JPA – Relacionamento Many to Many (Exercício)

Neste roteiro desenvolveremos uma aplicação baseada em JPA, usando o relacionamento **many to many**. O objetivo é gerar a seguinte estrutura:

1. Criar um projeto **JPA Project** chamado **Exercicio_ManytoMany**.

2. Criar as entidades definidas abaixo.

```
package br.com.fiap.entity;

import java.io.Serializable;
import java.util.HashSet;
import java.util.Set;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.persistence.UniqueConstraint;

@Entity
@Table(name="FUNCIONARIO", catalog="dbtarefas", uniqueConstraints =
	{
		@UniqueConstraint(columnNames="CODIGO_FUNCIONARIO")
	})
@NamedQuery(name="Funcionario.findAll", query="select f from Funcionario f")
public class Funcionario implements Serializable {

	private static final long serialVersionUID = 1L;

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Column(name="ID", unique=true, nullable=false)
	private Integer id;
```

```
	@Column(name="CODIGO_FUNCIONARIO", unique=true, nullable=false, length=10)
	private String matricula;

	@Column(name="NOME_FUNCIONARIO", unique=true, nullable=false, length=45)
	private String nome;

	@ManyToMany(fetch=FetchType.LAZY, cascade= CascadeType.ALL)
	@JoinTable(name="FUNCIONARIO_TAREFA", catalog="dbtarefas", joinColumns =
	{@JoinColumn(name="FUNCIONARIO_ID", nullable=false, updatable=false)},
	inverseJoinColumns = {@JoinColumn(name="TAREFA_ID", nullable=false,
updatable=false)})
	private Set<Tarefa> tarefas = new HashSet<>();

	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}
	public String getMatricula() {
		return matricula;
	}
	public void setMatricula(String matricula) {
		this.matricula = matricula;
	}
	public String getNome() {
		return nome;
	}
	public void setNome(String nome) {
		this.nome = nome;
	}
	public Set<Tarefa> getTarefas() {
		return tarefas;
	}
	public void setTarefas(Set<Tarefa> tarefas) {
		this.tarefas = tarefas;
	}

}


package br.com.fiap.entity;

import java.io.Serializable;
import java.util.HashSet;
import java.util.Set;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.ManyToMany;
import javax.persistence.Table;
```

```
@Entity
@Table(name="TAREFA", catalog="dbtarefas")
public class Tarefa implements Serializable{

	private static final long serialVersionUID = 1L;

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Column(name="ID", unique=true, nullable=false)
	private Integer id;

	@Column(name = "DESCRICAO", nullable = false, length=45)
	private String descricao;

	@Column(name = "DURACAO", nullable = false)
	private Integer duracao;


	@ManyToMany(fetch=FetchType.LAZY, mappedBy="tarefas")
	private Set<Funcionario> funcionarios = new HashSet<>();

	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}
	public String getDescricao() {
		return descricao;
	}
	public void setDescricao(String descricao) {
		this.descricao = descricao;
	}
	public Integer getDuracao() {
		return duracao;
	}
	public void setDuracao(Integer duracao) {
		this.duracao = duracao;
	}
	public Set<Funcionario> getFuncionarios() {
		return funcionarios;
	}
	public void setFuncionarios(Set<Funcionario> funcionarios) {
		this.funcionarios = funcionarios;
	}

}
```

3. Incluir as classes (entidades) no arquivo **persistence.xml** e mantê-lo como no modelo abaixo:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
<persistence-unit name="jpaPU" transaction-type="RESOURCE_LOCAL">
	<class>br.com.fiap.entity.Funcionario</class>
	<class>br.com.fiap.entity.Tarefa</class>
	<properties>
		<property name="javax.persistence.jdbc.driver"
			value="com.mysql.jdbc.Driver"/>
		<property name="javax.persistence.jdbc.url"
			value="jdbc:mysql://localhost:3306/dbtarefas"/>
		<property name="javax.persistence.jdbc.user" value="root"/>
		<property name="javax.persistence.jdbc.password" value="fiap"/>
		<property name="hibernate.hbm2ddl.auto" value="update" />
		<property name="hibernate.format_sql" value="true" />
		<property name="hibernate.dialect"
			value="org.hibernate.dialect.MySQLDialect" />


	</properties>
</persistence-unit>
</persistence>
```

4. Escrever a classe **Helper**, contendo métodos auxiliares para realizar a persistência:

```
package br.com.fiap.helper;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.Query;

import br.com.fiap.entity.Funcionario;

public class Helper {

	private EntityManager em;

	public Helper(EntityManager em){
		this.em = em;
	}

	public void salvar(Funcionario funcionario) throws Exception {
		try {
			em.getTransaction().begin();
			em.persist(funcionario);
			em.getTransaction().commit();
		} catch (Exception e) {
			throw e;
		} finally {
			em.close();
		}
	}
```

```
	//JPQL: Usando Query
	@SuppressWarnings("unchecked")
	public List<Funcionario> listarFuncionarios(){
		Query query = em.createQuery("select f from Funcionario f");
		return query.getResultList();
	}

	@SuppressWarnings("unchecked")
	public Funcionario buscarFuncionario(String numMatricula){
		Query query = em.createQuery("select f from Funcionario f where
matricula = :matricula");
		query.setParameter("matricula", numMatricula);
		Funcionario f = (Funcionario)query.getSingleResult();
		return f;
	}

	//JPQL: usando NamedQuery
	@SuppressWarnings("unchecked")
	public List<Funcionario> listarTodos(){
		Query query = em.createNamedQuery("Funcionario.findAll");
		return query.getResultList();
	}
}
```

5. Para testar a aplicação, utilize os métodos abaixo:

```
package br.com.fiap.programa;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

import br.com.fiap.entity.Tarefa;
import br.com.fiap.entity.Funcionario;
import br.com.fiap.helper.Helper;

public class Teste {
	public static void main(String[] args) {

		EntityManagerFactory  emf =
Persistence.createEntityManagerFactory("jpaPU");
		EntityManager em = emf.createEntityManager();

		//incluirFuncionario(em);
		//listarFuncionarios(em);
		//buscarFuncionario(em, "2000");

	}

	private static void incluirFuncionario(EntityManager em){
		Helper dao = new Helper(em);
```

```
		Funcionario funcionario = new Funcionario();
		funcionario.setMatricula("2000");
		funcionario.setNome("Alberto Santos");

		Tarefa tarefa = new Tarefa();
		tarefa.setDescricao("Teste Unitário");
		tarefa.setDuracao(100);
		tarefa.getFuncionarios().add(funcionario);

		funcionario.getTarefas().add(tarefa);

		try {
			dao.salvar(funcionario);
			System.out.println("Funcionario OK");
		} catch (Exception e) {
			System.out.println("ERRO ===>> " + e.getMessage());
		}
	}
	private static void listarFuncionarios(EntityManager em){
		Helper dao = new Helper(em);
		List<Funcionario> funcionarios = dao.listarTodos();
		for (Funcionario funcionario : funcionarios) {
			System.out.println(funcionario.getMatricula() + ": " +
funcionario.getNome());
		}
		em.close();
	}

	private static void buscarFuncionario(EntityManager em, String matricula){
		Helper dao = new Helper(em);
		Funcionario f = dao.buscarFuncionario(matricula);
		System.out.println(f.getMatricula() + ": " + f.getNome());
	}
}
```

Bom trabalho a todos!

Prof. Rafael Matsuyama