

Notas técnicas de JAVA Nro. 7– Tip Breve

(Lo nuevo, lo escondido, o simplemente lo de siempre pero bien explicado)

JAVA Basics: Diferencias conceptuales entre JavaBeans y Enterprise JavaBeans (EJB)

Tema: Java Beans, EJBs

Descripción: Este tip pretende esclarecer a qué se hace referencia exactamente cuando se habla de JavaBeans Vs. Enterprise JavaBeans, definirlos y diferenciarlos de otras componentes, como clases Java normales, Beans ActiveX y Servlets.

Nivel: Básico

Fecha pub: Abril 2005

*"Notas Técnicas de JAVA" se envía con frecuencia variable y absolutamente **sin cargo** como un servicio a nuestros clientes. Contiene notas/recursos/artículos técnicos desarrollados en forma totalmente objetiva e independiente. Teknoda es una organización de servicios de tecnología informática y **NO comercializa hardware, software ni otros productos**. Si desea suscribir otra dirección de e-mail para que comience a recibir los tips envíe un mensaje desde esa dirección a develop@teknoda.com, indicando su nombre, empresa a la que pertenece, cargo y país.*

Lista de Tips publicados hasta la fecha:

1. JAVA Basics: Cómo conformar un entorno de programación JAVA. Parte I: Selección e instalación de un IDE gratuito.
2. Una introducción a JDBC (Java Database Connectivity) (Acceso a bases de datos desde JAVA)
3. Manejo del error "Bad Magic Number"
4. Java Basics: Entendiendo la Java Virtual Machine
5. Organización de memoria en JAVA Vs. Modelo Tradicional
6. JAVA Basics: Entendiendo las applets

Próximos Tips:

Nivel Técnico avanzado

- JAVA Vs. C++

Nivel Básico

- JAVA Basics: Entendiendo los applets
- JAVA Basics: Mitos y Verdades sobre JAVA

Tabla de contenido

El objetivo del tip es explicar las diferencias conceptuales entre un JavaBean y un Enterprise Java Bean (EJB). Los siguientes puntos serán cubiertos en el mismo :

- I. Introducción
- II. JavaBeans vs. EJBs
- III. Más sobre JavaBeans
- IV. Más sobre Enterprise Java Beans (EJBs)
- V. Conclusión
- VI. Dónde obtener información

I. Introducción

Los “Java Beans” y los “Enterprise Java Beans” **NO son lo mismo**. Más aún, ninguno de ellos es una extensión o variante del otro. Es importante, por lo tanto, conocer exactamente a qué se hace referencia cuando se utilizan cada una de estas designaciones.

El aspecto que tienen en común JavaBeans y EJB’s, es que ambos son modelos de componentes dentro de una arquitectura JAVA, esto es, átomos o elementos de software reusables, codificados en JAVA, y ensamblados luego como bloques para construir aplicaciones.

Sin embargo, JavaBeans y EJB’s responden a propósitos completamente distintos, y sus “packages” (tipos básicos e interfaces) son completamente distintos también.

