03

Introdução ao ASP.NET MVC

Bem vindo ao curso de Asp.NET MVC 5

Vimos que existem muitas dúvidas sobre a diferença do Asp.NET MVC 5 para a versão 6 do framework, e a resposta é que eles são muito parecidos, a Microsoft tem evoluído demais o framework mas as diferenças são na parte interna. Então podemos fazer esse curso inteiro usando a versão do 5 pois a API é identica a versão 6 e não haverá problema algum.

Downloads

Segue o link do projeto inicial: Visual Studio 2013 (http://s3.amazonaws.com/caelum-online-public/asp-net-mvc5/CaelumEstoque.zip)

Existem muitos programadores que desenvolvem aplicativos web para o sistema operacional Windows. Muitos deles já desenvolveram utilizando a linguagem ASP ou, nos útimos anos, o famoso ASP.NET WebForms.

O WebForms, em sua época, revolucionou a maneira de se programar para Web. Programar com WebForms parecia programar com Visual Basic ou Delphi! Ou seja, os componentes eram arrastados pro formulário, e o programador podia colocar ações em eventos desses componentes, por exemplo, imprimir uma mensagem na tela quando acontecer um click no botão.

Como muita gente na época vinha desse tipo de programação de aplicativos Desktop, aprender WebForms não era tão difícil. Sem contar a grande sensação de produtividade, já que os componentes que existiam faziam as mais diversas coisas, como exibir dados de um banco de dados de maneira elegante, separando em páginas.

Mas infelizmente hoje percebemos que códigos feitos com WebForms são difíceis de manter. O motivo? Apesar de ser altamente produtivo, não existia alguma coisa que separasse o código que lida com interface, do código que lida com regra de negócio, do código que lida com banco de dados, e assim por diante... Na prática, os códigos WebForms são "macarrônicos", ou seja, são grandes e fazem muita coisa diferente.

No fim, o problema é que o WebForms não nos "obriga" a usar boas práticas de programação. Percebendo isso, a Microsoft resolveu criar uma alternativa ao WebForms: o chamado **ASP.NET MVC**! O MVC, um padrão de desenvolvimento muito utilizado no mundo web é conhecido por "forçar" o programador a separar as responsabilidades. Ou seja, código de interface separado de código de regra de negócio separado de código de banco de dados, e assim por diante. A vantagem disso? Sua aplicação viverá por mais tempo! Você conseguirá mantê-la pra sempre! Estudaremos mais sobre o padrão MVC no próximo capítulo.

Nesse curso estudaremos o framework ASP.NET MVC 5 e seus recursos!

Instalação do Visual Studio 2013

Antes de começarmos, precisamos instalar o ambiente de desenvolvimento para o Asp.Net MVC 5, o Visual Studio 2013. Nesse curso utilizaremos o Visual Studio Express 2013 for web (http://www.microsoft.com/en-us/download/details.aspx?id=40747 (http://www.microsoft.com/en-us/download/details.aspx?id=40747)), que é uma versão gratuita da ferramenta. Ou seja, para fazer o curso você não precisará comprar a versão paga da ferramenta (embora recomendado para empresas).

Depois de fazer o download do instalador, execute-o. Isso abrirá uma nova janela em que o Visual Studio pergunta se aceitamos o contrato de licença. Leia os termos e depois marque a opção `I agree to the License Terms and Privacy Policy` se você aceitar os termos da Microsoft.

[Imagem JanelaInstalacaoTermos.png]

