05

Salvando o Path da Capa do Livro

Transcrição

Podemos perceber no entanto que no banco de dados, o livro cadastrado não tem nenhuma referência com o caminho que informamos. Essas informações hoje estão desconectadas. Essa conexão pode ser criada através de um novo atributo na classe `Livro`. Como só queremos salvar o caminho onde o arquivo se encontra no servidor, basta criar um atributo com as seguintes características:

```
public class Livro {

    // Demais atributos acima, não alterar!

    private String capaPath;

    //  Crie também o getter e setter para capaPath

}
```

Agora que já podemos salvar o caminho do arquivo, vamos olhar novamente para o método `salvar` da classe `AdminLivrosBean`. Essa manipulação de arquivo que estamos fazendo pode começar a ficar muito comum no sistema, e ainda podemos considerar que outras partes do sistema também precisarão de upload de arquivos que talvez nem sejam capas de livro ou mesmo imagem.

Por isso, precisamos deixar o código que trata o envio do arquivo para o servidor reutilizável.

Vamos criar uma classe chamada `FileSaver` no pacote `br.com.casadocodigo.loja.infra` que será responsável por escrever esse arquivo no disco. Criaremos o método `write`, recebendo o tipo `Part` que será o arquivo que precisamos escrever no disco, e também uma `String` com o path referente ao negócio que está usando o `FileSaver`. Por exemplo, se estamos tratando de livro, podemos salvar na pasta padrão `/casadocodigo` mas dentro dela o path específico seria `/livros`.

Perceba que a pasta padrão no servidor pode mudar, então vamos deixar ela padronizada em um atributo `static` e `final` dentro do `FileSaver`.

Assim, resumindo nossa nova classe `FileSaver`, teremos o método `write` que receberá o arquivo e o caminho relativo para salvar, e retornará o caminho relativo usado para salvar já concatenado com o nome do arquivo. Ela terá também o caminho *default* do servidor. Ficando, ao final, conforme o código a seguir:

```
public class FileSaver {

    private static final String SERVER_PATH = "/casadocodigo";

    public String write(Part arquivo, String path) {
        String relativePath =  path + "/" + arquivo.getSubmittedFileName();
        try {
            arquivo.write(SERVER_PATH + "/" + relativePath);
```

```
            return relativePath;
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}
```

Feito isso, podemos voltar ao `AdminLivrosBean` e alterar o método `salvar` para usar o `FileSaver`. Ficando ao final, o método `salvar` da seguinte forma:

```
    @Transactional
    public String salvar() throws IOException {
        dao.salvar(livro);
        FileSaver fileSaver = new FileSaver(); // Nossa nova classe
        livro.setCapaPath(fileSaver.write(capaLivro, "livros")); // Já chamamos o método write e

        context.getExternalContext()
            .getFlash().setKeepMessages(true);
        context
            .addMessage(null, new FacesMessage("Livro cadastrado com sucesso!"));

        return "/livros/lista?faces-redirect=true";
    }
```

Novamente, execute um *Full Publish* e veja se tudo o que você fez está funcionando. Cadastre um novo livro com uma foto bacana para podermos exibir depois. Aqui estão algumas fotos de capas de livros para download.

https://s3.amazonaws.com/caelum-online-public/java-ee-webapp/capas.zip (https://s3.amazonaws.com/caelum-online-public/java-ee-webapp/capas.zip)