Chapter 9 Bean Persistence and Versioning

1. To declare a class so that its objects will be serializable, simply implement the java.io.Serializable interface.

2. The method writeObject is used to write an object to the ObjectOutputStream, and readObject is used to read an object from the ObjectInputStream. Attempting to store an object that does not support the Serializable interface would cause java.io.NotSerializableException.

3. To serialize a component in the Explore, choose Customize Bean in the context menu of the component in the Explorer.

4. The transient modifier is to mark a nonserializable property.

5. To load a bean directly without using the new operator, use the Beans.instantiate(…) method.

6. To set the JBuilder project to generate the code using the Beans.instantiate() method to instantiate beans, check the Use Beans.instantiate(…) option in the Code Style page of the Project Properties.

7. A new bean can be added to the ToolBox by placing its .jar file in the jars directory or by choosing File, LoadJar from the menu. To associate a source bean with a target bean in the BeanBox, select the source bean, then point the mouse to the target bean with mouse pressed.

8. The bean is saved using the writeObject method and loaded to the BeanBox using the readObject method by Java's built-in serialization system. You can implement these methods to add customized operations to preserve bean persistence. How do you customize serialization using the Externalizable interface?

   You can define your own mechanism to store and restore object data by implementing the Externalizable interface. This interface is a subinterface of Serializable and has two public methods, writeExternal and readExternal. Externalizable is a subinterface of Serializable?

9. To specify that the new class is compatible with its old version, add the old class's ID as a static final long constant named serialVersionUID in the new class, so that the new class uses the same ID as its old version.