

MBA⁺

MBA em Arquitetura e Desenvolvimento na Plataforma .NET

MBA⁺

Desenvolvimento Web

Profª . Rodolfo Fadino
 EMAIL rodolfo.fadino@gmail.com

2018

- Head de Tecnologia e sócio – tech.fit (Dieta e Saúde e Tecnonutri)
- Tecnologia em Processamento de Dados. FATEC-SP
- Microsoft MVP - Visual Studio and Development Technologies (2014-2018)
- Eventos:
 - Visual Studio Summit
 - TDC - The Developer Conference
 - ASP.NET Brasil
 - DevXperience
 - MVP Summit
- <https://rodolfofadino.com.br>
- +55 11 98040-7831 (whatsapp)
- rodolfo.fadino@gmail.com





Web



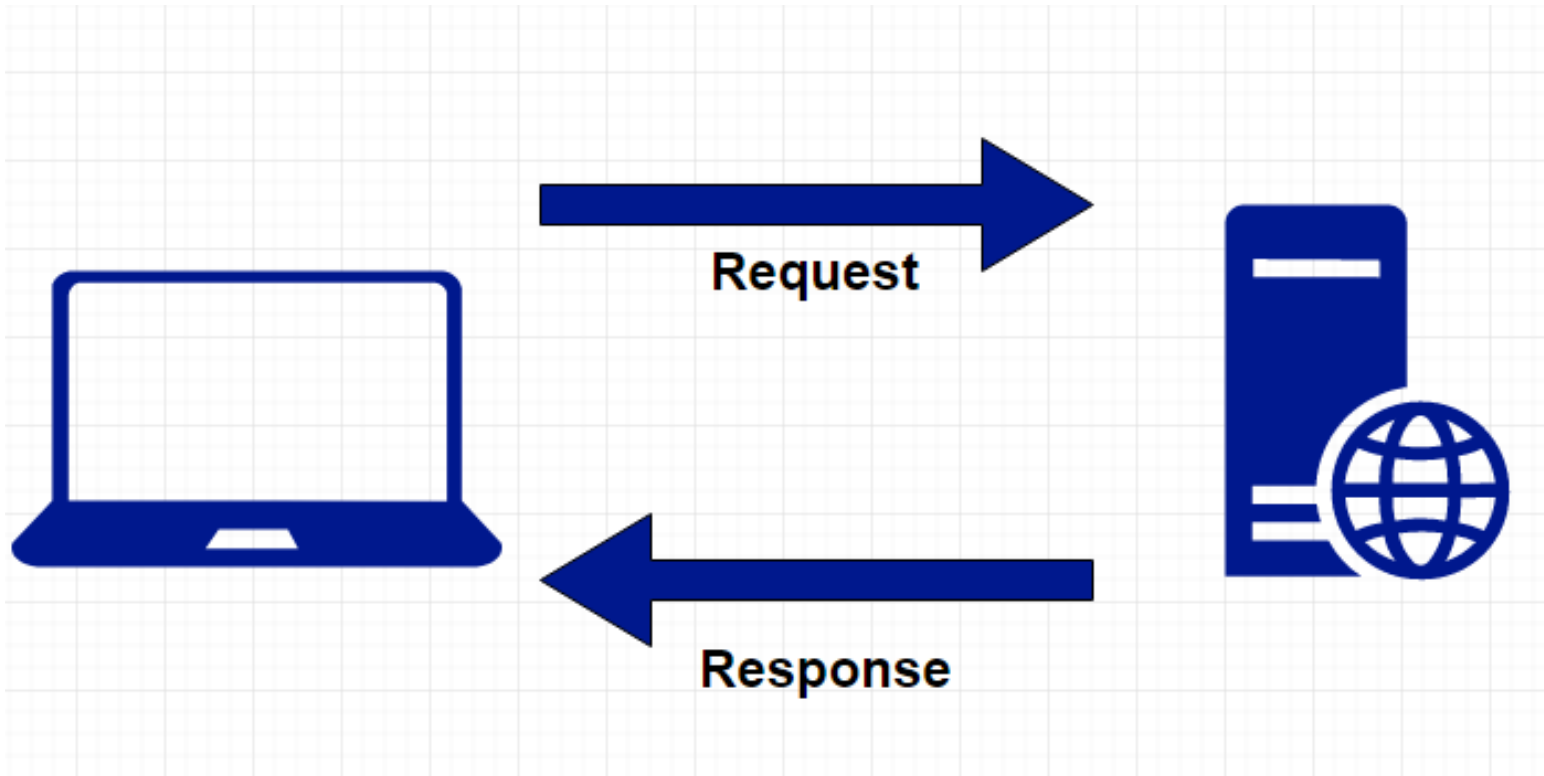
THIS IS THE WEB.



THIS WILL BE THE WEB.




Request e Response



Anatomia de uma Requisição HTTP

- DNS Lookup
- Initial Connection
 - Keep-Alive Header
- Time to First Byte
- Content Download

