Fundamentos .NET

C# BÁSICO

RUI FLEXA – MCPD

Origens do ASP.NET

ASP	• ASP clássico lançado em 1996
ASP.NET WebForms	• Lançado em 2002 junto com o .NET Framework
ASP.NET MVC	• Alternativa ao WebForms, lançado em 2009
ASP.NET Core	• Inicialmente com o nome .vNext e lançado no final de 2015 e com início de divulgação em 2016

Introdução

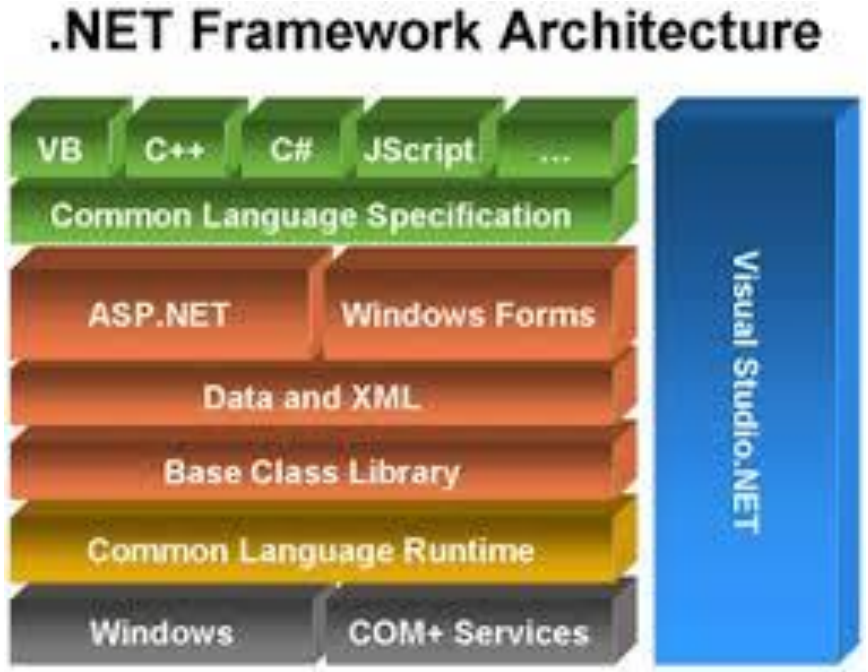
- C# foi desenvolvida pela Microsoft.
- Foi apresentada junto a plataforma .NET.
- C# é uma mistura de C++ e Java.
- Criador: Anders Hejlsberg (responsável pela arquitetura de software na Borland)

Características

- É orientada a objetos.
- Possui um alto nível de abstração.
- Possui *Garbage Collector (gerencia a alocação e liberação de memória para sua aplicação)*.
- Suporta tipagem dinâmica e estática.
- A tipagem dinâmica não exige declarações de tipos de dados (a partir da versão 3.0 do C#).
- A tipagem estática exige a declaração de quais dados poderão ser associados a cada variável antes da sua utilização.

