02

Verbos HTTP e Convenção de nomes das URLs

Utilizando os Verbos do HTTP

Até agora mapeamos todos os nosso métodos do `ProdutoController` utilizando a anotação `@Path`. Dessa forma ele atende as requisições feitas independente do verbo HTTP utilizado no `form`. Repare em como ficou a nossa URL após cadastrar um novo produto utilizando o método padrão, que é o `GET`:

http://localhost:8080/vraptor-produtos/produto/adiciona?produto.nome=Camiseta+Hollister&produto.valor=250%2C00&produto.quantidade=10 (http://localhost:8080/vraptor-produtos/produto/adiciona?produto.nome=Camiseta+Hollister&produto.valor=250%2C00&produto.quantidade=10).

Quando utilizamos o método `GET`, todas as informações são enviadas pela URL. Nem sempre queremos deixar tudo visível dessa forma. Além disso, existe um limite para o tamanho da URL e ainda corremos o risco de perder algumas informações se o conteúdo que formos adicionar for muito grande!

Podemos modificar nosso formulário para utilizar o método `POST`. Dessa forma, os dados não ficarão mais visiveis na URL e tudo é passado dentro do corpo do protocolo HTTP. A URL ficará: http://localhost:8080/vraptor-produtos/produto/adiciona (http://localhost:8080/vraptor-produtos/produto/adiciona), sem os dados visíveis!

É muito importante lembrar que isso não é uma medida de segurança, afinal os dados ainda podem ser consultados pelo navegador do usuário. O ganho dessa abordagem é que o corpo do `POST` é muito maior que a URL do `GET`, e evitamos deixar os dados sempre tão expostos.

Mas afinal, quando usar cada verbo? É muito comum utilizarmos o método **GET** para listagens ou visualização de algum de nossos recursos, sempre em operações que não modificam os valores do sistema.

O **POST** para adição de informações a um recuso existente ou adição de um novo recurso, como estamos fazendo agora com o `Produto`.

Além disso, existem outros verbos como o **PUT** e **DELETE**, que são utilizados para atualização e remoção, respectivamente.

Para utilizar o método `POST` na adição de produtos, tudo que precisamos fazer é adicionar o atributo `method="post"` em nosso `form` do `formulario.jsp`:

```
<form action="<c:url value='/produto/adiciona'/>" method="post">
    Nome: <input type="text" name="produto.nome"/>
    Valor: <input type="text" name="produto.valor"/>
    Quantidade: <input type="text" name="produto.quantidade"/>
    <input type="submit" value="Adicionar" />
</form>
```

Agora em nosso método `adiciona` do `ProdutoController`, vamos adicionar a anotação `@Post`:

```
@Path("/produto/adiciona") @Post
public void adiciona(Produto produto) {
    EntityManager em = JPAUtil.criaEntityManager();
```

```
    em.getTransaction().begin();
    ProdutoDao dao = new ProdutoDao(em);
    dao.adiciona(produto);
    em.getTransaction().commit();
}
```

Da mesma forma, podemos utilizar as anotações dos outros verbos HTTP como GET (`@Get`), PUT (`@Put`) ou DELETE (`@Delete`)

Vamos reiniciar o servidor e testar um novo cadastro de produtos!

Agora que vimos que tudo está funcionando, é possivel simplificar ainda mais o nosso código! Podemos passar o caminho diretamente pra anotação do verbo que escolhermos, assim não precisamos utilizar a anotação `@Path` :

```
@Post("/produto/adiciona")
public void adiciona(Produto produto) {
    EntityManager em = JPAUtil.criaEntityManager();
    em.getTransaction().begin();
    ProdutoDao dao = new ProdutoDao(em);
    dao.adiciona(produto);
    em.getTransaction().commit();
}
```

Convenção de nomes das nossas URLs

Até agora nomeamos as nossas urls da seguinte forma:

```
ProdutoController.lista()       =>  /produto/lista
ProdutoController.formulario()  =>  /produto/formulario
ProdutoController.adiciona()    =>  /produto/adiciona
```

Repare que a url sempre começa com o nome do `Controller` (sem o sufixo `Controller`), e logo em seguida usamos o mesmo nome do método. Essa é uma outra convenção do VRaptor, para simplificar ainda mais nosso código podemos anotar nossos métodos apenas com as anotações de verbo, como o `@Get` ou `@Post` , e por padrão o VRaptor vai nomear as url dessa mesma forma, considerando o nome do Controller e método. Vamos alterar nosso código:

```
@Controller
public class ProdutoController {

    @Get
    public List<Produto> lista() {
        EntityManager em = JPAUtil.criaEntityManager();
        ProdutoDao dao = new ProdutoDao(em);
        return dao.lista();
    }

    @Post
    public void adiciona(Produto produto) {
        EntityManager em = JPAUtil.criaEntityManager();
        em.getTransaction().begin();
        ProdutoDao dao = new ProdutoDao(em);
```

```
            dao.adiciona(produto);
            em.getTransaction().commit();
        }
    }
```