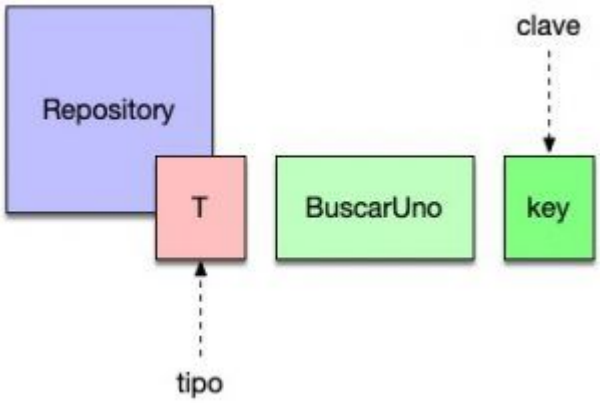
El concepto de Java Optional Repository llega en Java 8 y hace referencia a cómo podemos usar los tipos Optional dentro de nuestras clases clásicas de repositorio. El ejemplo mas clásico es el método BuscarUno o findOne() a nivel de Java Persistence API . Este método nos permite buscar un elemento concreto por clave primaria utilizando el repositorio.



En muchas ocasiones el programa puede dar problemas en el caso de que no se encuentre ningún elemento ya que cuando intentamos acceder a sus campos saltará una excepción de tipo NullPointer.

```
&amp;lt;br&amp;gt;&lt;br&gt;
package repositorios.ejemplo1;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import
javax.persistence.EntityManager;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import model.Cliente;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;public class ClienteRepository
{&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;       private EntityManager
entityManager;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;       public EntityManager getEntityManager()
{&amp;lt;br&amp;gt;&lt;br&gt;
                return entityManager;&amp;lt;br&amp;gt;&lt;br&gt;
```

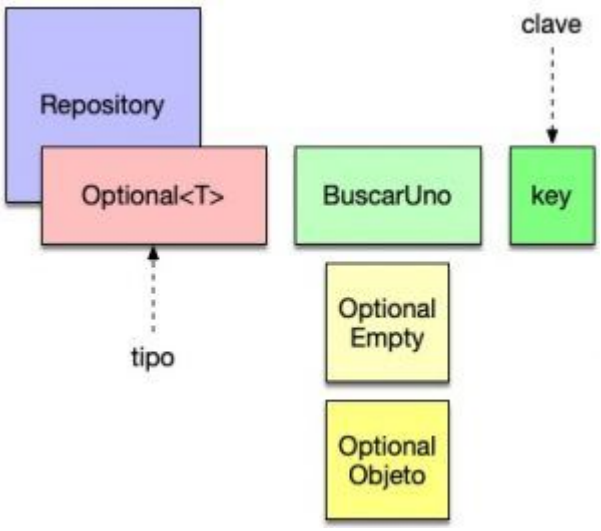
```
        }&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;       public void setEntityManager(EntityManager
entityManager) {&amp;lt;br&amp;gt;&lt;br&gt;
                this.entityManager =
entityManager;&amp;lt;br&amp;gt;&lt;br&gt;
        }&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;       public Cliente buscarUno(String dni)
{&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;               return
entityManager.find(Cliente.class,
&quot;1&quot;);&amp;lt;br&amp;gt;&lt;br&gt;
        }&amp;lt;br&amp;gt;&lt;br&gt;
}&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;


&amp;lt;br&amp;gt;&lt;br&gt;
package repositorios;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import
javax.persistence.EntityManager;&amp;lt;br&amp;gt;&lt;br&gt;
import
javax.persistence.EntityManagerFactory;&amp;lt;br&amp;gt;&lt;br&gt;
import javax.persistence.Persistence;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import model.Cliente;&amp;lt;br&amp;gt;&lt;br&gt;
import
repositorios.ejemplo1.ClienteRepository;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;public class Principal {&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;       public static void main(String[] args)
{&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;               EntityManagerFactory emf =
Persistence.createEntityManagerFactory(&quot;jpa001&quot;);&amp;lt;br&
amp;gt;&lt;br&gt;
```

```
                EntityManager em =
emf.createEntityManager();&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;                ClienteRepository repositorio = new
ClienteRepository();&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;                Cliente
cliente=repositorio.buscarUno(&quot;1&quot;);&amp;lt;br&amp;gt;&lt;br&
gt;
System.out.println(cliente.getNombre());&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;        }&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;}&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;
```

Java Optional Repository

Para evitar estos problemas podemos apoyarnos en Java 8 y hacer uso de los tipos Optional que nos obligan a comprobar si un tipo esta a null antes de acceder a la información que almacena.