Características

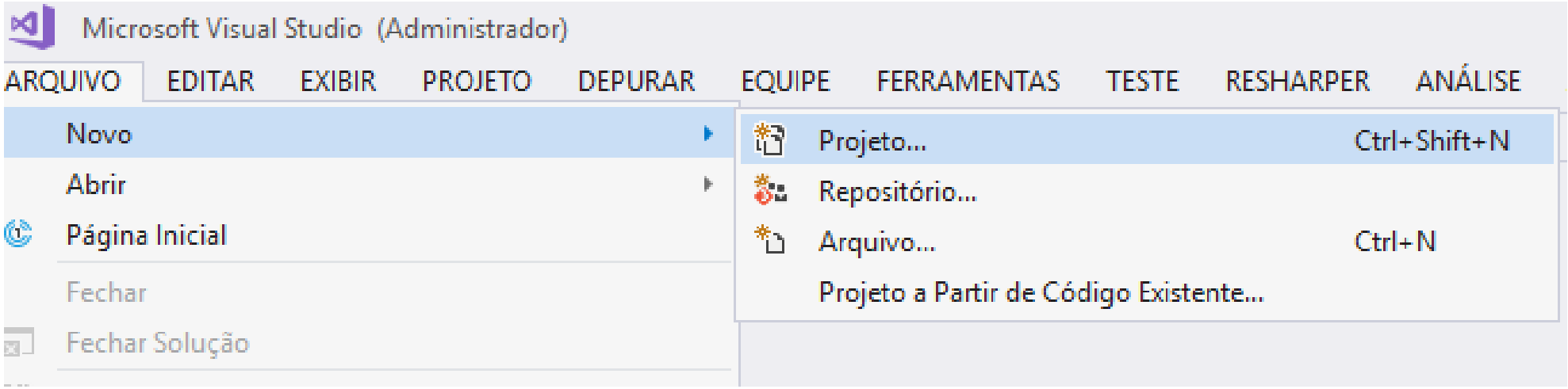
- C# está bastante vinculada ao framework .NET
  - O framework .NET suporta várias linguagens de programação.
- C# usa a biblioteca de classe do framework
  - .NET O framework .NET possui mais de 4 mil classes.



Ambiente de Desenvolvimento

- Windows 7
- Visual Studio 2017 Community
- SQL Server Local DB
- IIS Express

Criando um projeto no VS



Exercício 01 – O Bom e Velho Hello World!

Tecle CTRL + SHIFT + B para construir a aplicação e depois tecle a combinação Ctrl + F5 para executar

```
class Program
{
    static void Main(string[] args)
    {

        Console.WriteLine("Hello World!");
        Console.ReadKey();
    }
}
```

Exercício 01 – O Bom e Velho Hello World!

- Ctrl + F5 no Visual C# executa sem debugging. Isso força uma pausa no final da execução permitindo que você possa visualizar o resultado da execução
- Já o atalho CTRL + SHIFT + B é o mesmo que ir no menu Debug e depois clicar em “Build Solution”

Namespaces e Referências

- As classes são organizadas em namespaces. System é um namespace
- Console é uma classe do namespace System
- Declarar seus próprios namespaces pode ajudar no controle do escopo da classe e nomes de métodos em grandes projetos
- Os namespaces ajudam na manutenção de um programa
- Você faz referência a um namespace utilizando a palavra reservada “using
- WriteLine é um método da classe Console. É passada a string “Hello World!”.

```
using System;

namespace Exercicio01
{
    class Program
    {
        static void Main(string[] args)
        {

            Console.WriteLine("Hello World!");
            Console.ReadKey();
        }
    }
}
```

Observações

- C# é case-sensitive, ou seja, diferencia letras minúsculas de maiúsculas.
- Com o “//” é possível fazer comentários de uma linha.
- Começando com “/*” e fechando com “*/” é possível fazer comentários de múltiplas linhas.

Exercicio 02 – Obtendo Dados do Usuário

O método ReadLine() obtém os dados

O operador “+” concatena strings

```
namespace Exercicio02
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Nome:");
            string nome = Console.ReadLine();
            Console.WriteLine("Nome digitado foi:" + nome);
            Console.ReadKey();
        }
    }
}
```

Exercício 03 - Retornando o tipo de uma variável

Utilizar a função GetType().

Perceba que na execução o retorno será System.Single. Isso mesmo, float é um apelido para System.Single. Cada tipo em C# tem o apelido para o tipo .NET.

```
namespace Exercicio03
{
    class Program
    {
        static void Main(string[] args)
        {
            float pi = 3.14F;
            Console.WriteLine(pi.GetType());
            Console.ReadKey();
        }
    }
}
```

Exercicio 04 - Convertendo string para inteiro

Basta utilizar a função Parse()

```
namespace Exercicio04
{
    class Program
    {
        static void Main(string[] args)
        {
            string num = "1234";
            int n = int.Parse(num);
            Console.WriteLine(n);
            Console.ReadKey();
        }
    }
}
```

