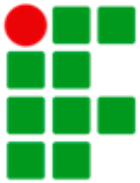

INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Bahia

Linguagem de Programação II

Hibernate (Framework de Persistência);

Roteiro

- Hibernate (Framework de Persistência):
  - Introdução ao Hibernate;
  - Padrões de Mapeamento Objeto-Relacional;
  - Tecnologias de Mapeamento no Hibernate:
    - Anotações JPA;
    - Arquivos de Mapeamento XML;
  - Configuração do Hibernate;
    - Arquivo “persistence.xml”;
    - Unidade de Persistencia (Persistence Unit);
  - Persistência de Dados com o Hibernate;
  - Exemplo Básico de Uso do Hibernate;
  - Criação de um Projeto com CRUD, Hibernate e Java SWING.

Introdução ao Hibernate

O **Hibernate** é um framework de mapeamento objeto-relacional (ORM) para Java, utilizado para facilitar a integração entre modelos de objetos e modelos de dados relacionais. Ele simplifica a persistência de dados em aplicativos Java, permitindo que os desenvolvedores trabalhem com bancos de dados de forma orientada a objetos.

Padrões de Mapeamento Objeto-Relacional

O **Hibernate** implementa padrões de mapeamento objeto-relacional para facilitar a integração entre modelos de objetos e modelos de dados relacionais. **Esses padrões incluem**:

Padrões de Mapeamento Objeto-Relacional

Entidades e Relacionamentos:

- O **Hibernate** permite mapear classes Java para tabelas de banco de dados e propriedades de classe para colunas de tabela. Além disso, suporta vários tipos de relacionamentos entre entidades, como **um para um**, **um para muitos** e **muitos para muitos**.

Padrões de Mapeamento Objeto-Relacional

Anotações de Mapeamento:

- O **mapeamento** pode ser definido diretamente nas classes de entidade usando anotações, o que simplifica a configuração e torna o código mais legível.

Tecnologias de Mapeamento no Hibernate

O **Hibernate** oferece várias tecnologias de mapeamento para atender às diferentes necessidades de desenvolvimento de aplicativos:

Tecnologias de Mapeamento no Hibernate

Anotações JPA:

O **Hibernate** é compatível com as anotações definidas na **especificação JPA** (Java Persistence API), como **@Entity**, **@Table**, **@Id**, **@GeneratedValue**, **@Column**, entre outras.

Tecnologias de Mapeamento no Hibernate

Arquivos de Mapeamento XML:

- Além das anotações, o **Hibernate** suporta o uso de arquivos de mapeamento XML (.hbm.xml) para configurar o mapeamento objeto-relacional. Isso proporciona uma maior flexibilidade em cenários complexos.

Configuração do Hibernate

A configuração do **Hibernate** envolve a definição das propriedades de conexão com o banco de dados e outras configurações relevantes. As etapas típicas de configuração incluem:

Configuração do Hibernate

Arquivo “persistence.xml”:

O **persistence.xml** é usado para configurar as unidades de persistência (Persistence Units) em aplicações JPA. Nele, definimos a unidade de persistência e configuramos propriedades como a conexão com o banco de dados.

Configuração do Hibernate

Unidade de Persistência (Persistence Unit):

- A **unidade de persistência** é um conjunto de configurações que define como as entidades serão gerenciadas pelo **Hibernate**. Nela, especificamos o nome da unidade de persistência, as classes de entidade e as propriedades de conexão.

Persistência de Dados com o Hibernate

Com o **Hibernate** configurado e as entidades devidamente mapeadas, podemos realizar operações de persistência de dados, como **criar**, **ler**, **atualizar** e **excluir** registros no banco de dados. Isso é feito por meio da **API** do **Hibernate**, que inclui classes como **EntityManagerFactory**, **EntityManager**, **EntityTransaction**, entre outras.

Persistência de Dados com o Hibernate

EntityManagerFactory e **EntityManager:**

O **EntityManagerFactory** é responsável por criar e gerenciar instâncias de **EntityManager**. O **EntityManager** é uma interface de contexto de persistência que fornece métodos para realizar operações de CRUD no banco de dados.

Persistência de Dados com o Hibernate

EntityTransaction:

A **EntityTransaction** é usada para gerenciar transações de banco de dados. Ela oferece métodos para iniciar, confirmar e reverter transações, garantindo atomicidade, consistência, isolamento e durabilidade (ACID).

Exemplo Básico de Uso do Hibernate

```
EntityManager gerente = Persistence.createEntityManagerFactory("hibernate")
        .createEntityManager();

try {
    Conta c = new Conta(numero, saldo, limite);

    gerente.getTransaction().begin();
    gerente.persist(c);
    gerente.getTransaction().commit();

    JOptionPane.showMessageDialog(this, "Cadastro realizado com sucesso!");

    LimparCampos();
} catch (Exception e) {
    JOptionPane.showMessageDialog(this, "Erro ao tentar cadastrar uma nova conta!");
} finally {
    if (gerente != null) {
        gerente.getEntityManagerFactory().close();
        gerente.close();
    }
}
```

```
EntityManager gerente = Persistence.createEntityManagerFactory("hibernate")
        .createEntityManager();

try {
    Conta c = new Conta(numero, saldo, limite);

    gerente.getTransaction().begin();
    gerente.persist(c);
    gerente.getTransaction().commit();

    JOptionPane.showMessageDialog(this, "Cadastro realizado com sucesso!");

    LimparCampos();
} catch (Exception e) {
    JOptionPane.showMessageDialog(this, "Erro ao tentar cadastrar uma nova conta!");
} finally {
    if (gerente != null) {
        gerente.getEntityManagerFactory().close();
        gerente.close();
    }
}
```

Exemplo Básico de Uso do Hibernate

Persistence.createEntityManagerFactory(“hibernate") cria um **EntityManagerFactory** usando a unidade de persistência chamada “**hibernate**" definida no arquivo **persistence.xml**.

createEntityManager() cria um **EntityManager**, que é usado para realizar operações de persistência, como persistir objetos no banco de dados.

Criação de um Projeto com CRUD, Hibernate e Java SWING

- Ver projeto **ProjetoBancoABC_**Hibernate no repositório.

Referências

- TUDOSE, Catalin; BAUER, Christian; KING, Gavin. **Persistence with Spring Data and Hibernate**. Manning Publications, 2023.

Obrigado!

- Canais de Comunicação;
- Horário de Atendimento.