Macoratti.net ASP .NET Core MVC 2 - Criando uma aplicação Desktop com Electron.NET

Neste artigo eu vou mostrar como criar uma aplicação ASP .NET Core 2 para Desktop usando os recursos do **Electron.NET.**

Aplicação ASP .NET Core 2 para desktop ????

Como assim !!!!

Isso mesmo vamos criar uma aplicação ASP .NET Core 2 para desktop usando o **Electron.NET.**

Mas quem é esse tal de Electron.NET ?

Esse tal de **Electron.NET** se baseia no **Electron** que permite você criar aplicações desktop multiplataforma usando JavaScript, Html e CSS.

Veja o que diz o site do **Electron**:

"Se você pode construir um website, você pode construir um aplicativo desktop. Electron é um framework para criação de aplicações nativas utilizando tecnologias web como JavaScript, HTML, e CSS. Ele cuida das partes mais difíceis para que você possa se concentrar somente com o principal da aplicação."

Veja detalhes no site : https://electronjs.org/

Assim a Electron.NET é uma biblioteca multiplataforma *(Windows, Linux e Mac)* para criar aplicativos de desktop usando JavaScript, HTML e CSS. Ela fornece um invólucro em torno do Electron com um aplicativo ASP .NET Core MVC.

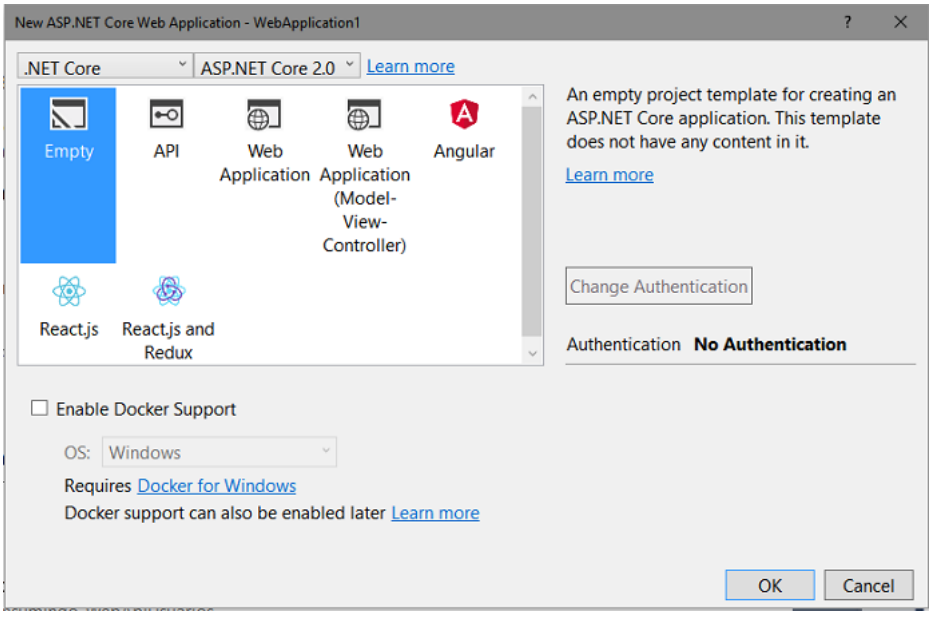
Nesse primeiro contato vamos criar uma aplicação ASP .NET Core MVC 2 para desktop que vai acessar o sistema local de arquivos e vai exibir e permitir executar arquivos mp3 e mp4.

'*Bora*' pra prática...

Artigo adaptado e traduzido do original: Electron.NET – Create a minimal MusicPlayer App with ASP.NET Core 2 for the Desktop

Criando uma aplicação ASP .NET Core MVC no VS 2017

Primeiro vamos criar uma aplicação ASP .NET Core usando o template **Empty** com o nome de **Electron_Demo**;



A seguir vamos incluir o middleware mvc no método **ConfigureServices** :

```
public void ConfigureServices(IServiceCollection services)
 {
     services.AddMvc();
 }
```

E configurar a rota no método **Configure**:

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
 {
     if (env.IsDevelopment())
     {
         app.UseDeveloperExceptionPage();
     }

     app.UseMvc(routes =>
     {
         routes.MapRoute(
             name: "default",
             template: "{controller=Home}/{action=Index}/{id?}");
     });
 }
```

Criando o controlador e a view Index

Vamos criar uma pasta chamada **Controllers** no projeto e nesta pasta incluir um controlador chamado **HomeController**.

Após criar a pasta via menu **Project -> New Folder ,** clique com o botão direito sobre a pasta **Controllers** e a seguir clique em **Add-> Controller;**

Selecione o template **MVC Controller - Empty** e clique em **Add**;

Informe o nome **HomeController** e clique em **Add**;

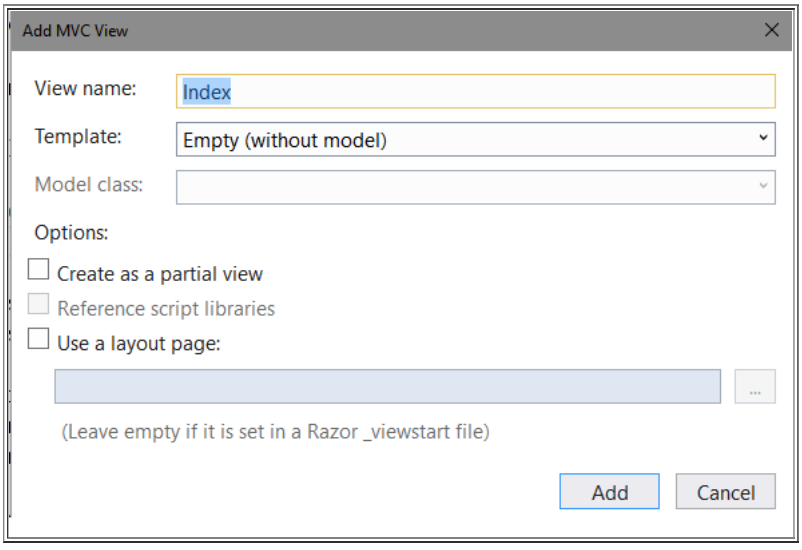
Informe o código abaixo :

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Threading.Tasks;
using ElectronNET.API;
using ElectronNET.API.Entities;
using Microsoft.AspNetCore.Mvc;

namespace Electron_Demo.Controllers
{
    public class HomeController : Controller
    {
         public IActionResult Index()
         {
               return View();
         }
    }
}
```

Clique com o botão direito sobre o método Action **Index** e a seguir clique em **Add View;**

Aceite os valores padrão da janela **Add MVc View** abaixo e clique em **Add**;



Será criada a view **Index.cshtml** na pasta **Views/Home** no projeto.

Instalando o pacote Electron.NET

Abra a janela do gerenciador de pacotes Nuget via menu **Tools-> Nuget Package Manager** e clique em **Package Manager Console;**

A seguir digite o comando para instalar o pacote Electron.NET : **Install-Package ElectronNET.API**

Com o pacote instalado altere o código do arquivo **Program.cs** conforme abaixo:

```
using ElectronNET.API;
using Microsoft.AspNetCore;
using Microsoft.AspNetCore.Hosting;

namespace Electron_Demo
{
    public class Program
    {
        public static void Main(string[] args)
        {
            BuildWebHost(args).Run();
        }

        public static IWebHost BuildWebHost(string[] args)
        {
            return WebHost.CreateDefaultBuilder(args)
                .UseElectron(args)
                .UseStartup<Startup>()
                .Build();
        }
    }
}
```

Pronto !

Agora podemos usar o **Electron.NET** em nosso projeto.

Para isso vamos incluir um código de inicialização no arquivo **Startup** para uma janela principal surja, além disso, queremos que a opção de segurança seja desativada. Isso nos permitirá acessar a lista de músicas.

Inclua o código destacado em azul conforme mostrado a seguir no arquivo **Startup**:

```
using ElectronNET.API;
using ElectronNET.API.Entities;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.Extensions.DependencyInjection;

namespace Electron_Demo
{
    public class Startup
    {
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddMvc();
        }

        public void Configure(IApplicationBuilder app, IHostingEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }

            app.UseMvc(routes =>
            {
                routes.MapRoute(
                    name: "default",
                    template: "{controller=Home}/{action=Index}/{id?}");
            });
            Bootstrap();
        }
```

```
        public async void Bootstrap()
        {
            var options = new BrowserWindowOptions
            {
                WebPreferences = new WebPreferences
                {
                    WebSecurity = false
                }
            };

            await Electron.WindowManager.CreateWindowAsync(options);
        }

    }
}
```

Agora vamos criar a aplicação para acessar as músicas do sistema de arquivos.