Exercício 05 - Exibindo várias variáveis

```
namespace Exercicio05
{
    class Program
    {
        static void Main(string[] args)
        {
            string carro = "Ferrari";
            float preco = 15000F;

            Console.WriteLine("O carro {0} custa R${1} reais", carro, preco);
            Console.ReadKey();
        }
    }
}
```

Exercício 06 - Tipagem implícita

```
namespace Exercicio06
{
    class Program
    {
        static void Main(string[] args)
        {
            var ano = 2018;
            var nome = "Edson Farias";

            Console.WriteLine("Nome: {0} | Ano: {1} ", nome, ano);
            Console.ReadKey();

        }
    }
}
```

Exercício 07 – Operações com Strings

```
string nome = "Gilmar Mendes";
string sobrenome = nome.Substring(6, 7);

string segundoNome = "Bruno";

//Contains
Console.WriteLine(nome.Contains("Gilmar"));

//Length
Console.WriteLine(nome.Length);

//Substring
Console.WriteLine(sobrenome);

//Concat
Console.WriteLine(string.Concat(segundoNome, "Marcel"));
```

Exercício 07 – Operações com Strings

```
//Trim
string nomeTrim = "Cesar            ";
Console.WriteLine(nomeTrim.TrimEnd());

//Split
string exSplit = "C# C++ Python";
string[] vet = exSplit.Split(' ');
for (int i = 0; i < vet.Length; i++)
    Console.WriteLine(vet[i]);

//Join
string exSplitJoin = "C# C++ Python";
string[] vetJoin = exSplitJoin.Split(' ');
Console.WriteLine(String.Join(".", vetJoin));
```

Exercício 08 – Exemplo com métodos

```
namespace Exercicio08
{
    class Program
    {

        static int somar(int a, int b)
        {
            return a + b;
        }

        static void Main(string[] args)
        {
            const int n = 2019;
            Console.WriteLine(n);
            Console.WriteLine(somar(2, 4));
            Console.ReadKey();
        }
    }
}
```

Exercício 09 - Expressões condicionais - if

```
int n1 = 200;
int n2 = 100;

if (n1 > n2)
    Console.WriteLine("n1 maior");
else
    Console.WriteLine("n2 maior");
```

Exercício 09 - Expressões condicionais - switch

```
int num = 2;
switch (num)
{
    case 1:
        Console.WriteLine("Number 1");
        break;
    case 2:
        Console.WriteLine("Number 2");
        break;
    default:
        Console.WriteLine("Valor nao encontrado");
        break;
}
```

Exercício 09 - Expressões condicionais - switch

```
string nome = "Flexa";

switch (nome)
{
    case "Gilmar":
        Console.WriteLine("Gilmar Nunes");
        break;
    case "Flexa":
        Console.WriteLine("Rui Flexa");
        break;

}
```

Loops (estruturas de repetição)

C# possui os seguintes laços (loops):

- do-while
- for
- while
- foreach-in

Exercício 10 – Loop do-while

```
//Exemplo do-while
int soma = 0;
int i = 1;
do
{
    soma += 1;
    i++;
} while (i <= 10);
```

Exercício 10 – Loop for

```
//Exemplo for
soma = 0; //reinicializando a variável soma para exemplo for
for (int j = 1; j <= 10; j++)
    soma += j;

Console.WriteLine(soma);
```

Exercício 10 – Loop while

```
//Exemplo while
soma = 0; //reinicializando a variável soma para exemplo while
int k = 1;
while (k <= 10)
{
    soma += k;
    k++;
}

Console.WriteLine(soma);
```

Exercício 11 – for-each

```
namespace Exercicio11
{
    class Program
    {
        static void Main(string[] args)
        {
            int[] vet = { 1, 2, 3 };

            foreach (int i in vet)
            {
                Console.WriteLine(i);
            }
            Console.ReadKey();
        }
    }
}
```

