cencal centro de formação profissional para a indústria de cerâmica

Programação em C#

António Ramos | Outubro 2022

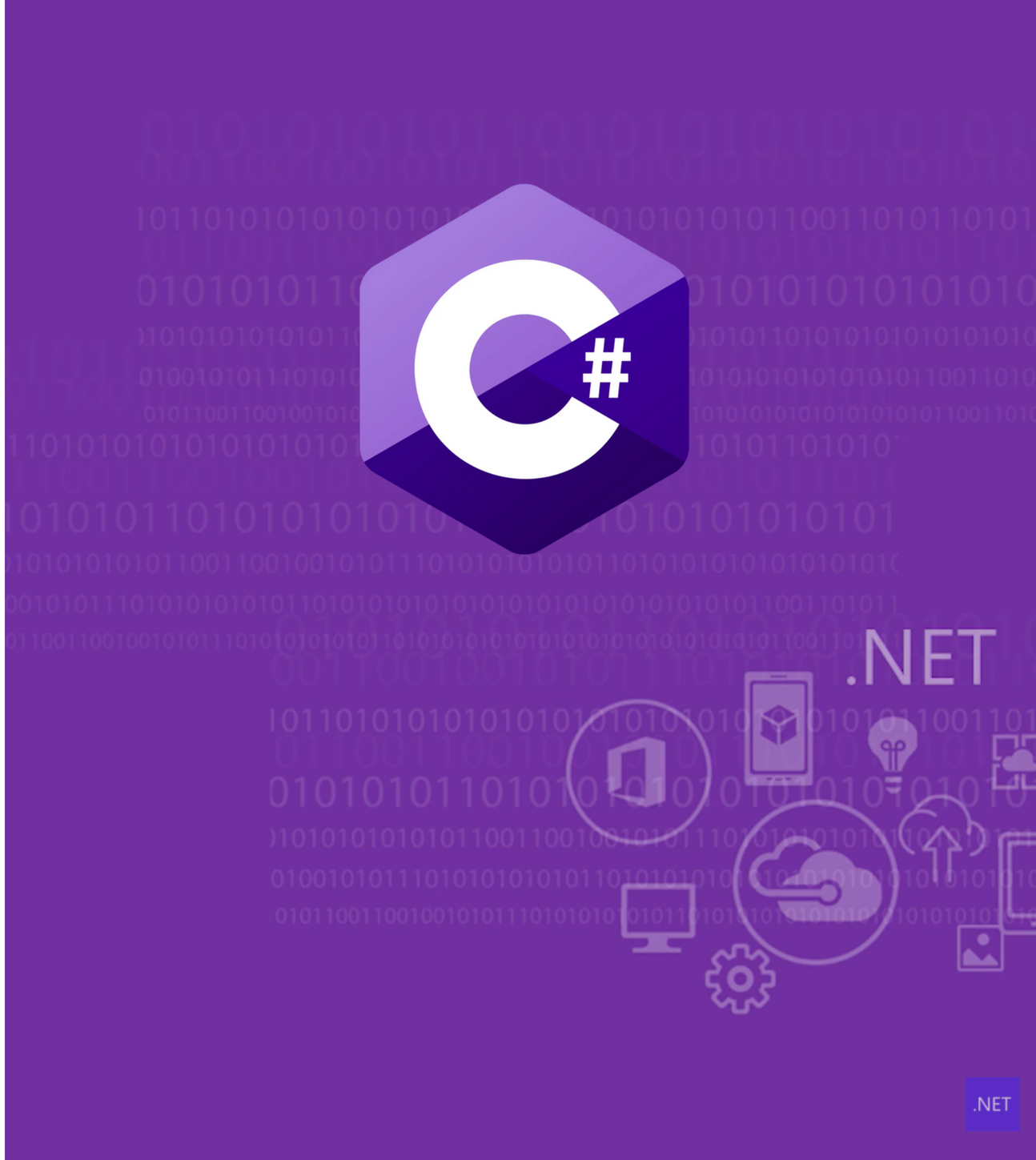
Co-Financiado por:

IEFP APICER associação portuguesa das indústrias de cerâmica e cristalaria ARO Associação Empresarial da Região Oeste REPÚBLICA PORTUGUESA POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

cencal centro de formação profissional para a indústria de cerâmica

O que é?

- Programação orientada a objetos (OOP)
- Framework .NET
- Diferentes tipos de aplicações
- Compilador



Vantagens

Programação Orientada a Objetos

Flexibilidade

Modular

Resolução de problemas

Framework .NET

Bibliotecas de pacotes

Reutilização

Desenvolvimento rápido e eficiente

Curva de Aprendizagem e Escabilidade

Curva de aprendizagem reduzida

Escalabilidade progressiva

.NET

Inicialmente o .NET era uma framework de código fechado, proprietária da Microsoft

Desde 2014 que foi disponibilizado publicamente e tornada opensource, sendo rescrita para uma framework aberta que permitisse ser executada em todos os sistemas operativos.

Tipos de Aplicações

- Web
- Mobile
- Desktop
- Microservices
- Cloud
- Machine Learning
- Game Development
- Internet of Things

Programa em C#

Namespaces

Integra blocos de código desenvolvidos

Criação personalizada de namespacess

Using directives

Simplica a utilização de namespaces sem especificação

Método Main()

Ponto de entrada das aplicações "Console"

Passagem de argumentos

Comentários

São ignoradas pelo compilador

Documenta e torna o programa mais elegivel

O que são variáveis?

