

CURSO DE JAVASERVER FACES

MANAGED BEANS EN JSF



Ing. Ubaldo Acosta

Por el experto: Ing. Ubaldo Acosta



CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

Hola, te saluda nuevamente Ubaldo Acosta. Espero que estés listo para comenzar con esta lección.

Vamos a estudiar el concepto de Managed Beans en JSF.

¿Estás listo? ¡Vamos!

CONCEPTO DE MANAGED BEANS EN JSF

- Un Managed Bean es una clase Java que sigue la nomenclatura de los JavaBeans
 - ✓ Los Managed Beans no están obligados a extender de ninguna otra clase
- Aunque JSF no define una clasificación para los Managed Beans, podemos definir las siguientes:
 - ✓ Beans de Modelo: Representan el Modelo en el patrón MVC
 - ✓ Beans de Control: Representan el Controlador en el patrón MVC
 - ✓ Beans de Soporte o Helpers: Contienen código por ejemplo de Convertidores
 - ✓ Beans de Utilerías: Tareas genéricas, como obtener el objeto *request*

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

Una JavaBean es una clase Java que sigue las siguientes convenciones:

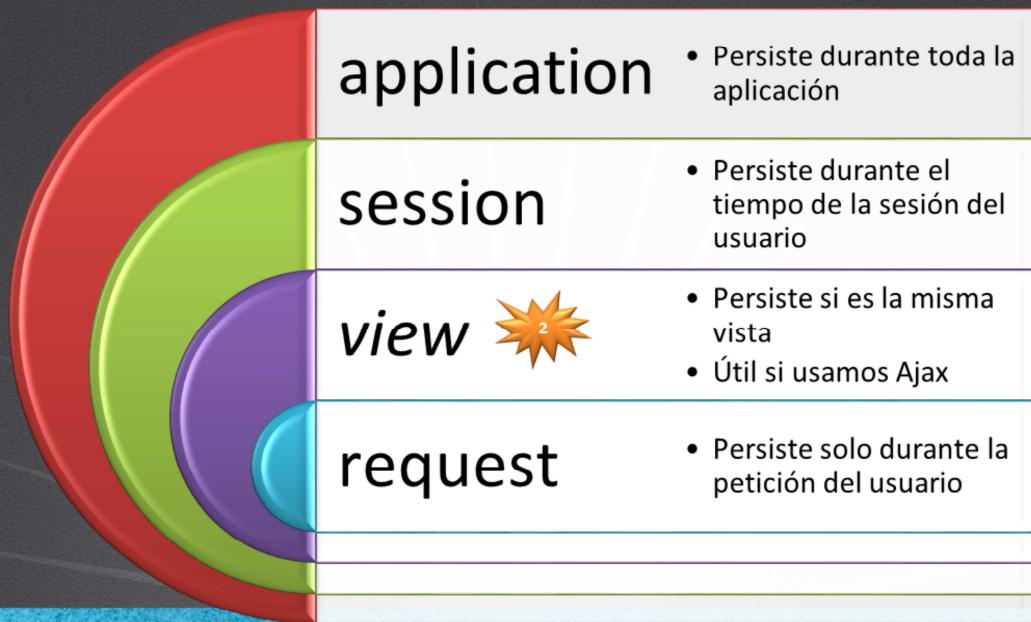
- Constructor vacío
- Atributos de clase privados
- Por cada atributo, se crean los métodos getters y setters

El Objetivo de los Managed Beans es controlar el estado de las páginas web. JSF administra automáticamente los Managed Beans:

- Crea las instancias
 - Por ello la necesidad de un constructor vacío
- Controla su ciclo de vida
 - JSF determina el ámbito o alcance (request, session, application, etc) de cada Managed Bean
- Llama los métodos getters o setters
 - Por ejemplo: <h:inputText value="#{empleado.apellidoPaterno}" al hacer submit llamará el método setApellidoPaterno()
 - Otro ejemplo: #{empleado.nombre} indirectamente llamará al método getNombre()

Los Managed Beans los podemos declarar de varias formas, ya sea utilizando anotaciones o en el archivo faces-config.xml.

