Apresentações

Lucas Rodrigues Monteiro

Bacharel em Sistemas da Informação e Analista de Sistemas

lucas.monteiro@educacao.senac.rio

21 97024-9609

Fórum: www.fasim.com.br

GitHub lucasrm1981

Currículo Lates

LinkedIn

> Variáveis no C#

> Regras para nomes de variáveis

A regra para nomear uma variável é que o nome dela sempre comece por uma letra ou _. No meio do nome podem-se usar números, mas não se devem usar caracteres especiais e também não pode ser uma palavra reservada.

Entende-se por palavras reservadas os comandos do C# e que são facilmente identificadas quando digitadas, por ficarem da cor azul.

Exemplos de palavras que não podem ser utilizadas são: if, for, while, string e etc.

> Variáveis no C#

> Modificadores

No caso das varáveis, os modificadores definem a visibilidade delas, se elas poderão ser acessadas por outras classes que não seja a sua própria, se serão acessadas somente por classes derivadas a classe que ela está e assim por diante.

Exemplo:

```
public int numero;
```

> Variáveis no C#

> Modificadores

No C# existem os seguintes modificadores:

Modificador	Funcionamento
public	O acesso não é restrito.
protected	O acesso é limitado às classes ou tipos derivados daclasse que a variável está.
Internal	O acesso é limitado ao conjunto de módulos(assembly) corrente.
protected internal	O acesso é limitado ao conjunto corrente ou tiposderivados da classe recipiente.
private	O acesso é limitado à classe que a variável está.

Quando uma variável é declarada sem um modificador de acesso o compilador atribui o modificador padrão private.

> Tipos de Dados

C# é uma linguagem de programação fortemente tipada, isso significa que todas as variáveis e objetos devem ter um tipo declarado.

Os tipos de dados são divididos em value types e reference types.

Os value types são derivados de System.ValueType e reference types são derivados de System.Object.

O valor atribuído a uma variável deve estar de acordo com o seu tipo declarado.

Por exemplo uma variável declarada com o tipo int não pode receber um valor numérico de ponto flutuante (de tipo double, float ou decimal).

> Variáveis no C#

> Variáveis Value Type

As variáveis value type contém dentro delas um valor, enquanto as reference type contém uma referência.

Isso significa que se copiar uma variável do tipo value type para dentro de outra o valor é copiado e, se o mesmo for feito com uma do tipo reference type será copiado apenas a referência do objeto.

Dentro de Value Type existem duas categorias: struct e enum.

- Struct: é dividida em tipos numéricos, bool e estruturas personalizadas pelo usuário.

> Variáveis no C#

> Tipos numéricos

_ struct

Tipo de dados	Intervalo
byte	0 ..255
sbyte	-128 ..127
short	-32,768 ..32,767
ushort	0 ..65,535
int	-2,147,483,648 ..2,147,483,647
uint	0 ..4,294,967,295
long	-9,223,372,036,854,775,808..9,223,372,036,854,775,807
ulong	0 ..18,446,744,073,709,551,615
float	-3.402823e38 ..3.402823e38
double	-1.79769313486232e308 ..1.79769313486232e308
decimal	-79228162514264337593543950335..79228162514264337593543950335
char	U+0000 .. U+ffff

> Variáveis no C#

> Variáveis Value Type

_ enum

permite criar um tipo que é formado por várias constantes.

Normalmente é usada quando em algum momento existe a necessidade de um atributo que pode ter múltiplos valores, como por exemplo, em uma aplicação de venda que tem a tela de pedidos: cada pedido tem a sua situação que pode ser Aberto, Faturado e Cancelado.

Para criar a classe Pedido, o tipo do atributo situação será int

> Variáveis no C#

> Variáveis Value Type

_ enum

```
class Pedido
    {
        public int numero;
        public DateTime dataHora;
        public int situacao;

    }
```

> Variáveis no C#

> Variáveis Value Type

_ enum

```
enum Situacao
  {
    Aberto,
    Faturado,
    Cancelado
  }
}
```

> Variáveis no C#

> Variáveis Value Type

_ enum

```
class Program
{
    static void Main(string[] args)
    {
        Pedido pedido = new Pedido();

        pedido.numero = 1;
        pedido.dataHora = DateTime.Now;
        pedido.situacao = (int) Situacao.Faturado;

        Console.WriteLine("Número do pedido: "
         + pedido.numero.ToString());
        Console.WriteLine("Hora: "
         + pedido.dataHora.ToString());
        Console.WriteLine("Situação: "
         + pedido.situacao.ToString());

        Console.ReadLine();

    }
}
```

> Variáveis Reference Types

As variáveis reference type armazenam apenas a referência do objeto.