- Representa uma localização na memória
- Declaração Inicial indica o tipo e o nome
- Acessíveis e modificadas através do nome

```
type variableName = value;
```

Variáveis

Int

Números inteiros desde o número
−2 147 483 648 até ao 2 147 483 647

Byte

Números inteiros desde o número
0 a 255

Variáveis

Float

Usado para números decimais, com uma aproximação de 7 casas decimais

Double

Usado para números decimais, com uma aproximação até 10 casas decimais

Variáveis

Decimal

Grande precisão até 28 casas decimais

Char

Utilizado para guardar caracteres Unicode

Variáveis

String

É utilizado para guardar uma coleção sequencial de objetos Chars.

Bool

Estados binários e apenas guarda um dos dois valores (verdadeiro ou falso).

Inicialização de Variáveis

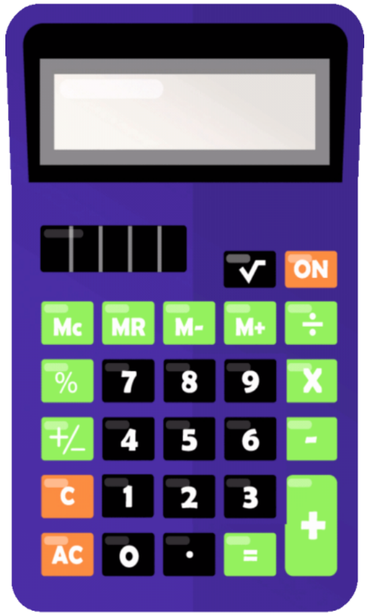
Atribuição de um valor efetuada na declaração de uma nova variável.

É boa prática as variáveis serem inicializadas

```
type variableName = value;
```

Operadores

- Atribuição (valor ou variável)
- Adição
- Subtração
- Multiplicação
- Divisão
- Módulo



Tipos de valor e tipos de referência

Tipos de valor – Variáveis que guardam o próprio valor.

Tipos de referência – Variáveis que guardam a referência do valor. Na prática, indica ao compilador onde pode ir buscar o valor.
Por defeito, têm o valor de null.

cencal centro de formação profissional para a indústria de cerâmica

Arrays, Strings e Lists

Array

Coleção / grupo de dados relacionados entre si.

```
int[] usersAge = {18, 19, 21, 26, 30};
```

Os parêntesis retos [] indicam ao compilador que esta variável de array.
Dentro dos parêntesis { }, estão os valores que serão guardados.

Array

A declaração e inicialização da variável pode ser efetuada separadamente.

```
int[] usersAge = new int[5];
usersAge = new [] {18, 19, 21, 26, 30};
```

Os valores individuais do array, são acessíveis através do seu índice.

```
Console.WriteLine(usersAge[3]);
```

String

Datatype que podem considerar como um pedaço de texto.

```
string mensagem = "Hello World";
```

Sinal (+) permite concatenar e juntar vários pedaços de texto

```
string mensagemConcat = "Hello" + "  " +"World";
```

List<T>

É uma classe genérica definida no namespace System.Collections.Generic

Semelhante ao array, mas usada para quando é necessário mais flexibilidade.

```
List<int> usersAgeList = new List<int>();
```

Para adicionarmos valores à lista usamos o método Add()

```
usersAgeList.Add(32);
```

Propriedades e Métodos

C# contém um conjunto de propriedades e métodos nativos úteis que podem ser utilizados nos Arrays, Strings e Listas.

Tanto os métodos e as propriedades são colocadas à frente do ponto final.

Para utilizar um método, colocamos os parêntesis ().

Propriedades e Métodos

- **Length**

  Array:

  - Indica o número de itens

  usersAge.Length

  String:

  - Indica o número de caracteres

Propriedades e Métodos

- **Count**

  Lista:

  - Devolve o número de itens da lista

usersAgeList.Count

Propriedades e Métodos

- **Sort()**

  Array:

  - Recebe o array como argumento e permite-o ordenar

```
Array.Sort(usersAge);
```

Propriedades e Métodos

- **IndexOf()**

Array:

- Usado para determinado se certo valor existe no Array. Caso exista este devolve o index da primeira ocorrência, se não, devolve -1

```
int[] usersAge = {18, 19, 21, 26, 30};

Array.IndexOf(usersAge, 21);
```

Propriedades e Métodos

- **Contains()**

  Listas:

  - Verifica se determinado item existe na lista. Este método retorna true/false.

```
usersAgeList.Contains(18)
```

Propriedades e Métodos

- **Add()**

  Listas:

  - Adiciona elementos à lista

```
usersAgeList.Add(32);
```

Propriedades e Métodos

- **Insert()**

  Listas:

  - Adiciona elementos à lista, numa posição especifica

```
usersAgeList.Insert(2,32);
```

Propriedades e Métodos

- **Remove()**

  Listas:

  - Remove o elemento da primeira instância onde ela ocorre

```
usersAgeList.Remove(32);
```

Propriedades e Métodos

- **Clear()**

  Listas:

  - Remove todos os elementos da lista

```
usersAgeList.Clear();
```

Propriedades e Métodos

- **Substring()**

  Strings:

  - Recebe dois argumentos, o primeiro como o index da string a remover e o segundo como a quantidade de caracteres a remover

```
string message = "Hello World";

string newMessage = message.Substring(2, 5);
```

Propriedades e Métodos

- **Equals()**

  Strings:

  - Compara se duas strings são iguais

```
stringXPTO.Equals(XPTOstring);
```

Propriedades e Métodos

- **Split()**

Strings:

- Método que divide a string baseado num separador que o utilizador define.

```
string nomes = "António, Maria, José,,Eduardo";

string [] separador= {","};

string [] arrayNomes= nomes.Split(separador, StringSplitOptions.None);
```