1: www.google.com.br - /



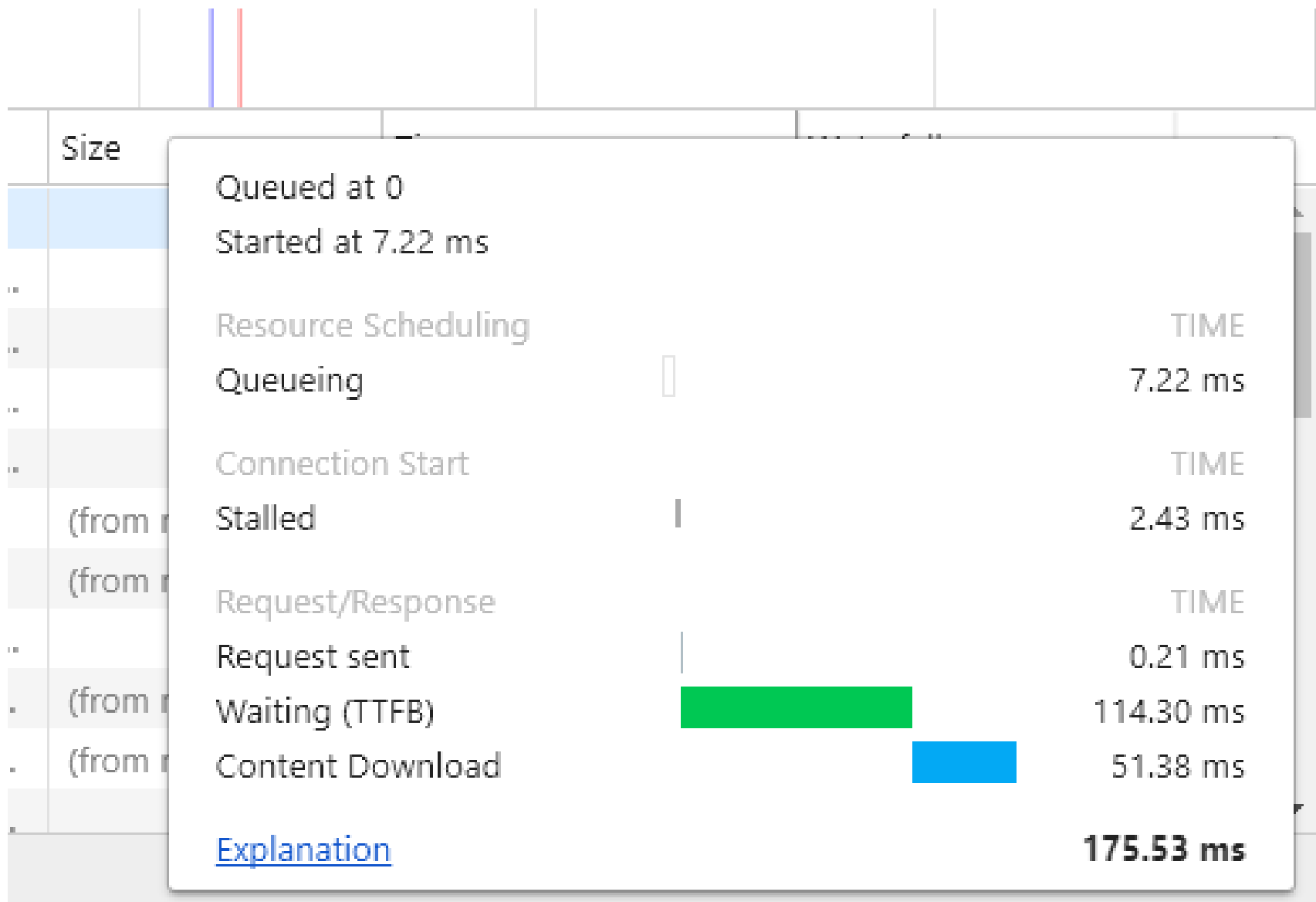
380 ms

Figura 3: Requisição

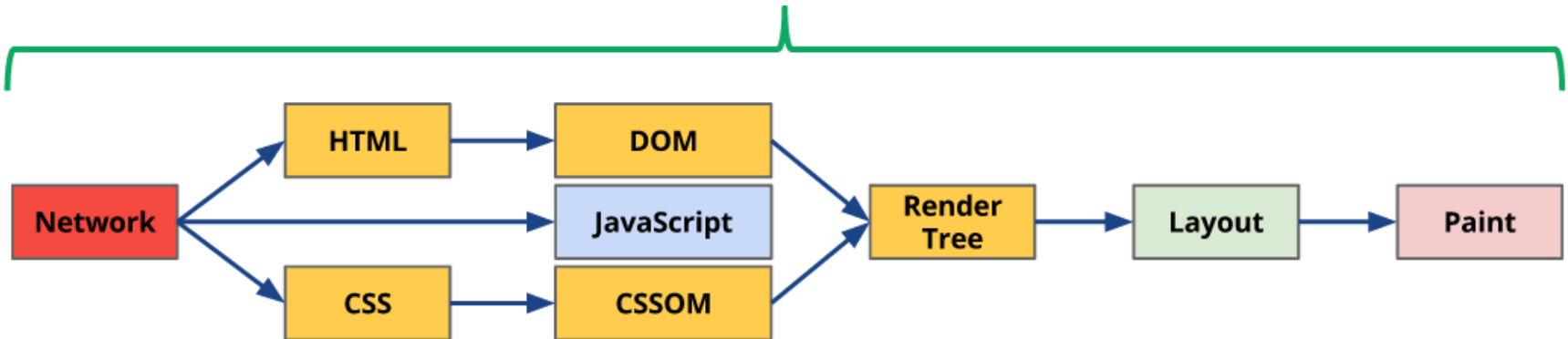
HTTP

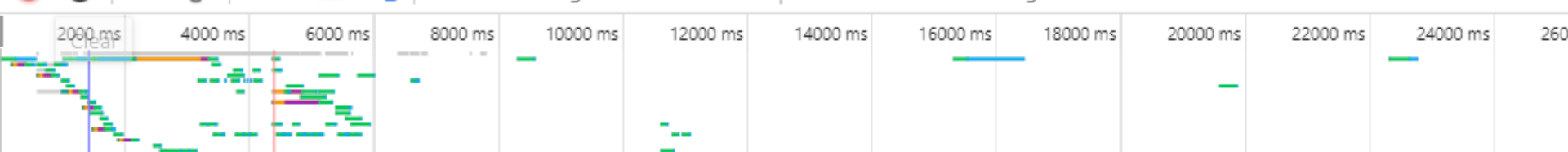


Anatomia de uma Requisição HTTP



Browser Rendering





Name	Status	Type	Initiator	Size	Time	Waterfall	10.00 s	15.00 s
www.uol.co...	200	docu...	Other	46.1 KB	365 ms			
detectadblo...	200	script	(index)	(from ...	8 ms			
home.css?v...	200	stylesh...	(index)	39.2 KB	134 ms			
uolhome.js	200	script	(index)	18.0 KB	349 ms			
adframe.js?v...	200	script	(index)	260 B	136 ms			
chartbeat_m...	200	script	(index)	6.4 KB	132 ms			
home.js?v=...	200	script	(index):13	136 KB	198 ms			
uoltm.js?id=...	200	script	(index):13	33.9 KB	415 ms			
b?c1=2&c2...	204	text/pl...	Other	248 B	106 ms			
collect?v=1...	200	gif	Other	63 B	105 ms			
uol-icones-...	200	font	(index)	(from ...	6 ms			
uol-text-reg...	200	font	(index)	(from ...	16 ms			
uollogo.wof...	200	font	(index)	(from ...	23 ms			
sprites-gera...	200	png	(index)	23.7 KB	343 ms			
data:image/...	200	gif	Other	(from ...	0 ms			
uol-text-bol...	200	font	(index)	(from ...	8 ms			
uol-text-lig...	200	font	(index)	(from ...	16 ms			
data:image/...	200	gif	(index)	(from ...	0 ms			
data:image/...	200	gif	(index)	(from ...	0 ms			
data:image/...	200	gif	(index)	(from ...	0 ms			
data:image/...	200	gif	(index)	(from ...	0 ms			
data:image/...	200	gif	(index)	(from ...	0 ms			