Criando a aplicação ASP .NET Core

Vamos alterar o código do método Index do controlador **HomeController** conforme abaixo:

```
using ElectronNET.API;
using ElectronNET.API.Entities;
using Microsoft.AspNetCore.Mvc;
using System.Collections.Generic;
using System.IO;
using System.Threading.Tasks;

namespace Electron_Demo.Controllers
{
    public class HomeController : Controller
    {
        public async Task<IActionResult> Index()
        {
            string caminhoMusica = await Electron.App.GetPathAsync(PathName.music);
            string[] arquivos_Mp3 = Directory.GetFiles(caminhoMusica, "*.mp3", SearchOption.TopDirectoryOnly);
            string[] arquivos_Mp4 = Directory.GetFiles(caminhoMusica, "*.mp4", SearchOption.TopDirectoryOnly);

            List<string> arquivosMusicas = new List<string>();
            arquivosMusicas.AddRange(arquivos_Mp3);
            arquivosMusicas.AddRange(arquivos_Mp4);

            return View(arquivosMusicas);
        }
    }
}
```

Neste código estamos acessando o sistema de arquivos e obtendo uma lista de músicas **mp3 e mp4** e retornando para a View Index.

Agora vamos alterar o código da View **Index** para receber a lista de arquivos e exibí-los com um botão para tocar a música.

Abra o arquivo **Index.cshtml** e altere o seu código como abaixo:

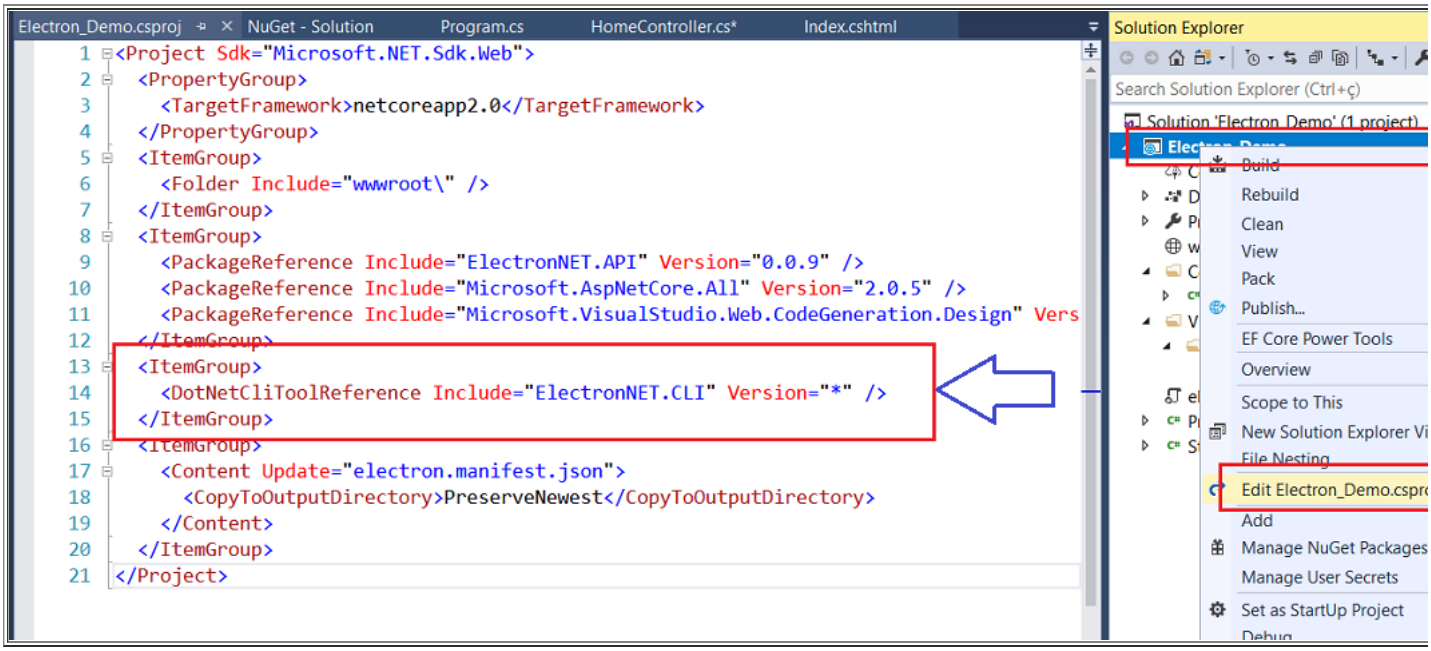
```
@model List<string>
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Meu Tocador de Músicas Electron</title>
    <meta charset="utf-8" />
</head>
<body>
    <h3>Minhas Músicas</h3>
    <ul>
        @foreach (string arquivoMusica in Model)
        {
            <li>
```

```
            @System.IO.Path.GetFileNameWithoutExtension(arquivoMusica )
            <button onclick="playMusic('@arquivoMusica.Replace(@"\", "/")')">Tocar</button>
        </li>
    }
</ul>
<script>
    function playMusic(arquivoMusica) {
        let audio = new Audio(`file:///${arquivoMusica}`);
        audio.play();
    }
</script>
</body>
</html>
```

