Herramientas de trabajo (5)

Miguel Angel Piña Avelino

Ingeniería de Software,
Facultad de Ciencias, UNAM

24 de septiembre de 2018

Índice

Java Persistence API

Persistence y Entity Manager

- EntityManager será la clase que nos permitirá hacer transacciones con la base de datos, es decir, guardar entidades en base de datos, modificarlas, consultarlas, etc.
- Tiene métodos, entre otros muchos, `persist()` para guardar una de nuestras entidades en base de datos, `createQuery()` para crear una consulta y obtener datos de la base de datos y `getTransaction()`, para obtener una "transaction".

Transacciones

A veces queremos hacer varias operaciones en base de datos y queremos que o bien todas ellas vayan bien, o bien que ninguna se haga. La "transaction.es una forma de conseguir esto. Se obtiene una "transactionz se llama a begin().

```
EntityManager manager = ...;
EntityTransaction tx = manager.getTransaction();
tx.begin();
// varias operaciones en base de datos.
...
// si todas van bien
tx.commit();
// si alguna va mal, en cuanto detectemos el fallo
tx.rollback();
```

Transacciones

Una vez llamado begin(), comenzamos a hacer todas las operaciones que queramos, que no se guardarán en base de datos, sino que quedarán anotadas de alguna forma. Si todas ellas van bien, llamamos al método commit(), con lo que todo lo que hemos hecho se guardará definitivamente en base de datos. Si alguna operación nos falla, en vez de deshacer todo lo que hayamos hecho hasta el momento, bastará con llamar a rollback(), y todo se deshará automáticamente.

EntityManager

¿Cómo obtenemos el EntityManager?. Si estamos usando un contenedor de aplicaciones completo, como JBoss o Glassfish, hay forma de obenerlo más o menos automáticamente (el contenedor nos lo pasará). Pero si usamos un contenedor no tan completo (como Apache Tomcat) o estamos en java de escritorio (no hay contenedor ninguno), debemos crear nosotros nuestro propio `EntityManager`. ¿Cómo lo hacemos?. Usando la clase `Persistence`

EntityManager

```
EntityManagerFactory factory =
Persistence.createEntityManagerFactory("eclipseLinkPersistenceUnit");
EntityManager manager = factory.createEntityManager();
```

Inserción

```
private void createEmployees() {
    System.out.println("CREATE EMPLOYEES :");
    // manager es el EntityManager obtenido anteriorment
    EntityTransaction tx = manager.getTransaction();
    tx.begin();

    try {
        manager.persist(new Employee("Jakab Gipsz"));
        manager.persist(new Employee("Captain Nemo"));
        tx.commit();
    } catch (Exception e) {
        e.printStackTrace();
        tx.rollback();
    }
}
```

Consulta

```
private void listEmployees() {
  System.out.println("Employees list :");

  // Nuevamente, manager es el EntityManager obtenido anteriormente.
  List<Employee> resultList = manager.createQuery(
  "Select a From Employee a", Employee.class).getResultList();
  System.out.println("num of employess:" + resultList.size());
  for (Employee next : resultList) {
    System.out.println("next employee: " + next);
  }
}
```

Consulta

```
private Employee queryForEmployee(String name) {
  System.out.println("QUERY FOR "+name);
  Query query = manager
  .createQuery("Select a from Employee a where a.name = :name");
  query.setParameter("name", name);
  Employee anEmployee = (Employee) query.getSingleResult();
  System.out.println("Result : " + anEmployee.toString());
  return anEmployee;
}
```

Modificación

```
EntityTransaction tx = manager.getTransaction();
tx.begin();

try {
  Employee anEmployee = manager.find(Employee.class, id);
  anEmployee.setName(newName);
  manager.persist(anEmployee);
  tx.commit();
} catch (Exception e) {
  e.printStackTrace();
  tx.rollback();
}
```

Borrado

```
EntityTransaction tx = manager.getTransaction();
tx.begin();

try {
  Employee anEmployee = manager.find(Employee.class, id);
  manager.remove(anEmployee);
  tx.commit();
} catch (Exception e){
  e.printStackTrace();
  tx.rollback();
}
```