364 requests



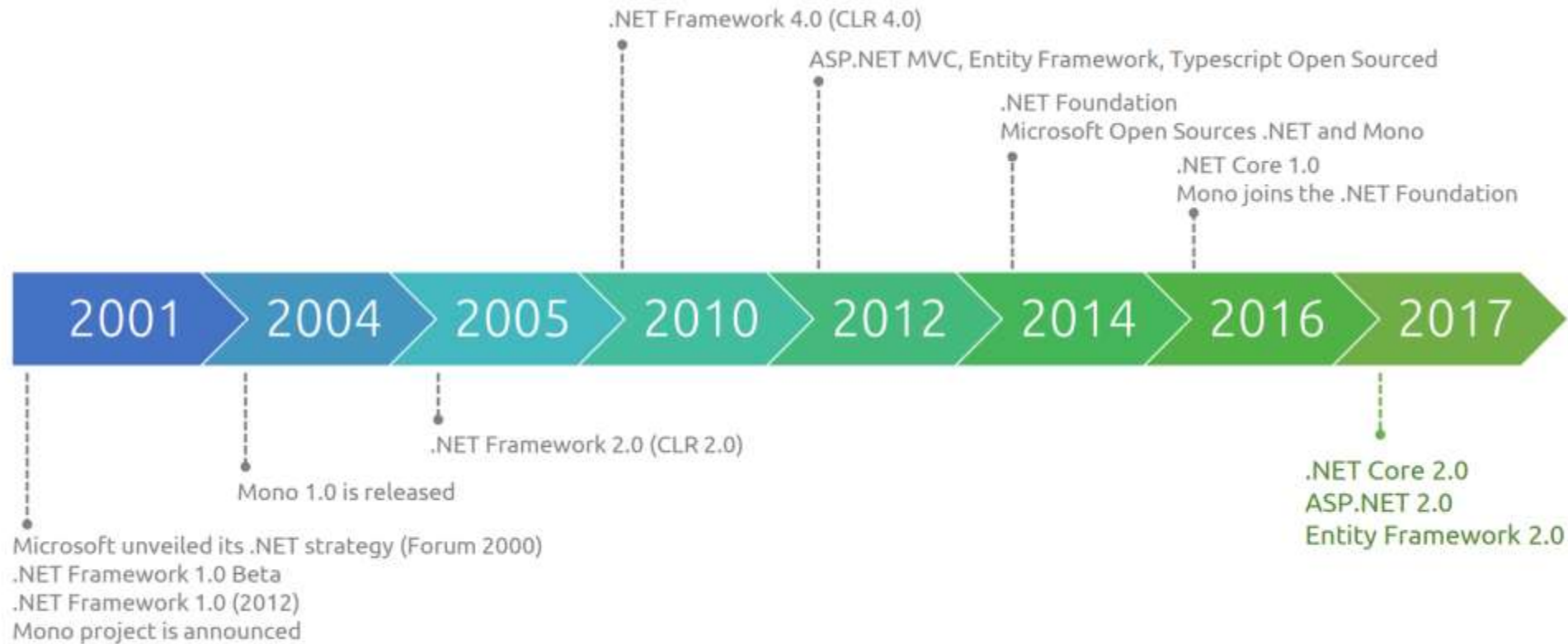
Escolha 4 sites que
você gosta e utiliza, analise:

- número de requisições
- tempo de carregamento
- tamanho



Vamos desenvolver para Web o/

História



System.Web Namespaces

.NET Framework (current version) | [Other Versions](#) ▼

The System.Web namespaces contain types that enable browser/server communication. Child namespaces include types that support ASP.NET forms authentication, application services, data caching on the server, ASP.NET application configuration, dynamic data, HTTP handlers, JSON serialization, incorporating AJAX functionality into ASP.NET, ASP.NET security, and web services.

- WebForms, MVC 1-4
- Performance
- Dificuldade de evoluir
- Manter compatibilidade

.NET Core

Cross-platform

Open source

Arquitetura de micro serviços

Containers

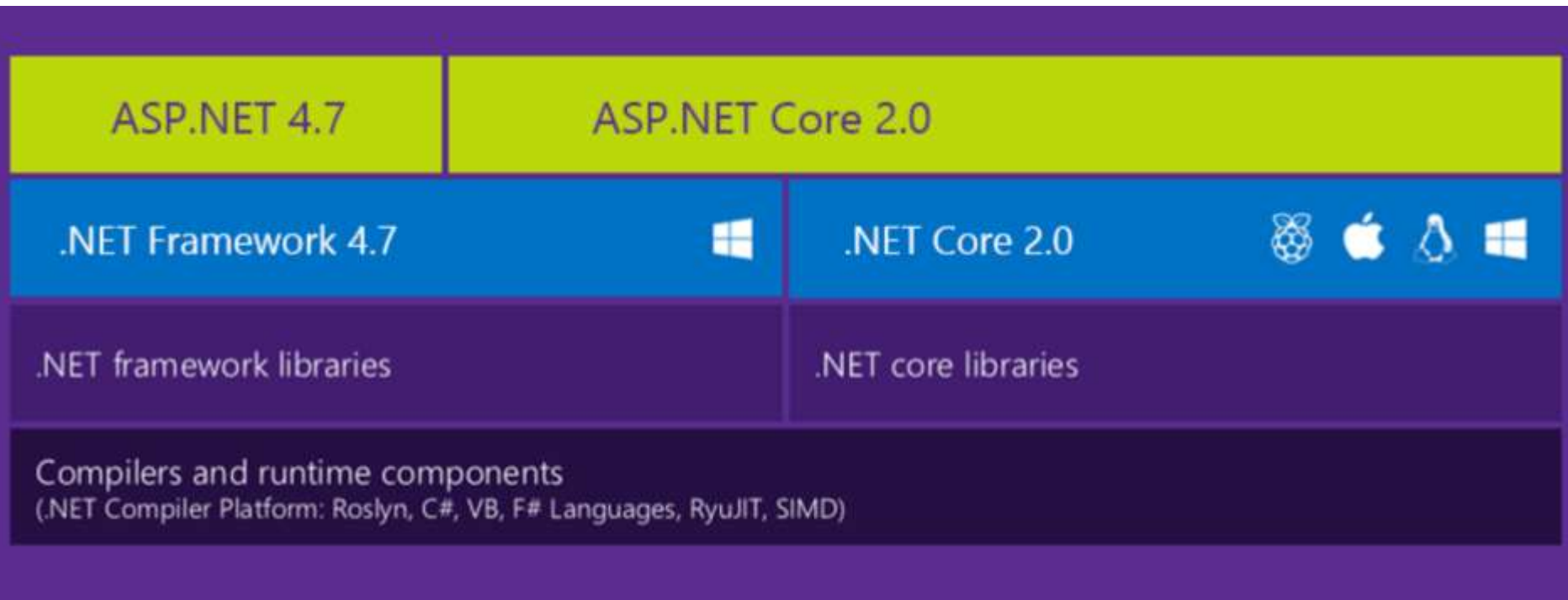
Design Modular

Variedade de ferramentas de desenvolvimento

Alta performance e escalabilidade



ASP.NET 4.7 e ASP.NET Core 2.0



Downloads

❓ Not sure where to start? See [Get started with .NET in 10 minutes.](#)