No código da View percorremos a lista de arquivos recebidos e exibímos uma relação de músicas com um botão para tocar a música usando a classe audio do HTML5.

Para poder executar o projeto usando o Electron.NET precisamos usar o Electron.NET CLI e para isso temos que editar o arquivo de projeto **Electron_Demo.csproj** e incluir uma referência a **ElectronNET.CLI** no projeto:

Para isso oclique com o botão direito do mouse sobre o projeto e a seguir clique em **Edit Electron_Demo.csproj** e inclua o código destacado abaixo:

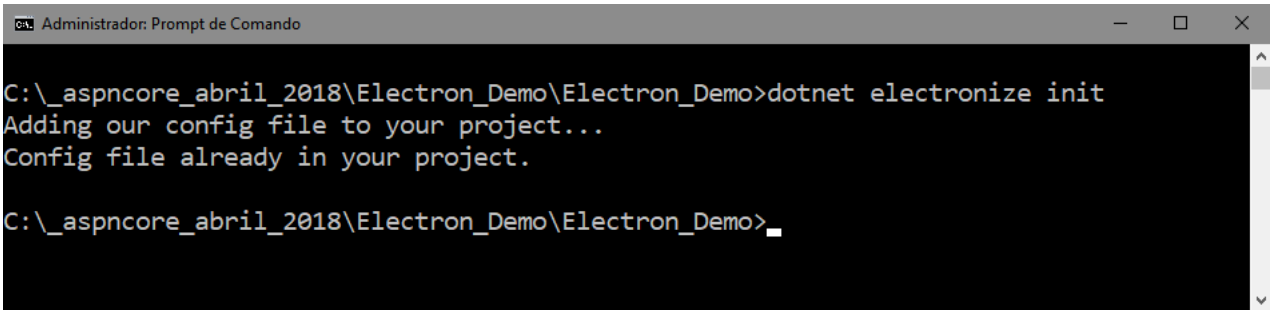



```
<Project Sdk="Microsoft.NET.Sdk.Web">
  <PropertyGroup>
    <TargetFramework>netcoreapp2.0</TargetFramework>
  </PropertyGroup>
  <ItemGroup>
    <Folder Include="wwwroot\" />
  </ItemGroup>
  <ItemGroup>
    <PackageReference Include="ElectronNET.API" Version="0.0.9" />
    <PackageReference Include="Microsoft.AspNetCore.All" Version="2.0.5" />
    <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Vers
  </ItemGroup>
  <ItemGroup>
    <DotNetCliToolReference Include="ElectronNET.CLI" Version="*" />
  </ItemGroup>
  <ItemGroup>
    <Content Update="electron.manifest.json">
      <CopyToOutputDirectory>PreserveNewest</CopyToOutputDirectory>
    </Content>
  </ItemGroup>
</Project>
```

Agora é só alegria...

