

Dell IT Academy

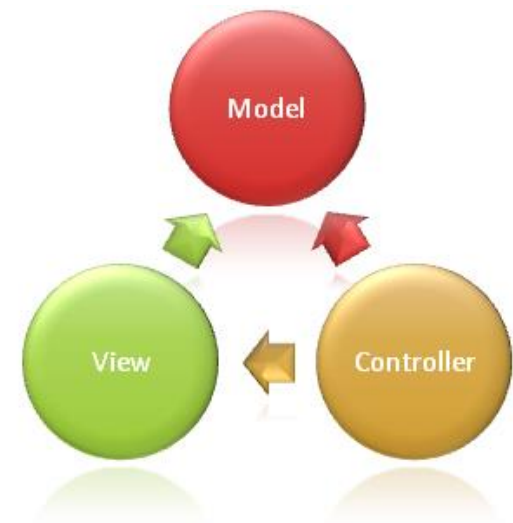


ASP.NET CORE MVC

VIEWS

Visão Geral do MVC

- Model: objetos que representam o domínio da aplicação
- **View: componentes que apresentam a interface de usuário**
- Controller: componentes que tratam a interação com o usuário, operam sobre modelos e selecionam a visão adequada



MVC View

- O conteúdo de uma View combina marcações da Web com a de scripts cliente e servidor
 - Sintaxe Razor
 - <https://learn.microsoft.com/en-us/aspnet/core/mvc/views/razor>
- Utiliza os dados enviados pelo controlador para preencher uma representação da interface do sistema

MVC View

- Documentos *.cshtml*
- Dois locais padrão:
 - Diretório */Views/[NomeDoControlador]* – para views específicas de um controlador
 - Diretório */Views/Shared* – para views compartilhadas entre os controladores
- Arquivos especiais:
 - *_Layout.cshtml* – define layout padrão das views
 - *_ViewImports.cshtml* – define importação de diretivas para as views
 - *_ViewStart.cshtml* – define código a ser executado antes do processamento das views

MVC View

- Controladores especificam a view através do método de factory *View()* que retorna um objeto do tipo *ViewResult*
- Exemplos:
 - Para view implícita associada ao controlador
 - `return View()`
 - Para view explícita
 - `return View("NomeDaView")`
 - Para view que recebe um objeto model
 - `return View(Objeto)`
 - Para view explícita que recebe um objeto model
 - `return View("NomeDaView", Objeto)`

Marcações *Razor*

- Marcações *Razor* iniciam com um @
 - Usar @@ para gerar o símbolo no html
- Comentários

```
@*Curso Microsoft MVC*@
```

- Código C#

```
@{ var list = new List<string>() { "WebForms", "Razor",  
                                   "Spark", "NHaml" };  
    DateTime hoje = DateTime.Now.Date;  
}
```


Marcações *Razor*

- Texto e marcação
- Repetição

Definido no bloco C#, texto é “***HtmlEncoded***”

```
<h3>Hoje: @hoje.ToShortDateString()</h3>
<h3>Hora no servidor: @ViewBag.Hora</h3>
<ul>
  @foreach (var item in list) {
    <li>@item</li>
  }
</ul>
```

Iterador

Definido no controlador

Definido no bloco C#

```
<h3>Hoje: 30/09/2013</h3>
<h3>Hora no servidor: 08:12</h3>
<ul>
  <li>WebForms</li>
  <li>Razor</li>
  <li>Spark</li>
  <li>NHaml</li>
</ul>
```

Marcações *Razor*

- Seleção
- *Plain text*
- Expressões explícitas

```
@(horaAgora < 12 ? "Bom dia" : "Boa tarde")
```

```
@if ( @horaAgora < 12 )  
{  
    @:Bom dia  
}  
else  
{  
    <text>Boa tarde</text>  
}
```

Tag Helpers

- Objetos auxiliares para a codificação das marcações de HTML a serem renderizadas no lado servidor
 - Criação de formulários
 - Criação de links com base nas ações de controladores
 - Carregamento de imagens
 - Validação de campos
 - etc
- Podem ser consumidos via elementos ou atributos
- Documentação:
 - <https://learn.microsoft.com/en-us/aspnet/core/mvc/views/tag-helpers/>
 - <https://learn.microsoft.com/en-us/aspnet/core/mvc/views/tag-helpers/built-in/>
 - <https://learn.microsoft.com/en-us/aspnet/core/mvc/views/working-with-forms>

Tag Helpers

- Exemplo: LabelTagHelper

```
<label class="caption" asp-for="PrimeiroNome"></label>
```

Irá renderizar

```
<label class="caption" for="PrimeiroNome">Primeiro Nome</label>
```

Compartilhamento de dados Controller-View

- Dois grandes meios:
 - Dados fortemente tipados via objeto viewmodel
 - Dados fracamente tipados:
 - **ViewData**: permite passar informações do método de ação para a view por meio de um dicionário
 - **ViewBag**: é um *wrapper* sobre o *ViewData* que permite passar informações do método de ação para a view por meio de propriedades

Compartilhamento de dados Controller-View

Controller

```
ViewBag.Data= DateTime.Now;
```

View

```
<h4>Hora: @ViewBag.Data.ToShortDateString()</h4>
```

Controller

```
ViewData["Hora"] = DateTime.Now;
```

View

```
<h4>Data: @(((DateTime)ViewData["Hora"]).  
ToShortTimeString())</h4>
```

Compartilhamento de dados Controller-View

- Ao criar uma *view* fortemente tipada, é incluída a declaração de qual o modelo que será utilizado via *@model*

```
@model MvcMusicStore.Models.Album
```

