JAVABEAN

MANEJO DE JAVABEANS CON JSP' S

- Los JSP's puede acceder a los JavaBeans.
- Un JavaBean es una clase de Java que sigue ciertas reglas básicas:
  - ✓ *Debe tener un constructor vacío.*
  - ✓ *Todos los atributos deben ser privados.*
  - ✓ *Se debe generar el método get y set para cada atributo.*
- Un JSP debe utilizar el nombre de la propiedad para acceder o modificar el atributo de un JavaBean.
- Indirectamente el JSP manda llamar al método get o set asociado con la propiedad indicada en el JSP.

Los JSP's nos van a permitir acceder a los JavaBeans que tengamos declarados en nuestra aplicación web. Un JavaBean simplemente es una clase pura de Java que sigue ciertas reglas. Una de las reglas es que debe de tener un constructor vacío y esa restricción es por lo siguiente, si tenemos una clase que no tiene un constructor vacío entonces deberemos tener algún constructor con cierto número de parámetros, por lo que cuando el JSP trate de acceder a este JavaBean debe saber cuántos parámetros se le deben de especificar para poder crear una estancia de objeto JavaBean, y esto implica mayor complejidad al momento de instanciar un objeto Java.

Por ello, los JavaBeans requieren de un constructor vacío, para que no haya necesidad de indicar cuales son los argumentos que se necesitan para poder instanciar una clase de un JavaBean.

Además de tener un constructor vacío, los JavaBeans deben tener atributos privados y por cada uno de los atributos declarados debemos agregar su método get y set para cada atributo. Vamos a ver más adelante que existen ciertas excepciones, por ejemplo, si únicamente queremos acceder a la propiedad pero no tenemos intenciones de modificar la propiedad, entonces podríamos tener únicamente su método get y viceversa, en dado caso que únicamente queramos modificar la propiedad agregamos el método set respectivo y podemos omitir su método get.

Por ello, los JavaBeans requieren de un constructor vacío, para que no haya necesidad de indicar cuales son los argumentos que se necesitan para poder instanciar una clase de un JavaBean.

Además de tener un constructor vacío, los JavaBeans deben tener atributos privados y por cada uno de los atributos declarados debemos agregar su método get y set para cada atributo. Vamos a ver más adelante que existen ciertas excepciones, por ejemplo, si únicamente queremos acceder a la propiedad pero no tenemos intenciones de modificar la propiedad, entonces podríamos tener únicamente su método get y viceversa, en dado caso que únicamente queramos modificar la propiedad agregamos el método set respectivo y podemos omitir su método get.

Un JSP debe de utilizar el nombre de la propiedad del JavaBean. Indirectamente el JSP al colocar el nombre de la propiedad se manda llamar el método get o set respectivo según indiquemos en la acción que utilicemos en nuestro JSP. Vamos revisar más adelante como realizar este código.

EJEMPLOS DE PROPIEDADES DE JAVABEANS

Nombre Propiedad	Nombre de los Métodos	Código en el JSP
nombreUsuario	getNombreUsuario setNombreUsuario	<jsp:getProperty … property="nombreUsuario" /> <jsp:setProperty … property="nombreUsuario" />
eliminado	isEliminado setEliminado	<jsp:getProperty … property="eliminado" /> <jsp:setProperty … property="eliminado" />
noTelefono	getTelefono setTelefono	<jsp:getProperty … property="telefono" /> <jsp:setProperty … property="telefono" />
codigo_postal	getCodigo_postal setCodigo_postal	<jsp:getProperty … property="codigo_postal" /> <jsp:setProperty … property="codigo_postal" />

Podemos observar los siguientes ejemplos, tenemos el nombre de la propiedad, el nombre de los métodos asociados get y set y el código que estamos utilizando dentro del JSP.

Por ejemplo, si tenemos una propiedad llamada nombreUsuario podemos observar que su método get se debe de escribir como sigue getNombreUsuario. Vamos a utilizar la notación de altas y bajas, o notación de camello, es decir la n se convierte en mayúscula y la letra u también ser convierte a mayúscula, debido a que estamos separando cada palabra del nombre del método.

