

# Noções básicas de



integração contínua e boas práticas de programação em geral

Matheus Neder – Arquiteto de Software BHS/Olé Consignado



# C# (C Sharp)

- Anders Hejlsberg (Turbo Pascal e Delphi)
- Microsoft .NET Framework
- ECMA 334 - ISO/IEC 23270
  - Mono
  - dotGNU
  - Portable.NET

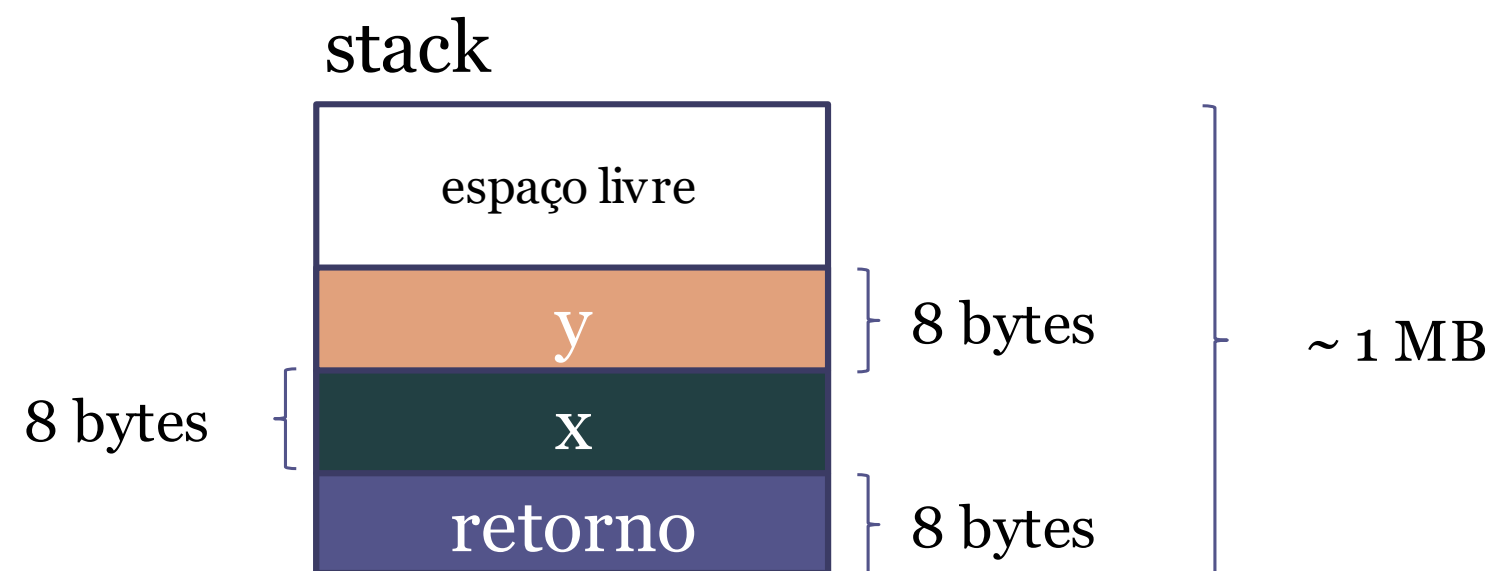


# Revisão relâmpago

- Tipos de valor e tipos de referencia e gerenciamento de memória
- Classes e estruturas e seus membros
  - Campos (atributos)
  - Métodos
  - Propriedades
- Herança e polimorfismo
- Interfaces

# Gerenciamento de memória

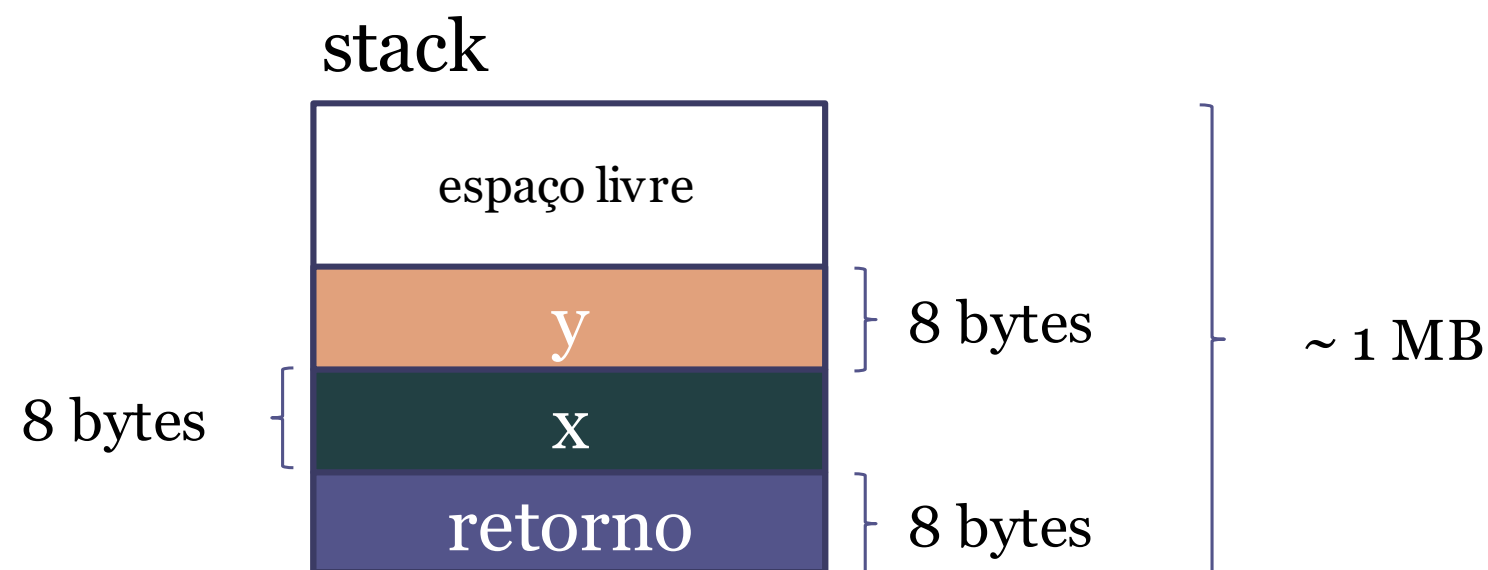
```
public long Soma(long x, long y)
{
    return x + y;
}
```



Observação: 8 bytes = sizeof(long)

# Gerenciamento de memória

Tipos de valor (value types) são armazenados na pilha (stack).

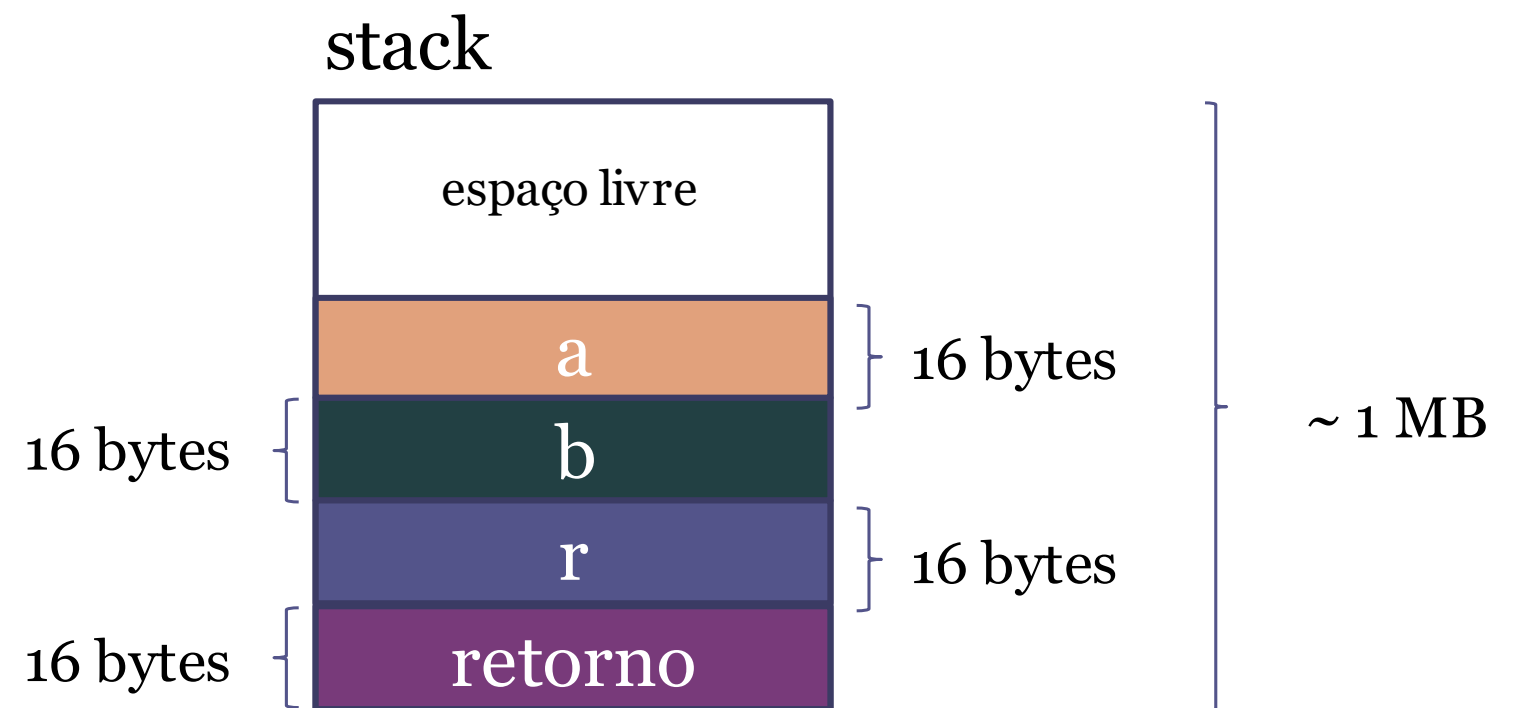


Observação: 8 bytes = sizeof(long)