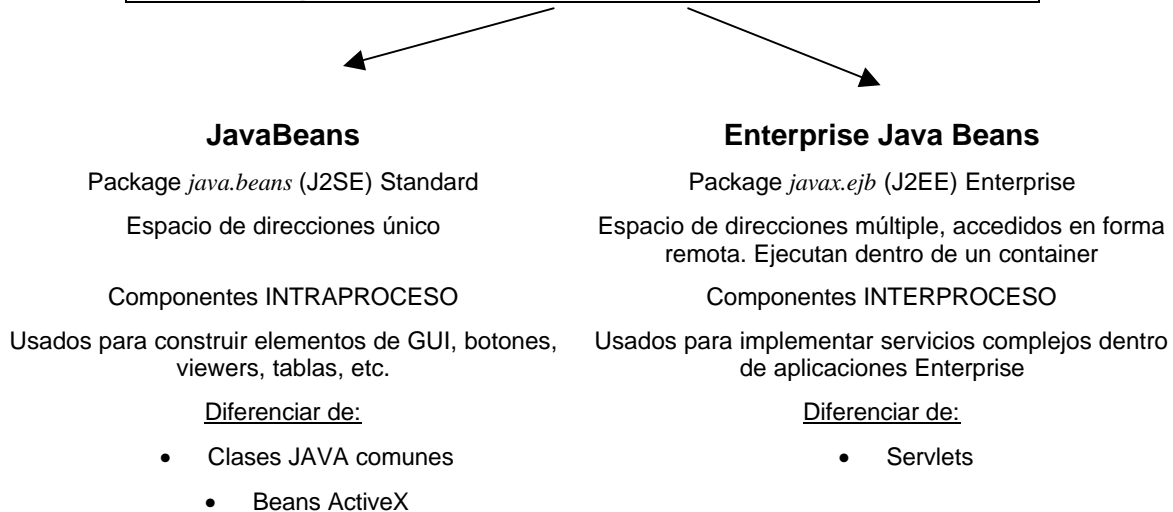
II. JavaBeans Vs. EJB’s

La especificación **JavaBeans** define los componentes de arquitectura para la plataforma **J2SE (Java 2 Standard Edition)**. Los JavaBeans son, en esencia, programas reusables que se desarrollan y se ensamblan fácilmente para crear aplicaciones sofisticadas y que pueden ser utilizados en cualquier aplicación que entienda el formato JavaBean. La especificación original de los JavaBeans está basada en el package *java.beans* que es un paquete estándar dentro del JDK. Los componentes que se construyen sobre la especificación JavaBeans son componentes **INTRAPROCESO** que viven dentro de un espacio de direcciones único, y que, **típicamente, se utilizan para manejar aspectos de la interfaz gráfica de usuario, botones, tablas, viewers HTML**, etc.

La especificación de los **EJB’s**, en cambio, está basada en el package *javax.ejb*, que **pertenece a la extensión J2EE (Java 2 Enterprise Edition)** del lenguaje. Responden a una arquitectura más compleja. Los componentes construidos de acuerdo a esta especificación son componentes **INTERPROCESO**, que viven en espacios de direcciones múltiples, como objetos distribuidos. Estos componentes son usados como objetos de negocio en aplicaciones transaccionales, y accedidos en forma remota. En cierta forma, los EJB’s se parecen más a los Servlets, que a los JavaBeans.



Beans: Componentes de Software reusables escritos en JAVA.....



III. Más sobre JavaBeans

Como explicáramos anteriormente, los JavaBeans son como bloques Lego de JAVA. Rápidamente se puede construir aplicaciones con tan sólo encastrar los distintos JavaBeans.

En cierta forma, son similares al control ActiveX. La principal diferencia entre ActiveX y JavaBean es que los ActiveX pueden ser desarrollados en cualquier lenguaje de programación pero ejecutados solo en plataformas Windows, mientras que los JavaBeans pueden ser desarrollados únicamente en JAVA, pero pueden correr en cualquier plataforma. En ambas tecnologías los componentes pueden ser propiamente contruidos o comprados a terceros.

Los JavaBeans son en esencia como clases normales de Java que adhieren a ciertas convenciones especiales. Cabe esclarecer entonces cuál es la diferencia ente un JavaBean y la instancia de una clase normal Java. Lo que diferencia a los beans de las típicas clases de Java es la **introspección**. Las Herramientas pueden reconocer patrones predefinidos en las firmas de métodos y definiciones de clase pudiendo “mirar dentro del bean” determinando sus propiedades y comportamiento. El estado de un bean puede ser manipulado al momento de ser ensamblado como parte de una gran aplicación. La aplicación ensamblada es referida en **tiempo de diseño** en contraste a **tiempo de ejecución**. En tanto a este esquema de trabajo, la firma de métodos debe seguir un cierto patrón que permita la introspección de herramientas para reconocer como el bean puede ser manipulado, ambos en tiempo de diseño y en tiempo de ejecución. No es necesario tener una herramienta constructora para poder crear o testear al bean. El patrón de firma fue diseñado para ser fácilmente reconocido por los lectores humanos tanto como para las herramientas constructoras. Lo primero que se debe aprender cuando se aprende a construir beans es como reconocer un método constructor que siga esos patrones.

No todo módulo de software útil debe ser un bean. Si el componente software va a ser manipulado visualmente es un candidato a bean (ej: botones, hojas de cálculo, etc.) dado que algunas

funcionalidades son mejor provistas a través de interfaces programadas. Por ejemplo los accesos SQL o JDBC API encajarían mejor en una biblioteca de clase que a través de un bean.

Para poder llevar a cabo los distintos propósitos, los JavaBeans soportan:

- **Customización:** permite al usuario alterar la apariencia y comportamiento del bean.
- **Propiedades:** permiten al bean ser manipulado programáticamente, en tanto como soportan su customización.
- **Persistencia:** permite la instanciabilidad de los objetos para ser transferidos y salvados indefinidamente.
- **Introspección** permitiendo ser analizados por herramientas para saber cómo funciona el bean.
- **Eventos:** permite que el bean dispare eventos e informe de los mismos a las herramientas informando cuáles puede manejar.