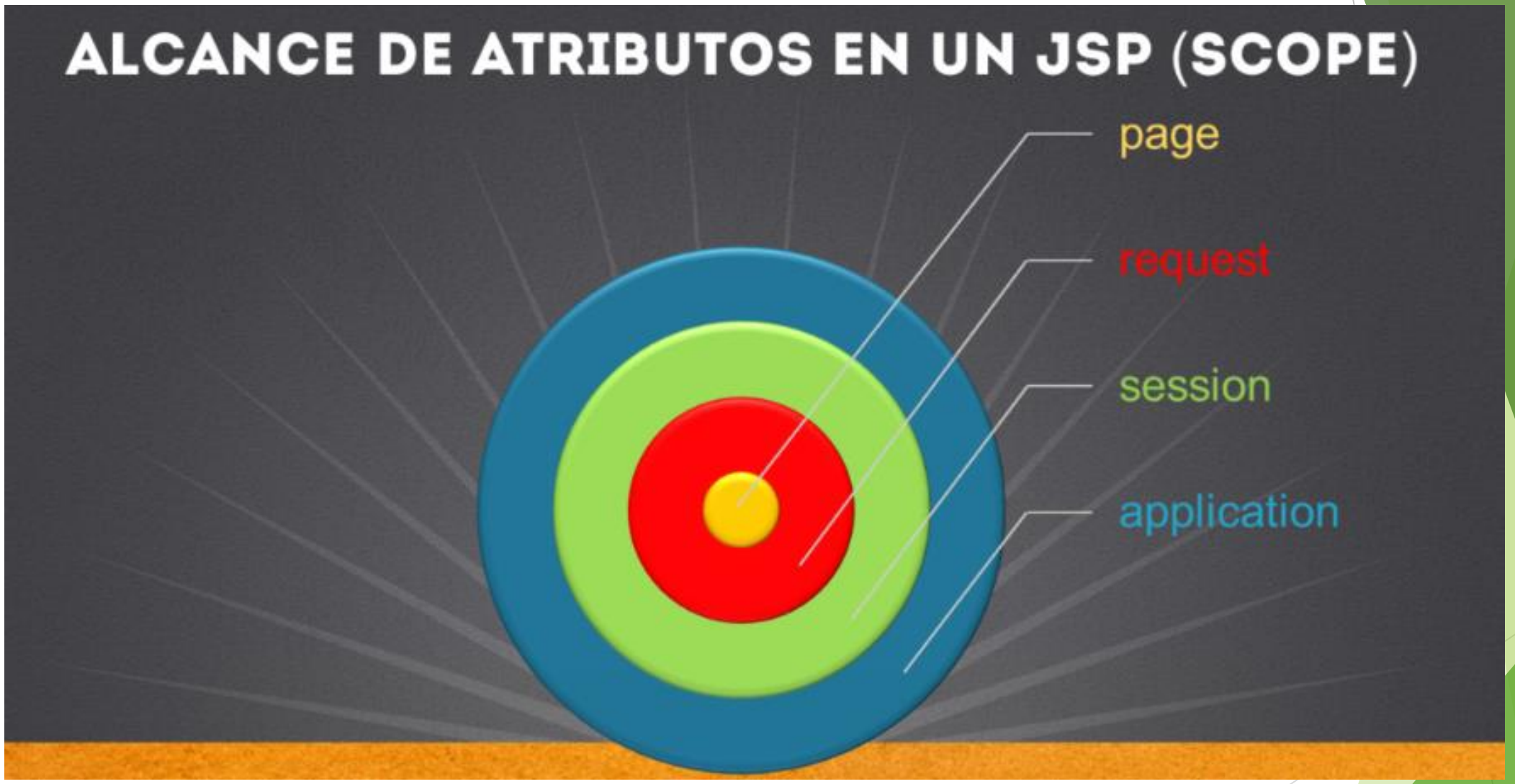
Esta es la notación de Java que debemos utilizar cuando creemos el método get y set respectivo de una propiedad.

