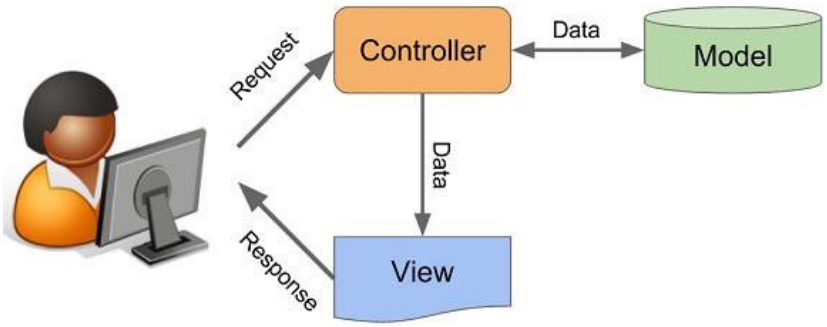
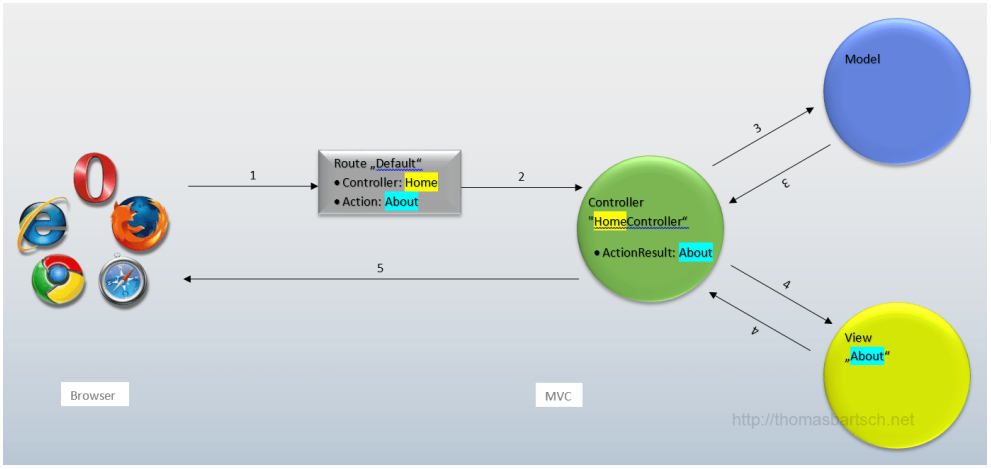
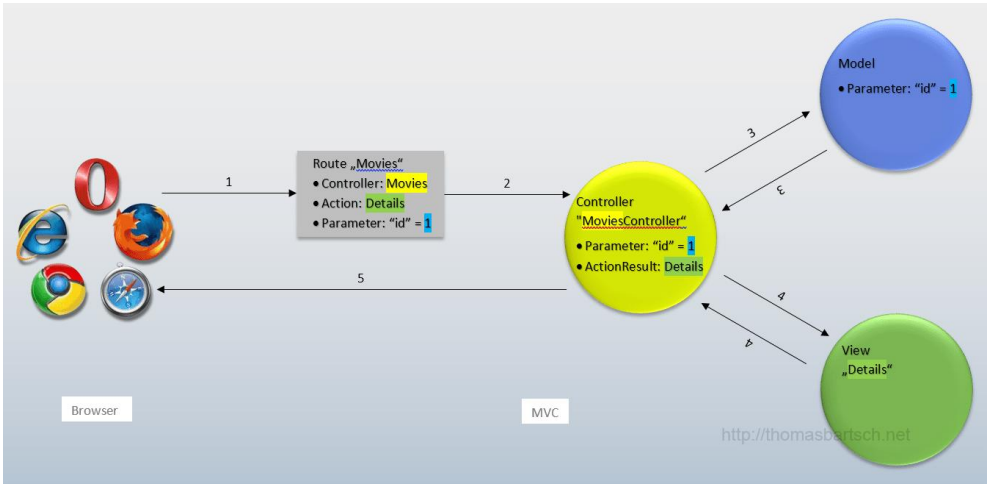
:: ASP.NET Core MVC :: Criar projeto de raiz ::

Objetivo:

- Criar um projeto Web ASP.NET Core MVC de raiz, sem usar o template predefinido.
- Explorar a anatomia de uma aplicação ASP.NET Core
- Configurar a aplicação
- TagHelper