# Gerenciamento de memória

```
public Coord Midpoint(Coord a, Coord b)
{
    Coord r = new Coord();
    ...
    return r;
}
```

```
struct Coord
{
    public double x;
    public double y;
}
```



Observação: 16 bytes = sizeof(Coord)

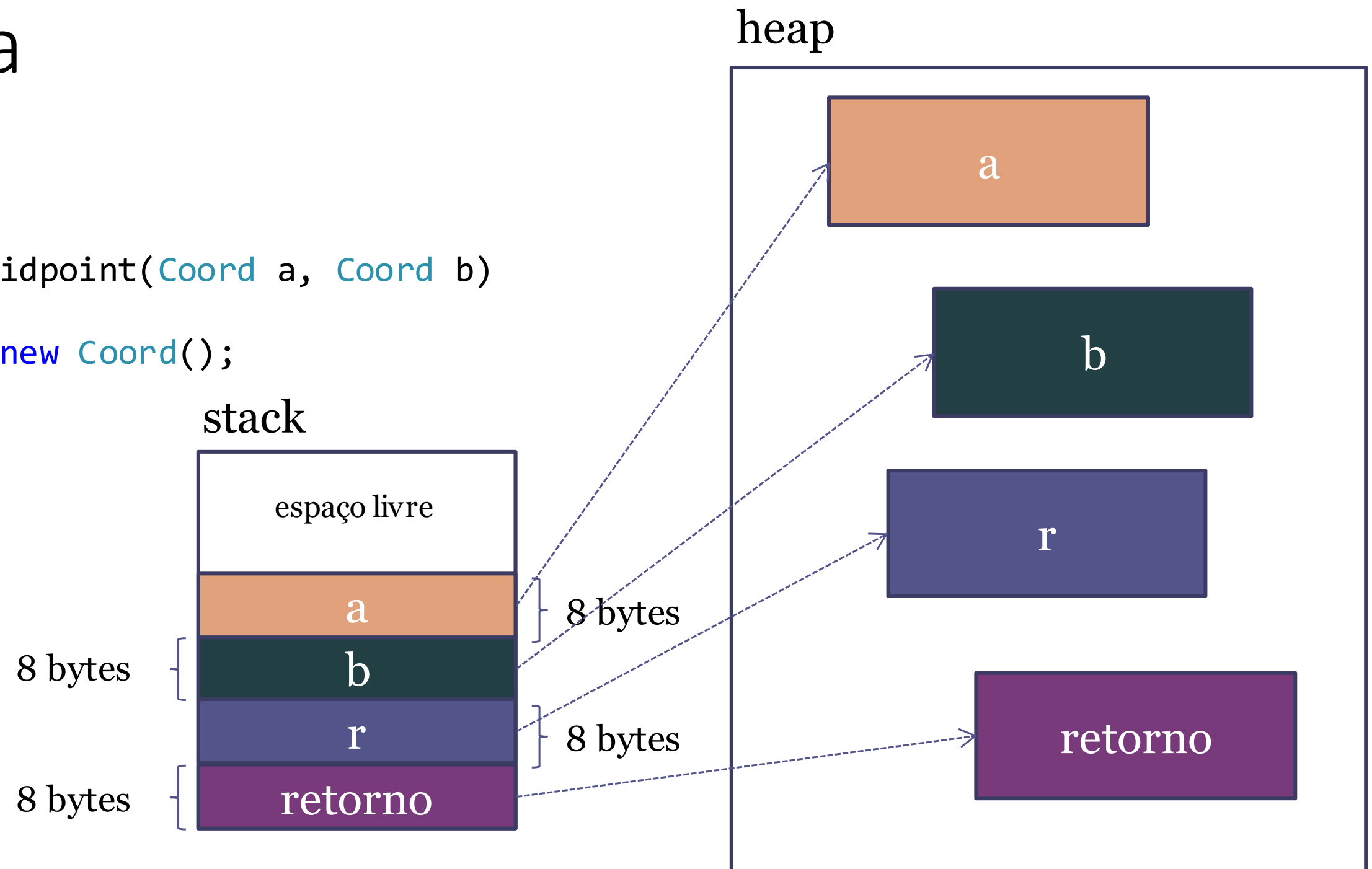
# Gerenciamento de memória

Considere uma nova versão do tipo *Coord*:

```
class Coord
{
    public double x;
    public double y;
}
```

# Memória

```
public Coord Midpoint(Coord a, Coord b)
{
    Coord r = new Coord();
    ...
    return r;
}
```

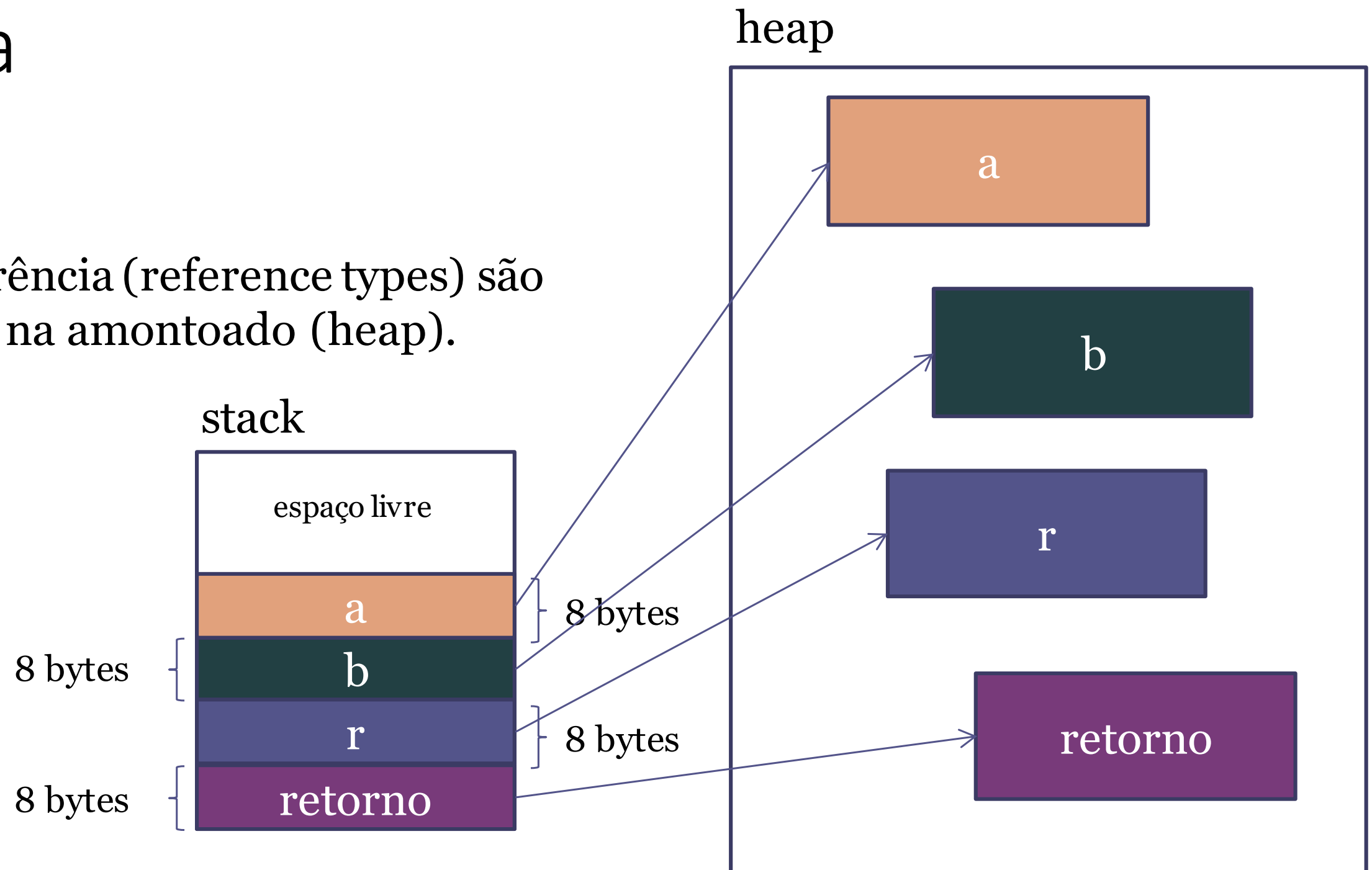


Observação: 8 bytes = tamanho do ponteiro



# Memória

Tipos de referência (reference types) são armazenados na amontoado (heap).