Una vez que tenemos los métodos get y set dentro de la clase del JavaBean, dentro del JSP lo que tenemos que hacer es utilizar la siguiente acción de los JSPs, escribiremos **jsp:** y dependiendo de la acción que queramos hacer será **getProperty** o **setProperty.** Posteriormente indicamos el nombre de la propiedad a utilizar, (Ej. nombreUsuario) e indirectamente se ejecutará el nombre del método asociado a la acción respectiva, ej. getNombreUsuario() y si estamos haciendo uso de la acción setProperty se manda a llamar el método setNombreUsuario(). Por ello es que decimos que en el JSP vamos a utilizar el nombre de la propiedad e indirectamente se va a mandar a llamar el método asociado dependiendo de la acción que estemos especificando.

Vamos a revisar algunos otros ejemplos. La propiedad eliminado es de tipo booleano y para las propiedades de tipo booleano en lugar de tener un método get vamos a convertirlo por un método is, (isEliminado). El método set no cambia debido a que para modificarlo se sigue utilizando la misma anotación (setEliminado) y en el código del JSP de igual manera no cambia, vamos a utilizar su método get que indirectamente va a mandar a llamar el método is que es el de lectura, o el action de setProperty e indicamos la propiedad y se manda llamar indirectamente el método setEliminado. De igual manera hay que notar que la propiedad está en minúsculas y el método que se manda a llamar es setEliminado y la E comienza con mayúscula.

Ahora vamos a ver un caso interesante en el cual muchas veces podemos aprovecharlo para hacer nombres de métodos que no necesariamente van a mapear a una propiedad de un JavaBean, ya que respetando la nomenclatura podemos mandar a llamar métodos como podemos observar. La propiedad noTelefono, podemos crear métodos llamados getTelefono y setTelefono y las acciones del JSP deben corresponder no al nombre de la propiedad, la puede incluso no existir, sino al nombre de los métodos eliminando la palabra get o set y convirtiendo la primera letra en minúscula, con esto podrá acceder a métodos que incluso no tengan mapeada una propiedad, sino posiblemente métodos que realicen cálculos dentro del JavaBean.

Y por ultimo, vemos otra anotación de una propiedad de un JavaBean, en este caso estamos utilizando la anotación que no es muy común incluso no es recomendable, pero en ocasiones ya existe código creado de esta manera getCodigo_postal. Lo que tenemos que hacer entonces es poner el nombre de la propiedad e indirectamente se va a mandar a llamar el método getCodigo_postal en este orden si tenemos guion bajo (_). Entonces no tenemos que hacer nada al respeto si no simplemente el único que se debe de convertir en mayúsculas es la primera letra, por lo tanto el método que se manda a ejecutar es getCodigo_postal y la p sigue siendo minúscula.


ALCANCE DE ATRIBUTOS EN UN JSP (SCOPE)
page
request
session
application

Vamos a revisar a continuación el tema de alcance de las variables en los JSPs.

Básicamente el alcance de una variable es la duración o tiempo de uso de una variable en una aplicación web. Podemos observar que tenemos cuatro ***alcances*** o en inglés ***scope.*** Según hemos revisado anteriormente, hemos utilizado el objeto request para agregar cierta información y también hemos utilizado el objeto session para compartir información que puede durar más que un request, es decir, una sesión puede administrar varias peticiones (request) y por lo tanto la información que almacenemos en una sesión dura más que el tiempo que si la almacenáramos a nivel del objeto request.

Observamos en la figura que tenemos dos alcances más y estos son todos los alcances que tenemos para compartir información entre los diferentes componentes ya sean Servlets o un JSPs.

Como podemos observar el alcance de page es el inferior. Este alcance únicamente va a durar durante el tiempo de traducción de los JSPs pero ni siquiera va a durar durante el tiempo de la petición (request) de nuestra petición, por lo tanto la información que esté almacenada en este alcance de page únicamente va a poder ser accedida en el JSP que está haciendo la traducción respectiva en ese momento.

El siguiente alcance es request. Como podemos observar el alcance de request tiene mayor duración que el alcance de page. Con el alcance de request la información que almacenemos va a existir durante todo el tiempo de nuestra petición web, esto significa desde que el cliente inicia la petición hasta que el servidor regresa la respuesta a nuestro cliente va a estar disponible en la aplicación web. Por lo tanto, el alcance de request tiene mayor duración que el alcance de page.

