07

Criando um projeto com JPA

Transcrição

Agora que já conhecemos um pouco da história da JPA, chegou a hora de colocarmos a "mão na massa", de começarmos a trabalhar, na prática, com a JPA, utilizando o Hibernate como implementação.

Utilizaremos o Eclipse como IDE (mas você pode usar a IDE de sua preferência). A ideia é criarmos um projeto com Maven para facilitar as dependências do Hibernate e as outras que decidirmos utilizar durante o treinamento.

Vamos começar selecionando a opção "Create a Maven Project" que está na aba lateral esquerda. Na próxima tela, marcaremos "Create a simple project (skip archetype selection) para que ele “pule” o *archetype,* e selecionaremos "Next". Em seguida, preencheremos: "Group Id" com "br.com.alura"; "Artifact Id" com "loja", isto é, nós criaremos uma aplicação como uma loja com cadastro de produto e outras coisas. O resto, podemos deixar com a opção padrão, e agora basta apertar "Finish".

Ele criará um projeto utilizando Maven, com uma estrutura de Diretórios e o arquivo `pom.xml`, onde configuraremos as dependências. O foco do nosso treinamento não é aprender sobre Maven (na Alura existe um treinamento de Maven). Basicamente, lidaremos com o `pom.xml`, encontraremos as configurações do projeto que preenchemos na tela de criação.

```
<modelVersion>4.0.0</modelVersion>
<groupId>br.com.alura</groupId>
```

```
<artifactId>loja</artifactId>
<version>0.0.1-SNAPSHOT</version>
```

COPIAR CÓDIGO

Precisaremos fazer duas configurações. Por padrão, no Eclipse, se não indicarmos ao Maven qual a versão do Java, ele considerará que é o Java 5. Então, colaremos a Tag `build` e um `plugin` do Maven para dizer: quero utilizar o Java na versão 11. Nós disponibilizaremos esse trecho de código para que não seja necessário digitar tudo manualmente.

```
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.8.0</version>
            <configuration>
                <release>11</release>
            </configuration>
        </plugin>
    </plugins>
</build>
```

COPIAR CÓDIGO

Então, se trata de um `plugin` para dizer ao Maven que queremos utilizar o Java 11. O que nos interessa é, logo embaixo da tag `build` , a parte de dependências do Maven. Portanto, abriremos a tag `<dependencies>` e adicionaremos uma dependência, `<dependency>` . No nosso caso, queremos utilizar a JPA, que é a especificação, com o Hibernate como implementação.

Basta adicionar apenas uma dependência do Hibernate e, automaticamente, ele adicionará outras dependências da JPA e de todas as outras que o Hibernate precisa. Continuando, precisamos colocar qual é o `groupId` , no caso

`org.hibernate`. E o `artifactId` que adicionaremos será o `hibernate-entitymanager`.

```
<dependencies>
    <dependency>
        <groupId>org.hibernate<groupId>
        <artifactId>hibernate-entitymanager</artifactId>
    </dependency>
</dependencies>
```

COPIAR CÓDIGO

Precisamos passar também qual é a versão da dependência - qual a versão do Hibernate que vamos utilizar. No momento de gravação deste curso, início de 2021, a última versão que estava disponível era a `5.4.27.Final` e é a que nós utilizaremos.

```
<dependencies>
    <dependency>
        <groupId>org.hibernate<groupId>
        <artifactId>hibernate-entitymanager</artifactId>
        <version>5.4.27.Final</version>
    </dependency>
</dependencies>
```

COPIAR CÓDIGO

Se olharmos o projeto, no "Maven Dependencies", perceberemos que ele já baixou uma série de dependências e adicionou ao nosso projeto. Dentre elas, temos o "hibernate-entitymanager", que foi o que acabamos de adicionar, e que, por sua vez, depende de todas as outras dependências, como o "hibernate-core", "hibernate-commons-anotations" e o "javax.persistence-api-2.2.jar", que é a JPA em si. Portanto, aí está a especificação.

De forma bem simples, adicionamos o Hibernate e a JPA como dependência uma aplicação que está utilizando o Maven. Além do Hibernate, precisaremos

de mais uma dependência, que é a dependência do driver do banco de dados que utilizaremos.

No nosso caso, vamos usar o H2, que é um banco de dados em memória, só para não perdermos tempo com instalação e configuração de banco de dados. Mas, é possível usar outros bancos de dados, por exemplo, o MySQL ou o Postgres. Então, vamos seguir colocando mais uma dependência `<groupId>com.h2database</groupId>`. E o `<artifactId>` é o `h2`. A versão do `h2` será a 1.4.200.

```
<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <version>1.4.200</version>
</dependency>
```

COPIAR CÓDIGO

Então, são duas dependências que precisamos adicionar ao nosso projeto quando trabalharmos com Hibernate. A dependência do Hibernate em si, que baixará todas aquelas dependências que ele tem, além da própria JPA, que é a especificação, e a dependência do banco de dados que estivermos utilizando. É possível também modificar para usar o MySQL, Postgres.

Com isso, já temos a nossa aplicação Java com Maven, utilizando as dependências do Hibernate e da JPA. Ele só está marcando um erro. Vamos corrigi-lo apertando o botão direito "Maven > Update Project", na próxima tela apertaremos "Ok". Resolvido, foi algum problema ao baixar as dependências.

Este é o nosso projeto Maven com Hibernate já baixado e adicionado como dependência do Maven, para não precisarmos baixar os JARs manualmente. O próximo passo seria configurar a JPA, criar o arquivo `persistence.xml`, criar as entidades, fazer o mapeamento e começar a trabalhar com a JPA. Como essa parte é um pouco mais complicada, deixaremos para a próxima aula.

Vejo vocês lá!! Um abraço!!