Observação: 8 bytes = tamanho do ponteiro

# Gerenciamento de memória

```
struct Coord
{
    public double x;
    public double y;
}

public void Swap(Coord coord)
{
    double aux = coord.x;
    coord.x = coord.y;
    coord.y = aux;
}
```

```
Coord coord = new Coord();
coord.x = 10.0;
coord.y = 5.0;

Swap(coord);

Console.WriteLine($"x: {coord.x}");
Console.WriteLine($"y: {coord.y}");
```

?

# Gerenciamento de memória

```
class Coord
{
    public double x;
    public double y;
}

public void Swap(Coord coord)
{
    double aux = coord.x;
    coord.x = coord.y;
    coord.y = aux;
}
```

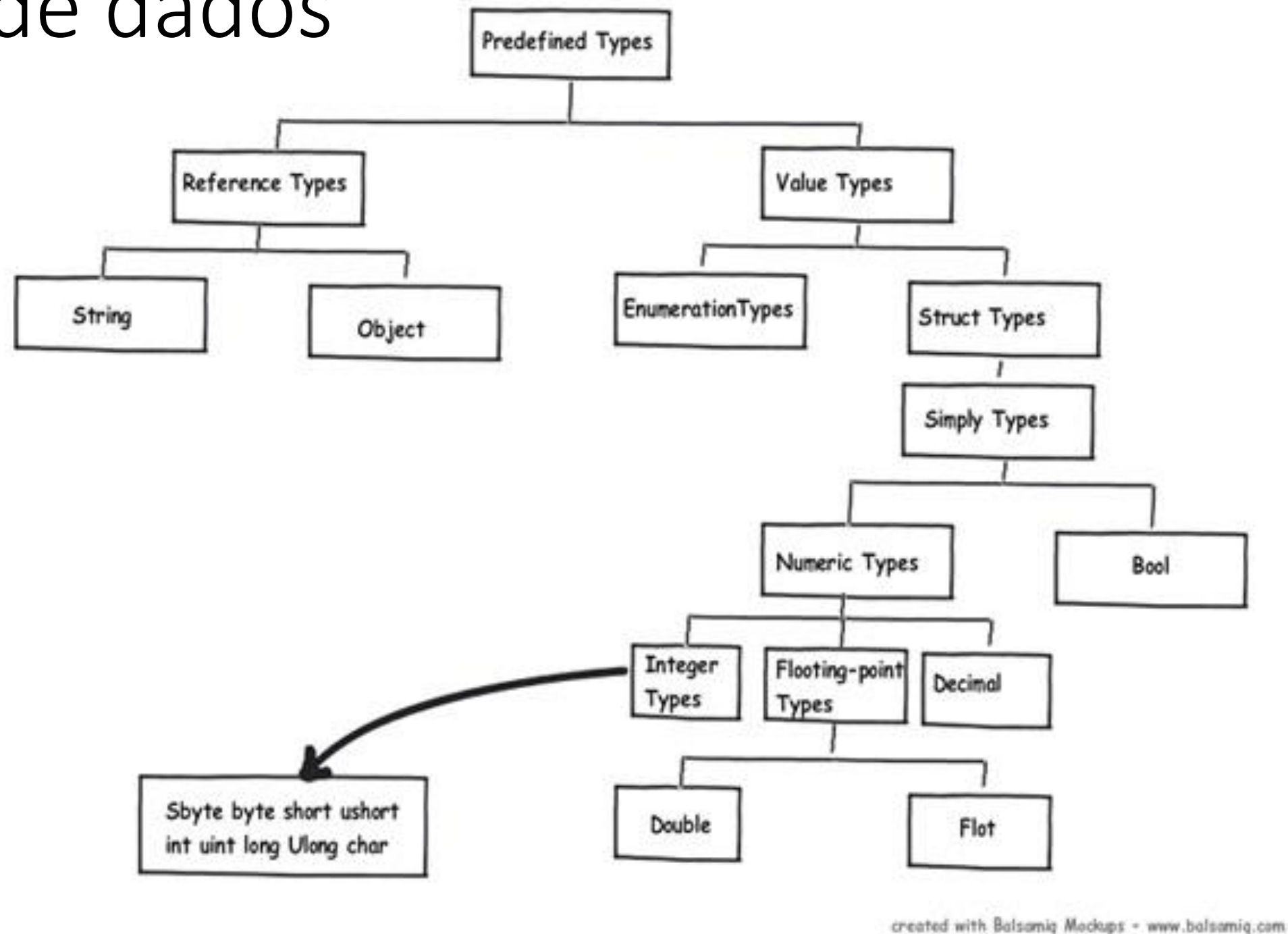
```
Coord coord = new Coord();
coord.x = 10.0;
coord.y = 5.0;

Swap(coord);

Console.WriteLine($"x: {coord.x}");
Console.WriteLine($"y: {coord.y}");
```

?

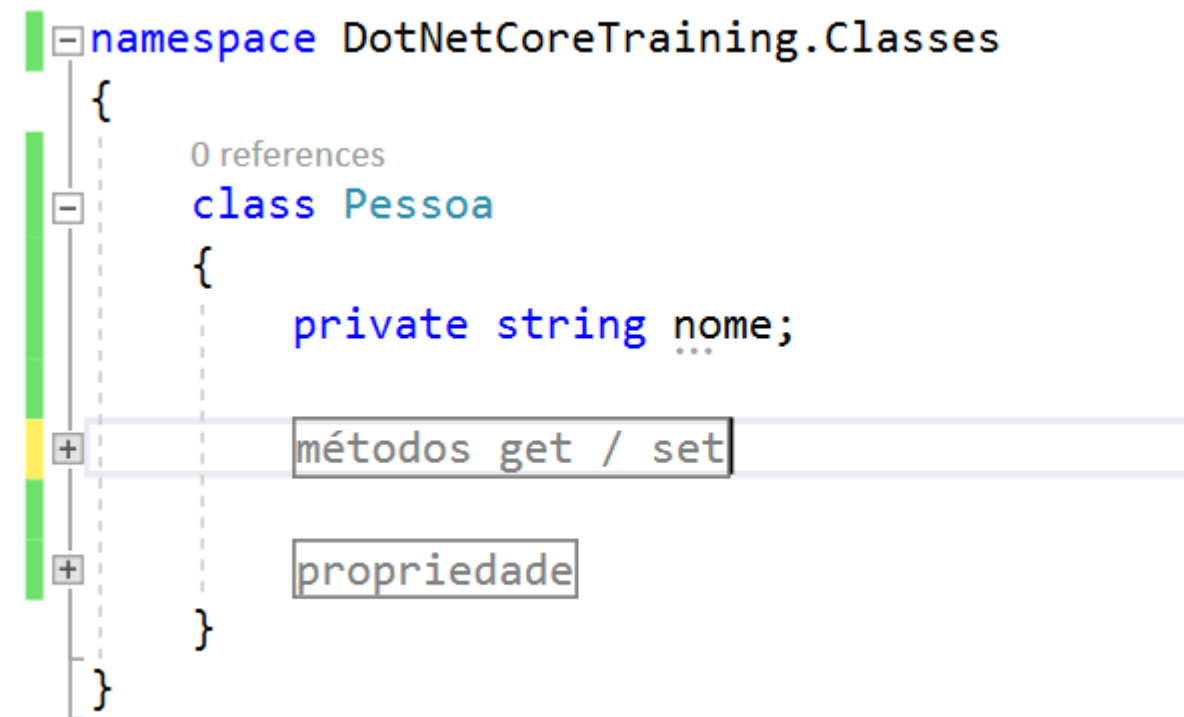
# Tipos de dados



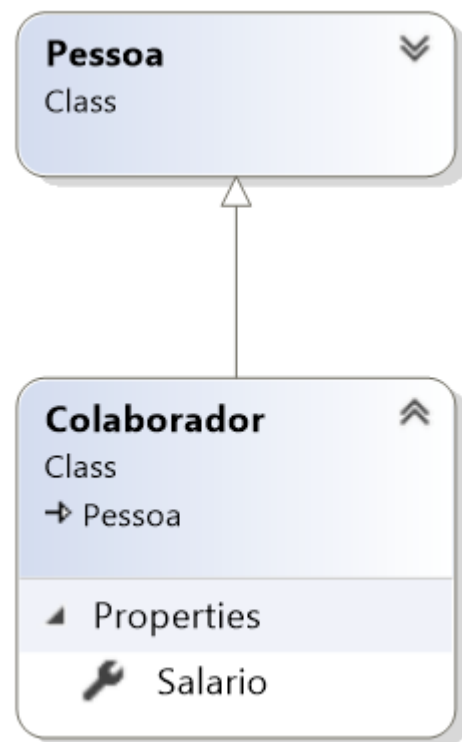
Fonte: <https://www.codeproject.com/Articles/76153/Six-important-NET-concepts-Stack-heap-value-types>

