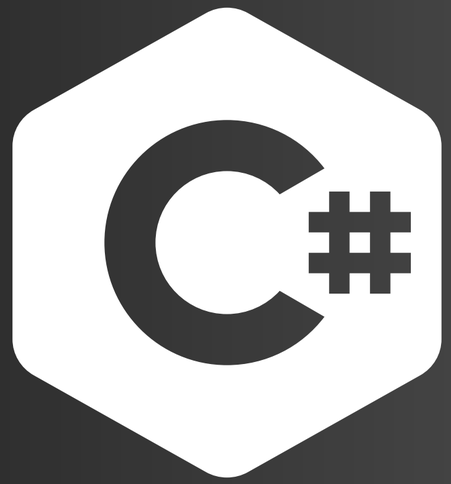


Concatenação e Conversão de Tipos de Dados

Prof. Wanderson Timóteo
www.wandersontimoteo.com.br

Wanderson Timóteo

O que vamos aprender

- Concatenação;
- Operador GetType();
- Palavra-chave typeof;
- Método Parse;
- Método ToString;
- Método Convert;

O que vamos aprender

- **Concatenação;**
- ~~Operador GetType();~~
- ~~Palavra-chave typeof;~~
- ~~Método Parse;~~
- ~~Método ToString;~~
- ~~Método Convert;~~

Concatenação

Concatenação é o processo de unir duas ou mais strings, ou cadeias de caracteres, em uma única string. Em programação, a concatenação é amplamente utilizada para combinar palavras, frases, ou valores de variáveis em uma mesma sequência de texto. Essa técnica é útil em diversos contextos, como para exibir mensagens completas, construir URLs, criar sentenças dinâmicas e muito mais.

```
string nome = "Wanderson";
string saudacao = "Olá, " + nome + "! Bem-vindo.";
Console.WriteLine(saudacao);
```

Interpolação de strings ($)

Em C#, permite incorporar variáveis diretamente em uma string.

```
string nome = "João";
string saudacao = $"Olá, {nome}! Bem-vindo.";
Console.WriteLine(saudacao);
```

Métodos como String.Concat e String.Join, são alternativas que permitem unir múltiplas strings ou arrays.

Wanderson Timóteo

Wanderson Timóteo

O que vamos aprender

- ~~Concatenação;~~
- **Operador GetType();**
- ~~Palavra-chave typeof;~~
- ~~Método Parse;~~
- ~~Método ToString;~~
- ~~Método Convert;~~

Operador GetType()

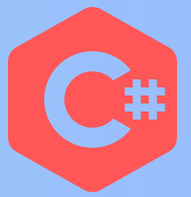
O método GetType() é usado para obter o tipo da instância de um objeto em tempo de execução.

```
int numero = 5;
Console.WriteLine(numero.GetType());
```

Wanderson Timóteo

O que vamos aprender

- ~~Operador GetType();~~
- **Palavra-chave typeof;**
- ~~Método Parse;~~
- ~~Método ToString;~~
- ~~Método Convert;~~



Palavra-chave typeof

A palavra-chave typeof é usada para obter o tipo de uma classe ou estrutura em tempo de compilação. No entanto, ele não verifica o tipo de uma instância, mas sim de um tipo em si.

```
Console.WriteLine(typeof(int));
```

O que vamos aprender

- ~~Operador GetType();~~
- ~~Palavra-chave typeof;~~
- **Método Parse;**
- ~~Método ToString;~~
- ~~Método Convert;~~

Método Parse

O método Parse é um método estático que pertence a tipos de dados específicos, como int, double, decimal, entre outros. Ele converte uma string em um valor do tipo especificado e é muito utilizado quando temos certeza de que a string é um valor válido que pode ser convertido.

```
string valorTexto = “100";
Console.WriteLine("A variável valorTexto é do tipo: " + valorTexto.GetType());

int numero = int.Parse(valorTexto);
Console.WriteLine(numero);
Console.WriteLine("A variável numeroParse é do tipo: " + numero.GetType());
```

Wanderson Timóteo

Método Parse

Características do Parse:

O método Parse é projetado para conversão de strings em tipos de dados primitivos, como int, double, float, entre outros. Ele interpreta a string como um valor do tipo especificado, mas exige que a string esteja no formato correto para o tipo de dado de destino.

Por exemplo:

- int.Parse("123") converte a string "123" para um valor int.
- double.Parse("123.45") converte a string "123.45" para um valor double.

Caso a string contenha um valor fora do formato esperado, como caracteres alfabéticos em uma string que deveria ser numérica ("abc" para int.Parse), uma exceção FormatException é lançada. Por isso, o Parse é mais seguro quando você tem certeza de que a string está no formato correto para o tipo de dado desejado.

