Técnicas de Programação Visual

Matheus Andreoli 2ºY Nº20

C#

Tipos:

**Tipos de referência:** variáveis de tipos de referência armazenam referências a seus dados, o último sendo conhecido como objetos. Com os tipos de referência, é possível que duas variáveis se refiram ao mesmo objeto e, portanto, possíveis operações em uma variável afetem o objeto referenciado pela outra variável.

**Tipos de valor:** As variáveis de tipos de valor contêm diretamente seus dados.

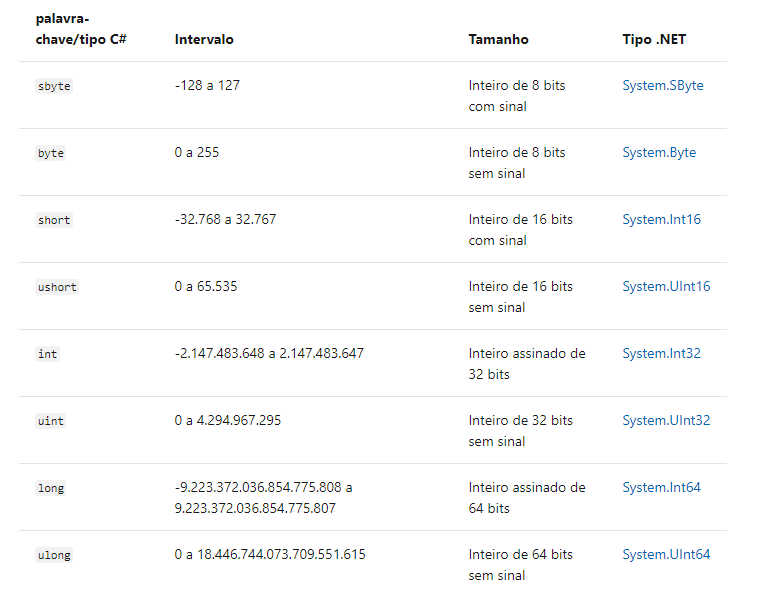
Os tipos de valor do C# são divididos em tipos simples, tipos de enum, tipos struct e tipos de valor anulável. Os tipos de referência do C# são divididos em tipos de classe, tipos de interface, tipos de matriz e tipos delegados.

O esboço a seguir fornece uma visão geral do sistema de tipo c#.

* Tipos de valor
  + Tipos simples
    - Integral com sinal: sbyte, short, int, long
    - Integral sem sinal: byte, ushort, uint, ulong
    - Caracteres Unicode: char
    - Ponto flutuante binário de IEEE: float, double
    - Ponto flutuante decimal de alta precisão: decimal
    - Booliano: bool
  + Tipos de enum
    - Tipos definidos pelo usuário do formulário enum E {...}
  + Tipos struct
    - Tipos definidos pelo usuário do formulário struct S {...}
  + Tipos de valor anuláveis
    - Extensões de todos os outros tipos de valor com um valor null
* Tipos de referência
  + Tipos de aula
    - Classe base definitiva de todos os outros tipos: object
    - Cadeia de caracteres Unicode: string
    - Tipos definidos pelo usuário do formulário class C {...}
  + Tipos de interface
    - Tipos definidos pelo usuário do formulário interface I {...}
  + Tipos de matriz
    - Unidimensional e multidimensional, por exemplo, int[] e int[,]
  + Tipos delegados
    - Tipos definidos pelo usuário do formulário delegate int D(...)

<https://docs.microsoft.com/pt-br/dotnet/csharp/tour-of-csharp/types-and-variables>

Tamanhos:

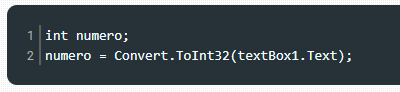


<https://docs.microsoft.com/pt-br/dotnet/csharp/language-reference/builtin-types/integral-numeric-types>

Método Convert:

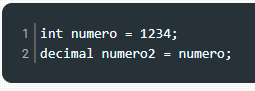
Uma situação comum para o desenvolvedor é a conversão de valores em uma aplicação, por exemplo, quando possui um formulário onde o usuário digita as informações e as mesmas são passadas para variáveis ou atributos de objetos para depois serem gravadas no banco de dados ou realizar alguma outra operação.

