

ASP.NET MVC Tutorial

Do original: <http://www.asp.net/mvc/tutorials/mvc-music-store/mvc-music-store-part-6>

Este tutorial é uma tradução do original criado e mantido pela Microsoft. Este material tem fins educativos e é fornecido na maneira que se apresenta.

Parte 6:: Usando Data Annotations para validar o model

Nós temos um problema com os nossos formulários de Criar e Editar: eles não estão fazendo nenhuma validação. Do jeito que está nós podemos deixar campos em branco ou digitar letras em campos de valores, e este erro irá estourar na camada de banco de dados com mensagens não muito clara para os usuários.

Nós podemos facilmente adicionar validação para nossa aplicação adicionando Data Annotations para nossas classes de modelo. Data Annotations nos permite descrever as regras que nós queremos aplicar para as propriedades do nosso modelo, e o ASP.NET MVC irá zelar para manter estas regras e exibir as mensagens adequadas para os usuários.

Adicionando validação para nossos formulários de Album

Nós iremos usar os seguintes atributos de Data Annotation

- **Required** – Indica que a propriedade é um campo requerido
- **DisplayName** – Define o nome que nós queremos usar nos campos do formulário e mensagens de validação
- **StringLength** – Define o tamanho máximo de um campo texto
- **Range** – Define o valor minimo e máximo de um campo numérico
- **Bind** – Lista campos para excluir e incluir ao fazer o databind automatico de campos do formulario para propriedades no model
- **ScaffoldColumn** – Nos permite esconder campos de formulários de edição

Nota: Para mais informação sobre validação do model usando atributos de Data Annotation veja a documentação em <http://go.microsoft.com/fwlink/?LinkId=159063>

Abra a classe Album e adicione as seguintes cláusulas using no topo da página conforme abaixo.

```
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Web.Mvc;
```

Em seguida atualize as propriedades para adicionar atributos de exibição e validação como mostrado abaixo.

```
namespace MvcMusicStore.Models
{
    [Bind(Exclude = "AlbumId")]
    public class Album
    {
        [ScaffoldColumn(false)]
        public int AlbumId { get; set; }
        [DisplayName("Gênero")]
        public int GeneroId { get; set; }
        [DisplayName("Artista")]
        public int ArtistaId { get; set; }
        [Required(ErrorMessage = "O título do album é obrigatório")]
        [StringLength(160)]
        public string Titulo { get; set; }
        [Required(ErrorMessage = "O preço é obrigatório")]
        [Range(0.01, 100.00,
            ErrorMessage = "O preço precisa ser entre 0.01 e 100.00")]
        public decimal Preco { get; set; }
        [DisplayName("URL da capa do Album")]
        [StringLength(1024)]
        public string AlbumArtUrl { get; set; }
        public virtual Genero Genero { get; set; }
        public virtual Artista Artista { get; set; }
    }
}
```