El siguiente alcance que tenemos es el de session, podemos observar que la sesión va a durar más que request y page. La información que almacenemos en una sesión nos va a permitir guardar datos entre diferentes peticiones y así compartir información entre diferentes componentes web como pueden ser diferentes JSPs y distintos Servlets. El alcance de request también nos va a permitir compartir información entre diferentes componentes JSPs y Servlets pero únicamente durante el tiempo en que dura una petición, en cambio la sesión nos permite hacer algo similar pero el tiempo que dura la información almacenada en una sesión es mayor y por lo tanto más componentes van a poder hacer uso de la información que se almacene en una sesión.

Por último, tenemos el alcance de ***aplicación***, esto es equivalente a manejar el contexto de ServletContext, este concepto se maneja en el tema de Servlets, entonces lo que es para un servlet el alcance de ServletContext en un JSP se llama ***application.*** Cada una de estas variables son variables implícitas que tenemos en un JSP, el alcance de la aplicación es el mayor alcance que podemos tener y la información que almacenemos en ese alcance, por ejemplo con el método setAtribute, va a durar durante todo el tiempo en que este nuestra aplicación esté arriba. Por ejemplo, si estamos utilizando un servidor de aplicaciones y agregamos información a este alcance va a durar todo el tiempo que el servidor este arriba y hasta que detengamos el servidor o que bajemos nuestra aplicación es cuando se va a destruir la información que tengamos almacenada en el alcance application.

USO BÁSICO DE LOS JAVABEANS

- jsp:useBean: Permite acceder a un bean en un alcance (scope) especificado

```
<jsp:useBean id="nombreBean" class="paquete.NombreClase" />
```

- jsp:setProperty: Permite modificar una o varias propiedades de un bean especificado

```
<jsp:setProperty name="nombreBean" property="nombreUsuario" value="Juan" />
```

- jsp:getProperty: Permite acceder a una propiedad de un bean especificado

```
<jsp:getProperty name="nombreBean" property="nombreUsuario" />
```

Por último, vamos a revisar el uso básico de los Java Beans desde los JSPs. Podemos utilizar las siguientes acciones en los JSPs:

- Tenemos la acción de **useBean,** esto nos va a permitir acceder a un bean en un alcance especificado. Podemos o no especificar el alcance, si no la especificamos dentro de nuestra sintaxis del **jsp:useBean,** el alcance por default es de tipo page. En cambio si queremos indicar alguno otro alcance tenemos que poner **scope=request, session o application.** La sintaxis que vamos a utilizar es similar a esta **<jsp:useBean id="nombre" class="paquete.NombreClase"/>**. Con esta sintaxis en pocas palabras es como si creáramos una nueva instancia y el **nombreBean** es el nombre de nuestra variable y el **paquete.NombreClase** es el tipo de la clase que estamos indicando para generar este nuevo objeto **nombreBean.**

- Posteriormente tenemos la acción de **setProperty**. Esta acción nos permite modificar una o varias propiedades de un bean especificado. Para esta acción especificamos el nombre del bean previamente definido con la acción jsp:useBean, posteriormente especificamos la propiedad a modificar, seguido del valor del mismo. Ej. **property="nombreUsuario"**, esto indirectamente va a mandar a llamar el método setNombreUsuario del JavaBean respectivo.

- Por último tenemos la acción de **getProperty.** Esta acción nos permite acceder a una propiedad del bean especificado. Utilizamos la acción **jsp:getProperty,** posteriormente indicamos el nombre del bean **nombreBean** que vamos a utilizar, es decir, el nombre de la instancia declarada previamente con la acción jsp:useBean. Posteriormente indicamos una propiedad a acceder, ej. **property="nombreUsuario"**, indirectamente se va a mandar a llamar el método getNombreUsuario().

Estas son las acciones básicas que vamos a utilizar con los JavaBeans, existen varias combinaciones que podemos utilizar al crear y utilizar los JavaBeans, por lo que vamos a ver a continuación algunos ejemplos para ir detallando el uso básico de los JavaBeans en los JSPs.