No exemplo, os dados vindos do formulário serão do tipo String e no caso de um campo que contenha número precisará ser convertido para inteiro para poder ser armazenado na variável. O conteúdo poderá ser convertido para inteiro utilizando o código:



Tipos de conversão:

1. **Conversão implícita:** é aquela em que não é necessário nenhum código especial, pois se trata de uma conversão segura, sem risco de perda de dados.

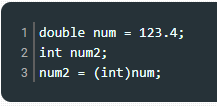


A **Tabela a seguir** contém os tipos que permitem conversões implícitas.

|  |  |
| --- | --- |
| **De** | **Para** |
| **sbyte** | short, int, long, float, double, ou decimal |
| **Byte** | short, ushort, int, uint, long, ulong, float, double, ou decimal |
| **short** | int , long, float, double, oudecimal |
| **ushort** | int, uint, long, ulong, float, double, ou decimal |
| **int** | long, float, double, oudecimal |
| **uint** | long, ulong, float, double, oudecimal |
| **Long** | float, double, oudecimal |
| **char** | ushort , int, uint, long, ulong, float, double, ou decimal |
| **float** | double |
| **ulong** | float, double, ou decimal |

1. **Conversão explícita (casts):**esse tipo de conversão necessita de um operador. É realizado quando há a necessidade de se converter um valor e pode ocorrer perda de informações.

Para fazer um cast, deve ser informado o tipo entre parênteses na frente da variável que será convertida

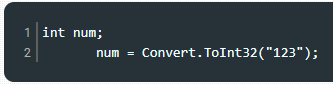


A **Tabela a seguir** contém as possíveis conversões explicitas que podem ser realizadas.

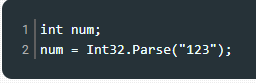
|  |  |
| --- | --- |
| **De** | **Para** |
| **sbyte** | byte, ushort, uint, ulong, ou char |
| **Byte** | Sbyte ou char |
| **short** | sbyte, byte, ushort, uint, ulong, ou char |
| **ushort** | sbyte, byte, short, ou char |
| **int** | sbyte, byte, short, ushort, uint, ulong ou char |
| **uint** | sbyte, byte, short, ushort, int ou char |
| **Long** | sbyte, byte, short, ushort, int, uint, ulong ou char |
| **ulong** | sbyte, byte, short, ushort, int, uint, long ou char |
| **char** | sbyte, byte ou short |
| **float** | sbyte, byte, short, ushort, int, uint, long, ulong, char ou decimal |
| **double** | sbyte, byte, short, ushort, int, uint, long, ulong, char, float ou decimal |
| **decimal** | sbyte, byte, short, ushort, int, uint, long, ulong, char, float ou double |

1. **Conversões definidas pelo usuário:**são conversões realizadas por métodos criados pelos desenvolvedores que contém o código que converta as informações para a maneira desejada.
2. **Conversões com classes auxiliares:**são as conversões em que, por exemplo, o desenvolvedor utiliza a classe System.Convert para converter para o tipo desejado ou o método Parse dos tipos de dados.

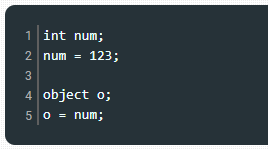
Classe Convert:



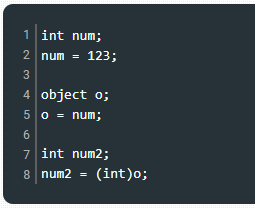
Método Parse:



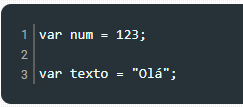
1. **Boxing**: é uma conversão implícita que converte um valor de tipo de valor para um tipo de objeto, ou seja, de um Value Type para Reference Type.



1. **Unboxing** é quando acontece o inverso, ou seja, quando há uma conversão de um tipo de objeto para um tipo de valor, de Reference Type para Value Type.



1. **Var**: essa palavra reservada pode ser usada como um tipo de dados para declaração de variáveis. Quando ela é usada, o compilador irá escolher qual é o tipo de dados mais apropriado para a variável.



<https://www.devmedia.com.br/conversao-de-tipos-de-variaveis-no-csharp/29849>