# Classes e estruturas

- Campos (atributos)
- Métodos
- Propriedades



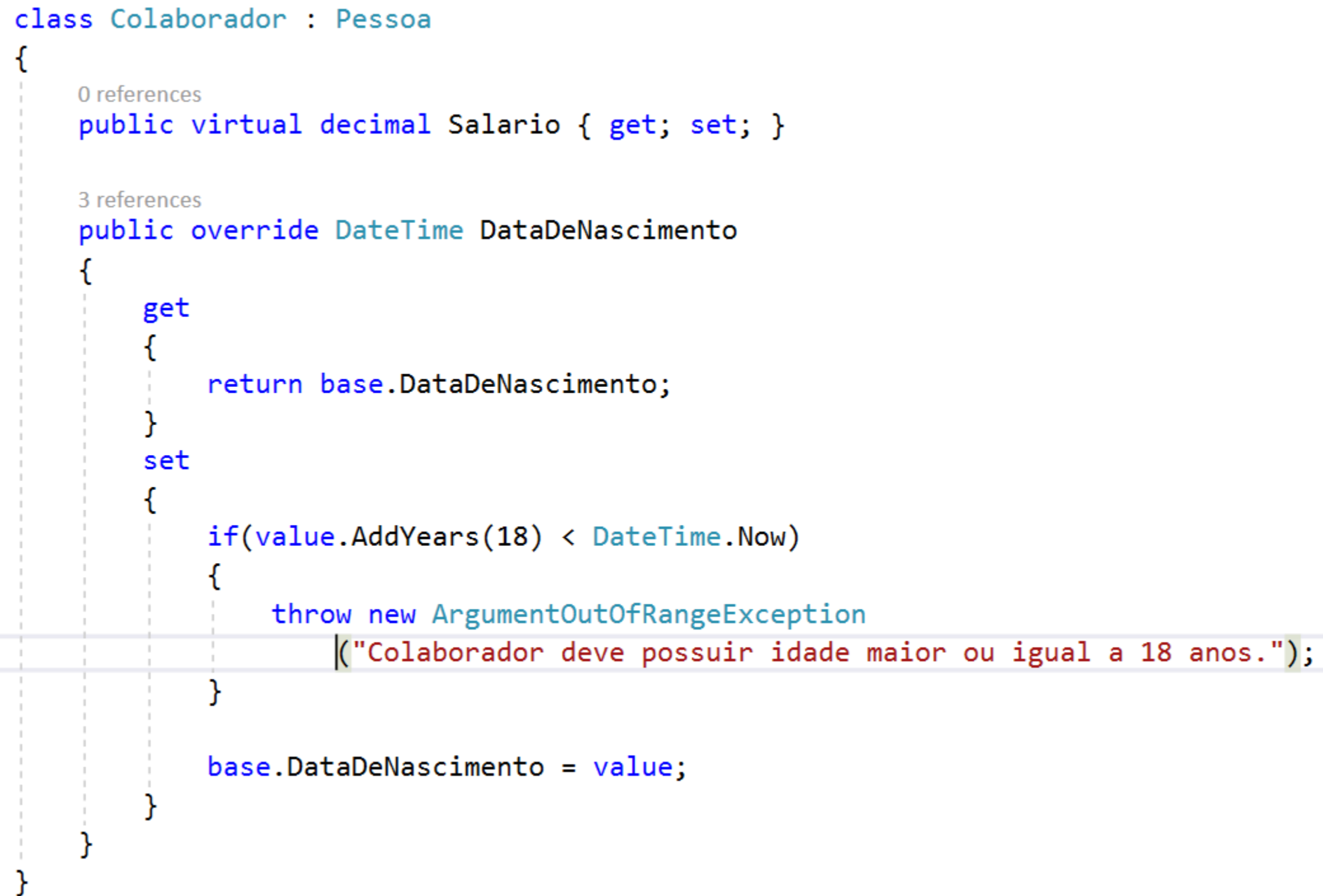
# Herança e polimorfismo



```
namespace DotNetCoreTraining.Classes
{
    0 references
    class Colaborador : Pessoa
    {
        0 references
        public decimal Salario { get; set; }
    }
}
```

Obs.: Herança é possível apenas para tipos de referencia

# Herança e polimorfismo



```
class Colaborador : Pessoa
{
    0 references
    public virtual decimal Salario { get; set; }

    3 references
    public override DateTime DataDeNascimento
    {
        get
        {
            return base.DataDeNascimento;
        }
        set
        {
            if(value.AddYears(18) < DateTime.Now)
            {
                throw new ArgumentOutOfRangeException
                    |("Colaborador deve possuir idade maior ou igual a 18 anos.");
            }

            base.DataDeNascimento = value;
        }
    }
}
```

# Herança e polimorfismo

1 reference

```
public void CorrigirDataDeNascimento(Pessoa pessoa, DateTime novaDataDeNascimento)
{
    pessoa.DataDeNascimento = novaDataDeNascimento;
}
```

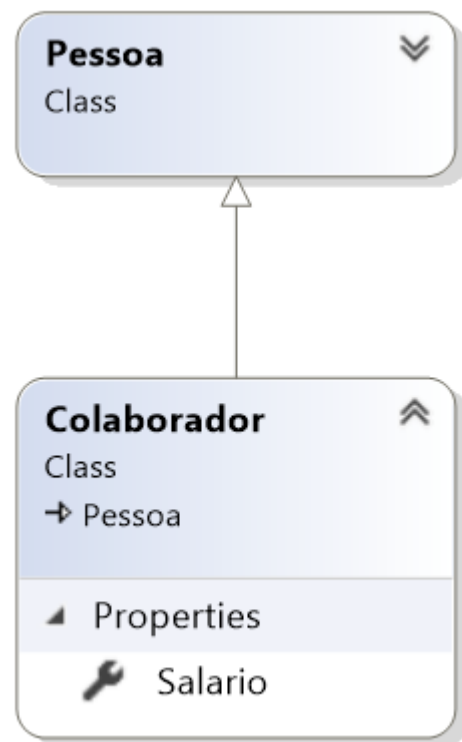
2 references

```
public void Executar()
{
    try
    {
        var joseDaSilva = new Colaborador()
        {
            Nome = "José da Silva",
            DataDeNascimento = DateTime.Parse("1980-04-30"),
            Salario = 500
        };

        CorrigirDataDeNascimento(joseDaSilva, DateTime.Parse("2000-04-30"));
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
        Console.ReadLine();
    }
}
```



# Herança e polimorfismo



```
namespace DotNetCoreTraining.Operacoes
{
    0 references
    class Exemplo
    {
        0 references
        public void Adicionar(Pessoa pessoa)
        {
            // logica para adicionar

            pessoa.Salario;
        }
    }
}
```

# Interfaces

2 references

class RepositorioPessoas

{

0 references

public Pessoa[] ObterPessoas()

{

var query = @"

select top 100 nome

, salario

, data\_de\_nascimento

, tipo

from pessoas";

return Db.Query<Pessoa>(query);

}

1 reference

public void Atualizar(Pessoa pessoa)

{

var query = "update pessoas set ...";

Db.Query(query);

}

}

# Interfaces

1 reference

class Exemplo02

```
{  
    private RepositorioPessoas repositorioDePessoas;  
  
    0 references  
    public Exemplo02(RepositorioPessoas repositorioDePessoas)  
    {  
        this.repositorioDePessoas = repositorioDePessoas;  
    }  
  
    0 references  
    public void CorrigirDataDeNascimento(Pessoa pessoa, DateTime novaDataDeNascimento)  
    {  
        pessoa.DataDeNascimento = novaDataDeNascimento;  
  
        repositorioDePessoas.Atualizar(pessoa);  
    }  
}
```

**RepositorioPessoas**

Class

Methods

Atualizar

ObterPessoas

# Interfaces

1 reference

```
interface IRepositoryPessoas
{
    2 references
    void Atualizar(Pessoa pessoa);
    1 reference
    Pessoa[] ObterPessoas();
}
```

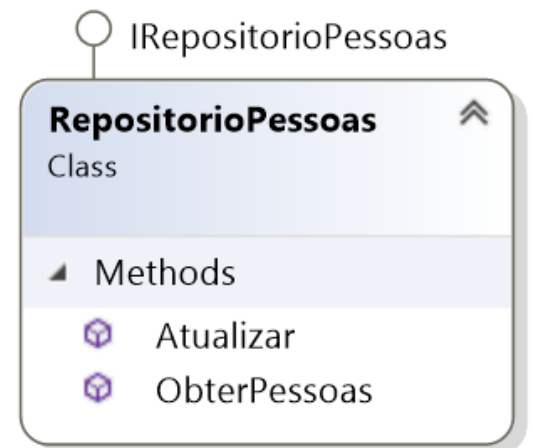
2 references

```
class RepositorioPessoas : IRepositoryPessoas
{
    1 reference
    public Pessoa[] ObterPessoas()
    {
        var query = @"
        select top 100 nome
            , salario
            , data_de_nascimento
            , tipo
        from pessoas";

        return Db.Query<Pessoa>(query);
    }

    2 references
    public void Atualizar(Pessoa pessoa)
    {
        var query = "update pessoas set ...";

        Db.Query(query);
    }
}
```