Windows

Linux

macOS

Build Apps

Run Apps

.NET
Core



.NET Core SDK

The smallest download to build .NET apps, using command line tools and any editor.

[Download .NET Core SDK 2.1.101](#)

Visual Studio

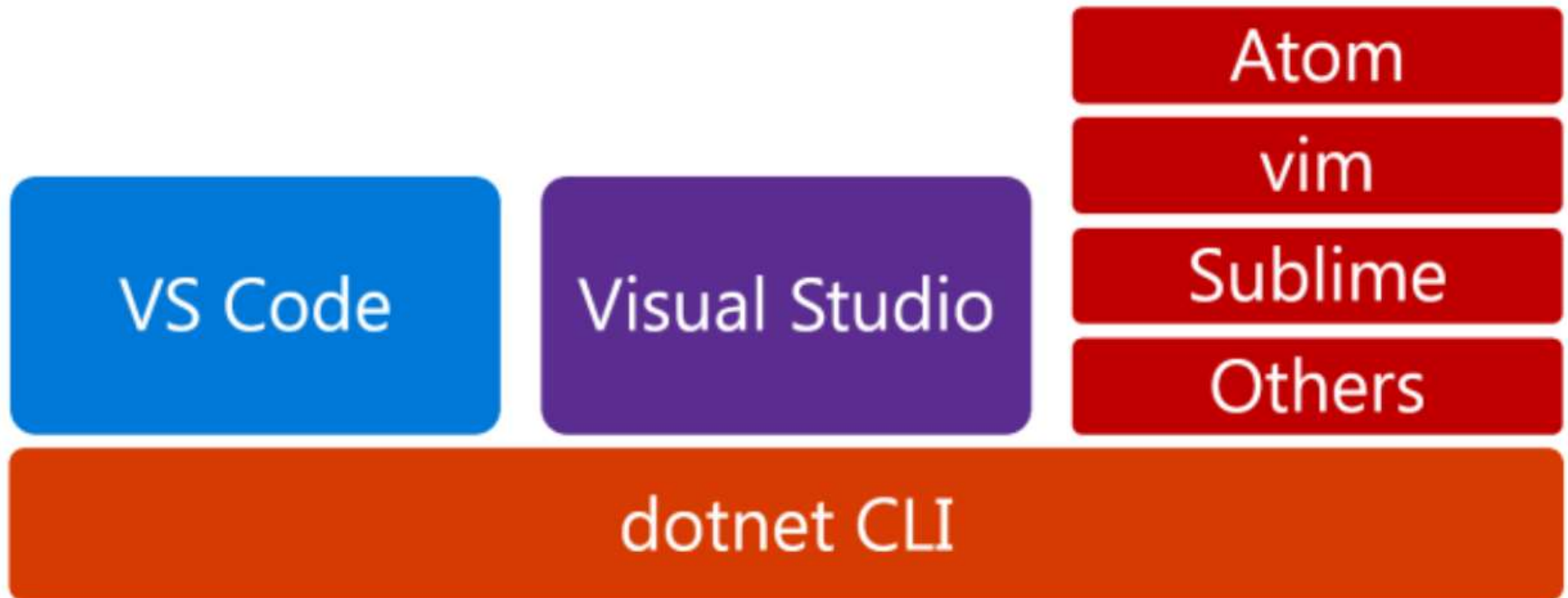
A fully-featured editor for developing .NET apps on Windows. Includes .NET Core and .NET Framework.

[Download Visual Studio](#)

- Sublime
- Visual Studio
- Visual Studio Code
- Atom
- Notepad ++
- Vi

.NET Core command-line interface (CLI) tools

FIAP



.NET Core command-line interface (CLI) tools

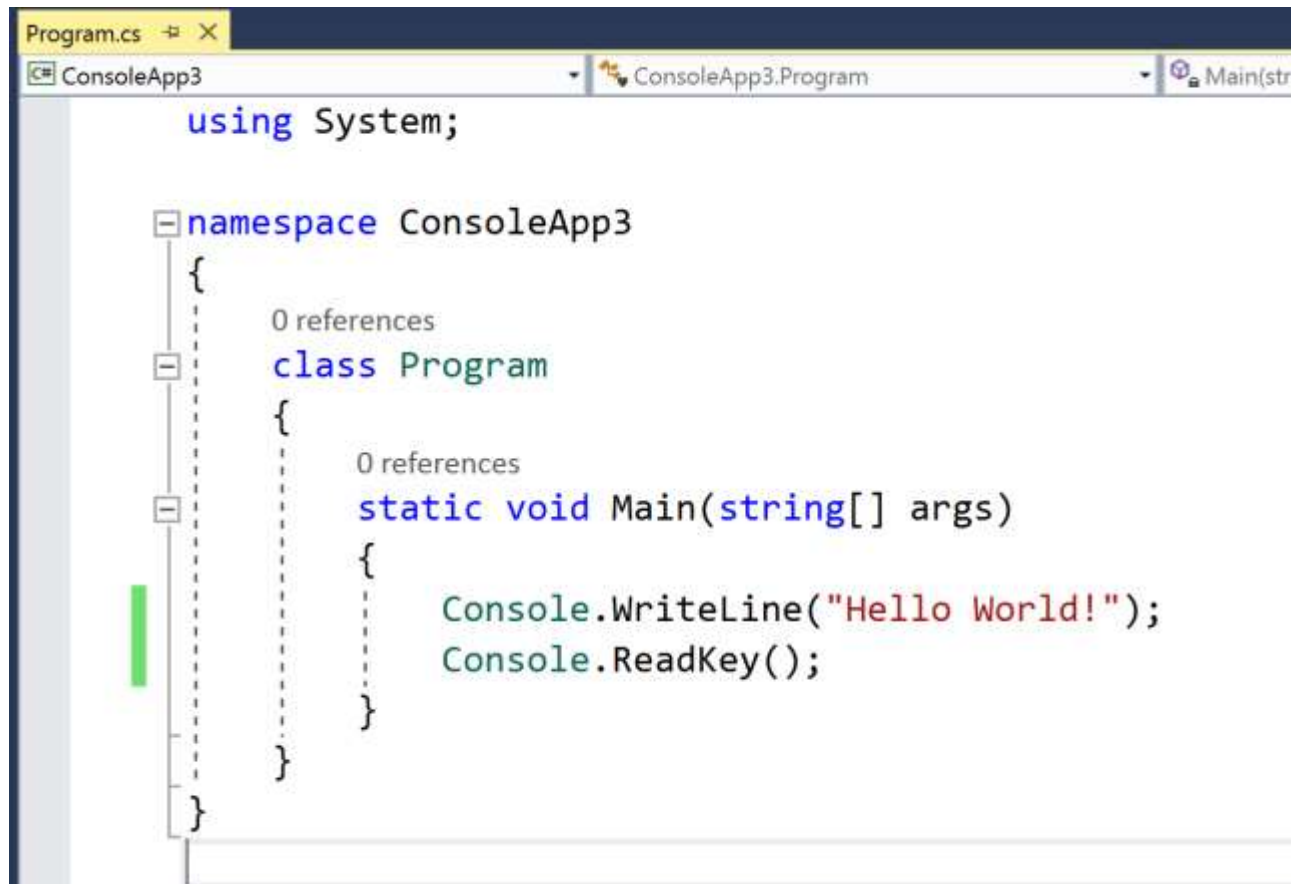
- new
- restore
- build
- publish
- run
- test
- vstest
- pack
- migrate
- clean
- sln
- help
- store

