

## Montando a View de Cadastro

### Transcrição

Quando terminamos de inserir os produtos no carrinho e queremos finalizar a compra clicando no botão "Finalizar Pedido", somos direcionados para o resumo de compra, que conterà informações do pedido e do cadastro do cliente.

#### Resumo do Pedido

Nº do Pedido: 29007	
<b>Seus Dados</b> Harry Porto (11) 1234-5678	harry.porto@alura.com.br <b>Endereço de Entrega</b> Rua dos Alfeneiros, no. 4 (embaixo da escada) - Little Whinging - Surrey - Reino Unido
<b>Item</b> Algoritmos em Java	<b>Quantidade</b> 1

Contudo, as informações que dispomos são falsas e servem apenas de modelo. Queremos exibir informações que foram coletadas do cliente, para isso precisamos criar uma tela de cadastro. Faremos uma modificação no fluxo de navegação e trocaremos o botão "Finalizar Pedido" da tela de carrinho por outra que direcione o usuário para uma tela de cadastro.

#### Meu Carrinho

Item	Preço Unitário	Quantidade	Subtotal
 Algoritmos em Java	R\$ 49,90	- 1 +	R\$ 49,90
Total: 1 itens			Total: R\$ 49,90

Adicionar Produtos Finalizar Pedido

Acessaremos `Cadastro.cshtml`, isto é, nossa view de cadastramento de usuários. Teremos todos os elementos que compõe o formulário, que até o momento não passa de um conjunto de tags html, portanto precisamos criar os campos para serem preenchidos.

Incluiremos o elemento `<form>` no começo do código. Tal elemento precisa de algumas informações, como a `action`, que é o endereço para onde enviaremos as informações, ou seja, `/pedido/resumo`. Inseriríamos, ainda, o método `post()` cuja função é enviar dados e realizar mudanças na aplicação.

```
@{
    ViewData["Title"] = "Cadastro";
}

<h3>Cadastro</h3>

<form action="/pedido/resumo" method="post">
```

```
</form>
```

```
<div class="panel panel-default">
```

Incluiremos em `<form>` a `<div>` que contém as informações do cadastro de usuário.

```
@{
    ViewData["Title"] = "Cadastro";
}

<h3>Cadastro</h3>

<form action="/pedido/resumo" method="post">

<div class="panel panel-default">
    <div class="panel-body">
        <div class="col-md-4">
            <div class="form-group">
                <label class="control-label" for="nomeCliente">Nome do Cliente</label>
                <input type="text" class="form-control" id="nomeCliente" placeholder="Nome do Client
            </div>
            <div class="form-group">
                <label class="control-label" for="email">Email</label>
                <input type="email" class="form-control" id="email" placeholder="Email">
            </div>
            <div class="form-group">
                <label class="control-label" for="telefone">Telefone</label>
                <input type="text" class="form-control" id="telefone" placeholder="Telefone">
            </div>

<****!****>

                class="btn btn-success">
                    Finalizar Pedido
                </button>
            </div>
        </div>
    </div>
</div>

</form>
```

Estamos trabalhando com a forma html que será executada no browser. Contudo, existe outra maneira de criar o formulário por meio da *tag helper*. Para isso, trocaremos o atributo `action` para `asp-action`. Assim feito, mudaremos o nome da nossa antiga `action` para uma que se encontra em `PedidoController.cs`, que será a `resumo`.

```
@{
    ViewData["Title"] = "Cadastro";
}

<h3>Cadastro</h3>

<form asp-action="resumo" method="post">
```

```
<div class="panel panel-default">

<****!****>
```

Começaremos a colocar os campos do formulário e associá-los ao nosso modelo, contudo ainda não definimos um modelo para esta view, faremos isso nesse momento. No começo do código, criaremos uma diretiva `model`, em seguida, o nome da classe `Cadastro`.

```
@{
    ViewData["Title"] = "Cadastro";
}

@model Cadastro

<h3>Cadastro</h3>

<form asp-action="resumo" method="post">

    <div class="panel panel-default">

<****!****>
```

Precisamos associar cada campo do formulário com uma propriedade da classe `Cadastro`. Faremos esse procedimento ao modificar cada um dos elementos `input`. O primeiro é aquele que recebe `nomeCliente`.

```
<input type="text" class="form-control" id="nomeCliente" placeholder="Nome do Cliente">
</div>
```

Declaremos como tag helper a propriedade que está associada a este `input`, faremos isso ao criar um novo atributo chamado `asp-for`. Assim feito, criaremos a associação entre uma propriedade do modelo: `@Model.Nome`

```
<input type="text" class="form-control" id="nomeCliente" placeholder="Nome do Cliente"
    asp-for="@Model.Nome">
</div>
```

Faremos o mesmo procedimento para todos os `inputs` do formulário, são eles `Email`, `Telefone`, `Endereço`, `Complemento`, `Bairro`, `Município` e `CEP`.

```
<input type="text" class="form-control" id="nomeCliente" placeholder="Nome do Cliente"
    asp-for="@Model.Nome">
    <span asp-validation-for="@Model.Nome" class="text-danger"></span>
</div>
<div class="form-group">
    <label class="control-label" for="email">Email</label>
    <input type="email" class="form-control" id="email" placeholder="Email"
        asp-for="@Model.Email">
    <span asp-validation-for="@Model.Email" class="text-danger"></span>
</div>
<div class="form-group">
    <label class="control-label" for="telefone">Telefone</label>
```

```
<input type="text" class="form-control" id="telefone" placeholder="Telefone"
      asp-for="@Model.Telefone">
<span asp-validation-for="@Model.Telefone" class="text-danger"></span>
</div>
</div>
```

```
<****!****>
```

O elemento responsável pelo estado ( UF ) não se trata de um input, mas de select. Temos a caixa de listagem com todos os estados brasileiros disponíveis. Incluiremos `@Model.UF`.

```
<div class="form-group">
  <label class="control-label" for="UF">UF</label>
  <select class="form-control single-select" name="UF"
        asp-for="@Model.UF">
    <option selected value="">[Selecione o estado]</option>
    <option value="AC">Acre</option>
    <option value="AL">Alagoas</option>
    <option value="AP">Amapá</option>
    <option value="AM">Amazonas</option>
```

```
<****!****>
```

Nas próximas aulas prepararemos o envio do modelo para a view.