# Interfaces

```
1 reference
class Exemplo02
{
    private IRepositoryPessoas repositorioDePessoas;

    0 references
    public Exemplo02(IRepositoryPessoas repositorioDePessoas)
    {
        this.repositorioDePessoas = repositorioDePessoas;
    }

    0 references
    public void CorrigirDataDeNascimento(Pessoa pessoa, DateTime novaDataDeNascimento)
    {
        pessoa.DataDeNascimento = novaDataDeNascimento;

        repositorioDePessoas.Atualizar(pessoa);
    }
}
```

## IRepositoryPessoas

Interface

### Methods



Atualizar



ObterPessoas



<https://goo.gl/forms/6UP0BiX2INTol2xx2>



Mono

# Cross platform, open source .NET framework



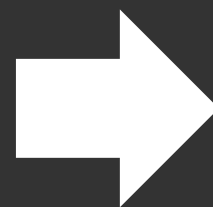
European Computer Manufacturers Association

Standard ECMA-335 (CLI) / ECMA-334 (C#)

Ximian / Novell



Mono



# Cross platform, open source .NET framework



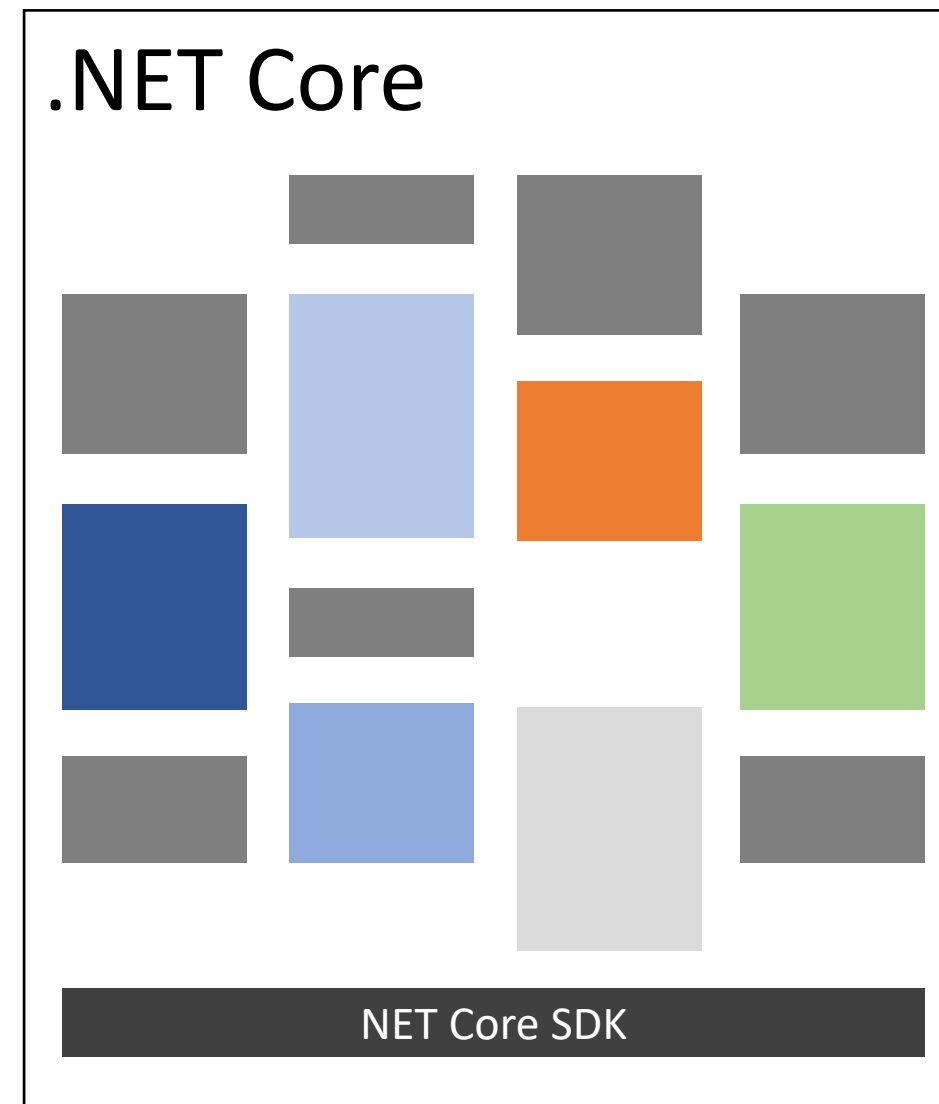
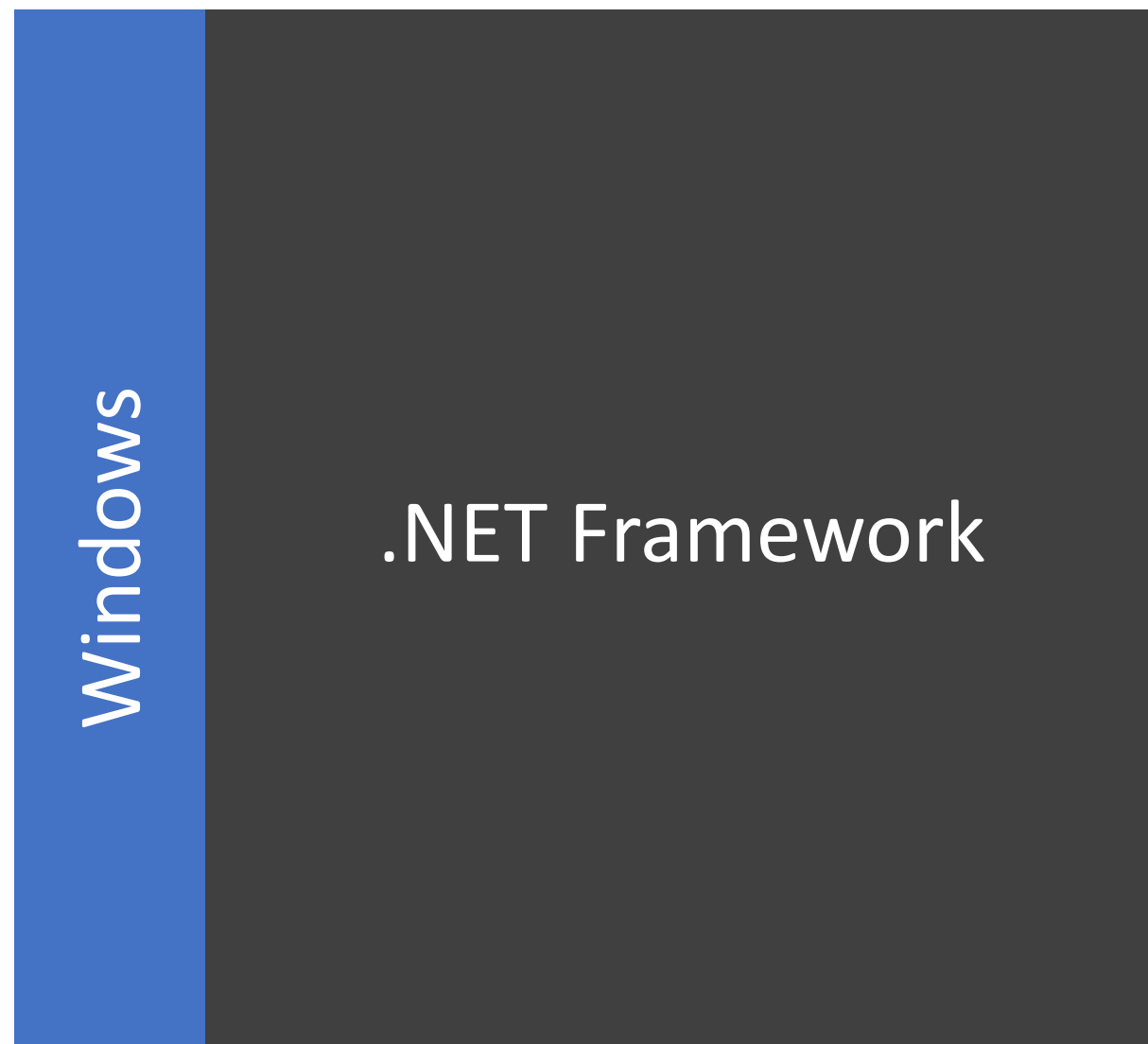
European Computer Manufacturers Association

Standard ECMA-335 (CLI) / ECMA-334 (C#)