Criando um projeto

- `dotnet new`
- `dotnet restore`
- `dotnet build`

Console => Web Site

- Vamos criar um console em .NET Core



```
Program.cs
using System;

namespace ConsoleApp3
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
            Console.ReadKey();
        }
    }
}
```

Console => Web Site

Install-package Microsoft.ASPNETCore.All

ou

dotnet add package Microsoft.ASPNETCore.All

Console => Web Site

Adicionamos o método que será responsável por criar um WebHost no Program.cs, notem que precisaremos de uma classe Startup

1 reference

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

Console => Web Site

Na criação do WebHost foi utilizado uma classe chamada Startup como configuração, é nela que iremos colocar o que nossa app terá no pipeline para responder as requests.

1 reference

```
public class Startup
```

```
{
```

0 references

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
```

```
{
```

```
    app.Run(async (context) =>
```

```
    {
```

```
        await context.Response.WriteAsync("Olá Fiap!!!");
```

```
    });
```

```
}
```

```
}
```

Console => Web Site

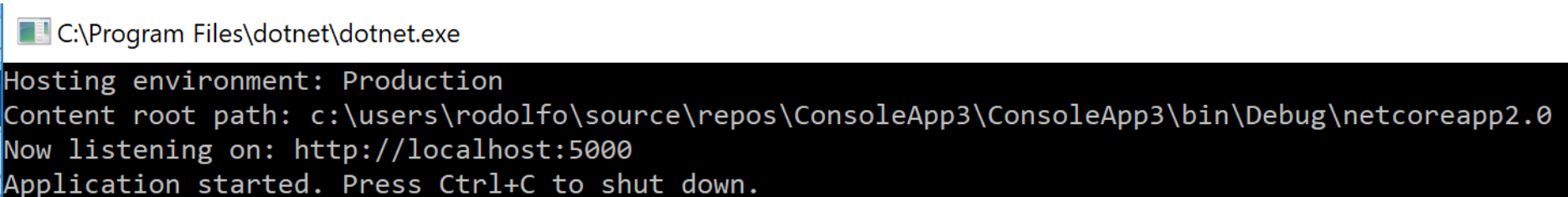
Após criar a Startup.cs, vamos utilizar o método que cria o WebHost na main do nosso Program.cs

```
0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        BuildWebHost(args).Run();
    }

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

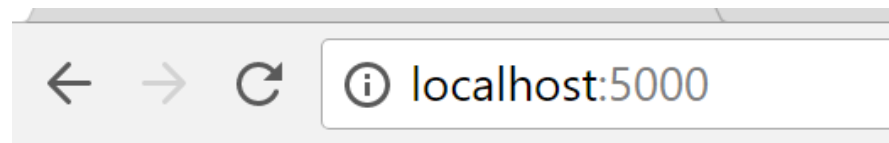
Console => Web Site

- Com isto, ao executar nosso Console, teremos um servidor web ouvindo e respondendo na porta 5000 o/



C:\Program Files\dotnet\dotnet.exe

```
Hosting environment: Production
Content root path: c:\users\rodolfo\source\repos\ConsoleApp3\ConsoleApp3\bin\Debug\netcoreapp2.0
Now listening on: http://localhost:5000
Application started. Press Ctrl+C to shut down.
```



OlÃ; Fiap!!!

| Console => Web Site

FIAP

Console => Web Site

Tínhamos um console

Temos um servidor web respondendo um texto

Como eu transformo um console em um site?

Middlewares

Utilizaremos o MVC, um middleware que facilita nosso desenvolvimento web.

Mas o que é MVC? Model View Controller?

MVC (1978, Smalltalk Xerox)

- **Model**

Sempre que você pensar em manipulação de dados, pense em model. Ele é **responsável** pela **leitura** e **escrita** de **dados**, e também de suas **validações**.

