**PERSISTENCE**

Q. How can configuration settings of a bean be made persistent?

A JavaBean can be made persistent by serialization. There are three different ways of making a class serializable: automatic serialization implemented by the serializable interface, customized serialization by selectively selecting which parts to serialize, and customized file formatting implemented by the externalizable interface. The most common form of setting up persistence is with automatic serialization.

Automatic serialization is done by implementing java.io.Serializable into the class. As provided in code, you can see this being implemented in EXAMPLE 1. This tells the Java Virtual Machine (JVM) that you are making sure this class will work with default serialization. There are a few things that are particular about automatic serialization that must be followed in order to work: If there is a super class then implementing serializable is not required, the class must have a no-argument constructor, all methods and variables will be serialized unless specified using the transient modifier.

Selective serialization is easiest done by using the transient modifier on a specific method. As long as a bean is implementing the Serializable interface, you can use a transient modifier to force it to not be serialized. You can also see this in EXAMPLE 1 where the transient modifier is set to the email private variable. Another way to selectively choose to serialize your data is by overriding the writeObject and readObject methods of Serializable. If you need more control over the Serializable method then this is a better way to go.

The third way to obtain persistence is by implementing the Externalizable interface within your bean. This is for complete and full control over your serialization. In order to utilize the Externalizable interface you need to override two of its methods: readExternal and writeExternal. This is shown in EAXMPLE 2. As a requirement, this class will also need a no-argument constructor a well.