Ficha de Trabalho N.º 7

ASP.NET - WebMatrix

1. Iniciar o WebMatrix
2. Abrir o web site WebPagesMovie.
3. Clicar em Files para abrir o ambiente de trabalho Files.
4. Abrir o ficheiro Filmes.cshtml.
5. Na seção body da página alterar a entrada WebGrid para lhe acrescentar uma coluna.

```
@grid.GetHtml(
    tableStyle: "grid",
    headerStyle: "head",
    alternatingRowStyle: "alt",
    columns: grid.Columns(
        grid.Column(format: @<a href="~/EditarFilme?id=@item.ID">Editar</a>),
        grid.Column("Título"),
        grid.Column("Género"),
        grid.Column("Ano")
```

6. A nova coluna é esta:

```
grid.Column(format: @<a href="~/EditarFilme?id=@item.ID">Editar</a>),
```

   a. O objetivo desta coluna é mostrar um link (element HTML <a>) com o texto Editar. Quando clicamos em Editar numa das linhas da tabela de Filmes, é criado um link deste tipo:

   http://localhost:32122/EditarFilme?id=5

   com um id diferente para cada filme.

   b. O link vai invocar uma página EditarFilme passando-lhe a string ?id=5

   c. O objeto WebGrid mostra linhas, uma para cada registo da tabela da base de dados e colunas, uma para cada campo da tabela. À medida que cada linha do objeto WebGrid está a ser construída, o objeto item contém o registo da tabela para aquela linha. Isto torna possível aceder aos dados de uma determinada linha em código. A expressão item.ID dá o valor do ID do registo corrente da base de dados. Podemos obter qualquer valor do registo usando item.Título, item.Género, etc.

   d. Em cada linha, o link Editar foi então criado usando código HTML do tipo:

   Href=”/EditarFilme?id=2”, sendo naturalmento o valor do id diferente para cada linha

7. Criar um novo ficheiro com o nome EditarFilme.cshtml, que será a página chamada sempre que o utilizador clicar em Editar numa linha.
8. Substituir o código existente pelo seguinte:

POPH QUALIFICAR É CRESCER · QREN QUADRO DE REFERÊNCIA ESTRATÉGICO NACIONAL · UNIÃO EUROPEIA Fundo Social Europeu · Governo da República Portuguesa

```
<!DOCTYPE html>
<html>
  <head>
   <meta charset="utf-8" />
   <title>Editar Filme</title>
    <style>
      .validation-summary-errors{
        border:2px dashed red;
        color:red;
        font-weight:bold;
        margin:12px;
      }
    </style>
  </head>
</head>
<body>
  <h1>Editar Filme</h1>
    @Html.ValidationSummary()
  <form method="post">
    <fieldset>
      <legend>Informação do Filme</legend>

      <p><label for="title">Título:</label>
         <input type="text" name="title" value="@title" /></p>

      <p><label for="genre">Género:</label>
         <input type="text" name="genre" value="@genre" /></p>

      <p><label for="year">Ano:</label>
         <input type="text" name="year" value="@year" /></p>

      <input type="hidden" name="movieid" value="@movieId" />

      <p><input type="submit" name="buttonSubmit" value="Submeter Alterações" /></p>
    </fieldset>
  </form>
</body>
</html>
```

9. Tal como na página Adicionar Filme, o atributo value das textbox são pré preenchidos. Mas em vez de usarmos a forma Request.Form[“title”], usamos a variável title.

    a. Existe também um elemento <input type=”hidden”: serve para guardar o campo ID sem o mostrar.

10. Esta página tem duas funcionalidades: a primeira, quando a página é requisitada, o código obtém o Id do filme através da URL da página. A segunda é validar os dados das textbox e atualizar o registo na base de dados.

11. Para concretizar a primeira funcionalidade adicionar o seguinte código ao início da página:

```
@{
    var title = "";
    var genre = "";
    var year = "";
    var movieId = "";

    if(!IsPost){
```

```
    if(!Request.QueryString["ID"].IsEmpty()){
        movieId = Request.QueryString["ID"];
        var db = Database.Open("WebPagesMovie");
        var dbCommand = "SELECT * FROM filmes WHERE ID = @0";
        var row = db.QuerySingle(dbCommand, movieId);
        title = row.Título;
        genre = row.Género;
        year = row.Ano;
    }
    else{
        Validation.AddFormError("Não foi selecionado nenhum filme.");

    }
}
}
```

a. Após ter a certeza que existe um Id válido para um filme, o código procura na base de dados o registo correspondente a esse Id.

b. Usamos o comando db.QuerySingle para obter um único registo da base de dados e esse registo é colocado na variável row. De seguida as variáveis title, genre e year são atualizadas com os valores daquele filme.

12. Guardar as alterações e testar no browser.

13. Para a segunda funcionalidade da página, atualizar o registo na base de dados, adicionar o seguinte código antes de fechar a chaveta do bloco @:

```
    if(IsPost){
    Validation.RequireField("title", "Tem de digitar um título");
    Validation.RequireField("genre", "Género: campo necessário");
    Validation.RequireField("year", "Não digitou o ano");
    Validation.RequireField("movieid", "Não foi submetido nenhum Id!");

    title = Request.Form["title"];
    genre = Request.Form["genre"];
    year = Request.Form["year"];
    movieId = Request.Form["movieId"];

    if(Validation.IsValid()){
        var db = Database.Open("WebPagesMovie");
        var updateCommand = "UPDATE Filmes SET Título=@0, Género=@1, Ano=@2 WHERE Id=@3";
        db.Execute(updateCommand, title, genre, year, movieId);
        Response.Redirect("~/Filmes");
    }
}
```

14. Usamos o comando SQL

    UPDATE tabela SET col1=”valor”, col2=”valor”,… WHERE ID = valor

    Para atualizar o registo na base de dados.

15. Adicionar a seguinte linha após o tag </form>, para crirar um link que permite regressar à página da listagem de filmes:

```
<p><a href="~/Filmes">Regressar à lista de Filmes</a></p>
```