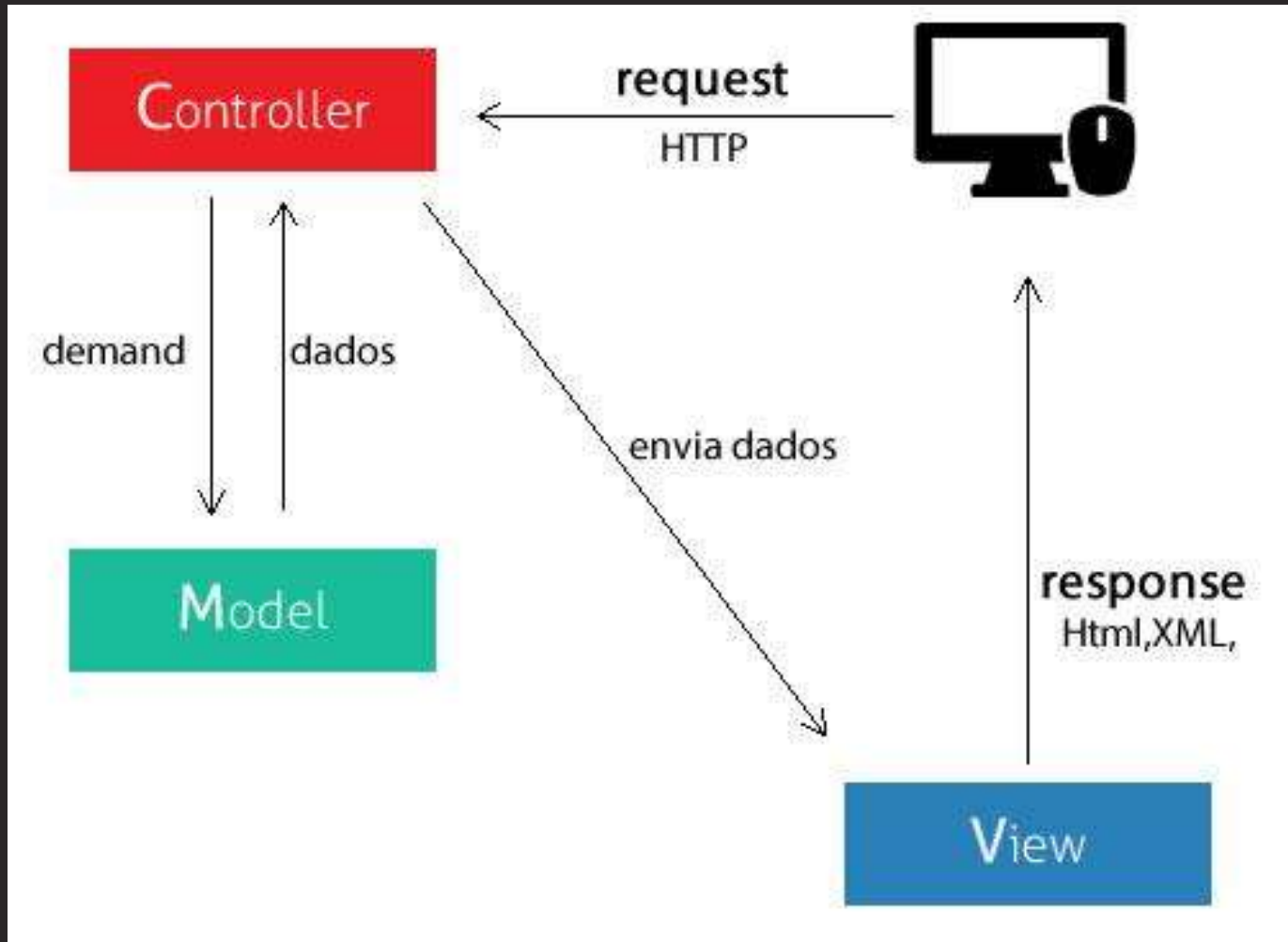
- **View**

Simples: a camada de interação com o usuário. Ela apenas faz a **exibição** dos **dados**, sendo ela por meio de um **html** ou **xml**.

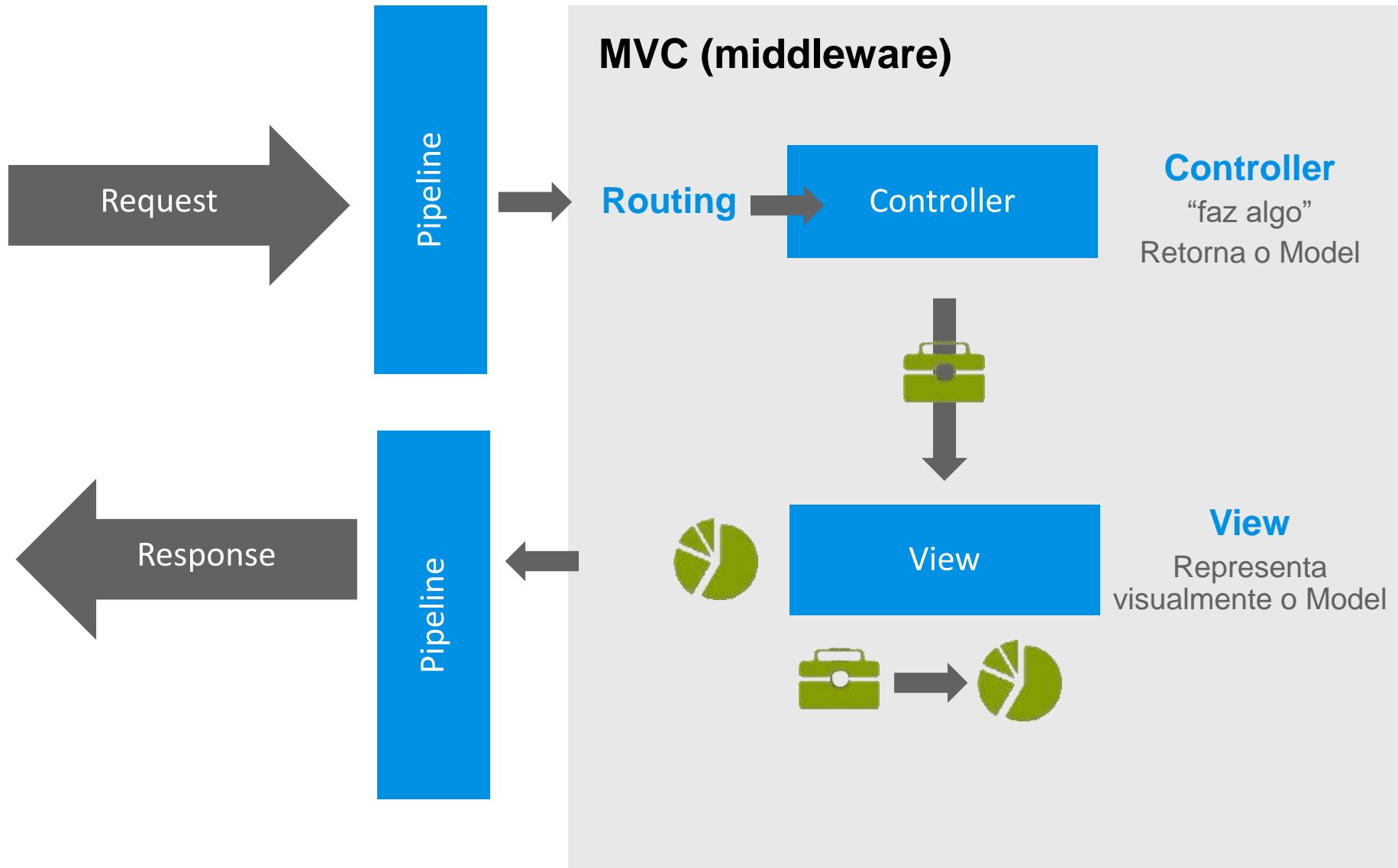
- **Controller**

O responsável por **receber** todas as **requisições** do **usuário**. Seus métodos chamados actions são responsáveis por uma página, controlando qual model usar e qual view será mostrado ao usuário.