IV. Más sobre Enterprise JavaBeans

La especificación de Enterprise JavaBeans (EJB) 1.1 define el modelo para el desarrollo y “deployment” (despliegue) de componentes servidores, luego utilizados para construir aplicaciones orientadas a objetos y transaccionales. Estos componentes de servidor, llamados Enterprise JavaBeans contienen y proveen servicios remotos a clientes distribuidos a través de la red. Como explicáramos anteriormente, son los bloques constructores esenciales de la plataforma J2EE. Los EJB’s son bien distintos a los JavaBeans; de hecho, podría pensarse que funcionalmente están más cerca de los servlets. No obstante, como trataremos más adelante, se diferencian también de los servlets en múltiples aspectos.

Los EJB’s corren en un entorno especial llamado **EJB container**. El container alberga y administra un enterprise bean en la misma forma que el Java Web server alberga a un Servlet, o un browser HTML alberga a un Java Applet. Un enterprise bean no puede funcionar afuera del EJB container. El container maneja cada aspecto del bean en tiempo de ejecución incluyendo: accesos remotos al bean, seguridad, persistencia, transacciones, concurrencia, y acceso a un pool de recursos.

Para que un enterprise bean pueda ser utilizado debe ser desplegado, o “deployado”, en el container.

El container aísla al bean del acceso directo de las aplicaciones clientes. Cuando una aplicación cliente invoca un método remoto de un enterprise bean, el contenedor primero intercepta automáticamente la invocación para asegurar que la persistencia, transacción, y seguridad son aplicadas correctamente a cada operación cliente invocada en el bean. Por tal motivo el desarrollador de beans no debe programar este tipo de lógica dentro del código del bean en sí. El desarrollador de Enterprise bean puede concentrarse en el encapsulamiento de las reglas de negocio, mientras que el container se ocupa de todo el resto.

Los Enterprise JavaBeans clientes pueden ser aplicaciones standalone, servlets, applets, u otros enterprise beans.

A los efectos de completar la diferenciación de los EJB’s, cabe contrastarlos con los servlets. Esto implica tener en cuenta:

- Ambos se encuentran dentro del “contenedor” del software de terceros, el cual provee su soporte de runtime: en el caso de los Servlets, es un web-server add-on.; en el caso de los EJB's, éste es un servidor de aplicaciones EJB.
- Al igual que los applets, ni los Servlets ni los EJB's usan main() como su punto de entrada. Ellos tienen otros puntos de entrada predefinidos.
- Los Servlets generalmente usan HTTP como protocolo client-server. Los EJB's usan RMI (Remote Method Invocation) que es un protocolo mucho más rico.
- Generalmente, hay una sola instancia de un Servlet corriendo en el contenedor web a la cual acceden todos los clientes. En cuanto a los EJBs, típicamente hay una instancia de bean de sesión por cliente y una instancia de bean de entidad compartida por todos los clientes. Esto último se debe a que los beans de entidad corresponden a un registro específico en la base de datos.
- Los Servlets, generalmente, generan contenido para browsers mientras que los EJBs pueden soportar funciones de lógica de negocios más complejas, basándose en la utilización de RMI y evitando las restricciones HTTP.

V. Conclusión

Las principales diferencias entre Enterprise JavaBeans y JavaBeans son:

1. Los primeros son componentes de Servidores y los segundos de escritorio.
2. Ambos pertenecen a distintas ediciones de la plataforma JAVA: J2EE y J2SE respectivamente.
3. Las diferencias entre sus funcionalidades y origen de necesidad son abismales.

VI. Dónde Obtener Información Adicional

Sitio de Sun: <http://java.sun.com>

Foro de desarrolladores : <http://forum.java.sun.com>

Copyright 2005 Teknoda S.A. Abril 2005 JAVA es marca registrada de Sun. SAP, R/3 y ABAP son marcas registradas de SAP AG. AS/400 es marca registrada de IBM. Todas las marcas mencionadas son marcas registradas de las empresas proveedoras.

La información contenida en este artículo ha sido recolectada en la tarea cotidiana por nuestros especialistas a partir de fuentes consideradas confiables. No obstante, por la posibilidad de error humano, mecánico, cambios de versión u otro, Teknoda no garantiza la exactitud o completud de la información aquí volcada.

Dudas o consultas develop@teknoda.com