Wanderson Timóteo

O que vamos aprender

- ~~Operador GetType();~~
- ~~Palavra-chave typeof;~~
- ~~Método Parse;~~
- **Método Convert;**
- ~~Método ToString;~~

Método Convert

O método Convert faz parte da classe estática System.Convert e permite a conversão entre uma variedade maior de tipos de dados. Ele tenta converter um objeto (não necessariamente uma string) para o tipo especificado.

```
string valorEmTexto = “100";
Console.WriteLine("A variável valorTexto é do tipo: " + valorEmTexto.GetType());

int numeroConvertido = Convert.ToInt32(valorEmTexto);
Console.WriteLine(numeroConvertido);
Console.WriteLine("A variável numeroConvertido é do tipo: " + numeroConvertido.GetType());
```

Método Convert

Características do Parse:

- Oferece uma gama mais ampla de métodos como Convert.ToInt32, Convert.ToDouble, Convert.ToDecimal, etc.
- Aceita valores null e retorna 0 no caso de tipos numéricos quando null é passado, diferentemente do Parse, que lança uma exceção.
- Lança exceções de acordo com o tipo do valor fornecido, como FormatException, InvalidCastException, entre outras, caso o valor não possa ser convertido.

Wanderson Timóteo

Principais Diferenças entre Parse e Convert

Característica	Parse	Convert
Uso principal	Apenas para conversão de strings para um tipo	Para converter diferentes tipos de dados
Classe	Método específico do tipo de dado (ex.: int)	Métodos da classe System.Convert
Valores nulos	Lança exceção se a string for null	Converte null para o valor padrão (ex.: 0 para tipos numéricos)
Exceções	Lança FormatException se a string não for válida	Pode lançar diferentes exceções, dependendo do tipo de dado
Flexibilidade	Menos flexível (somente strings)	Mais flexível (aceita vários tipos de entrada)

Prof. Wanderson Timóteo E-mail: wanderson.timoteo@mt.senac.br

Wanderson Timóteo

O que vamos aprender

- ~~Operador GetType();~~
- ~~Palavra-chave typeof;~~
- ~~**Método Parse;**~~
- **Método ToString;**
- ~~Método Convert;~~

Método ToString

O método ToString converte um valor de qualquer tipo em uma representação de string. Ele é um método muito flexível, disponível para quase todos os tipos, incluindo tipos primitivos e não primitivos.

Método ToString

Características do ToString:

1. Converte Qualquer Tipo para String;
2. Personalização de Formato;
3. Uso em Interpolação de Strings;
4. Evita Exceções em Conversões;
5. Cultura e Formato;

Wanderson Timóteo

Método ToString

1. Converte Qualquer Tipo para String

O ToString é herdado de System.Object, a classe base de todos os tipos em C#. Isso significa que quase todos os tipos de dados podem ser convertidos para string chamando ToString.

```
int numero = 123;
Console.WriteLine(numero.GetType());

string numeroComoString = numero.ToString();
Console.WriteLine(numeroComoString.GetType());
```

Wanderson Timóteo

Método ToString

2. Personalização de Formato

Para tipos numéricos e DateTime, o ToString permite que você defina um formato personalizado. Por exemplo, para datas e valores monetários, ele aceita padrões de formatação que permitem exibir o valor de forma específica.

```
DateTime dataAtual = DateTime.Now;
Console.WriteLine(dataAtual);

string dataFormatada = dataAtual.ToString("dd/MM/yyyy");
Console.WriteLine(dataFormatada);
```

Método ToString

3. Uso em Interpolação de Strings

ToString é automaticamente chamado durante a interpolação de strings, quando você insere variáveis dentro de uma string usando $"".

```
double preco = 29.99;
Console.WriteLine($"O preço é: {preco.ToString("C2")}");
```

O "C" (de Currency) instrui o ToString a formatar o valor numérico como uma moeda, de acordo com a cultura ou localização padrão do sistema. Isso significa que o símbolo da moeda (como $, €, R$, etc.) e o formato numérico variarão conforme o idioma e o local configurados. O número "2" após o "C" indica o número de casas decimais que devem ser exibidas. No caso de "C2", o valor será exibido com duas casas decimais, comum em representações de valores monetários.

Método ToString

4. Evita Exceções em Conversões

Ao contrário do Parse, o ToString não lança exceções. Ele pode retornar valores padrão por exemplo, null.ToString() não lança exceção diretamente, mas você deve ter cuidado ao chamá-lo em variáveis null.

Wanderson Timóteo

Método ToString

5. Cultura e Formato

ToString pode aceitar um parâmetro de CultureInfo para adaptar a saída de acordo com configurações de localização, como idioma e formato monetário.

```
//using System.Globalization;
decimal valor = 1234.56m;
string valorUS = valor.ToString("C", new System.Globalization.CultureInfo("en-US"));
Console.WriteLine(valorUS);

string valorBR = valor.ToString("C", new System.Globalization.CultureInfo("pt-BR"));
Console.WriteLine(valorBR);
```

Wanderson Timóteo

Referências

- https://learn.microsoft.com/pt-br/dotnet/api/system.convert.tostring?view=net-8.0
- https://learn.microsoft.com/pt-br/dotnet/api/system.datetime?view=net-8.0
- https://learn.microsoft.com/pt-br/dotnet/visual-basic/programming-guide/concepts/linq/concatenation-operations
- https://learn.microsoft.com/pt-br/dotnet/core/tutorials/cli-templates-create-project-template
- https://learn.microsoft.com/pt-br/visualstudio/get-started/csharp/tutorial-console?view=vs-2022
- https://learn.microsoft.com/pt-br/visualstudio/debugger/debug-using-the-just-in-time-debugger?view=vs-2022
- https://learn.microsoft.com/pt-br/dotnet/standard/managed-execution-process
- https://learn.microsoft.com/pt-br/dotnet/core/tools/