MVC (1978, Smalltalk Xerox)



Como o ASP.NET MVC funciona



- Para começar a utilizar o MVC em nossos projetos vamos precisar adicionar ele no pipeline de nossa aplicação, para isto vamos utilizar o seguinte código:

0 references

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    app.UseMvc();
}
```


- Também será necessário adicionar via injeção de dependência o serviço do MVC

1 reference

```
public class Startup
```

```
{
```

0 references

```
public void ConfigureServices(IServiceCollection services)
```

```
{
```

```
    services.AddMvc();
```

```
}
```

- Ao tentar acessar nossa aplicação, ela não saberá qual Controller e Action deverá ser executado, pois falta o roteamento



Não foi possível encontrar a página deste **localhost**

Nenhuma página da web foi encontrada para o endereço da Web:**http://localhost:5000/**

Pesquise [localhost 5000](#) no Google

HTTP ERROR 404

- O Roteamento pode ser configurado no AddMVC da seguinte maneira:

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

- Caso seja necessário criar mais rotas, elas devem ser adicionadas da rota mais especifica, para a mais genérica (de cima para baixo)

```
app.UseMvc(routes =>
{
    routes.MapRoute(
        name: "palestras",
        template: "trilha/{nomedatrilha}/",
        defaults: new { controller = "Palestras", action = "Trilha" }
    );
    routes.MapRoute(
        name: "default",
        template: "{controller=Home}/{action=Index}/{id?}");
});
```

- Com o ASP.NET Core, os controllers de Web e de API foram unificado em um tipo só para herdar.
- Notem que qualquer método publico em um controller pode ser acessível e retornar.

0 references

```
public class HomeController : Controller
{
    0 references
    public string Index()
    {
        return "olá";
    }
}
```

| Controllers

FIAP

- Por convenção no exemplo abaixo ele irá procurar uma View com o nome Index.cshtml dentro de uma pasta chamada Home

0 references

```
public IActionResult Index()  
{  
    return View();  
}
```

- É possível sobrescrever a convenção par retornar uma view especifica em determinados cenários

```
public class HomeController : Controller
{
    0 references
    public IActionResult Index()
    {
        return View("NomeDaViewCustomizada");
    }
}
```


- Desambiguação: em determinados momentos teremos duas actions com o mesmo nome, podemos utilizar um filtro determinando qual action funcionara no verbo http GET e qual será utilizada no POST

0 references

```
public class ProdutosController : Controller
```

```
{
```

```
    [HttpGet]
```

0 references

```
    public IActionResult Edit(int id)...
```

```
    [HttpPost]
```

0 references

```
    public IActionResult Edit(Produto produto)...
```

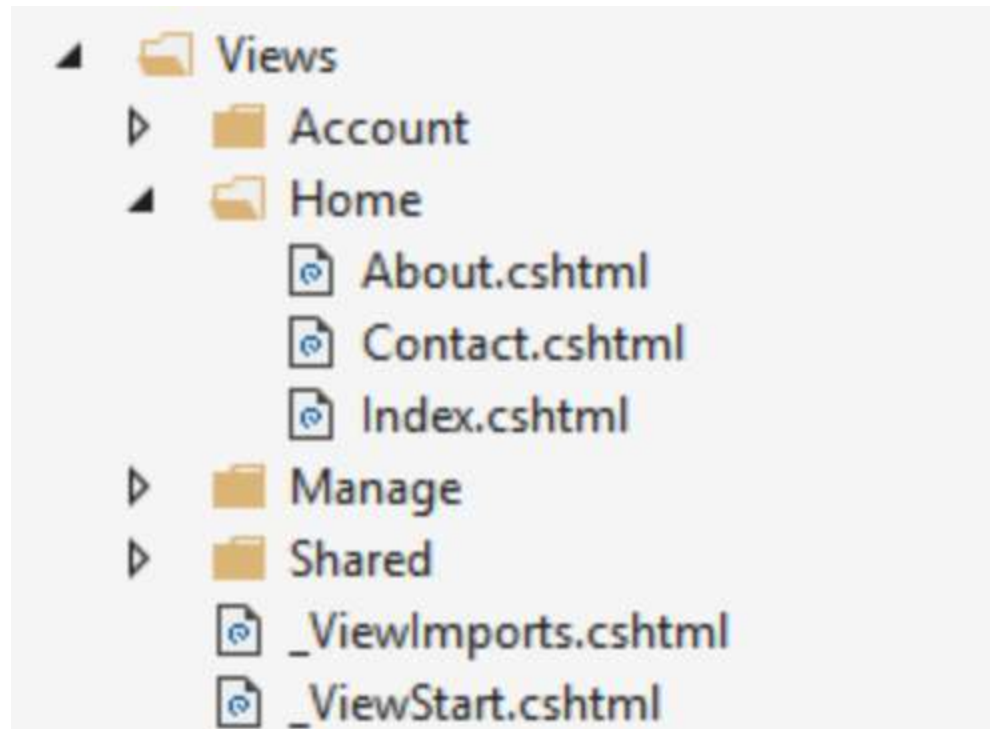
```
}
```

- Model Binder, mecanismo do ASP.NET que transforma os parâmetros da requisição e atribui os dados em um Objeto (Model)