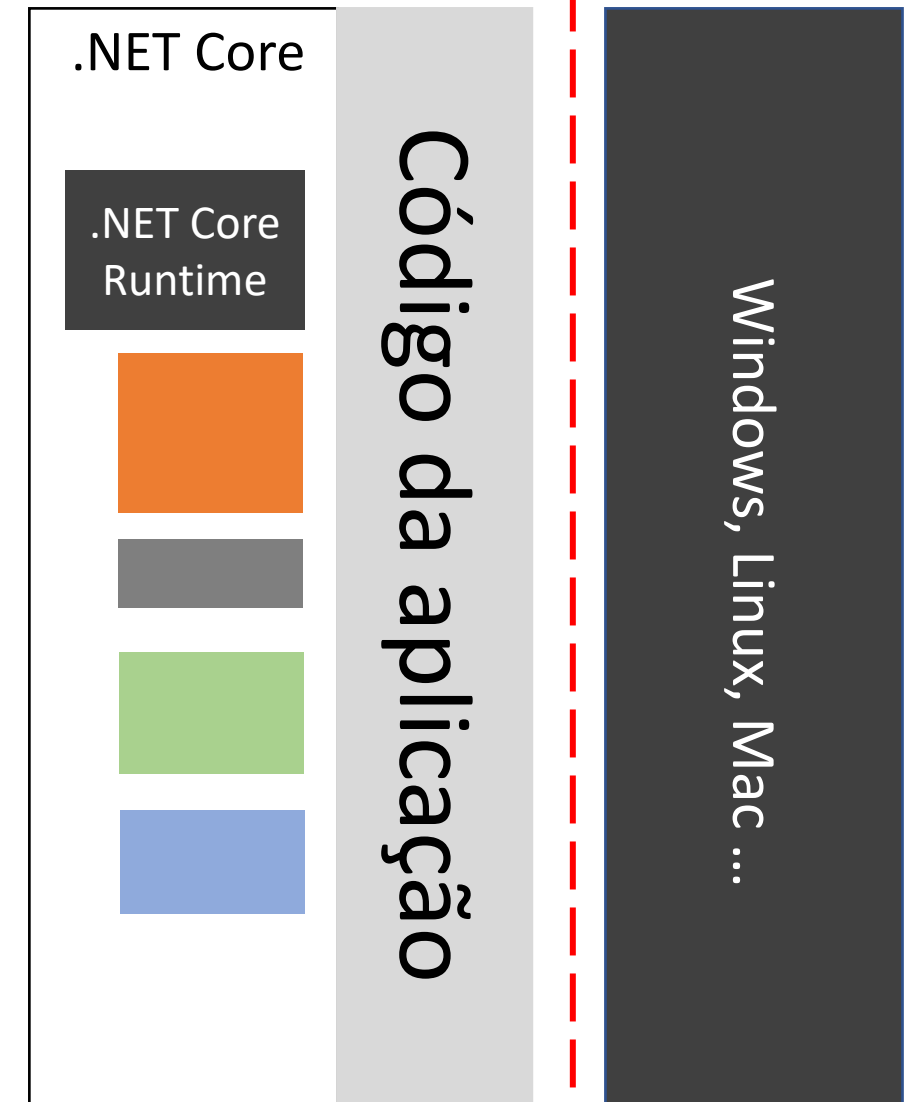
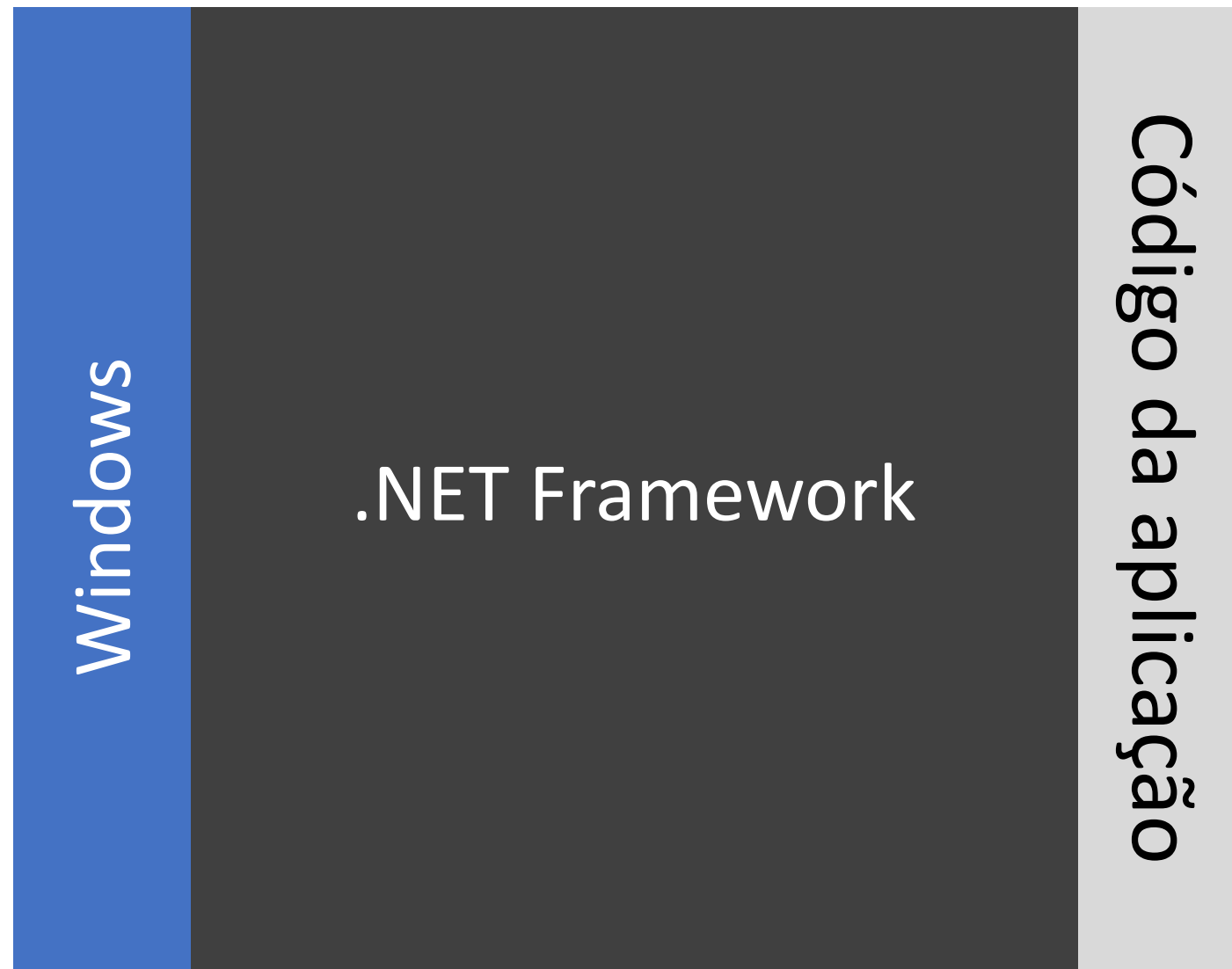
Ximian / Novell