Exercício 12 - Tratamento de erros - try-catch - finally

```
namespace Exercicio12
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                int n = 0;
                int div = 10 / n;
                Console.WriteLine(div);
                string linguagem = "Java Love!";

                if (linguagem == "Java Love!")
                    throw new LinguagemException("Texto Exceção");

            }
            catch (LinguagemException)
            {
                Console.Write("Exceção capturada: Divisao por zero");
            }
            finally
            {
                Console.WriteLine("Valeus! Flws!");
            }
            Console.ReadKey();
        }
    }
}
```

Exercício 12 - Classe de exceção customizada

```
namespace Exercicio12
{
    public class LinguagemException: Exception
    {
        public LinguagemException(string msg)
        {
            Console.WriteLine(msg);
        }
    }
}
```

Exercício 13 – Vetores (Array)

```
namespace Exercicio13
{
    class Program
    {
        static void Main(string[] args)
        {
            int[] vet = new int[10];

            for (int i =0;i < 10; i++ )
                vet[i] = i;

            for (int i = 0; i < vet.Length; i++)
                Console.WriteLine(vet[i]);

            Console.ReadKey();

        }
    }
}
```

Exercício 14 – Ordenar um array

```
namespace Exercicio14
{
    class Program
    {
        static void Main(string[] args)
        {
            int[] vet = new int[4];
            for (int i = 4, j = 0; i > 0; i--, j++)
            {
                vet[j] = i;
            }
            Array.Sort(vet);
            for (int i = 0; i < vet.Length; i++)
            {
                Console.WriteLine(vet[i]);
            }
            Console.ReadKey();

        }
    }
}
```

Exercício 15 - Matrizes

```
namespace Exercicio15
{
    class Program
    {
        static void Main(string[] args)
        {
            int[,] m = new int[2, 2];
            m[0, 0] = 1;
            m[0, 1] = 2;
            m[1, 0] = 3;
            m[1, 1] = 4;
            for (int i = 0; i < 2; i++)
                for (int j = 0; j < 2; j++)
                    Console.WriteLine(m[i, j]);
            Console.ReadKey();
        }
    }
}
```

Exercício 16 - Enums

```
namespace Exercicio16
{
    public enum mes { janeiro, feveiro, marco, abril }

    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine((int)mes.janeiro);
            Console.ReadKey();
        }
    }
}
```

Exercício 17 – Criando classes/Trabalhando com herança

```
namespace Exercicio17
{
    class Pessoa
    {

        public Pessoa(string nome, int idade)
        {
            this.Nome = nome;
            this.Idade = idade;
        }

        public int Idade { get; set; }
        public string Nome { get; set; }

    }
}
```

Exercício 17 – Criando classes/Trabalhando com herança

O C# permite métodos com nomes iguais, mas parâmetros diferentes

```
static void MostrarDados(string nome)
{
    Console.WriteLine(nome);
}

static void MostrarDados(string nome, int idade)
{
    Console.WriteLine("Nome: {0} | Idade: {1}", nome, idade);
}

static void SetNome(string nome = "Sem nome")
{
    Console.WriteLine(nome);
}
```

Exercício 17 – Criando classes/Trabalhando com herança

O C# permite métodos com nomes iguais, mas parâmetros diferentes

Exercício 18 – Sobrescrevendo métodos e polimorfismo

REFERENCIAS

- Introdução a C# e .NET Framework. https://docs.microsoft.com/en-us/dotnet/csharp/getting-started/introduction-to-the-csharp-language-and-the-net-framework
- Apostila C# Caelum. https://www.caelum.com.br/apostila-csharp-orientacao-objetos/o-que-e-c-e-net/
- ARAUJO, Everton Coimbra de Araujo. C# e Visual Studio – Desenvolvimento de Aplicações Desktop
- JUNIOR, Carlos Olavo de Azevedo Camacho – Desenvolvimento em Camadas com C# .NET
- Introdução a C# - Marcos Castro. Disponível em https://pt.slideshare.net/mcastrosouza/introduo-a-linguagem-c-29621255