

Redirecionando requisições

Uma outra funcionalidade importante na nossa aplicação é a de deletar contas. Vamos colocar então um link para deletar uma conta na listagem:

```
<td><a href="removeConta?id=${conta.id}">Remover</a></td>
```

Repare no link acima. Veja que estamos passando para a variável `id`, o ID da conta. Com esse ID, fica fácil deletar a conta depois. Vamos agora para o método do nosso controller.

```
@RequestMapping("/removeConta")
public String remove(Conta conta) {
    ContaDAO dao = new ContaDAO();
    dao.remove(conta);
    return "para onde ir???";
}
```

A pergunta que aparece é: para onde redirecionar depois da deleção? Geralmente, aplicações web como essas devolvem o usuário para a listagem.

Uma das formas que poderíamos fazer esse redirecionamento é enviar o usuário diretamente para a página que lista as contas (`conta/lista.jsp`). Mas, essa não é uma boa abordagem, porque precisaríamos, outra vez, disponibilizar a lista das contas para o JSP, algo que já fazemos na ação de listar as contas, o método `lista` na classe `ContaController`.

Já que o método `lista` faz esse trabalho, poderíamos, ao invés de redirecionar a execução para o JSP, enviá-la para essa ação. Para isso, o retorno do método deve ser um pouco modificado. Vamos continuar devolvendo uma `String`, mas essa `String` deve indicar que queremos chamar uma outra ação.

Podemos fazer um redirecionamento no lado do servidor (`forward`) ou pelo navegador, no lado do cliente (`redirect`). Para fazer um redirecionamento no lado do servidor basta usar o prefixo `forward:` no retorno:

```
@RequestMapping("/removeConta")
public String remove(Conta conta) {
    ContaDAO dao = new ContaDAO();
    dao.remove(conta);
    return "forward:listaContas";
}
```

Para fazer um redirecionamento no lado do cliente usamos o prefixo `redirect:`:

```
@RequestMapping("/removeConta")
public String remove(Conta conta) {
    ContaDAO dao = new ContaDAO();
    dao.remove(conta);
    return "redirect:listaContas";
}
```

Redirecionar para ações já existentes é bem comum, e prático. Nos ajuda a escrever menos código!