# .NET Core vs .NET Framework



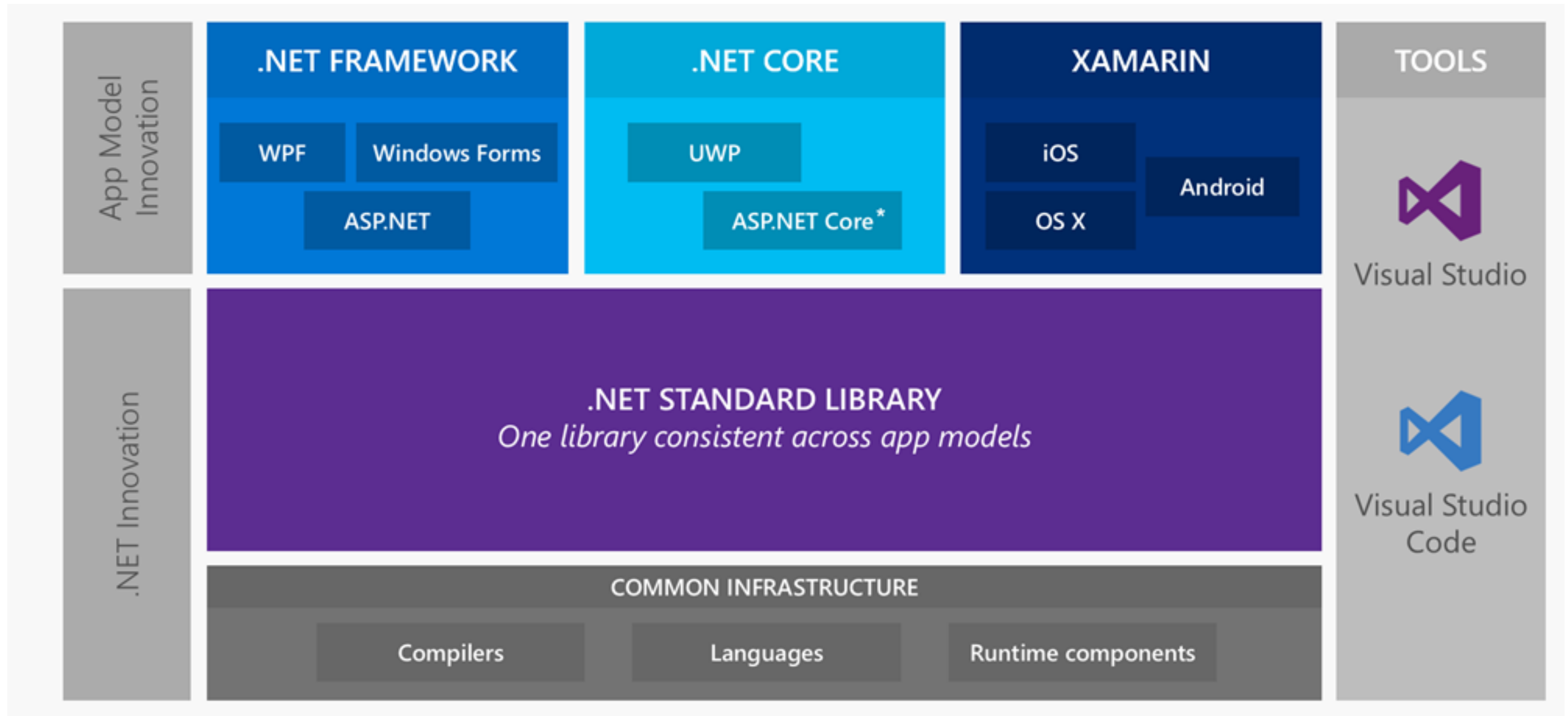
# .NET Core vs .NET Framework



# .NET Standard



# .NET Standard



<https://www.slideshare.net/dotnet18/moving-forward-with-aspnet-core>

# .NET Standard

<b>.NET Standard</b>	<a href="#"><u>1.0</u></a>	<a href="#"><u>1.1</u></a>	<a href="#"><u>1.2</u></a>	<a href="#"><u>1.3</u></a>	<a href="#"><u>1.4</u></a>	<a href="#"><u>1.5</u></a>	<a href="#"><u>1.6</u></a>	<a href="#"><u>2.0</u></a>
<b>.NET Core</b>	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0
<b>.NET Framework</b>	4.5	4.5	4.5.1	4.6	4.6.1	4.6.1	4.6.1	4.6.1
<b>Mono</b>	4.6	4.6	4.6	4.6	4.6	4.6	4.6	5.4
<b>Xamarin.iOS</b>	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.14
<b>Xamarin.Mac</b>	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.8
<b>Xamarin.Android</b>	7.0	7.0	7.0	7.0	7.0	7.0	7.0	8.0
<b>UWP</b>	10.0	10.0	10.0	10.0	10.0	10.0.16	10.0.16	10.0.16
<b>Windows</b>	8.0	8.0	8.1					

<https://github.com/dotnet/standard/blob/master/docs/versions.md>

# .NET Core

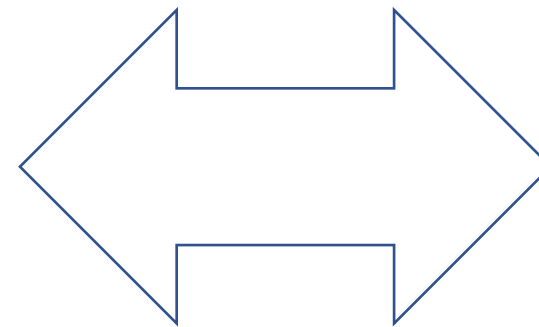
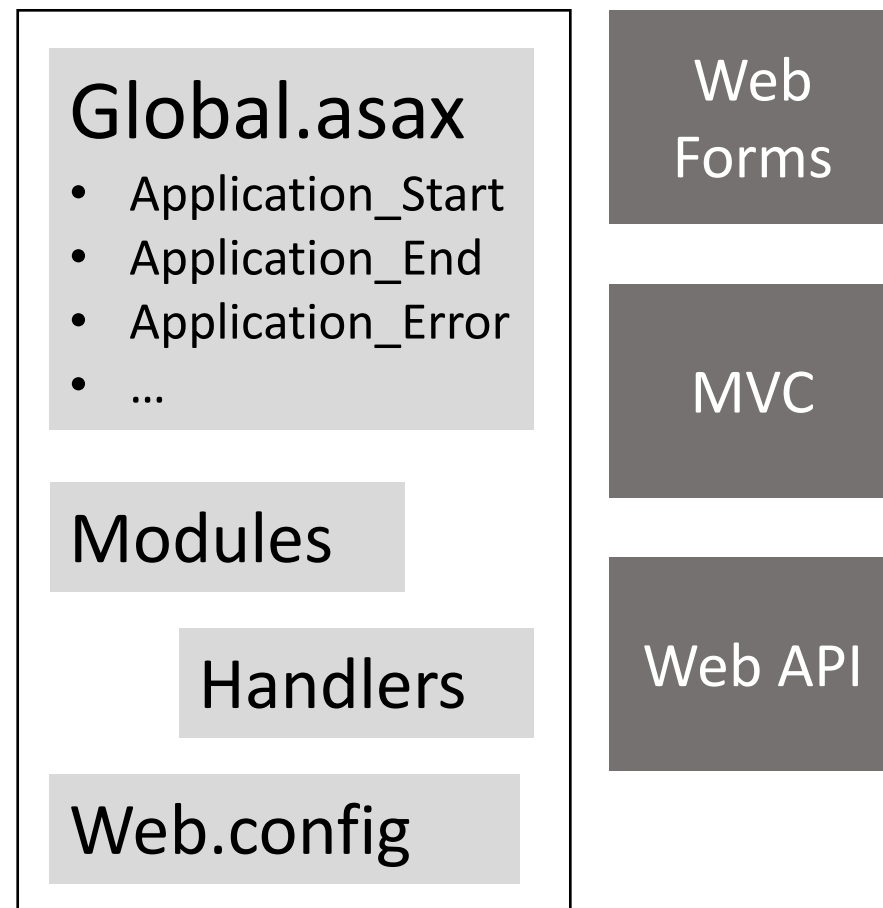
- Plataforma de desenvolvimento de propósitos gerais
- Executa sobre Windows, Linux, macOS, nuvem ou dispositivos embarcados
- Open source
- Compatível com .NET Framework, Xamarin e Mono via .NET Standard.

# ASP.NET Core



O que mudou?

# ASP.NET Core



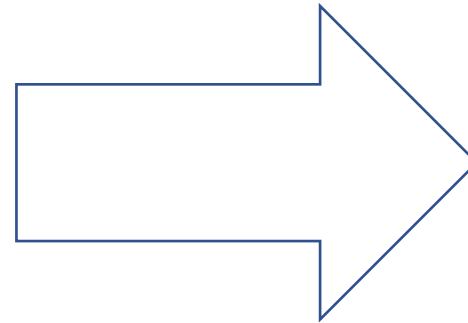


# ASP.NET Core

ASP  
.NET

MVC

Web API 2

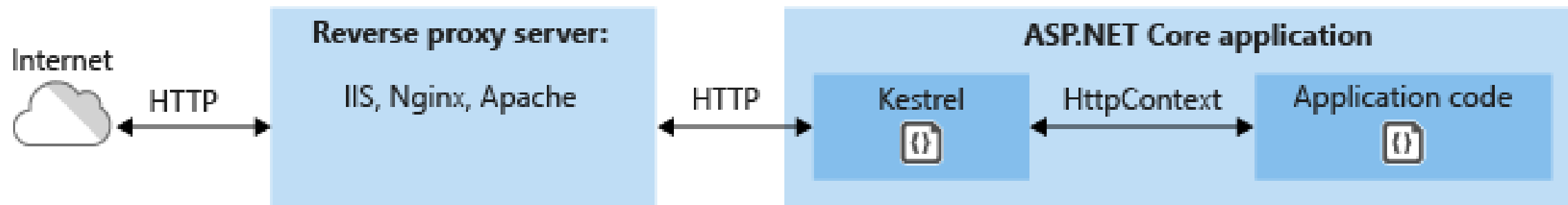


ASP.NET  
Core

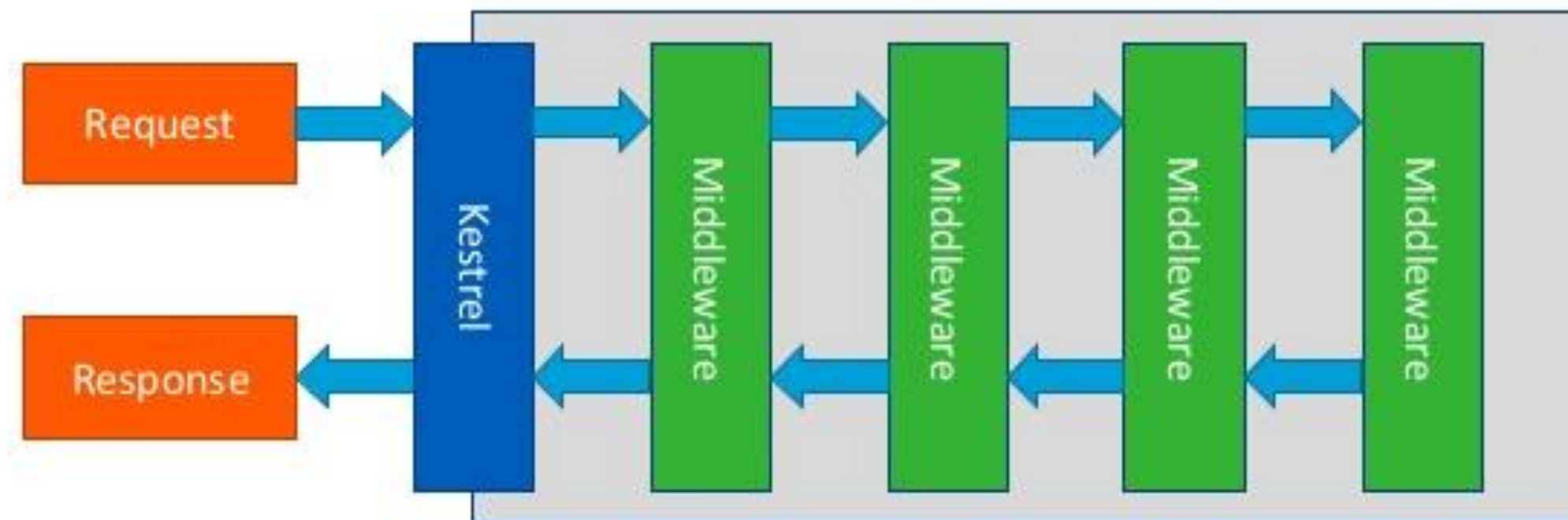
MVC

Escolha ruim para o nome da biblioteca ou estaria a Microsoft tentando ressuscitar o ASP.NET MVC?