```
[HttpPost]
0 references
public IActionResult Edit(Produto produto)
{
    if (ModelState.IsValid)
    {
        //Salvar
    }

    return View();
}
```

- Views são arquivos .cshtml que utilizam C# e uma linguagem de marcação chamada Razor.
- Por convenção elas são organizadas em pastas de acordo com a estrutura de Controllers da aplicação.



- Podemos passar dados dos controllers para as views de diferentes maneiras, as principais são:
- 1) ViewData
- 2) ViewBag
- 3) Model

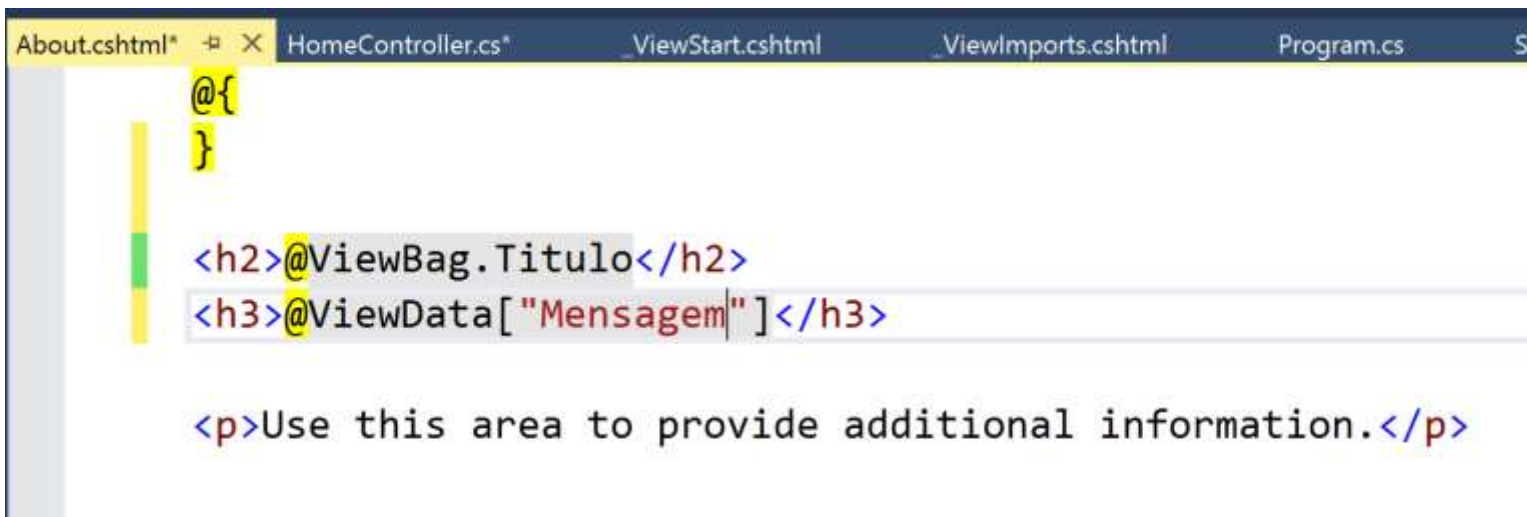
0 references

```
public IActionResult About()  
{  
    ViewData["Mensagem"] = "Your application description page.";  
    ViewBag.Titulo = "Titulo da pagina";  
    return View();  
}
```

- 1) ViewData
- 2) ViewBag

0 references

```
public IActionResult About()  
{  
    ViewData["Mensagem"] = "Your application description page.";  
    ViewBag.Titulo = "Titulo da pagina";  
    return View();  
}
```



The screenshot shows a code editor with several tabs: About.cshtml*, HomeController.cs*, _ViewStart.cshtml, _ViewImports.cshtml, and Program.cs. The active tab is About.cshtml*, which contains the following HTML code:

```
@{  
}  
  
<h2>@ViewBag.Titulo</h2>  
<h3>@ViewData["Mensagem"]</h3>  
  
<p>Use this area to provide additional information.</p>
```

Views

- 3) Model

0 references

```
public class HomeController : Controller
{
    0 references
    public IActionResult Index()
    {
        var pessoa = new Pessoa
        {
            Nome = "Rodolfo Fadino",
            Id = 123
        };

        return View(pessoa);
    }
}
```



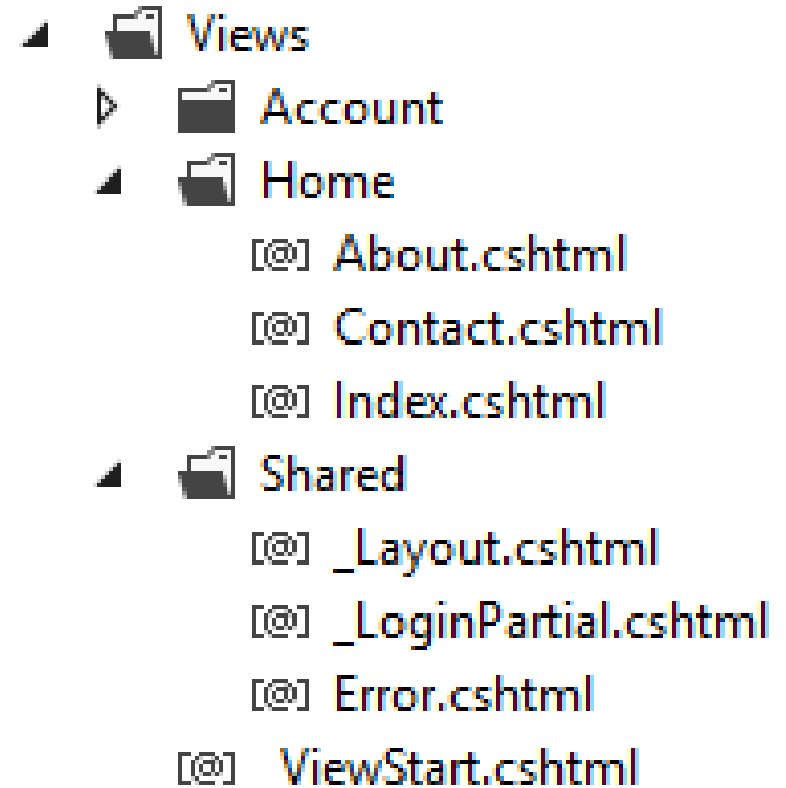
The screenshot shows a code editor with two files open: `HomeController.cs` and `Index.cshtml`. The `HomeController.cs` file shows the `Index` method, which creates a `Pessoa` object and returns `View(pessoa)`. The `Index.cshtml` file shows the view template, which uses the `@model` directive to specify the `Pessoa` model. The template contains two paragraphs: the first displays the `Id` property of the model, and the second displays the `Nome` property.

```
Index.cshtml* X Pessoa.cs About.cshtml HomeController.cs _ViewStart.c
@model WebApplication1.Controllers.Pessoa

<p>
    @Model.Id
</p>

<p>
    @Model.Nome
</p>
```

- View Engine => Razor
- Helpers
- Partial
- Layout
- Section





TM

FIFA WORLD CUP

<Projeto Rússia/>

até 4 pessoas 😊

cadastro de times

web site

API

...



FIFA WORLD CUP

MBA⁺

Copyright © **2018** Prof. Rodolfo Fadino Junior

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor Rodolfo.