Os tipos de referência são: class, interface, delegate, object, string e Array.

Tipo object: todos os tipos são derivados da classe Object, sendo assim é possível converter qualquer tipo para object.

Tipo string: é utilizado para se armazenar caracteres e uma string deve estar entre aspas.

```
string nome =
“SENAC";
```

Para concatenar uma ou mais strings é usado o sinal de +

```
string a =
"C#";
string b =
".net";

string c = a
+ b;
```

> Variáveis no C#

> Variáveis Reference Types

Outro recurso também é a possibilidade de se obter um determinado caractere de uma string

```
string a = "C#";
string b = ".net";

string c = a + b;

char d = c[3];
```

A variável d vai ficar com o valor da letra n que é a letra que está no índice 3, conforme informado no código.

> Variáveis no C#

> Variáveis Reference Types

Outro recurso também é a possibilidade de se obter um determinado caractere de uma string

```
char d =
"C#.net"[3];
```

Pegando um caractere de uma palavra.

> Variáveis Reference Types - Variáveis no C#

> Tipo array

Utilizado ao trabalhar com coleções de dados que possuem um tamanho previamente definido, armazenam múltiplas variáveis de um mesmo tipo.

A grande vantagem do array está no fato de serem estruturas indexadas, ou seja, os itens do array podem ser acessados através de índices, o que garante grande velocidade para essas operações.

```
tipo[] nomeDoArray = {item, item, item};
```

É adicionado colchetes [] após o tipo da variável e o seu conteúdo fica entre chaves {}, separado por vírgula ,.

Por exemplo, para um array do tipo string contendo formas geométricas, a declaração seria da seguinte forma:

```
string[] array = {"Quadrado", "Círculo", "Triângulo", "Retângulo"};
```

> Variáveis Reference Types - Variáveis no C#

> Tipo array

No código acima criamos um array do tipo string contendo as seguintes formas: "Quadrado", "Círculo", "Triângulo" e "Retângulo".
Agora para obter algum dos itens armazenados no array, buscamos o seu valor pelo seu índice no array.

```
string[] array = {"Quadrado", "Círculo", "Triângulo", "Retângulo"};
string forma = array[2];
```

Lembrando que um índice começa do zero, no exemplo acima a variável forma irá buscar o valor do terceiro índice do array, retornando o valor "Triângulo".

```
string[] frutas = {“Laranja”, “Pera”, “Banana”};
Console.WriteLine(frutas.Length);
```

Para se obter o tamanho de um array deve-se usar a propriedade Length. Com essa propriedade se obtém o número de índices em um array.

> Constantes no C#

> constantes

Em algumas situações trabalhamos com algum valor que não precisa ser alterado. Por exemplo uma medida que é padrão. Em casos como esses, utilizamos constantes, que tem características semelhantes a uma variável, mas que tem de receber um valor em sua declaração que não será alterado

```
const double medida = 10.5;
```

Para declarar uma constante usamos a palavra reservada const antes de informar seu tipo e seu nome.

> Visualização e Entrada de Dados

Quando desejamos passar alguma informação para o usuário pelo console utilizamos o comando Console.WriteLine();

```
Console.WriteLine("Type a number, and then press Enter");
int num = Convert.ToInt16(Console.ReadLine());
```

Quando desejamos receber alguma informação do usuário pelo console utilizamos o comando Console.ReadLine();

```
Console.WriteLine("Type a nome, and then press Enter");
string nome = Console.ReadLine();
```

> Links de referência

https://docs.microsoft.com/pt-br/dotnet/csharp/whats-new/csharp-version-history

https://www.microsoft.com/pt-br/store/collections/visualstudio?atc=true&icid=SMB_CP1_visualstudio&rtc=2

https://visualstudio.microsoft.com/pt-br/vs/pricing/?tab=individual

https://visualstudio.microsoft.com/pt-br/vs/community/