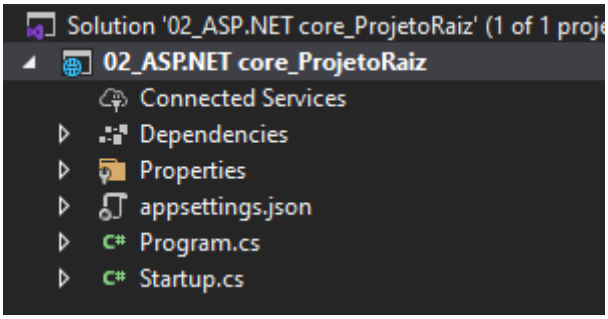






ASP.NET Core Empty

1. Create a new Project
2. ASP.NET Core Empty
3. Definir nome, exemplo: 02_ASP.NET core_ProjetoRaiz
4. Escolha de Framework e outros parâmetros: .NET 5.0 (Current)



Todas as aplicações ASP .NET Core são aplicações console, e, os outros tipos de aplicações como MVC, SPA, etc, são construídas sobre a aplicação de Console.

A aplicação console inicia com a execução do ficheiro *Program.cs*, que tem o método estático *Main* que é chamado sempre que a aplicação é iniciada.

Program.cs

```
namespace _02_ASP.NET_core_ProjetoRaiz
{
    public class Program
    {
        public static void Main(string[] args)
        {
            CreateHostBuilder(args).Build().Run();
        }

        public static IHostBuilder CreateHostBuilder(string[] args) =>
            Host.CreateDefaultBuilder(args)
                .ConfigureWebHostDefaults(webBuilder =>
                {
                    webBuilder.UseStartup<Startup>();
                });
    }
}
```

2. Faz Build e executa

1. Cria o Host

O método *UseStartup* informa ao host onde procurar pela classe *Startup*. O host, então, espera que a classe de inicialização defina o método de serviços *Configure* e *ConfigureServices*.

A classe *Startup* é uma classe simples, com funções principais:

- Configurar o pipeline de requisições que lida com todas as requisições feitas à aplicação;

- Configurar os serviços para injeção de dependência;

Para realizar essas tarefas ela usa os métodos *ConfigureServices*() e *Configure*().

Configurar aplicação para: ASP.NET Core MVC

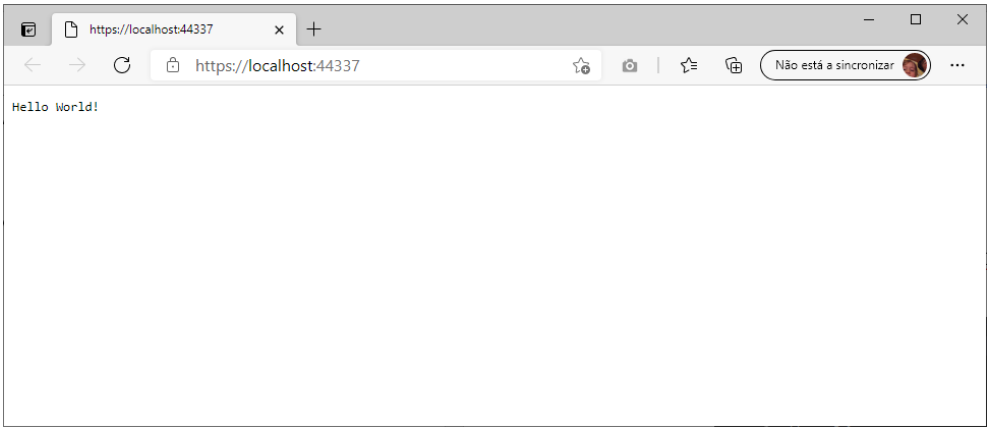
Startup.cs

```
namespace _02_ASP.NET_core_ProjetoRaiz
{
    public class Startup
    {
      // This method gets called by the runtime. Use this method to add services to the container.
      // For more information on how to configure your application, visit https://go.microsoft.com/fwlink/?LinkID=398940
        public void ConfigureServices(IServiceCollection services)
        {
        }

        // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
        public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }

            app.UseRouting();

            app.UseEndpoints(endpoints =>
            {
                endpoints.MapGet("/", async context =>
                {
                        await context.Response.WriteAsync("Hello World!");
                });
            });
        }
    }
}
```