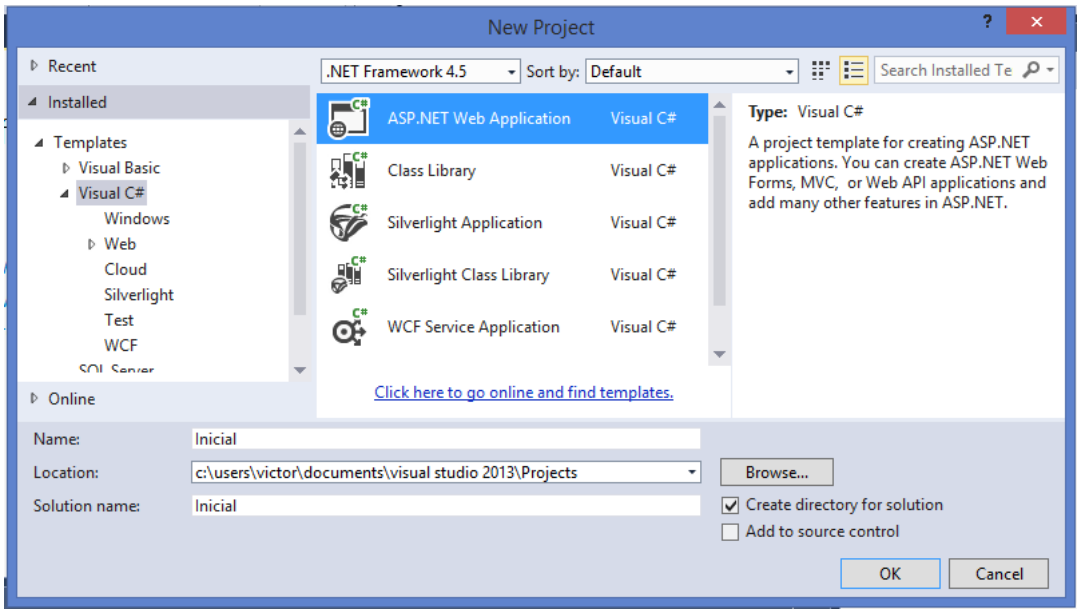
E depois clique no botão install. Com isso o instalador começará a fazer o download dos componentes adicionais do Visual Studio e executará a instalação em sua máquina.

[Imagem JanelaInstalacaoDownload.png]

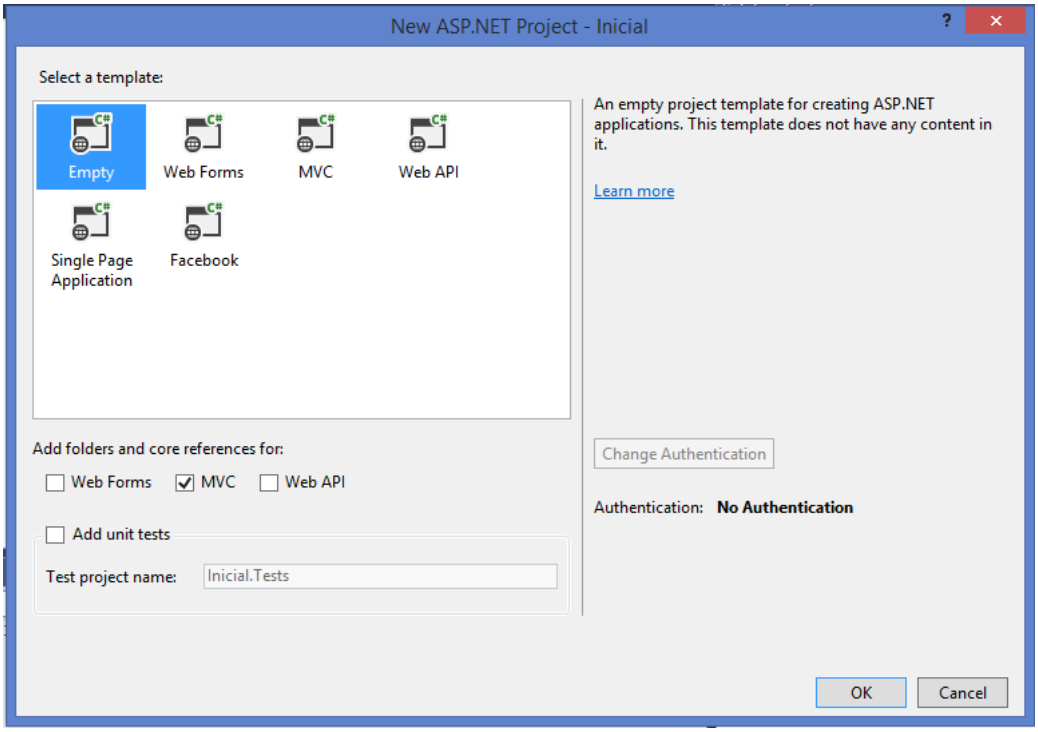
Quando a instação for concluída, execute o Visual Studio que acabou de ser instalado. Na primeira execução ele perguntará se você deseja se logar na sua conta da microsoft para compartilhar configurações entre máquinas. Se você quiser compartilhar as suas informações com a Microsoft, faça o login em sua conta, senão clique no link `Not now, maybe later .` Com isso estamos prontos para utilizar o Visual Studio para desenvolver nossas aplicações com o Asp.Net MVC.

