UNIVERSIDAD DEL QUINDÍO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

Información general	
FECHA :	2015-02-16
DURACIÓN ESTIMADA EN MINUTOS:	90
DOCENTE:	Christian Andrés Candela
GUÍA NO.	04
Nombre de la guía:	Parametrización de atributos en una entidad

Información de la Guía

Objetivos: Aprender a parametrizar de forma adecuada los atributos de una entidad.

Conceptos Básicos: Manejo de Eclipse, Java, Bases de Datos, JDBC, XML, Glassfish.

Contextualización Teórica: Se puede restringir, validar o limitar la información a ser almacenada en un determinado atributo de una entidad. Esto se logra mediante el uso de anotaciones que permiten indicar al proveedor JPA los elementos a tener en cuenta para un determinado atributo.

Una de las anotaciones básicas para la manipulación de un atributo es la anotación @Column. En primera instancia la anotación @Column nos permite especificar el nombre que se le dará al atributo a nivel de base de datos. Ejemplo:

```
@Column(name=”mi_campo”)
private int miCampo;
```

En este caso, name nos permite indicar al proveedor JPA que el nombre con el cual será conocido el atributo miCampo a nivel de base de datos es mi_campo. De igual forma se puede especificar si el atributo podrá o no ser nulo, si debe o no ser único, si se puede o no insertar, actualzar, la longitud o tamaño del campo, la precisión (número de decimales) y la escala o número de dígitos. Todo esto a través del uso de atributos en la notación @Column así:

- unique: Indica si el campo es o no único.

```
@Column(name=”mi_campo”,unique=true)
```

- nullable: Permite definir si el campo puede o no ser nulo. Es decir, si se debe llenar obligatoriamente o no.

```
@Column(name=”mi_campo”,nullable=false)
```

- length: Permite definir la longitud máxima del campo. Es usado en los campos de tipo string.

```
@Column(name=”mi_campo”, length=29)
```

- precision: Determina el número de dígitos de la parte entera que puede tener un campo de tipo BigDecimal.

@Column(name="mi_campo", precision=8)

- scale: Determina el número de dígitos de la parte decimal que puede tener un campo de tipo BigDecimal.

@Column(name="mi_campo", precision=8, scale=2)

- insertable: Determina si el campo podrá o no ser insertado.

@Column(name="mi_campo",insertable=true)

- updatable: Indica si el campo puede o no ser actualizado.

@Column(name="mi_campo",updatable=true)

Se debe tener en cuenta que elementos tales como la longitud no debe ser usada en atributos con tipo de dato complejo. De igual forma no tendría sentido usar el atributo precision en un campo de tipo String. El uso inadecuado de este tipo de elementos en un campo que no lo soporta puede causar errores en la aplicación.

Además de esta anotación, también podemos hacer uso de otras que permiten de forma fácil establecer restricciones, sin embargo, estas otras restricciones trabajan a nivel lógico no físico como si lo hace la anotación @Column. Entre las más comunes tenemos:

@NotNull: Indica si un campo puede o no ser nulo y el mensaje de error en caso de que el campo no cumpla con la restricción.
@Size.max: Permite indicar el número máximo de caracteres de un String.
@Min, @Max: Permite indicar el máximo o mínimo valor a ser asignado a un campo numérico.
@Digits: Permite indicar el la precisión y la escala de un campo numérico.

Precauciones y Recomendaciones: Recuerde verificar que el servidor de aplicaciones soporte el motor de base de datos que usará, de igual forma debe verificar que eclipse este haciendo uso del JDK y no del JRE y recuerde adicionar al workspace el servidor de aplicaciones Glassfish antes de crear cualquier proyecto. También puede ser importante verificar que los puestos usados por Glassfish no estén ocupados (Para ello puede hacer uso del comando ***netstat -npl*** o ***netstat -a***).

Artefactos: Se requiere tener instalado el JDK y un IDE para el desarrollo de aplicaciones (Eclipse Luna), un servidor de aplicaciones que cumpla con las especificaciones de JEE, para esta práctica Glassfish y el motor de base de datos Mysql.

Evaluación o Resultado: Se espera que el alumno logre crear entidades estableciendo adecuadamente las restricciones y configuraciones a cada uno de los atributos de la entidad.

Procedimiento

1. Para empezar será necesario crear una base de datos en mysql. Si ya ha creado una obvie este paso.

2. Adicione al workspace el servidor de aplicaciones GlassFish.

3. En eclipse cree un Proyecto de tipo Maven Proyect con el arquetipo proyecto-archetype.

4. Configure la conexión a su base de datos. Si ya posee una puede omitir este paso.

5. Para cada uno de los atributos de las entidades creadas en la guía 3 aplique las validaciones y parametrizaciones que considere pertinentes mediante el uso de la anotación @Column

6. Corra su aplicación en el servidor de aplicaciones y verifique los resultados en su base de datos.

7. Cree un segundo conjunto de entidades equivalentes a las de la guía 3, pero en esta ocasión parametricelo y validelo sin hacer uso de la anotación @Column

8. Corra su aplicación en el servidor de aplicaciones y verifique los resultados en su base de datos.

9. Corra su aplicación en el servidor de aplicaciones y verifique los resultados en su base de datos.

10. Como podrá observar algunos aspectos de configuración requieren explícitamente del uso de @Column. Complete las entidades con la anotación @Column

11. Corra su aplicación en el servidor de aplicaciones y verifique los resultados en su base de datos.

12. Tome su documento de identidad y cree una entidad que represente todos y cada uno de los atributos que en el encuentre. Adicione las restricciones y parametrizaciones que considere necesarias.