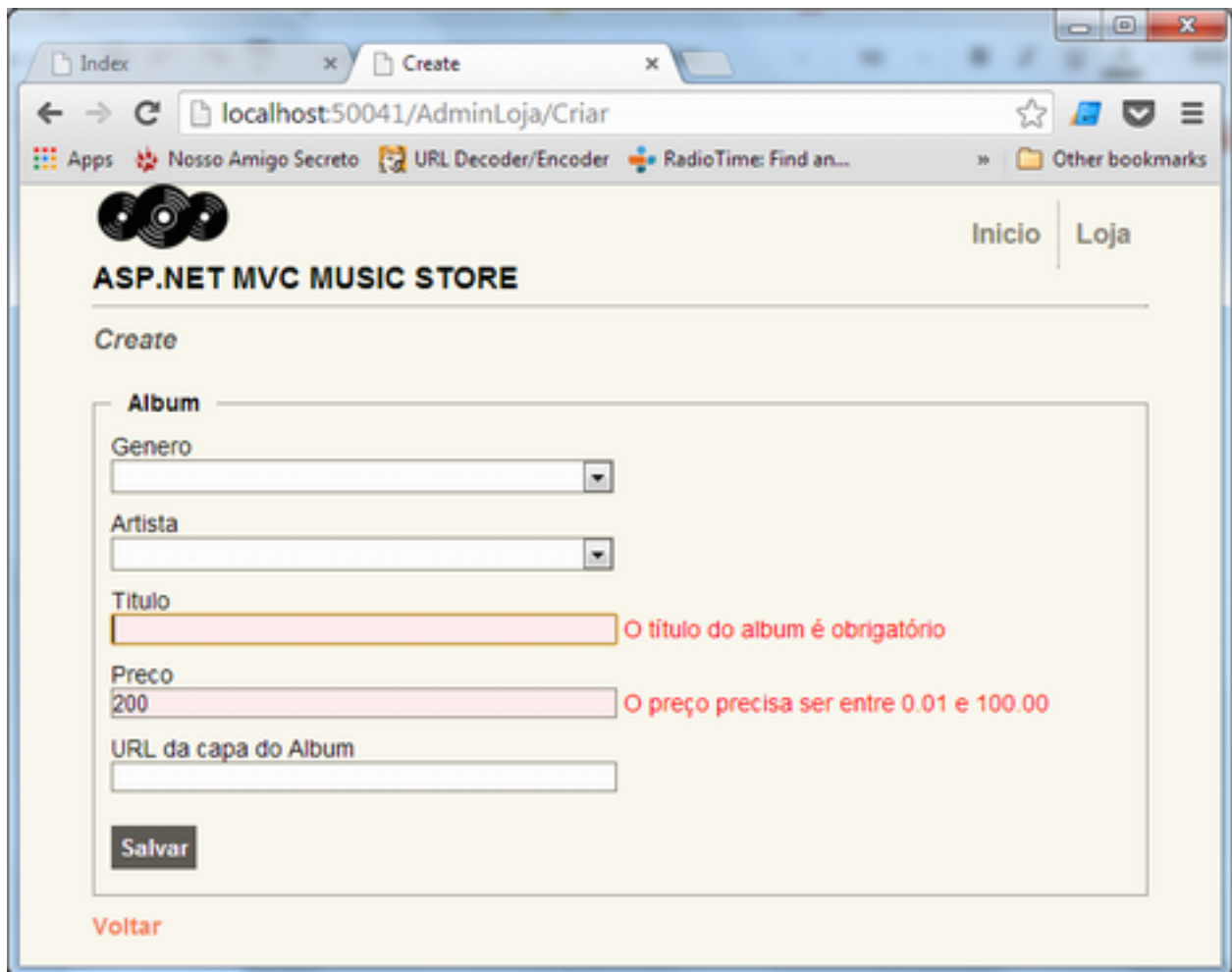
Aproveitando que nós estamos aqui vamos alterar também as propriedades de Genero e Artista para propriedades virtuais. Isto permite que o framework Entity carregue estas propriedades somente quando for solicitado (lazy-loading).

```
public virtual Genre Genre { get; set; }
public virtual Artist Artist { get; set; }
```

Depois de ter adicionado estes atributos para nosso model Album, nossas telas de Criar e Editar irão validar os campos e exibir os nomes dos campos que nós escolhemos. (exemplo URL da capa do Album ao invés de AlbumArtUrl). Execute a aplicação e navegue para /AdminLoja/Criar.

The screenshot shows a web browser window with two tabs: 'Index' and 'Create'. The address bar displays 'localhost:50041/AdminLoja/Cria'. The page has a header with a logo of three vinyl records, the title 'ASP.NET MVC MUSIC STORE', and navigation links 'Inicio' and 'Loja'. Below the header, the page is titled 'Create'. A form titled 'Album' contains the following fields: 'Genero' (a dropdown menu), 'Artista' (a dropdown menu), 'Titulo' (a text input), 'Preco' (a text input), and 'URL da capa do Album' (a text input). A 'Salvar' button is located below the form. At the bottom left of the page, there is a 'Voltar' link.

Em seguida vamos quebrar algumas regras de validação. Entre o preço 200 e deixe o título em branco. Quando nós clicarmos no botão Salvar, nós iremos ver o formulário com os erros de validação para cada campo que não cumpriu com as regras que nós definimos para cada campo.



Testando a validação do lado cliente

A validação do lado do servidor é muito importante da perspectiva da aplicação porque os usuários podem burlar a validação do lado cliente e comprometer a integridade dos dados. Por outro lado aplicações web que somente apresentam validações do lado servidor exibem três problemas principais:

- 1. O usuário tem que aguardar o formulário ser enviado e validado no servidor, e aguardar a resposta ser enviada novamente ao navegador.
- 2. O usuário não obtém um feedback no instante que altera o campo para saber se o problema foi corrigido e se agora os valores estão corretos.
- 3. Nós desperdiçamos recursos do servidor para fazer esta lógica de validação ao invés de aproveitar os recursos da máquina do usuário.


Os templates gerados pelo ASP.NET MVC 3 possuem validação do lado cliente já embutido, não necessitando de trabalho adicional da nossa parte.

Por exemplo, ao digitar uma única letra no campo Título já satisfaz os requisitos de validação e então a mensagem de erro é removida imediatamente.

Index x Create x

localhost:50041/AdminLoja/Criar

Apps Nosso Amigo Secreto URL Decoder/Encoder RadioTime: Find an... Other bookmarks

 [Início](#) | [Loja](#)

ASP.NET MVC MUSIC STORE

Create

Album

Genero

Artista

Título

Preço
 O preço precisa ser entre 0.01 e 100.00

URL da capa do Album

[Voltar](#)