ALCANCE DE LOS MANAGED BEANS EN JSF



CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

Los JSF proveen los siguientes alcances o ámbitos (scopes) para almacenar información en forma de un mapa o tabla (llave, valor).

JSF 2 agregó algunos otros alcances, llamados view, custom.

El alcance de **custom** es simplemente un estructura de datos *map* que enlaza objetos (values) con llaves (keys). El tiempo de vida del *map* es administrado por el implementador.

Las anotaciones para el manejo de alcances JSF son:

- @ RequestScoped (ámbito por default)
- @ ViewScoped
- @ SessionScoped
- @ ApplicationScoped

Las anotaciones de alcance se utilizan después de la anotación del Bean, por ejemplo:

```
@ManagedBean @SessionScoped
```

```
public class UsuarioForm{ }
```

CDI Y EL ALCANCE DE BEANS EN JSF



CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

JSF y CDI (Context and Dependency Injection de Java EE) tienen en común los alcances de request, session y application. Aunque se encuentran en distintos paquetes.

Paquete para JSF: `javax.faces.bean`

Paquete para CDI: `javax.enterprise.context`

Además, CDI agrega un alcance más, conocido como **conversation**. Este alcance permite guardar información entre páginas relacionadas y la información almacenada es eliminada hasta que se cumple cierta meta.

Las anotaciones para el manejo de alcances CDI son:

- `@ RequestScoped` (ámbito por default)
- `@ ConversationScoped`
- `@ SessionScoped`
- `@ ApplicationScoped`

EJEMPLO DE USO DE MANAGED BEANS

1. Creación del Bean

```
@ManagedBean  
@SessionScoped  
public class EmpleadoForm{...}
```

Uso en la página JSF con EL (Expression Language):

```
#{{ empleadoForm.atributo }}
```

2. Creación del Bean con nombre personalizado y notación CDI

```
@Named  
public class EmpleadoForm{...}
```

Uso en la página JSF con EL:

```
#{{ empleadoForm.atributo }}
```

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

Podemos crear Managed Beans que posteriormente serán utilizados en las páginas JSF, comúnmente utilizando Expression Language de JSF (EL).

En el caso 1, se utilizan las anotaciones de JSF. Esto genera un objeto de tipo EmpleadoForm llamado empleadoForm (el nombre empieza con minúsculas), además JSF agrega este objeto al alcance de sesión.

Por lo que el bean creado, se puede utilizar directamente utilizando EL:
`#{{ empleadoForm.atributo }}`

En el caso 2, se utiliza la notación CDI de Java EE (Versión Empresarial de Java). El proceso es muy similar a la notación de JSF. Para configurar los beans CDI, se debe agregar un archivo beans.xml al directorio WEB-INF, que puede quedar en blanco. Este archivo se utiliza para declarar cuestiones específicas en el manejo de CDI.

Si el despliegue de nuestra aplicación se va a realizar en un Servidor JEE, se recomienda utilizar CDI, en cambio si se va a utilizar un Servidor Web como Tomcat, se puede utilizar las anotaciones de JSF.

INYECCIÓN DE MANAGED BEANS

JSF soporta inyección de dependencias de manera muy simple.

La inyección se puede hacer de 2 maneras:

- Con anotaciones dentro del Managed Bean:

```
@ManagedProperty(value= "#{nombreBean}")
```

- En el archivo faces-config.xml utilizando la siguiente etiqueta dentro de un managed bean:

```
<managed-property>#{nombreBean}</managed-property>
```

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

En muchas ocasiones una página JSF puede utilizar varios Managed Beans, de ahí la necesidad de relacionar distintos Managed Beans.

La inyección de dependencias (Dependency Injection – DI) es un derivado de IoC. Uno de los iniciadores de este principio fue Spring framework y al día de hoy es una práctica muy utilizada en arquitecturas JEE.

Podemos lograr la inyección de dependencias de 2 maneras:

- 1) Con anotaciones: Manejando la anotación @ManagedProperty en la propiedad de la clase Managed Bean a injectar.
- 2) Con el archivo faces-config.xml: Agregando la etiqueta <managed-property> dentro del Managed Bean declarado.