Uma instância de *Album*

```
@model IEnumerable<MvcMusicStore.Models.Album>
```

Uma coleção de *Album*

Compartilhamento de dados Controller-View

- Acesso por meio da propriedade *@Model*

```
@model MvcMusicStore.Models.Album
...
<h2>Delete Confirmation</h2>
<h3>Delete the album title <strong>@Model.Title</strong> ?</h3>
```


Apresentação do Model

- Anotações permitem definir como as propriedades do Model devem ser renderizadas :
 - Display
 - DataType
 - DisplayFormat
- Disponíveis no namespace
System.ComponentModel.DataAnnotations

Exemplo de anotações

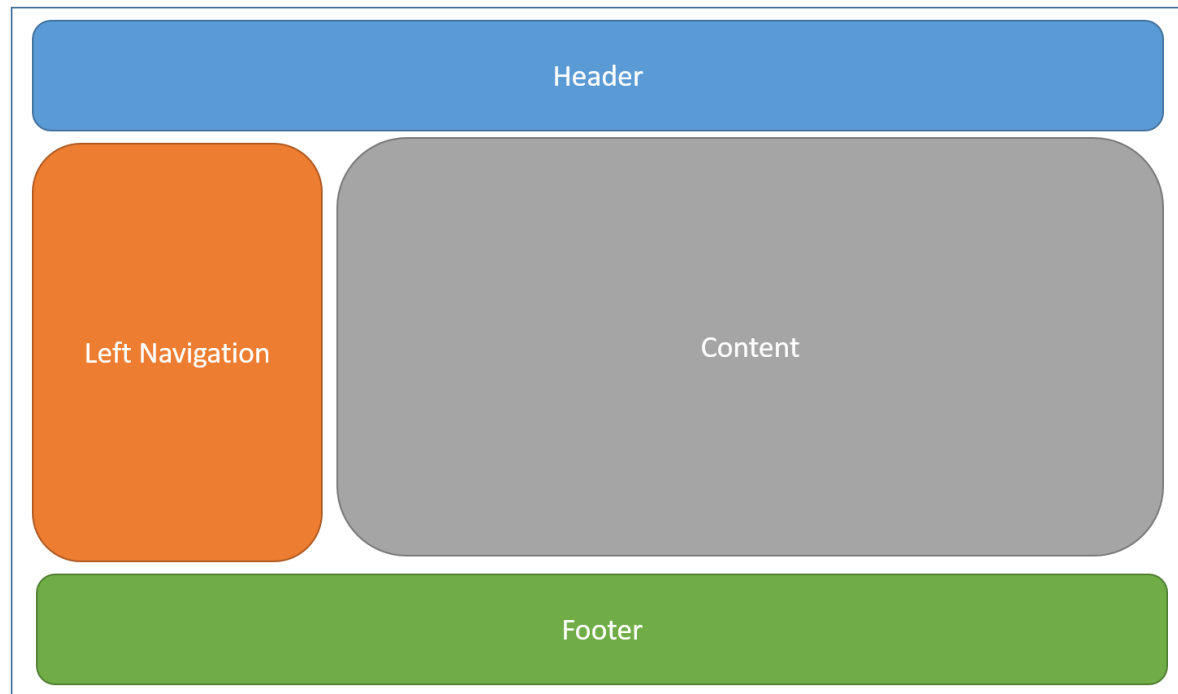
```
[Display(Name="Data de nascimento")]  
[DisplayFormat(DataFormatString="{0:d}",  
                ApplyFormatInEditMode=true)]  
public DateTime dnasc { get; set; }
```

```
[Display(Name="Data de nascimento")]  
[DataType(DataType.Date)]  
public DateTime dnasc { get; set; }
```

LAYOUT

Layout Page

- Permite definir um padrão visual comum a todas as páginas
- Evita repetição de códigos nas *views*
- Facilita a manutenção



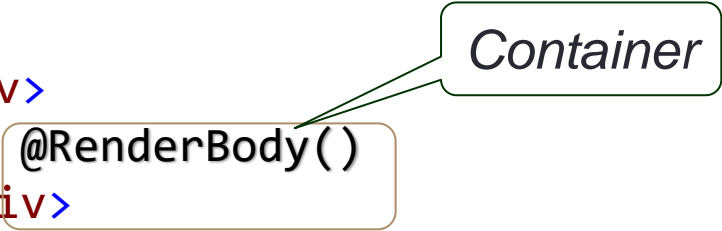
Layout Page

- *Layout page*
 - Define conteúdo estático e dinâmico
 - Define um ou mais *containers* para renderização de conteúdos
- *View Page (com layout)*
 - Define conteúdo para o os *containers*

Layout Page

- Template que pode ser aplicado a um conjunto de páginas

```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>@ViewBag.Title</title>
</head>
<body>
    <div>
        @RenderBody()
    </div>
    @RenderSection("scripts", required: false)
</body>
</html>
```



Ligação de views a layouts

- Layout padrão para a aplicação é definido no arquivo **_Layout.cshtml**
- Arquivos de layout normalmente devem ser criados na pasta **Views | Shared**
- Para especificar o layout, utilizar a propriedade **Layout** na view page:

```
@{  
    Layout = "_Layout";  
}
```

Ligação de views a layouts

```
@{  
    ViewBag.Title = "Uso de layouts";  
    Layout = null;  
}
```

Não utiliza layout

```
@{  
    ViewBag.Title = "Uso de layouts";  
    Layout = "~/Views/Shared/_LayoutPageTeste.cshtml";  
}
```

Arquivo de layout a ser utilizado