



# Aula 10

Prof: Henrique Augusto Maltauro

# Desenvolvendo Algoritmos

# Visual Studio

# Visual Studio

É um ambiente de desenvolvimento da Microsoft, para desenvolvimento de software especialmente dedicado ao .NET Framework e às linguagens Visual Basic, C, C++, C# e F#.

# Visual Studio

- C# (C Sharp)

É uma linguagem de programação, multiparadigma, de tipagem forte, desenvolvida pela Microsoft como parte da plataforma .NET.

A sua sintaxe orientada a objetos foi baseada no C++ mas inclui muitas influências de outras linguagens de programação, como Object Pascal e, principalmente, Java.

# Visual Studio

- Paradigma de Programação

É um meio de classificar as linguagem de programação, baseado em suas funcionalidades.

# Visual Studio

- Tipagem Forte

Determina que todas as variáveis ou constantes do código devem possuir os seus tipos definidos de formas explícitas.

# Visual Studio

- C# (C Sharp)

Importante lembrar, no C# todas as linhas de instruções, devem terminar com ponto e vírgula.

Só não se utiliza ponto e vírgula, no início e fim da definição de um bloco de código, que é representado pelas chaves.

**Abrir o Visual  
Studio Community**



# Variáveis e Constantes

# Variáveis e Constantes

Antes de qualquer coisa, dois conceitos precisam estar bem definidos na cabeça de vocês para que a gente possa dar continuidade.

A **declaração** de uma variável ou constante é o momento que eu defino um nome para ela.

A **atribuição** de uma variável ou constante é o momento que eu dou um valor para ela.

# Variáveis e Constantes

- Variável

É um objeto, gravado em alguma região da memória do computador, capaz de armazenar uma informação e de ter a sua informação alterada a qualquer momento.

# Variáveis e Constantes

- Estrutura das Variáveis em C#

Tipo da Variável

string

identificador

Nome da Variável

=

Valor

valor;

# Variáveis e Constantes

- Constante

É um objeto, gravado em alguma região da memória do computador, capaz de armazenar uma informação e a sua informação não pode ser alterada.

No C# nós temos duas maneiras diferentes de definir uma constante.

# Variáveis e Constantes

- Constante: C#

**const**

O valor da constante deve ser definido no momento da declaração dela.

**readonly**

O valor da constante pode ser definido no momento da execução do código.

# Variáveis e Constantes

- Estrutura das Constantes em C#

The diagram illustrates the syntax for declaring a constant in C#. It shows the code `const string identificador = valor;` with brackets and labels identifying each component: `const` is the **Definição da Constante** (Definition of the Constant); `string` is the **Tipo da Constante** (Type of the Constant); `identificador` is the **Nome da Constante** (Name of the Constant); and `valor` is the **Valor** (Value). The equals sign and semicolon are not labeled.

```
const string identificador = valor;
```

Definição da Constante

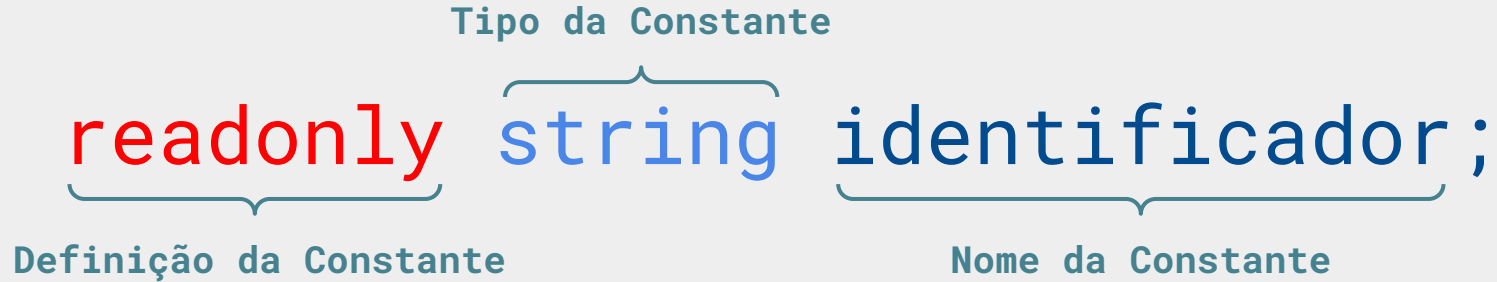
Tipo da Constante

Nome da Constante

Valor

# Variáveis e Constantes

- Estrutura das Constantes em C#

  
Definição da Constante      Tipo da Constante      Nome da Constante

  
Valor



# Tipos de Dados

# Tipos de Dados

Os dados manipulados na programação podem possuir naturezas distintas, ou seja, podem ser números, letras, frases, etc.

Dependendo da natureza de um dado, algumas operações podem ou não fazer sentido quando aplicadas a eles.

Por exemplo, não faz sentido falar em somar duas letras.

# Tipos de Dados

O tipo de um dado define o conjunto de valores ao qual aquele dado pertence, bem como o conjunto de todas as operações que podem atuar sobre qualquer valor daquele conjunto de valores.

