CETEJ35 - Java Web - JAVA_XXX (2024_01)

Criando, Alterando e Excluindo

✓ **Feito:** Ver ✓ **Feito:** Gastar pelo menos 20 minutos na atividade ✓ **Feito:** Passar pela atividade até o fim

Fecha: segunda-feira, 2 dez. 2024, 00:00

Nesta aula nós finalizamos a implementação das quatro operações CRUD. Isso significa que nosso usuário será capaz de criar, alterar, excluir e listar as cidades em uma base de dados. Observe que ainda estamos usando uma base local, baseada em uma lista em memória. Nós vamos evoluir esse projeto até integrarmos essa base com um banco de dados.

EXCLUINDO

Para excluir uma cidade, tudo que precisamos é remover a `Cidade` correspondente da lista cidades. Para isso, vamos começar alterando o botão `EXCLUIR` na página `crud.ftlh` para que ele envie uma solicitação para o `CidadeController`.

Assim, vamos adicionar o atributo `href` à *tag* `a`, usada para criar o botão. Vamos enviar a solicitação para a URL `/exclu` Adicionalmente, precisamos informar qual cidade queremos excluir. Nesse caso, sabemos que, teoricamente, o nome e único dentro de um estado. Por isso, vamos enviar o nome e o estado como parâmetros na requisição. A Figura abaixo mostra como ficou o código do botão.

Na verdade, na forma como está o código, não tem qualquer validação durante a inserção. Portanto, ele vai aceitar valores vazios, cidades repetidas, etc. Teremos uma aula específica sobre validação, na qual iremos resolver isso.

```
40          <tbody>
41              <#list listaCidades as cidade >
42                  <tr>
43                      <td>${cidade.nome}</td>
44                      <td>${cidade.estado}</td>
45                      <td>
46                          <div class="d-flex d-justify-content-center">
47                              <a class="btn btn-warning mr-3">ALTERAR</a>
48                              <a href="/excluir?nome=${cidade.nome}&estado=${cidade.estado}" class="btn btn-danger">EXCLUIR</a>
49                          </div>
50                      </td>
51                  </tr>
52              </#list>
53          </tbody>
```

A linha 48 da Figura acima mostra que estamos enviando dois parâmetros, `nome` e `estado`, para a URL `/excluir`. Os valores desses parâmetros são dinâmicos, uma vez que os valores da tabela são gerados automaticamente (Veja linha 41). Portanto, tudo que precisamos fazer é extrair o nome e o estado da cidade atual no *loop*, e apresentar esses valores usando a sintaxe de interpolação do Freemarker (`${}`).

Agora, na classe `CidadeController`, precisamos criar um método que receba esses parâmetros e que também esteja mapeado com a URL `/excluir`. Para isso, vamos criar um método chamado `excluir()`, que retorna uma `String` e recebe duas `String` como parâmetro: `nome` e `estado`.

Vamos adicionar aos parâmetros a anotação `org.springframework.web.bind.annotation.RequestParam`. Essa anotação mapeia os parâmetros enviados pelo formulário com os parâmetros do método `excluir()`. Observe o nome dos parâmetros no formulário e no método excluir são iguais. Isso é necessário para que o Spring Boot faça o mapeamento automático entre eles.

```
38      public String excluir(
39              @RequestParam String nome,
40              @RequestParam String estado) {
```

Dentro do método, vamos usar o método `removeIf()` do *Collections Framework* para remover a `Cidade` da lista. Para isso, vamos usar o atributo `cidades`, do tipo `Set` . O método `removeIf()` usa um predicado (Lambda) como parâmetro. Para iterar pela lista, vamos usar uma variável chamada `cidadeAtual`. Tudo que precisamos fazer é verificar se o `nome` e `estado` da `cidadeAtual` é igual ao `nome` e `estado` passado como parâmetro para o método `excluir()`.

> Não esqueça que comparação de String em Java é feita pelo método `equals()`.

```
42          cidades.removeIf(cidadeAtual ->
43                  cidadeAtual.getNome().equals(nome) &&
44                  cidadeAtual.getEstado().equals(estado));
```

Agora, vamos redirecionar o resultado do método para o método `listar()`, que vai carregar a página e mostrar os dados atualizados. Por fim, adicionamos a anotação que mapeia a URL enviada pelo botão `EXCLUIR`, na página `crud.ftl`, com o método `excluir()`, na `CidadeController`. Veja como ficou o código completo do método `excluir()`Í.

```
37      @GetMapping("/excluir")
38      public String excluir(
39              @RequestParam String nome,
40              @RequestParam String estado) {
41
42          cidades.removeIf(cidadeAtual ->
43                  cidadeAtual.getNome().equals(nome) &&
44                  cidadeAtual.getEstado().equals(estado));
45
46          return "redirect:/";
47      }
```

Você já pode executar o projeto, inserir, listar e excluir cidades. Na próxima Seção vamos criar o último método: alterar.

> O código desenvolvido nesta Seção está disponível no **Github**, na branch `semana03-30-crud-excluir`.

Retroceder Avançar

◄ Atividade I WebConf

Seguir para...

Verificação de aprendizado - Criando, Alterando e Excluindo ►

Contate o suporte do site

Você acessou como RAFAEL ROCHA DA SILVA PROENCA (Sair)
CETEJ35 - Web (2024_01)

Tema
Adaptable
Boost
Clássico

Campus
Apucarana
Campo Mourão
Cornélio Procópio
Curitiba
Dois Vizinhos
Francisco Beltrão
Guarapuava
Londrina
Medianeira
Pato Branco
Ponta Grossa
Reitoria
Santa Helena
Toledo

UTFPR
Ajuda
Chat UTFPR
Calendário Acadêmico
Biblioteca
e-Mail
Nuvem (OwnCloud)
Produção Acadêmica
Secretaria Acadêmica
Sistemas Corporativos
Sistema Eletrônico de Informação - SEI
Suporte ao usuário
Criação de curso
Comunidade

Português - Brasil (pt_br)
Deutsch (de)
English (en)
Português - Brasil (pt_br)

Resumo de retenção de dados

Baixar o aplicativo móvel.

Dê um feedback sobre este software

Universidade Tecnológica Federal do Paraná - UTFPR
Suporte ao usuário