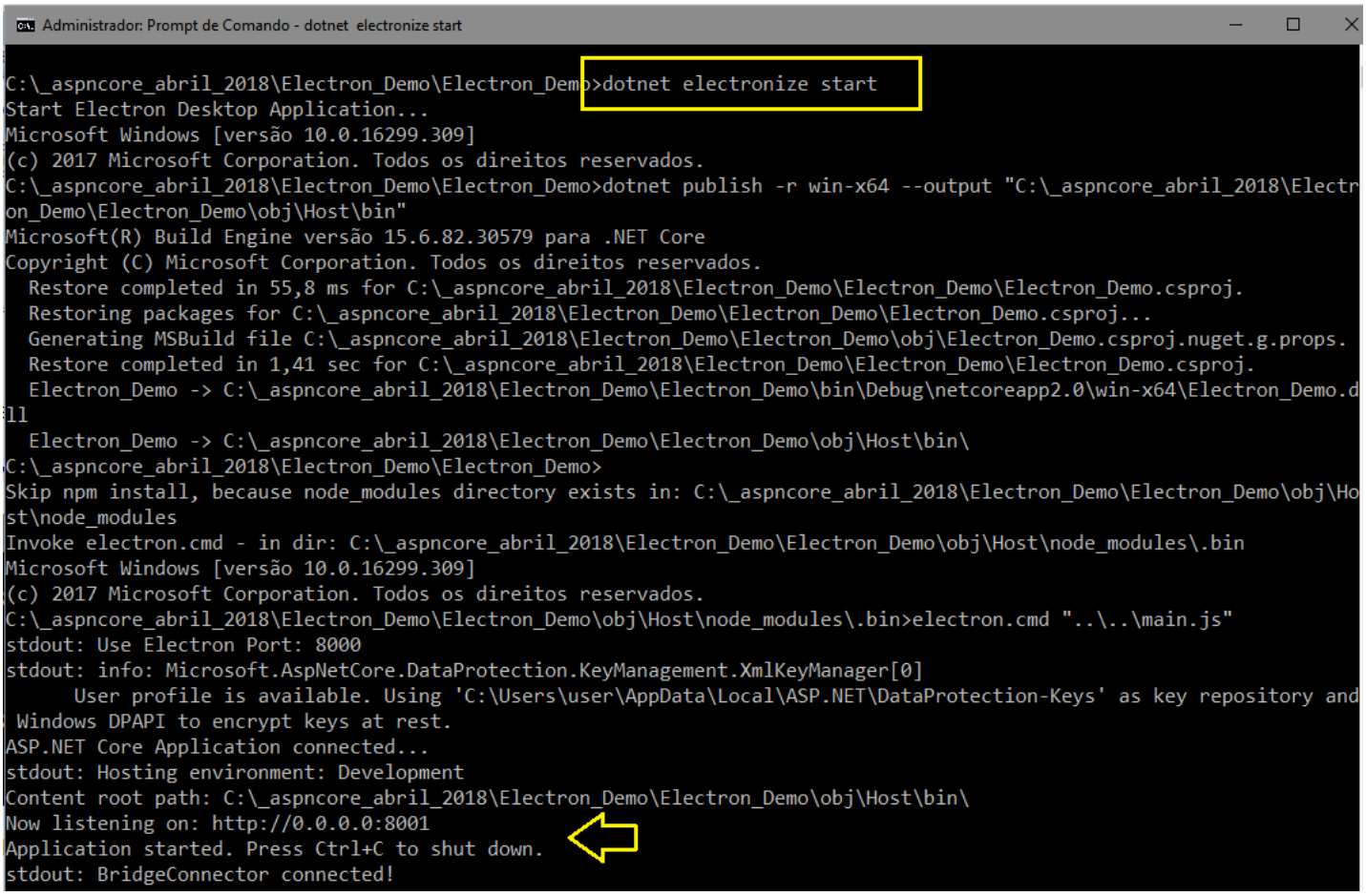
Na primeira vez que vamos executar o projeto temos que abrir uma janela de comandos na pasta do projeto e digitar:

dotnet electronize init

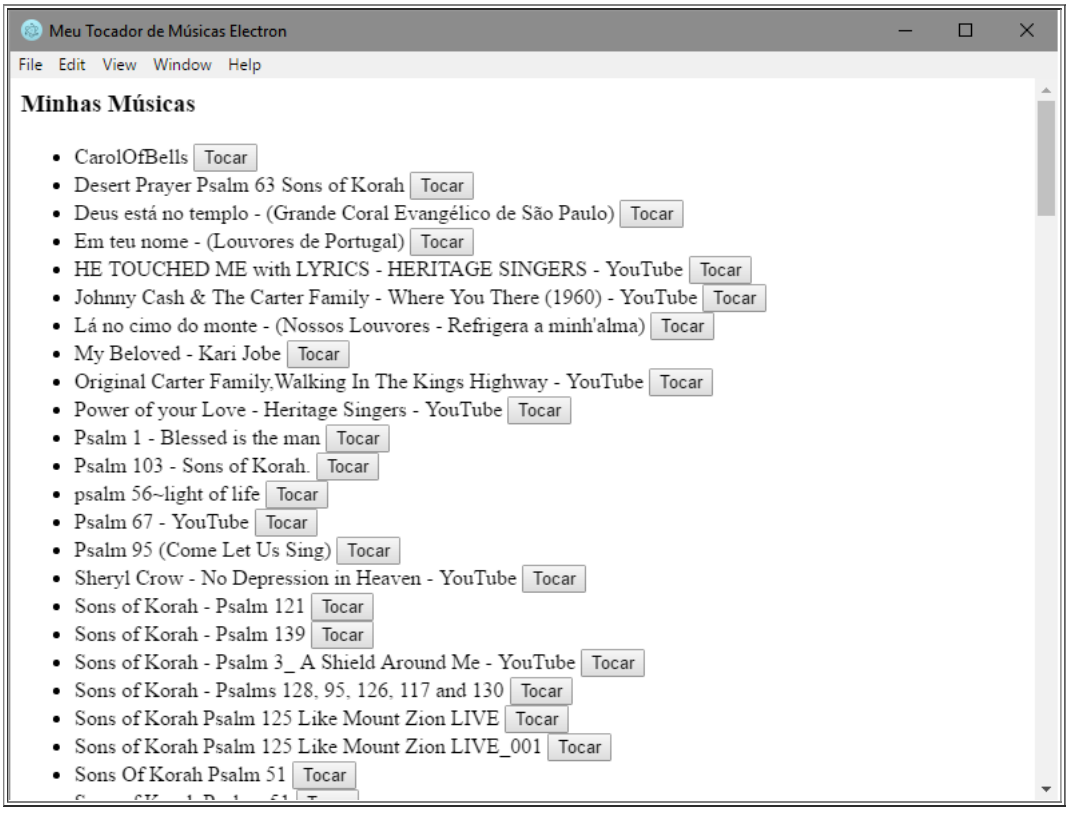


Para executar a aplicação ASP .NEt Core MVC 2 como uma aplicação desktop digite o comando :

dotnet electronize start



A seguir você verá a aplicação exibindo a relação de arquivos mp3 e mp4 e o botão **Tocar**:



Você acabou de criar sua primeira ASP .NET Core MVC 2 como uma aplicação **desktop** usando o Electron.NET

Parabéns...

Pegue o projeto aqui : **Electron_Demo.*zip*** *(sem as referências...)*

Até o próximo artigo...

"E disse-lhes: Ide por todo omundo, pregai o evangelho a toda criatura. Quem crer e for batizado será salvo; mas quem não crer será condenado."
Marcos 16:15,16

Gostou ? Compartilhe no Facebook Compartilhe no Twitter

Referências:

- Seção VB .NET do Site Macoratti.net
- Super DVD .NET - A sua porta de entrada na plataforma .NET
- Super DVD Vídeo Aulas - Vídeo Aula sobre VB .NET, ASP .NET e C#
- Seção C# do site Macoratti.net
- Super DVD C#
- Super DVD Visual Basic
- Curso Básico VB .NET - Vídeo Aulas
- Curso C# Básico - Vídeo Aulas
- ASP .NET Core - Criando uma aplicação com Angular 2 - Macoratti.net
- ASP .NET Core - Criando uma aplicação Web no ... - Macoratti.net
- ASP .NET Core - Macoratti.net
- ASP .NET Core - Iniciando com ASP .NET Core MVC e ... - Macoratti

---

José Carlos Macoratti