No `C#` nós temos 18 tipos de dados `internos` diferentes, mas nós só vamos nos atentar a 15 deles, e durante as nossas aulas vamos fazer uso real de apenas 8 deles.

# Tipos de Dados

- C#: bool

Representa um valor **booleano**, que pode ter o seu valor entre **true** ou **false**.

# Tipos de Datos

- C#: bool

```
bool identificador = true;  
bool identificador = false;
```

# Tipos de Dados

- C#: byte

Representa um valor **numérico inteiro** de 8 bits, que pode ter o seu valor entre **0** a **255**.

# Tipos de Dados

- C#: byte

```
byte identificador = 0;  
byte identificador = 255;
```

# Tipos de Dados

- C#: sbyte

Representa um valor **numérico inteiro** de 8 bits, que pode ter o seu valor entre **-128** a **127**.



# Tipos de Dados

- C#: sbyte

```
sbyte identificador = -128;  
sbyte identificador = 127;
```

# Tipos de Dados

- C#: short

Representa um valor **numérico inteiro** de 16 bits, que pode ter o seu valor entre **-32.768** a **32.767**.

# Tipos de Dados

- C#: short

```
short identificador = -32768;  
short identificador = 32767;
```

# Tipos de Dados

- C#: ushort

Representa um valor **numérico inteiro** de 16 bits, que pode ter o seu valor entre **0** a **65.535**.

# Tipos de Dados

- C#: ushort

```
ushort identificador = 0;  
ushort identificador = 65535;
```

# Tipos de Dados

- C#: int

Representa um valor **numérico inteiro** de 32 bits, que pode ter o seu valor entre **-2.147.483.648** a **2.147.483.647**.

# Tipos de Datos

- C#: int

```
int identificador = -2147483648;  
int identificador = 2147483647;
```

# Tipos de Dados

- C#: uint

Representa um valor **numérico inteiro** de 32 bits, que pode ter o seu valor entre **0** a **4.294.967.295**.



# Tipos de Dados

- C#: uint

```
uint identificador = 0;  
uint identificador = 4294967295;
```

# Tipos de Dados

- C#: long

Representa um valor **numérico inteiro** de 64 bits, que pode ter o seu valor entre **-9.223.372.036.854.775.808** a **9.223.372.036.854.775.807**.

# Tipos de Datos

- C#: long

```
long identificador = -9223372036854775808;  
long identificador = 9223372036854775807;
```

# Tipos de Dados

- C#: ulong

Representa um valor **numérico inteiro** de 64 bits, que pode ter o seu valor entre **0** a **18.446.744.073.709.551.615**.

# Tipos de Datos

- C#: ulong

```
ulong identificador = 0;  
ulong identificador = 18446744073709551615;
```

# Tipos de Dados

- C#: float

Representa um valor **numérico real** de 32 bits, com precisão de até 9 dígitos, que pode ter o seu valor entre  $\pm 1.2 * 10^{-38}$  a  $\pm 3.4 * 10^{+38}$ .

# Tipos de Dados

- C#: float

```
float identificador = -15.123f;  
float identificador = 845.234f;
```

# Tipos de Dados

- C#: double

Representa um valor **numérico real** de 64 bits, com precisão de até 17 dígitos, que pode ter o seu valor entre  $\pm 2.2 * 10^{-308}$  a  $\pm 1.7 * 10^{+308}$ .



# Tipos de Datos

- C#: double

```
double identificador = -15.123d;  
double identificador = 845.234d;
```

# Tipos de Dados

- C#: decimal

Representa um valor **numérico real** de 128 bits, com precisão de até 29 dígitos, que pode ter o seu valor entre  $\pm 1.0 * 10^{-28}$  a  $\pm 7.9 * 10^{+28}$ .

# Tipos de Datos

- C#: decimal

```
decimal identificador = -15.123m;  
decimal identificador = 845.234m;
```

# Tipos de Dados

- C#: char

Representa uma **cadeia de caracteres**, ou seja, um texto de 16 bits, isso dá um tamanho máximo de 1 caractere, definido sempre com aspas simples.

# Tipos de Dados

- C#: char

```
char identificador = 'a';
```

# Tipos de Dados

- C#: string

Representa uma **cadeia de caracteres**, ou seja, um texto, que pode ser definido como um objeto que representa uma coleção sequencial de **char**.

O seu tamanho máximo é de 2 GB, o que dá mais ou menos 1 bilhão de caracteres, definido sempre com aspas duplas.

# Tipos de Dados

- C#: string

```
string identificador = "texto";
```

# Tipos de Dados

- C#: object

Representa um objeto, que pode ser, literalmente, qualquer coisa.

Tudo dentro do C# é um objeto.



# Tipos de Dados

- C#: object

```
object identificador = "texto";  
object identificador = 'a';  
object identificador = 845.234f;  
object identificador = 5414;
```

# Tipos de Dados

No momento, os tipo de dados em `C#` que vamos ter maior uso nas nossas aulas serão:

- `bool`
- `int`
- `long`
- `float`
- `double`
- `decimal`
- `char`
- `string`