Programação Avançada - AULA 13

Matheus Moresco
Engenharia de Software - 5º Período
2025/01

Introdução

- Introdução ao JPA.
  - O que é?
  - Como usar?
  - Provedor hibernate.
  - Vantagens do JPA.

O que é JPA?

- O JPA é uma especificação da linguagem Java para o mapeamento objeto-relacional (ORM – Object-Relational Mapping).
- Ele permite que você armazene, recupere, atualize e delete objetos Java diretamente em um banco de dados relacional, sem precisar escrever SQL manualmente.

Objetivos do JPA

Usando o JPA para persistência de dados podemos:

- Reduzir o uso direto de SQL
- Tornar o código mais limpo e orientado a objetos
- Facilitar o desenvolvimento de sistemas que usam banco de dados

Por que usar JPA?

- Facilita a leitura e escrita no banco de dados
- Elimina grande parte do código SQL manual
- Permite foco na lógica da aplicação
- É uma API padronizada
- Suporte a vários bancos: MySQL, PostgreSQL, H2, Oracle...

Como funciona o JPA?

- Você define **entidades** (classes que representam tabelas)
- Usa **anotações** para mapear os campos da classe
- JPA cuida de:
  - Conexão com banco
  - Inserções, atualizações, exclusões
  - Consultas via JPQL ou Criteria API

Provedores JPA

O JPA é apenas uma especificação. Para usá-lo, é preciso de um provedor que implemente essa especificação.

Exemplos de provedores:

- **Hibernate** (mais popular)
- **EclipseLink**
- **Apache OpenJPA**

Entidade JPA

- Uma **entidade JPA** é uma **classe Java que representa uma tabela no banco de dados**. Cada instância da classe representa uma **linha** dessa tabela.
- Requisitos para ser uma Entidade:
  - A classe deve ser anotada com @Entity
  - Deve ter um identificador único com @Id
  - Deve ter um construtor público sem argumentos
  - Opcionalmente, pode ter @Table, @Column, etc.

```
@Entity
public class Pessoa {
  @Id
  @GeneratedValue
  private Long id;
  private String nome;
}
```

Arquivo persistence.xml

- Com o JPA podemos definir um arquivo de configuração que será responsável por armazenar as informações necessárias para conectar com o banco de dados.

```
<persistence-unit name="meuPU">
  <class>modelo.Pessoa</class>
  <properties>
    <property name="jakarta.persistence.jdbc.url" value="jdbc:h2:mem:meubanco"/>
    <property name="hibernate.dialect" value="org.hibernate.dialect.H2Dialect"/>
    <property name="hibernate.hbm2ddl.auto" value="update"/>
  </properties>
</persistence-unit>
```

Como salvar uma objeto

- Salvar uma entidade com JPA é simples e elegante, você cria o objeto Java e usa o EntityManager para persistir no banco de dados. A JPA cuida do resto!
- Etapas para salvar uma entidade com JPA:
  1. Ter a entidade mapeada
  2. Criar o **EntityManager**
  3. Criar o objeto e salvar com **persist**()
  4. Encerrar recursos

```
import jakarta.persistence.*;

public class Main {
  public static void main(String[] args) {
    EntityManagerFactory emf = Persistence.createEntityManagerFactory("meuPU");
    EntityManager em = emf.createEntityManager();

    Pessoa pessoa = new Pessoa();
    pessoa.setNome("João");

    em.getTransaction().begin();
    em.persist(pessoa);
    em.getTransaction().commit();

    em.close();
    emf.close();
  }
}
```

Vantagens e desvantagens

- Vantagens do JPA
  - Reduz o uso de SQL
  - Portabilidade entre bancos
  - Orientado a objetos
  - Integração com frameworks (Spring, Jakarta EE)

- Limitações e Cuidados
  - Nem tudo deve ser feito com JPA
  - Controle de transações pode exigir atenção
  - Performance precisa de tuning (fetch, cache, etc)

Exemplo prático

- Criar mesmo projeto de exemplo da ultima aula, mas usar o JPA com hibernate para persistência dos dados.