# ASP.NET Core



<https://imasters.com.br/dotnet/configuracao-e-deploy-de-aplicativos-asp-net-core-2-0-no-iis>



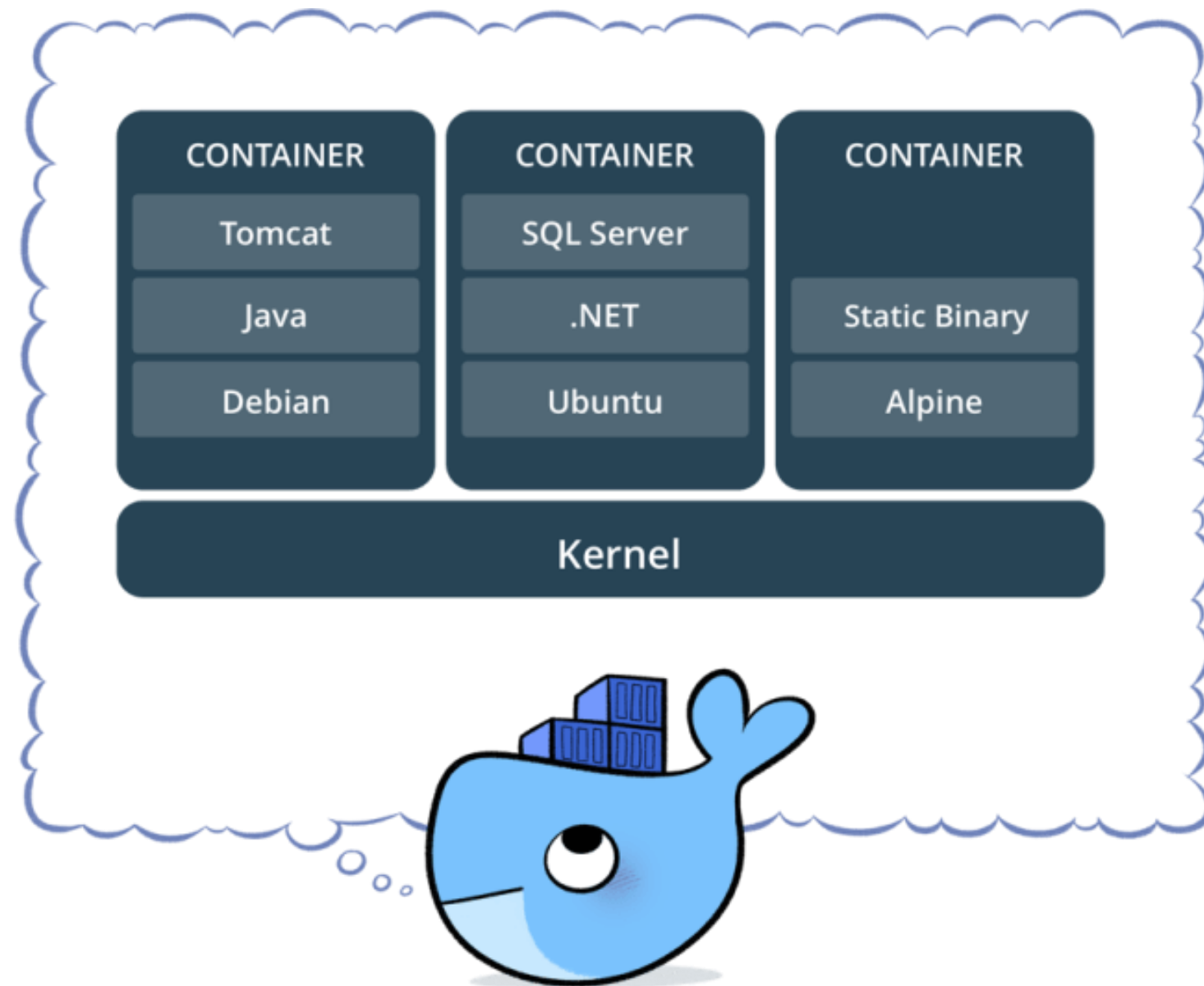
<https://www.slideshare.net/Avanade-Nederland/introduction-to-aspnet-core>

# hands-on



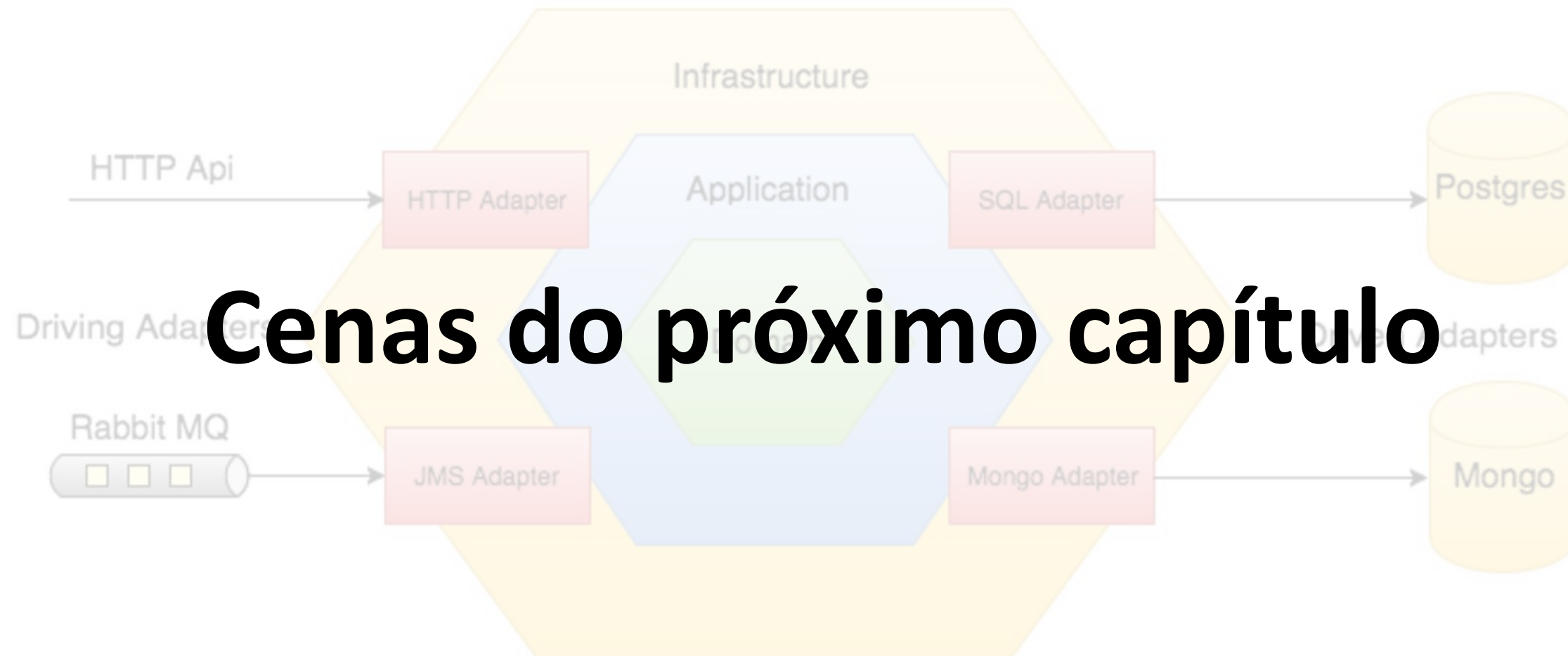
<http://espacojacyra.com.br/mao-na-massa-colaboradoras-participam-de-treinamento-na-chocolandia/>

# Docker



<https://blog.docker.com/2017/08/docker-101-introduction-docker-webinar-recap/>

# Arquitetura Hexagonal



**Cenas do próximo capítulo**

<https://gumtreeuk.github.io/presentations/gumtree-tech-talks/microengines-241116/index.html>