Primeiro Projeto

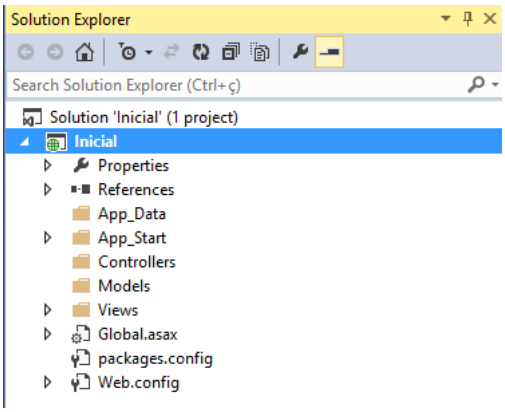
Vamos criar um projeto do ASP.NET MVC. Abra o Visual Web Developer e aperte `Ctrl+Shift+N` , a janela `New Project` será aberta, selecione a linguagem `Visual C# e ASP.NET Web Application .`



Na próxima janela, procure o campo `Select a template` e escolha a opção `Empty` , isso criará um projeto web vazio. Agora para que o projeto contenha as bibliotecas do Asp.Net MVC, procure o campo `Add folders and core references for` e nele marque a opção `MVC` e depois clique no botão `Ok` . Isso é tudo que precisamos para criar um novo projeto Asp.Net MVC 5!



Após criar o projeto, abra o solution explorer (Ctrl+Alt+L), repare que diversos arquivos e pastas foram automaticamente criados pela IDE. Essa é a estrutura de diretórios esperada de uma aplicação ASP.NET MVC. É importante conhecer essa estrutura, pois você precisará sempre seguir as convenções já pré-definidas. Essas convenções, aliás, são muito comuns ao longo do ASP.NET MVC: elas lhe pouparão muito trabalho!

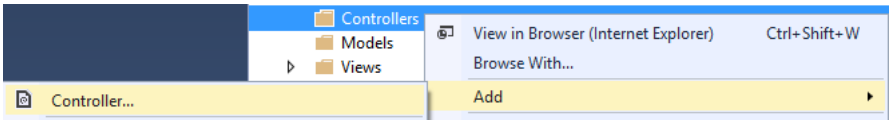


Entre as pastas criadas na estrutura inicial, duas são mais importantes. A pasta `Controllers` : é nela que guardaremos nossas classes responsáveis por tratar as requisições que vem do browser do usuário. É aqui que pegaremos os dados que o usuário nos envia através de formulários, por exemplo. E a pasta `Views` , onde os arquivos que serão utilizados para renderizar a resposta para o usuário são colocadas. Por exemplo, é aqui que guardaremos o HTML que renderiza o formulário.

