C# Fundamentals

Métodos e funções

Deal

Métodos e funções:

Métodos (Guia de Programação em C#)

08/03/2021 • 9 minutos para o fim da leitura •

Um método é um bloco de código que contém uma série de instruções. Um programa faz com que as instruções sejam executadas chamando o método e especificando os argumentos de método necessários. No C#, todas as instruções executadas são realizadas no contexto de um método.

O `Main` método é o ponto de entrada para cada aplicativo C# e é chamado pelo Common Language Runtime (CLR) quando o programa é iniciado. Em um aplicativo que usa instruções de nível superior, o `Main` método é gerado pelo compilador e contém todas as instruções de nível superior.

Métodos e funções:

Assinaturas de método

Os métodos são declarados em uma classe, structou interface especificando o nível de acesso, como `public` ou `private` , modificadores opcionais, como `abstract` ou `sealed` , o valor de retorno, o nome do método e qualquer parâmetro de método. Juntas, essas partes são a assinatura do método.

> ⓘ **Importante**
>
> Um tipo de retorno de um método não faz parte da assinatura do método para fins de sobrecarga de método. No entanto, ele faz parte da assinatura do método ao determinar a compatibilidade entre um delegado e o método para o qual ele aponta.

Os parâmetros de método estão entre parênteses e separados por vírgulas. Parênteses vazios indicam que o método não requer parâmetros. Essa classe contém quatro métodos:

C# Copiar

```
abstract class Motorcycle
{
    // Anyone can call this.
    public void StartEngine() {/* Method statements here */ }

    // Only derived classes can call this.
    protected void AddGas(int gallons) { /* Method statements here */ }

    // Derived classes can override the base class implementation.
    public virtual int Drive(int miles, int speed) { /* Method statements here */ return 1; }

    // Derived classes must implement this.
    public abstract double GetTopSpeed();
}
```

Métodos e funções:

Acesso de método

Chamar um método em um objeto é como acessar um campo. Após o nome do objeto, adicione um ponto final, o nome do método e parênteses. Os argumentos são listados dentro dos parênteses e são separados por vírgulas. Os métodos da classe `Motorcycle` podem, portanto, ser chamados como no exemplo a seguir:

C# Copiar

```
class TestMotorcycle : Motorcycle
{

    public override double GetTopSpeed()
    {
        return 108.4;
    }

    static void Main()
    {

        TestMotorcycle moto = new TestMotorcycle();

        moto.StartEngine();
        moto.AddGas(15);
        moto.Drive(5, 20);
        double speed = moto.GetTopSpeed();
        Console.WriteLine("My top speed is {0}", speed);
    }
}
```

Métodos e funções:

Parâmetros do método vs. argumentos

A definição do método especifica os nomes e tipos de quaisquer parâmetros obrigatórios. Quando o código de chamada chama o método, ele fornece valores concretos, chamados argumentos, para cada parâmetro. Os argumentos devem ser compatíveis com o tipo de parâmetro, mas o nome do argumento (se houver) usado no código de chamada não precisa ser o mesmo que o parâmetro nomeado definido no método. Por exemplo:

C# Copiar

```
public void Caller()
{
    int numA = 4;
    // Call with an int variable.
    int productA = Square(numA);

    int numB = 32;
    // Call with another int variable.
    int productB = Square(numB);

    // Call with an integer literal.
    int productC = Square(12);

    // Call with an expression that evaulates to int.
    productC = Square(productA * 3);
}

int Square(int i)
{
    // Store input argument in a local variable.
    int input = i;
    return input * input;
}
```

Métodos e funções:

- Links de documentação oficial:
- Documentação:
- https://docs.microsoft.com/pt-br/dotnet/csharp/programming-guide/classes-and-structs/methods

- https://docs.microsoft.com/pt-br/dotnet/csharp/programming-guide/classes-and-structs/local-functions


Dúvidas?