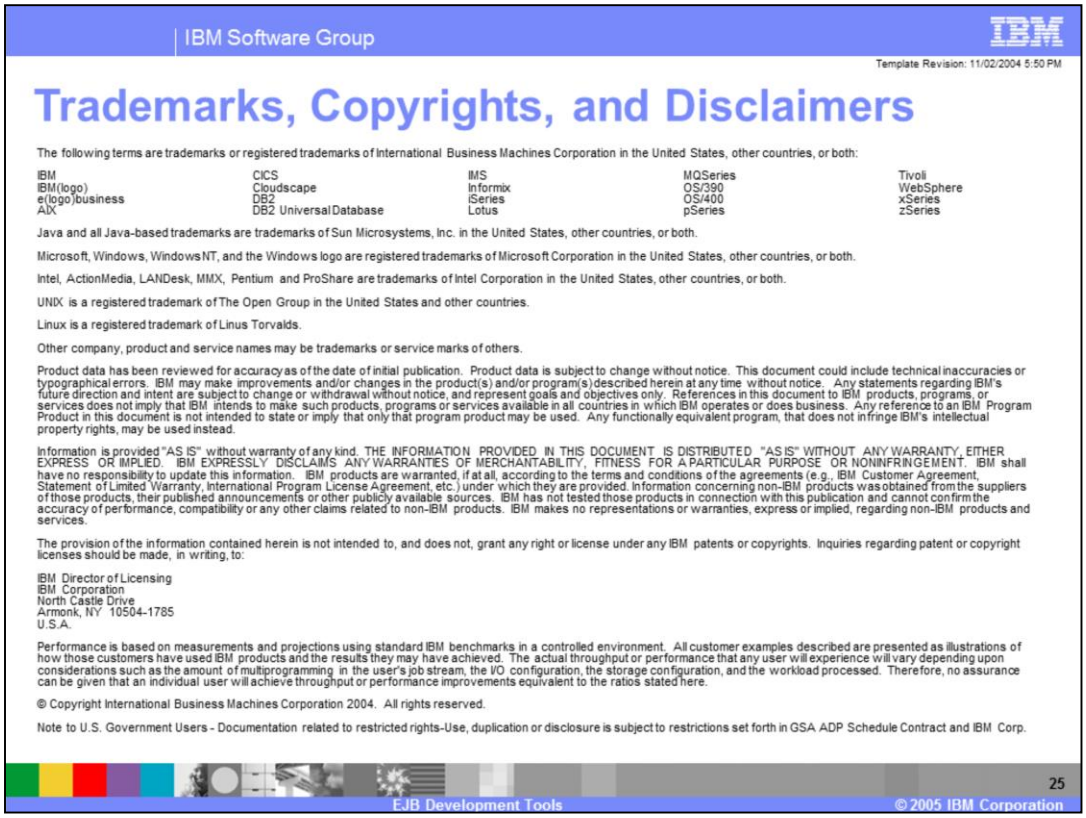
The universal test client allows you to unit test Session and Entity Beans by automatically publishing the beans to any test server in the environment. This feature has a different look and feel when compared with previous versions. It is recommended that you consult the help available for this tool 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.



This section will provide a summary of the presentation.



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.



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