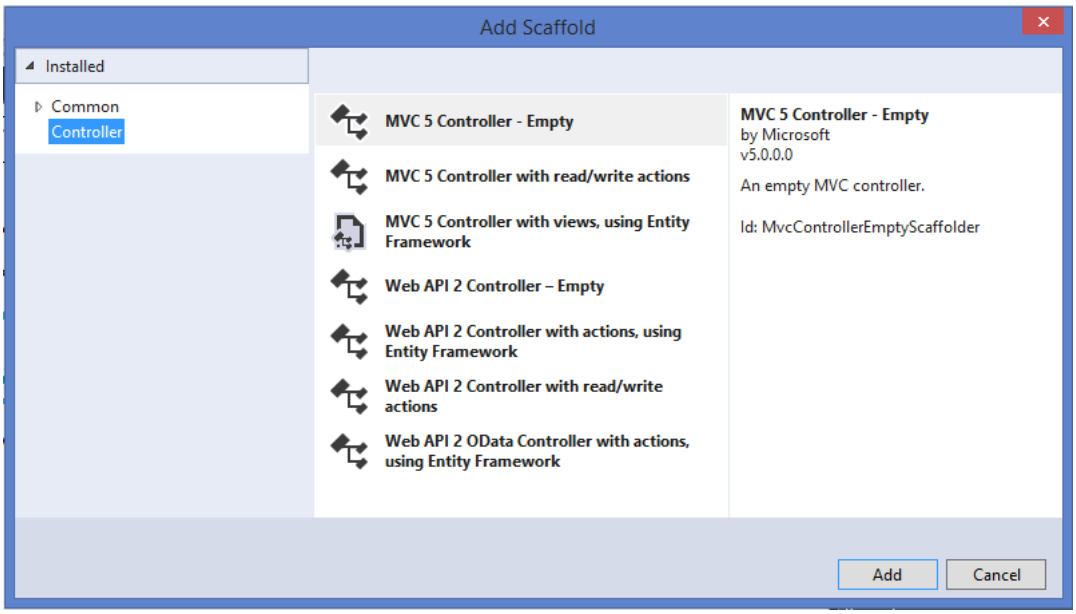
Primeira página do projeto

Agora que já aprendemos como criar o projeto inicial do Asp.Net MVC, vamos aprender como criar a página inicial da aplicação. Para isso, criaremos uma nova classe que será responsável por tratar todas as requisições para a página inicial. Uma classe que trata requisições web é chamada de **Controller** pelo Asp.Net MVC e deve ser criada dentro da pasta controllers da aplicação.

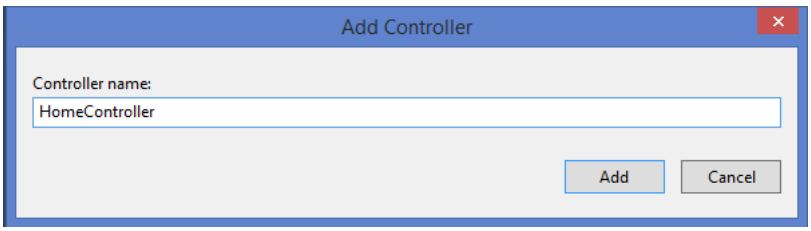
Podemos utilizar o próprio Visual Studio para criar o controller. Para isso, clique com o botão direito na pasta `Controllers` do projeto e selecione a opção `Add > Controller` :



Dentro da janela que é aberta, a `Add Scaffold` selecione a opção `MVC 5 Controller - Empty` e depois clique no botão `Add`.



Na próxima janela, precisamos dar um nome para a classe que será criada. Como esse é o controller inicial da aplicação, nós o chamaremos de `Home`, porém, por convenção, controllers do Asp.Net MVC tem o nome terminado pela palavra `Controller`, então o nome final da classe será `HomeController`.



Com isso, o Visual Studio criará uma nova classe na pasta `Controllers` do projeto com o seguinte código:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace Inicial.Controllers
{
    public class HomeController : Controller
    {
        //
        // GET: /Home/
        public ActionResult Index()
        {
            return View();
        }
    }
}
```

Veja que essa classe herda de uma classe chamada `Controller` do namespace `System.Web.Mvc`. Todo controller do Asp.Net MVC precisa herdar dessa classe.

Para atender requisições web, o Asp.Net MVC utiliza os métodos públicos implementados dentro dos controllers, esses métodos são conhecidos como **actions**. As actions do MVC sempre devolvem um objeto do tipo `ActionResult`. Para decidir qual é a action que deve ser utilizada para atender uma determinada requisição, o Asp.Net MVC utiliza um conjunto de regras conhecido como **Rota Padrão**.

Segundo a rota padrão, quando o navegador tenta utilizar uma url da forma `/Home/Index`, a primeira parte da url (`Home`) indica qual é o nome do controller que atenderá a requisição e a segunda parte indica o nome da action. Então nesse exemplo, a action `Index` do `HomeController` seria utilizada.

A rota padrão também possui valores padrão para os nomes do controller e da action. Quando não informamos o nome da action, a action `Index` é utilizada e quando não informamos o nome do controller, o `HomeController` é utilizado. Então podemos acessar o método `Index` do `HomeController` através de três endereços diferentes: `/Home/Index`, `/Home` e `/`.

Agora que já sabemos como criar e acessar controllers, precisamos aprender como implementar uma action e o como montar a resposta HTML que será devolvida para o navegador que acessou a aplicação.

Na implementação das actions, precisamos colocar o código que acessa as regras de negócio implementadas pela aplicação e, ao terminar de executar as lógicas de negócio, precisamos avisar o Asp.Net MVC que queremos executar a lógica de visualização da aplicação. Fazemos isso através do método `View` que é herdado da classe `Controller`:

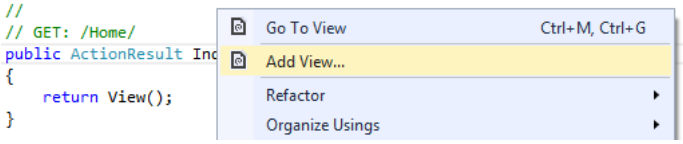
```
public ActionResult Index()
{
    // Chama a lógica de negócio da aplicação

