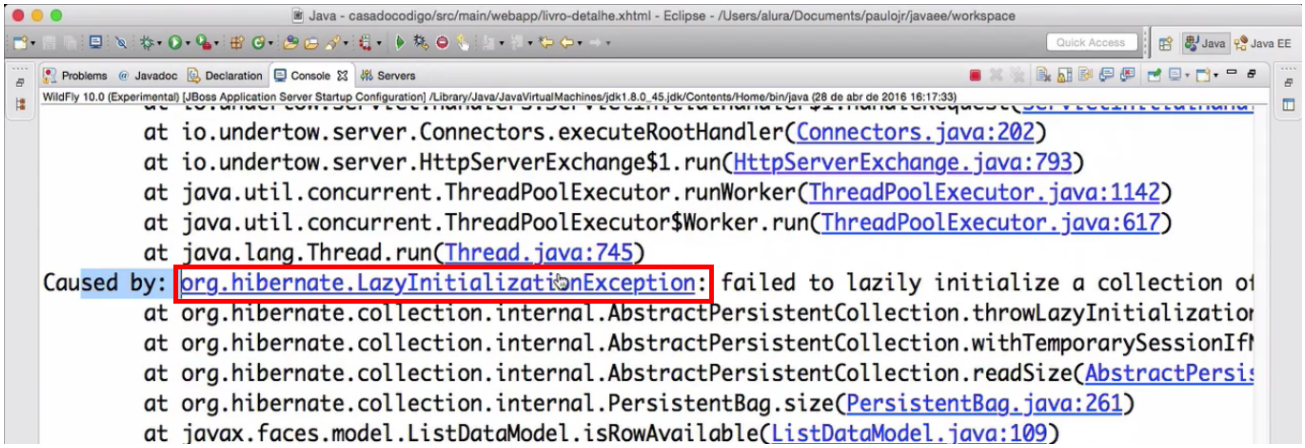
10

Carregando os Autores do Livro

Transcrição

Fizemos o direcionamento do link para a página do `Detalhe` e ele exibiu apenas o título.

Se olharmos o console do Eclipse, percebemos uma exception chamada de `LazyInitializationException` .



Ao realizarmos a busca pelo `Livro` , o **JPA** buscou apenas pelo próprio `Livro` , e não pelos `Autores` relacionados a ele. Só quando precisamos dos `autores` na tela, o **JPA** tentou carregá-los. Porém a conexão com o banco de dados já havia sido fechada. Mas temos várias formas de resolver um `LazyInitializationException` . Começaremos fazendo de uma forma mais elegante, buscando o livro e junto deste, trazer os autores. Faremos isto no `LivroDao` fazendo uso do **JPQL**, que já possui um atributo chamado `join fetch` . O `fetch` tem a capacidade de realizar a busca dos `autores` bem no momento em que buscamos o `livro` . Para isso, vamos alterar o método `buscaPorId()` da classe `LivroDao` para buscar com **JPQL**. Nosso código ficará assim:

```
public Livro buscarPorId(Integer id) {
    String jpql = "select l from Livro l join fetch l.autores "
            + "where l.id = :id";
    return manager.createQuery(jpql, Livro.class)
            .setParameter("id", id)
            .getSingleResult();
}
```

Faça um novo *Full Publish** e perceba que o detalhe do Livro é exibido quase completamente.

Mas ainda não são exibidas algumas informações, como o nome dos autores e as informações do número de páginas. Percebemos que faltavam fazer pequenos ajustes no código.

O explicação do vídeo passado já contêm o código com as correções feitas.

Agora, as informações que faltavam serão exibidas:

Em seguida, queremos mudar a capa. Já sabemos carregar a capa do livro usando o mapeamento `/file/*` que fizemos no *Servlet*, agora, usaremos esse mapeamento para substituir a capa fixa que temos pela capa que enviada pelo banco de dados.