5. Executar

   Debug | **Start Debugging** ou **Start Without Debugging**

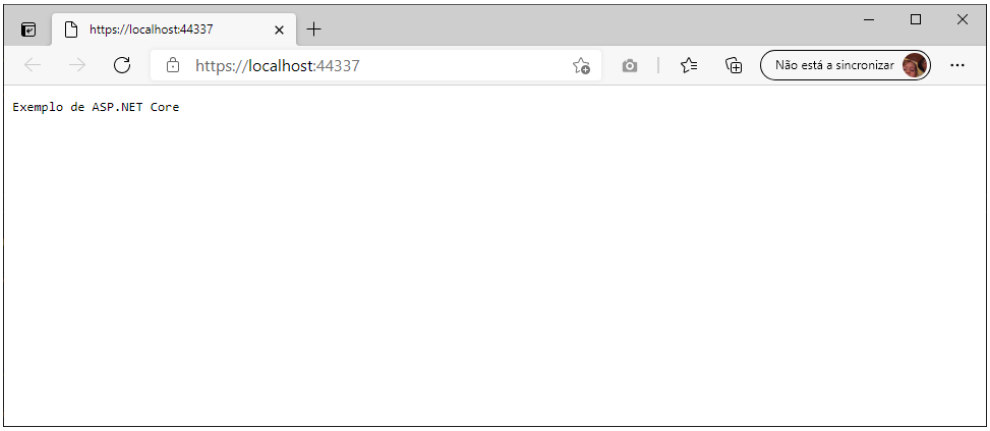
   F5 (executa com debug) ou **CTRL F5** (executar sem debug)



6. Startup.cs

Alterar para:

```
await context.Response.WriteAsync("Exemplo de ASP.NET Core");
```



Executar novamente, ou fazer refresh da página

7. Configurar aplicação para: ASP.NET Core MVC

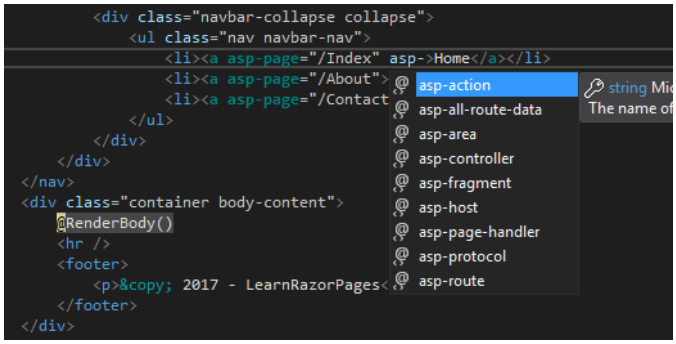
Startup.cs

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


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

    app.UseRouting();
    app.UseStaticFiles();

     app.UseEndpoints(endpoints =>
     {
         endpoints.MapControllerRoute(
             name: "default",
             pattern: "{controller=Home}/{action=Index}/{id?}");
     });

 }
```

Taghelpers

Os Tag helpers são componentes para automatizar a geração de HTML nas páginas do Razor.

Os Tag helpers direcionam para tags HTML específicas. Substitui tags/atributos do HTML.



Ativar os TagHelpers

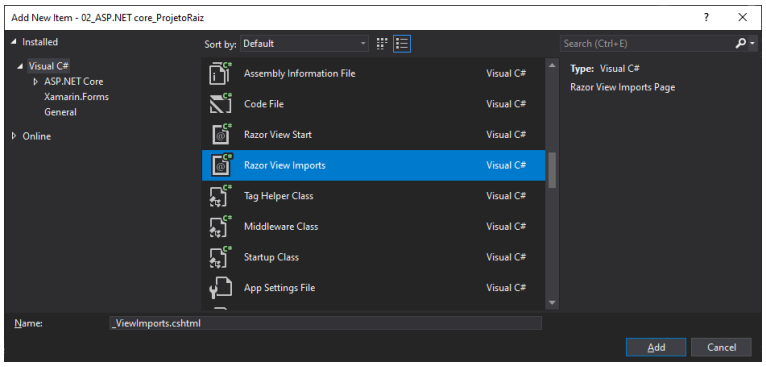
Para ativar os TagHelpers, adicionar a diretiva `@addTagHelper` ao ficheiro `_ViewImports.cshtml` dentro da pasta View

```
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
```

Criar o ficheiro `_ViewImports.cshtml`

Pasta Views (tecla direita do rato)

Add | New Item – Razor e escolher



Adicionar a diretiva:

```
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
```

Definições globais

Para definir de forma global, que todas as páginas usem o layout (templates, masterpage) podemos definir dentro da `Views\_ViewStart.cshtml`

Views_ViewStart.cshtml

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

O exemplo anterior pressupõe a existência na pasta Views do ficheiro `Views\Shared\_Layout.cshtml` com a definição do layout (template, masterpage) comum a várias páginas.

Para ativar o uso de tagHelpers (substitui tags/atributos do HTML, por exemplo, `asp-controller`, `asp-action`) podemos definir dentro da `Views\_ViewImports.cshtml`

Views_ViewImports.cshtml

```
@using _01_PrimeiraAPPaspNetCoreMVC
@using _01_PrimeiraAPPaspNetCoreMVC.Models
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
```