    // Depois redireciona para a visualização
    return View();
}
```

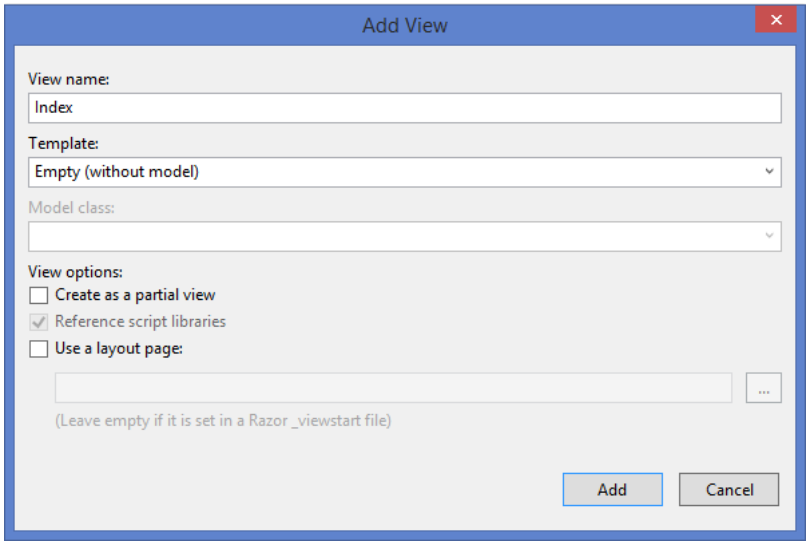
Para implementarmos a lógica de visualização da aplicação, precisamos seguir mais uma convenção do Asp.Net MVC. Arquivos de visualização possuem a extensão `cshtml` e devem ser colocados dentro da pasta `Views` da aplicação. Todas as views de um determinado controller, devem ficar dentro de uma subpasta de `Views` com o mesmo nome do controller, no caso do `HomeController`, todas as views seriam colocadas dentro da pasta `Views/Home`. O nome do arquivo deve ser `<nome da action>.cshtml`, `Index.cshtml` no caso do método `Index` do `HomeController`.

Dentro do `cshtml`, precisamos colocar todo o código HTML que será devolvido para o navegador. Todo o código colocado dentro desse arquivo é interpretado por um componente do Asp.Net MVC chamado **Razor**. Vamos então implementar o código da página inicial da aplicação. Para isso utilizaremos novamente um assistente do Visual Studio.

Clique com o botão direito do mouse sobre o método `Index` e selecione a opção `Add View`.



Dentro da janela que é aberta pelo Visual Studio, desmarque a opção `Use a layout page` e depois clique em `Add`.



Com isso o Visual Studio criará um novo arquivo na pasta `Views/Home` chamado `Index.cshtml` com o seguinte código:

```
@{
    Layout = null;
}

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Index</title>
</head>
<body>
    <div>
    </div>
</body>
</html>
```

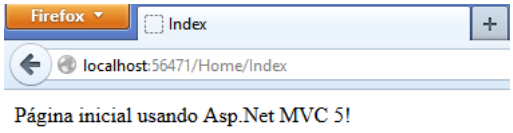
Vamos modificar esse arquivo para mostrar uma mensagem na página inicial da aplicação:

```
<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Index</title>
</head>
<body>
    Página inicial usando Asp.Net MVC 5!
</body>
</html>
```

Agora vamos apertar o `F5` para executar a aplicação no servidor do Visual Studio. Isso abrirá o navegador do seu sistema utilizando um endereço da forma: `http://localhost:<numero>` , ou seja, não estamos informando nem o controller e nem a action nessa url, o Asp.Net MVC executará a lógica do método `Index` do `HomeController` . A mesma lógica seria executada caso acessássemos as urls `http://localhost:<numero>/Home` ou `http://localhost:<numero>/Home/Index` .

Com isso terminamos a nossa primeira página de uma aplicação Asp.Net MVC 5!



Página inicial usando Asp.Net MVC 5!