07

Alterando empresa

Transcrição

Estamos terminando nossa funcionalidade de edição: já mostramos os dados da empresa que estarão sujeitos à alteração pelo usuário e no navegador já exibimos o formulário populado, só resta construirmos o Serlvet `AlteraEmpresa` onde os dados serão alterados.

Para isso, usaremos o método POST que enviará as informações da empresa e atualizaremos o modelo. Por fim, faremos o redirecionamento e os dados serão novamente direcionados.

detalhes do fluxo (https://s3.amazonaws.com/caelum-online-public/986-servlets-parte1/07/7_3_1_detalhes+de+fluxo.png)

Na pasta `src`, criaremos um Servlet com o mapeamento `alteraEmpresa`, e o seu nome será `AlteraEmpresaServlet`. Esse novo Servlet terá apenas o método `doPost()`, afinal os dados serão enviados diretamente no corpo da requisição e não por meio dos parâmetros.

Após a criação do Servlet, limparemos os comentários do código:

```
@WebServlet("/alteraEmpresa")
public class AlteraEmpresaServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServ
```

```
    }
}
```

COPIAR CÓDIGO

Precisamos ler os dados da requisição. A abordagem será parecida com a que tivemos em `novaEmpresaServlet`, isto é, leremos os parâmetros e realizaremos o *parsing* da data:

```
@WebServlet("/alteraEmpresa")
public class AlteraEmpresaServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServl

        System.out.println("Alterando empresa");

        String nomeEmpresa = request.getParameter("nome");
        String paramDataEmpresa = request.getParameter("data")

        Date dataAbertura = null;
        try {
            SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/
            dataAbertura = sdf.parse(paramDataEmpresa);
        } catch (ParseException e) {
            throw new ServletException(e);
        }
```

COPIAR CÓDIGO

Precisamos saber qual empresa deverá ser alterada, e atualmente o parâmetro sendo utilizando é `nome`. Como mencionamos nas aulas anteriores, a forma mais adequada de identificar uma empresa é pelo seu `id`. Portanto, em `formAlteraEmpresa.jsp` adicionaremos mais um `<input>`.

```
<body>

    <form action="${linkServletNovaEmpresa }" method="post">

        Nome: <input type="text" name="nome" value="${empresa.n
        Data Abertura: <input type="text" name="data"  value="$
        <input type="text" name="id" value="${empresa.id }">
        <input type="submit" />
    </form>

</body>
```

COPIAR CÓDIGO

Dessa forma, a alteração também deve enviar na requisição o `id`, de forma que não haja dúvidas de qual empresa estamos nos referindo. Agora, em `AlteraEmpresaServlet` leremos o parâmetro e faremos o *parsing*. Por fim, para testarmos as atribuições, iremos imprimir o `id` no console usando o `System.out.println()`:

```
        System.out.println("Alterando empresa");

        String nomeEmpresa = request.getParameter("nome");
        String paramDataEmpresa = request.getParameter("data");
        String paramId = request.getParameter("id");
        Integer id = Integer.valueOf(paramId);

        Date dataAbertura = null;
        try {
            SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/
            dataAbertura = sdf.parse(paramDataEmpresa);
        } catch (ParseException e) {
            throw new ServletException(e);
        }

        System.out.println(id);
```

COPIAR CÓDIGO

De volta ao navegador, ao acessarmos a página da lista de empresas (http://localhost:8080/gerenciador/listaEmpresa (http://localhost:8080/gerenciador/listaEmpresa)), clicaremos sobre o link "edita" na linha da empresa "Caelum", e seremos direcionados para o formulário. Dessa vez teremos três campos: "Nome", "Data Abertura" e um campo não nomeado populado com o valor "2", que corresponde ao "id", além do botão "Enviar".

Ao enviarmos a requisição para o servlet `alteraEmpresa`, poderemos verificar o *id* `2` impresso na tela, o que significa que nosso Servlet já recebeu as informações para o que o modelo seja atualizado.

Como a classe `Banco` é a responsável pelo nosso modelo, escreveremos `Banco banco = new Banco()`. Dessa forma, criaremos a nossa ferramenta para acessar a camada de persistência do nosso modelo.

Em seguida, precisaremos alterar nossa empresa no banco de dados. Nesse ponto, caso estivéssemos trabalhando com um banco real, usaríamos um SQL de atualização, isto é, um *update*. Como nossos dados estão todos em memória, teremos um pouco mais de trabalho.