Veámoslos en código:

```
&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;package
repositorios.ejemplo2;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import
java.util.Optional;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import
javax.persistence.EntityManager;&amp;lt;br&amp;gt;&lt;br&gt;
import
javax.persistence.EntityManagerFactory;&amp;lt;br&amp;gt;&lt;br&gt;
import javax.persistence.Persistence;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import model.Cliente;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;public class ClienteRepository
{&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;	private EntityManager
entityManager;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;	public EntityManager getEntityManager()
{&amp;lt;br&amp;gt;&lt;br&gt;
		return entityManager;&amp;lt;br&amp;gt;&lt;br&gt;
	}&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;	public void setEntityManager(EntityManager
entityManager) {&amp;lt;br&amp;gt;&lt;br&gt;
		this.entityManager =
entityManager;&amp;lt;br&amp;gt;&lt;br&gt;
	}&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;	public ClienteRepository(EntityManager
entityManager) {&amp;lt;br&amp;gt;&lt;br&gt;
		super();&amp;lt;br&amp;gt;&lt;br&gt;
		this.entityManager =
entityManager;&amp;lt;br&amp;gt;&lt;br&gt;
	}&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;	public Optional&amp;amp;lt;Cliente&amp;amp;gt;
```

```
buscarUno(String dni) {&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;               Cliente cliente =
entityManager.find(Cliente.class, dni);&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;               if (cliente == null)
{&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;                       return
Optional.empty();&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;               } else {&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;                       return
Optional.of(cliente);&amp;lt;br&amp;gt;&lt;br&gt;
                }&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;       }&amp;lt;br&amp;gt;&lt;br&gt;
}&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;
```

Ahora la clase de repositorio hace uso de JPA pero apoyandose en tipos Optional de Java 8 de tal forma que cuando queramos acceder desde un cliente nos obligará a comprobar si existe un valor antes de imprimirlo por la pantalla.

```
&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;package repositorios;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import
java.util.Optional;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import
javax.persistence.EntityManager;&amp;lt;br&amp;gt;&lt;br&gt;
import
javax.persistence.EntityManagerFactory;&amp;lt;br&amp;gt;&lt;br&gt;
import javax.persistence.Persistence;&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;import model.Cliente;&amp;lt;br&amp;gt;&lt;br&gt;
import
repositorios.ejemplo2.ClienteRepository;&amp;lt;/p&amp;gt;&lt;br&gt;
```

```
&amp;lt;p&amp;gt;public class Principal2 {&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;       public static void main(String[] args)
{&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;               EntityManagerFactory emf =
Persistence.createEntityManagerFactory(&quot;jpa001&quot;);&amp;lt;br&
amp;gt;&lt;br&gt;
                EntityManager em =
emf.createEntityManager();&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;               ClienteRepository repositorio = new
ClienteRepository(em);&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;
Optional&amp;amp;lt;Cliente&amp;amp;gt;
cliente=repositorio.buscarUno(&quot;1&quot;);&amp;lt;/p&amp;gt;&lt;br&
gt;
&amp;lt;p&amp;gt;               if (cliente.isPresent())
{&amp;lt;br&amp;gt;&lt;br&gt;
System.out.println(cliente.get().getNombre());&amp;lt;br&amp;gt;&lt;br
&gt;
                }&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;       }&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;}&amp;lt;/p&amp;gt;&lt;br&gt;
&amp;lt;p&amp;gt;
```

Aprendamos a usar Java Optional Repository como un refactor importante a la hora de construir nuestras clases clásicas de repositorio.

1. Java Optional Stream y reference methods
2. Java Optional ifPresent y como utilizarlo
3. Java 8 Optional y NullPointerExceptions
4. Java 8