No arquivo `livro-detalhe.xhtml` , procure pela tag `<img class="imagemLivroNinja-principal" ... >` . Nessa tag, temos um atributo chamado `src` que está fixo para uma imagem. Vamos trocar o `src` para que fique com o seguinte valor:

```
#{request.contextPath}/file/#{livroDetalheBean.livro.capaPath}
```

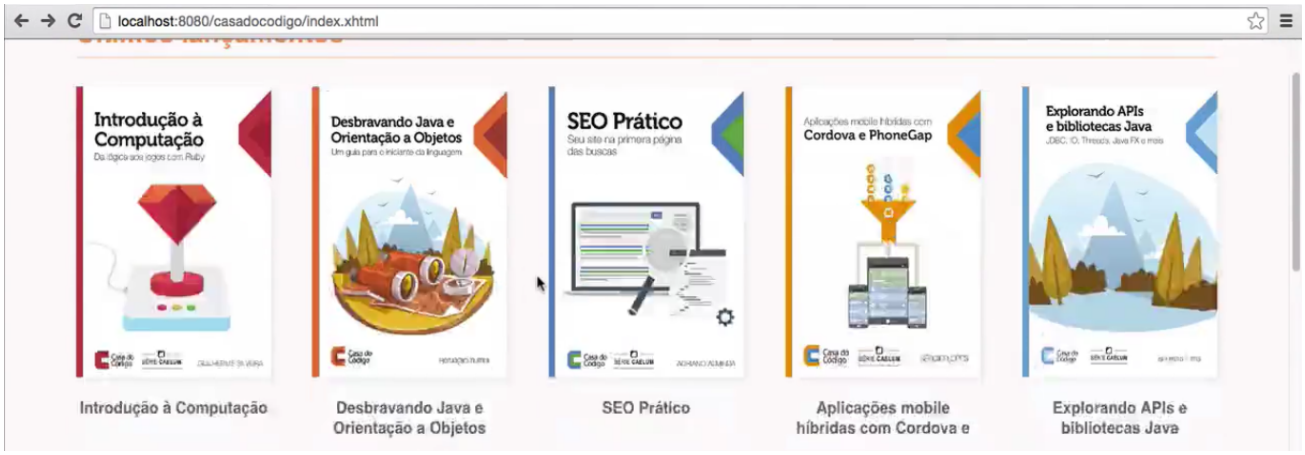
Você pode trocar o `alt` e o `title` que a tag `<img>` pelo título do livro. Essa tarefa fica para você.

Mais abaixo dessa tag, teremos outra importante, chamada `<image>`. Nessa tag temos um atributo diferente, chamado `xlink:href`. Quando olhamos a tela pelo navegador, veremos que uma imagem é diretamente a capa do livro, mas as demais simulam um *tablet* e um *smartphone*. É possível usar esta funcionalidade com o `xlink`. Por isso **tenha cuidado**, para não alterar os demais valores acima ou abaixo da tag. Apenas modifique o `xlink:href` para o mesmo valor da `<img>` anterior:

```
<!-- altere apenas o que está dentro de href -->
<image xlink:href="#{request.contextPath}/file/#{livroDetalheBean.livro.capaPath}" ... >
```

Volte ao navegador e pressione `Ctrl + F5` e veja se a imagem é exibida corretamente. Até aqui, tudo deve estar funcionando corretamente.

Então estamos praticamente pronto com nossa tela de `livro-detalhe.xhtml`, pois já temos a maior parte do que precisamos para o correto funcionamento da nossa tela. E veja como está elegante, exibindo os dados corretos dos livros.



Tente voltar na *Home* e clicar em outro livro, para ver os dados desse livro também corretamente. Só faremos um pequeno ajuste, falta exibir um livro. Isto ocorreu porque a contagem dos resultados começa da posição `0`. No método `demaisLivros()`, do `LivroDao.java`, vamos alterar o valor de `6` para `5`:

```
public List<Livro> demaisLivros() {
        String jpql = "select l from Livro l order by l.id desc";
        return manager.createQuery(jpql, Livro.class)
```

```
                .setFirstResult(5);
                .getResultList();

    }
```

Nós escolhemos solucionar o `LazyInitializationException` fazendo o **carregamento antecipado** e realizando o `join fetch` dos autores. Mas continuaremos analisando este recurso nos livros.