Aproveitaremos o método `buscaEmpresaEpelaId()`, que criamos anteriormente, para buscar `empresa`. Contudo, a empresa buscada possui ainda os dados antigos, e precisaremos preenchê-la com as informações mais recentes. Logo, usaremos `empresa.setNome(nomeEmpresa)`. Em seguida, incluiremos a `dataAbertura`, a mesma que utilizamos no parsing.

```
Date dataAbertura = null;
        try {
            SimpleDateFormat sdf = new SimpleDateFormat("dd/...
```

```
            dataAbertura = sdf.parse(paramDataEmpresa);
        } catch (ParseException e) {
            throw new ServletException(e);
        }

        System.out.println(id);

        Banco banco = new Banco();
        Empresa empresa = banco.buscaEmpresaPelaId(id);
        empresa.setNome(nomeEmpresa);
        empresa.setDataAbertura(dataAbertura);

    }

}
```

COPIAR CÓDIGO

Já chamamos a alteração via `AlteraEmpresaServlet` e estamos atualizando o modelo. Agora precisamos realizar um redirecionamento para o navegador, para que os dados novos sejam listados e exibidos.

Esse redirecionamento será feito por meio de `reponse.sendRedirect()`, que receberá como parâmetro `listaEmpresas`, isto é, o destino do redirecionamento.

```
        System.out.println(id);

        Banco banco = new Banco();
        Empresa empresa = banco.buscaEmpresaPelaId(id);
        empresa.setNome(nomeEmpresa);
        empresa.setDataAbertura(dataAbertura);

        response.sendRedirect("listaEmpresas");

    }
```

```
}
```

COPIAR CÓDIGO

Tudo pronto para realizarmos um teste. No navegador, acessaremos a página da lista de empresas, que contém "Alura" e "Caelum". Clicaremos sobre o link "edita" e alteraremos o nome "Caelum" para "Facebook". Também modificaremos a data para "01/01/2001".

Não alteraremos o campo "id", pois trata-se da identificação da empresa definida pelo banco de dados e não podemos mudar a camada de persistência simulada. Feito isso, clicaremos sobre o botão "Enviar".

Seremos redirecionados para a lista de empresas e já teremos o nome "Facebook" no lugar de "Caelum".

> Lista de Empresas:
>
> Alura - 31/08/2018 edita remove
>
> Facebook - 01/01/2001 edita remove

A alteração funcionou e as novas informações da empresa são exibidas na lista com sucesso.

Contudo, não é interessante que o campo de id esteja sendo exibido dessa maneira ao editarmos os dados de uma empresa, já que essa é uma informação que não deve ser disponibilizada para o usuário.

Em `FormAlteraEmpresa`, temos `<input>` que representa a `id`. Atualmente estamos utilizando um o tipo (" `type` ") `text`.

```
<form action="${linkServletNovaEmpresa }" method="post">

    Nome: <input type="text" name="nome" value="${empresa.n
    Data Abertura: <input type="text" name="data"  value="$
    <input type="text" name="id" value="${empresa.id }">
    <input type="submit" />
</form>
```

COPIAR CÓDIGO

Contudo, existem o tipo `hidden`, que significa "escondido" ou "oculto". É ele que usaremos para esconder o campo de *id* na tela do navegador:

```
<form action="${linkServletNovaEmpresa }" method="post">

    Nome: <input type="text" name="nome" value="${empresa.n
    Data Abertura: <input type="text" name="data"  value="$
    <input type="hidden" name="id" value="${empresa.id }">
    <input type="submit" />
</form>
```

COPIAR CÓDIGO

Ao acessarmos novamente a tela de edição, o campo *id* não está mais visível. Entretanto, consultando o código fonte da página, confirmaremos que ele ainda existe no formulário.

```
<!DOCTYPE html
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <form action="/gerenciador/alteraEmpresa" method="post">
```

```
            Nome: <input type="text" name="nome" value"Alura" />
            Data Abertura: <input type="text" name="data" value="3
            <input type="submit name="id" value="1">
            <input type="submit" />
        </form>

    </body>
    </html>
```

COPIAR CÓDIGO

Se submetermos a requisição, esse campo continua a ser enviado normalmente, ainda que não esteja mais visível na página. Podemos verificar o funcionamento dessa metodologia alterando uma empresa, veremos que o id continua sendo enviado e a alteração é executada com sucesso.

Nosso objetivo foi alcançado: fizemos a criação dos objetos e realizamos a leitura de todos eles de forma que podemos realizar a alteração na empresa correta por meio de seu id. Além disso, a remoção dos objetos também funciona som sucesso.