Fetch Type

DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Fetch Type

Es requerido especificar un Fetch Type al usar cualquiera de las asociaciones (relaciones) ya que define cuando se recupera la información de la BD y se carga en la memoria.

Cuando Hibernate accede a una entidad que se relaciona con otra mediante asociaciones, ¿cuándo recupera la información?

Este comportamiento puede configurarse utilizando estos tipos de FetchType:

- EAGER.
- LAZY.

```java
@Entity
public class Address {
@Id
@GeneratedValue(strategy = GenerationType.AUTO)
private int id;
private String street;
private int houseNumber;
private String city;
private int zipCode;
@ManyToOne(fetch = FetchType.LAZY)
private Person person;
}
```

Eager

Es un patrón de diseño en el cual la **inicialización de los datos ocurre en el momento.**

Eager Loading - Carga ansiosa

En Eager Loading o carga ansiosa, la carga en memoria de registros y sus registros asociados mapeados a través de sus entidades se realizan en una consulta.

En el siguiente ejemplo podemos ver la carga ansiosa configurada en las clases Cart e Items:

```
@OnetoMany(mappedBy = “cart” ,cascade = CascadeType.ALL,fetch =
FetchType.EAGER)
private Set <Item> items;
```

Resultados

En el ejemplo de Cart e Items, cuando hacemos una búsqueda por id y está configurado con carga ansiosa, podemos ver en consola la siguiente salida:

```
{}
Hibernate: select cart0_.id as id1_1_0_, items1_.cart_id as
cart_id4_4_1_, items1_.id as id1_4_1_, items1_.id as id1_4_2_,
items1_.cantidad as cantidad2_4_2_, items1_.cart_id as
cart_id4_4_2_, items1_.descripcion as descripc3_4_2_ from cart
cart0_ left outer join items items1_ on
cart0_.id=items1_.cart_id where cart0_.id=?
```

Lazy

Es un patrón de diseño que se utiliza para definir la **inicialización de un objeto tanto como sea posible.**

Lazy Loading - Carga perezosa

En Lazy Loading o carga perezosa, la carga en memoria de registros y sus registros asociados mapeados a través de sus entidades no se inicializará y cargará hasta una llamada explícita al método get.

Podemos ver en la siguiente imagen la carga perezosa configurada en las clases Cart e Items:

```
@OnetoMany(mappedBy = “cart” ,cascade = CascadeType.ALL,fetch =
FetchType.EAGER)
private Set <Item> items;
```

Resultados

En el ejemplo de Cart e Items, cuando hacemos una búsqueda por id y está configurado con carga perezosa, podemos ver en consola la siguiente salida:

```
{}
```

```
Hibernate: select cart0_.id as id1_1_0_ from cart cart0_ where
cart0_.id=?
Hibernate: select items0_.cart_id as cart_id4_4_0_, items0_.id
as id1_4_0_, items0_.id as id1_4_1_, items0_.cantidad as
cantidad2_4_1_, items0_.cart_id as cart_id4_4_1_,
items0_.descripcion as descripc3_4_1_ from items items0_ where
items0_.cart_id=?
```

Eager vs Lazy

Eager Loading

El uso de Eager Loading en escenarios donde hay carga de información innecesaria puede generar problemas de rendimiento.

Lazy Loading

Lazy Loading proporciona tiempos más pequeños de carga y menor consumo de memoria con respecto a Eager Loading. Lazy Loading es el fetch por defecto en @OneToMany y @ManyToMany

DigitalHouse>