ARCHIVO FACES CONFIG.XML

- Uso del archivo faces-config.xml:
 - Configuración Centralizada
 - Reglas de navegación
 - Declaración de Managed Beans
 - Inyección de Dependencias
 - Definir configuraciones regionales
 - Registro de validadores
 - Registro de listeners
 - Entre otros puntos...
- Ubicación del archivo de configuración: **WEB-INF/faces-config.xml**

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

El archivo faces-config.xml nos permite agregar la configuración de JSF de manera centralizada, incluyendo las características descritas.

El formato general del archivo JSF depende de la versión a utilizar, en este caso es un ejemplo de la versión JSF 2.2.

```
<?xml version='1.0' encoding='UTF-8'?>
<faces-config xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
  http://xmlns.jcp.org/xml/ns/javaee/web-facesconfig_2_2.xsd" version="2.2">
  ...
</faces-config>
```

Con el uso de anotaciones, cada vez es menos indispensable utilizar este archivo, aunque todavía suele utilizarse para tareas como el manejo regional (varios lenguajes en la aplicación web), declaración de variables para integración con frameworks como Spring entre otras tareas.

EJEMPLO DE DECLARACIÓN DE BEANS

```
<?xml version="1.0"?>
<faces-config ...>
    <managed-bean>
        <managed-bean-name>empleadoBean</managed-bean-name>
        <managed-bean-class>beans .EmpleadoBean</managed-bean-class>
    </managed-bean>

    <managed-bean>
        <managed-bean-name>empleadoForm</managed-bean-name>
        <managed-bean-class>forms .EmpleadoForm</managed-bean-class>
        <managed-bean-scope>session</managed-bean-scope>
        <managed-property>"#{empleadoBean}"</managed-property>
    </managed-bean>
</faces-config>
```

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

Podemos observar un ejemplo de configuración de Managed Beans utilizando el archivo faces-config.xml.

El primer bean declarado, empleadoBean, se encuentra en el paquete beans. Al no definir un scope, se crea por default en el alcance de request.

El segundo bean declarado (empleadoForm) se encuentra en el paquete forms. Dicho bean tiene una dependencia con empleadoBean, por lo que se manda a llamar el método setEmpleadoBean de la clase EmpleadoForm para injectar la dependencia.

Esta forma de configurar beans es más extensa que al utilizar anotaciones, pero permite tener centralizada la configuración en un solo archivo.

EXPRESSION LANGUAGE (EL)

El Lenguaje de Expresión (EL) de JSF nos permite simplificar el manejo de expresiones en las páginas JSF

El lenguaje EL enlaza la Vista con el Modelo del patrón MVC

```
#{}{nombreBean.propiedad}
```

¿Qué sucede al evaluar una expresión EL?

- Se busca el bean a utilizar en cierto alcance
- Se manda a llamar el método get o set de la propiedad indicada



CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

El lenguaje de Expresión (EL) nos permite evaluar y crear más fácilmente expresiones en nuestras páginas JSF.

En JSP's también existe un lenguaje de expresión, pero tiene una diferencia en su sintaxis, por ejemplo:

- En JSP: \${nombreBean.propiedad}
- En JSF: #{}{nombreBean.propiedad}

EL permite enlazar la Vista con el Modelo, por lo que desde una página JSF podemos acceder fácilmente a los Managed Beans de JSF.

Cuando una EL encuentra un bean a utilizar, lo busca en cierto alcance. El orden de búsqueda es: request, view, session, application

Una vez que se ha ubicado el Bean a utilizar, se manda a llamar el método get o set de la propiedad indicada, por ejemplo:

```
nombreBean.getPropiedad();
```

OBJETOS IMPLÍCITOS EN EL

El Lenguaje EL nos permite acceder fácilmente a objetos implícitos (ya instanciados) en las páginas JSF, algunos ejemplos son:

```
cookie: Map  
facesContext: FacesContext  
header: Map  
headerValues: Map  
param: Map  
paramValues: Map  
request (JSF 1.2): ServletRequest or PortletRequest  
session (JSF 1.2): HttpSession or PortletSession  
requestScope: Map  
sessionScope: Map  
applicationScope: Map  
initParam: Map  
view: UIViewRoot
```

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

Al igual que en los JSPs, la tecnología JSF permite acceder a varios de los objetos implícitos disponibles en sus propias páginas.

