05

Taglibs do Spring Security

Transcrição

Na página inicial da nossa aplicação, os links para listagem e cadastro de produtos ainda é visível para quem não está logado, ou seja, estão públicos e isso não é bom. Devemos esconde-los e somente exibi-los se o usuário estiver logado.

O *Spring* nos fornece uma *Taglib* que faz exatamente o que queremos fazer. Esta que pode ser importada na página `home.jsp` da seguinte maneira:

```
<%@ taglib uri="http://www.springframework.org/security/tags" prefix="security" %>
```

Ao importar a *Taglib*, podemos fazer uso da tag `<security:authorize>` que tem o atributo `access` no qual podemos usar o método `isAuthenticated` também fornecido pela própria *Tablib* do *Spring*. Assim devemos encontrar o seguinte trecho de código:

```
<div id="header-content">
    <nav id="main-nav">
        <ul class="clearfix">
            <li><a href="${s:mvcUrl('PC#listar').build() }" rel="nofollow">Listagem de Produtos
            <li><a href="${s:mvcUrl('PC#form').build() }" rel="nofollow">Cadastro de Produtos</
            <li><a href="/cart" rel="nofollow">Carrinho</a></li>
            <li><a href="/pages/sobre-a-casa-do-codigo" rel="nofollow">Sobre Nós</a></li>
        </ul>
    </nav>
</div>
```

E esconder os links de listagem e cadastro de produtos da seguinte forma:

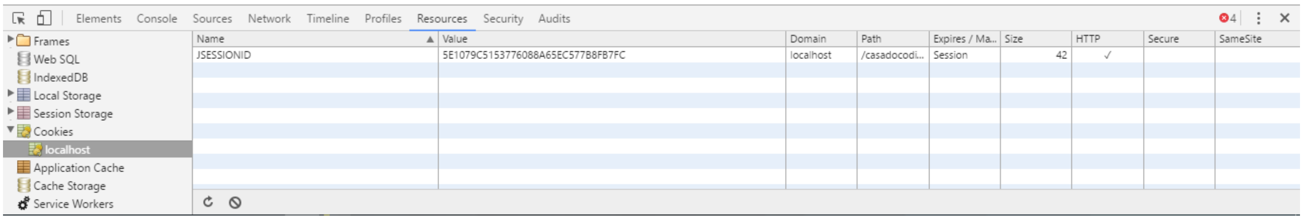
```
<security:authorize access="isAuthenticated()">
    <li><a href="${s:mvcUrl('PC#listar').build() }" rel="nofollow">Listagem de Produtos</a></li
    <li><a href="${s:mvcUrl('PC#form').build() }" rel="nofollow">Cadastro de Produtos</a></li>
</security:authorize>
```

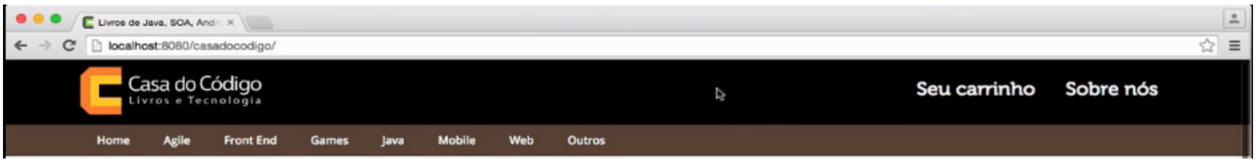
O menu completo da página `home.jsp` ficará da seguinte forma:

```
<div id="header-content">
    <nav id="main-nav">
        <ul class="clearfix">
            <security:authorize access="isAuthenticated()">
                <li><a href="${s:mvcUrl('PC#listar').build() }" rel="nofollow">Listagem de Produ
                <li><a href="${s:mvcUrl('PC#form').build() }" rel="nofollow">Cadastro de Produto
            </security:authorize>
            <li><a href="/cart" rel="nofollow">Carrinho</a></li>
            <li><a href="/pages/sobre-a-casa-do-codigo" rel="nofollow">Sobre Nós</a></li>
        </ul>
    </nav>
</div>
```

Mas ao atualizarmos a página, não teremos nenhum resultado visual já que estamos logados e não implementamos nenhuma forma de deslogar da aplicação ainda. Como faremos para verificar se os links realmente não aparecem para os usuários não logados? A forma mais simples é apagando o **Cookie** de sessão do *Spring*.