Por ejemplo, si queremos acceder a un bean llamado empleadoBean, en el alcance de sesión, podemos utilizar cualquiera de las siguientes notaciones:

- 1) #{empleadoBean.propiedad}
- 2) #{sessionScope.empleadoBean.propiedad}
- 3) #{sessionScope["empleadoBean"].propiedad}

Debido a que el bean de empleado se encuentra en el alcance de sesión, podemos recuperarlo ya sea directamente o a través de la variable implícita sessionScope.

Además la variable sessionScope, al ser un mapa, podemos utilizar la notación de arreglos para acceder a sus elementos.

EL también nos permite acceder a propiedades de manera anidada, por ejemplo:

#{empleadoBean.direccion.calle}

En este último ejemplo, solo debemos asegurarnos que los valores anidados (objeto dirección) no sean nulos, de lo contrario dará un NullPointerException.

OPERADORES EL

El Lenguaje EL cuenta con operadores para evaluar expresiones:

Aritméticos: +, -, *, / (div), % (mod)

Relacionales: == (eq), != (ne), < (lt), > (gt), <= (le), >= (ge)

Lógicos: && (and), || (or), ! (not)

Condicionales: x ? y : z

Empty: vacíos (regresa verdadero si el valor de la variable es null o sin elementos al manejar String, Arreglos ,Mapas o Colecciones)

Extracto de Código

```
<h:panelGroup rendered="#{bean.propiedad ne empty and !bean.ocultar} " >
...
</h:panelGroup>
```

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

EL cuenta con una serie de operadores para evaluar expresiones, aunque se considera buena práctica no utilizar de manera excesiva esta característica ya que se rompe con el patrón de diseño MVC al agregar lógica en la Vista.

Como podemos observar en el extracto de código, el elemento **panelGroup**, el cual es similar a un elemento **div** de HTML, se muestra de manera condicionada utilizando la propiedad de rendered. Si la condición resulta verdadera el elemento se muestra.

En el ejemplo se utilizan los operadores ne, empty, and y de negación para controlar esta lógica. Este es un ejemplo muy práctico que podemos adoptar para mostrar elementos de manera condicional en las páginas JSF.

EJERCICIOS CURSO DE JAVASERVER FACES

- **ABRIR LOS ARCHIVOS DE EJERCICIOS EN PDF.**
- **EJERCICIO:** Model Managed Bean
- **EJERCICIO:** Backing Managed Bean



CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

CURSO ONLINE

JAVASERVER FACES (JSF)

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida

CURSO DE JAVASERVER FACES

www.globalmentoring.com.mx

En Global Mentoring promovemos la Pasión por la Tecnología Java. Te invitamos a visitar nuestro sitio Web donde encontrarás cursos Java Online desde Niveles Básicos, Intermedios y Avanzados, y así te conviertas en un experto programador Java.

Además agregamos nuevos cursos para que continúes con tu preparación como programador Java profesional. A continuación te presentamos nuestro listado de cursos:

- | | |
|--|--|
| <ul style="list-style-type: none">✓ Programación con Java✓ Fundamentos de Java✓ Programación con Java✓ Java con JDBC✓ HTML, CSS y JavaScript✓ Servlets y JSP's✓ Struts Framework | <ul style="list-style-type: none">✓ Hibernate Framework y JPA✓ Spring Framework✓ JavaServer Faces✓ Java EE (EJB, JPA y Web Services)✓ JBoss Administration✓ Android con Java✓ HTML5 y CSS3 |
|--|--|

Datos de Contacto:

Sitio Web: www.globalmentoring.com.mx

Email: informes@globalmentoring.com.mx