Abrindo as ferramentas do desenvolvedor do navegador, podemos navegar até a sessão `resourses -> cookies -> localhost` e apagar o *Cookie* com o nome `JSESSIONID`. Assim, ao atualizarmos a página não veremos mais os links que ocultamos anteriormente.





Mostrando dados do usuário logado.

Uma caracteristica interessante para adicionarmos na nossa aplicação será mostrar qual usuário está logado no momento. O lugar em que o nome do usuário costuma ser exibido nas aplicações é dentro do painel de administração. No caso, pode ser a página de listagem dos produtos, no canto superior, do lado direito do menu principal.

Para exibirmos dados do usuário logado, as tags de segurança do *Spring* nos ajudam de uma forma bastante simples. Existe uma tag chamada `security:authentication`, na qual através do atributo `property` podemos usar o valor `principal` para recuperar dados do usuário atualmente logado.

Vamos abrir a página `lista.jsp` e encontrar o seguinte treco de código:

```
<div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
  <ul class="nav navbar-nav">
    <li><a href="${s:mvcUrl('PC#listar').build() }">Lista de Produtos</a></li>
    <li><a href="${s:mvcUrl('PC#form').build() }">Cadastro de Produtos</a></li>
  </ul>
</div>
```

Logo abaixo da tag `ul` criaremos uma nova `ul` com as classes padrões do bootstrap para criação de um menu alinhado à direita da barra de menu.

```
<ul class="nav navbar-nav navbar-right">
  <li><a href="#">NOME DO USUÁRIO</li>
</ul>
```

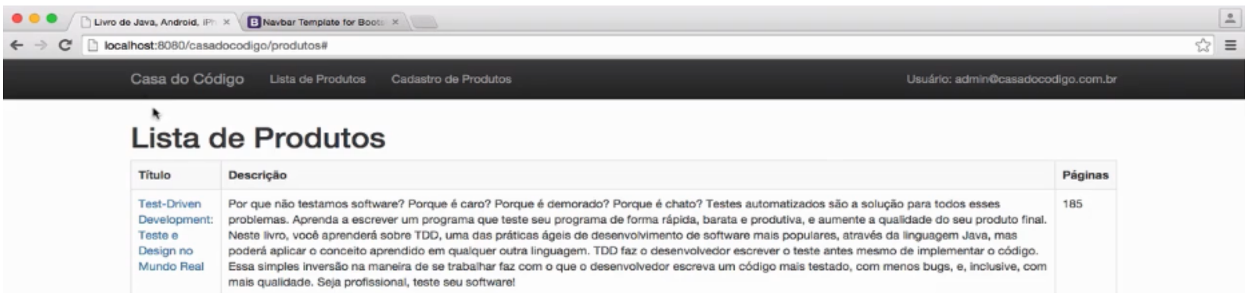
Neste novo trecho de código, substituiremos `NOME DO USUÁRIO` pela tag `security:authentication` . Usaremos o atributo `property` e através do valor `principal` , vamos recuperar o `username` atual do usuário, da seguinte forma:

```
<ul class="nav navbar-nav navbar-right">
  <li><a href="#"><security:authentication property="principal.username"/></a></li>
</ul>
```

Visualmente, já podereamos verificar o resultado após logar na aplicação.

Uma variação do código anterior é usar outra variável para guardar os dados do usuário. Podemos fazer isto da seguinte forma para alcançar o mesmo resultado:

```
<ul class="nav navbar-nav navbar-right">
  <li>
    <a href="#">
        <security:authentication property="principal" var="usuario"/>
        Usuário: ${usuario.username}
    </a>
  </li>
</ul>
```



Outra variação de código que pode ser usada para exibir (ou não) informações nas páginas `jsps` é o uso do `hasRole` na tag `security:authorize` , passando a *role* requerida para o acesso. No caso de ocultar os links da página `home.jsp` , como fizemos antes, podemos fazer dessa maneira:

```
<security:authorize access="hasRole('ROLE_ADMIN')">
    <li><a href="${s:mvcUrl('PC#listar').build() }" rel="nofollow">Listagem de Produtos</a></li
```

```
    <li><a href="${s:mvcUrl('PC#form').build() }" rel="nofollow">Cadastro de Produtos</a></li>
</security:authorize>
```

Neste caso o uso do parâmetro para o `hasRole` precisa ser prefixado com o **ROLE_**.