**C# e .NET**

**O que é C# (C SHARP)?**

É uma linguagem de programação

**O que é .NET?**

(DOTNET) é uma plataforma de desenvolvimento para criar diversos tipos de aplicações, utilizando algumas linguagens de programação, tais como, C# (C SHARP), VB (VISUAL BASIC) e F# (F SHARP).

**O que tem no .NET?**

* **BCL – BASE CLASS LIBRARY**

Nessa biblioteca tem um conjunto de funcionalidades para utilizar nos nossos programas.

* **CLR – COMMON LANGUAGE RUNTIME (MÁQUINA VIRTUAL)**

É uma máquina virtual que executa os programas realizados em .net, possui também o garbage collection é um processo de coletar os objetos não utilizados e são automaticamente desalocados da memória.

**Qual a diferença de .NET Framework, .NET Core e Xamarin?**

.Net Framework é usado para desenvolvimento de aplicações Windows usando o Windows Forms (Sistema de telas), WPF (Sistema de telas) e de Aplicações Web usando ASP .NET MVC.

.NET Core é usado para criar aplicações Web para diferentes tipos de sistemas operacionais (Windows, Linux e Mac).

Xamarin é um framework que permite criar aplicações nativas (As aplicações nativas são desenvolvidas para um sistema específico (Android, iOS, Windows) e são descarregadas para um dispositivo móvel a partir de uma loja de aplicações.) para Android, IOS.

**O que é aplicações console?**

Aplicações console não possuem formulários, interface gráfica, elas rodam dentro de uma janela do MSDOS. Normalmente esse tipo de aplicação não possui nenhuma interação com o usuário, possuindo certa limitação para execução de determinadas funcionalidades.

**Como criar aplicações desktop no .NET Core?**

É necessário usar outra biblioteca de tela, como React JS.

Compilação e Interpretação

O C# é uma linguagem pré-compilada que executa em uma máquina virtual. O programador escreve o código que é pré-compilado, gerado um bytecode (Common Intermediate Language (CIL)) que é rodado em uma máquina virtual, depois disso é convertido em código de máquina que é executado no computador.

**Como é a estrutura de uma aplicação em C#?**

Uma aplicação construída em uma linguagem orientada a objeto como o C#, é formada por um conjunto de classes, essas classes podem ser agrupadas em Namespace **(Agrupamento lógico de classes relacionadas)**, em uma aplicação pode ter várias classes que vão representar as entidades de negócio (Cliente, produtos, etc.), elas ficam agrupadas em um Namespace chamado Entities (Entidades), da mesma forma pode ter outras classes, como, serviços do sistema, que ficam organizados em um Namespace chamado services e a classe repositories que armazena os dados.

1. **Using System;**

É a biblioteca onde tem funções básicas de manipulações de programas.

1. **Namespace;**

É onde as classes ficam organizadas, não é obrigatório o uso, porém por questão de organização é recomendado deixar nossas classes dentro do namespace.

1. **Class program;**

Todo código fonte escrito em c# tem que estar dentro de uma classe.

1. **Static void Main (string [] args);**

É uma declaração padrão em c#, para identificar o entre point da aplicação, ou seja, o ponto de entrada, onde a execução começa.

**A tabela a seguir lista os tipos de valores internos do C#:**

**TIPOS INTERNOS (REFERÊNCIA C#)**

|  |  |
| --- | --- |
| Palavra-chave Type do C# | Tipo .NET |
| bool | System.Boolean |
| byte | System.Byte |
| sbyte | System.SByte |
| char | System.Char |
| decimal | System.Decimal |
| double | System.Double |
| float | System.Single |
| int | System.Int32 |
| uint | System.UInt32 |
| nint | System.IntPtr |
| nuint | System.UIntPtr |
| long | System.Int64 |
| ulong | System.UInt64 |
| short | System.Int16 |
| ushort | System.UInt16 |

**A tabela a seguir lista os tipos de referência interna do C#:**

**TABELA 2**

|  |  |
| --- | --- |
| Palavra-chave Type do C# | Tipo .NET |
| object | System.Object |
| string | System.String |
| dynamic | System.Object |
|  |  |

Nas tabelas anteriores, cada palavra-chave de tipo C# da coluna esquerda (exceto Nint e nuint e Dynamic) é um alias para o tipo .NET correspondente. Eles são intercambiáveis. Por exemplo, as declarações a seguir declaram variáveis do mesmo tipo:

C#

int a = 123;

System.Int32 b = 123;

Os nint nuint tipos e nas duas últimas linhas da primeira tabela são inteiros de tamanho nativo. Eles são representados internamente pelos tipos do .NET indicados, mas, em cada caso, a palavra-chave e o tipo .NET não são intercambiáveis. O compilador fornece operações e conversões para nint e nuint como tipos inteiros que ele não fornece para os tipos de ponteiro System.IntPtr e System.UIntPtr. Para obter mais informações, consulte nint e nuint tipos.

A void palavra-chave representa a ausência de um tipo. Você o usa como o tipo de retorno de um método que não retorna um valor.

**C# Orientado a objetos**

**O que é?**

Programação Orientada a Objetos (também conhecida pela sua sigla POO) é um modelo de análise, projeto e programação de software baseado na composição e interação entre diversas unidades chamadas de 'objetos'

**O que é uma classe?**

É uma descrição que abstrai um conjunto de objetos com características similares. Mais formalmente, é um conceito que encapsula abstrações de dados e procedimentos que descrevem o conteúdo e o comportamento de entidades do mundo real, representadas por objetos.

**O que é um atributo?**

Os Atributos em Programação Orientada a Objetos são os elementos que definem a estrutura de uma classe. Os atributos também são conhecidos como variáveis de classe, e podem ser divididos em dois tipos básicos: atributos de instância e de classe. Os valores dos atributos de instância determinam o estado de cada objeto.

**O que é métodos?**

É uma função é uma sub-rotina (ou procedimento ou função) associada a um objeto, e que possui acesso aos seus dados, as variáveis membro. Ele é executado por um objeto ao receber uma mensagem, como (cadastrar, excluir, alterar).

**O que é modificadores de acesso?**

São as palavras-chaves usadas para especificar a acessibilidade de um método dentro da classe.

public -> Atributos e métodos visíveis para todas as classes

private -> Atributos e métodos visíveis apenas onde foi criado

protected -> Atributos e métodos visíveis em classes onde são criados ou herdados

Exemplo:

public string nome; -> fica visível quando o objeto é instanciado.

private double valor; -> não fica visível quando o objeto é instanciado.

**O que é encapsulamento?**

Ele serve para restringir a visibilidade dos métodos e atributos de uma classe, ou seja, essa informação fica totalmente guardada (como um remédio dentro de uma capsula).

**O que é um construtor?**

É um método com o mesmo nome da classe, onde ele é executado assim que instancia um objeto.

**O que é get e set?**

Os métodos GET e SET são técnicas padronizadas para gerenciamento sobre o acesso dos atributos. Nesses métodos determinamos quando será modificado um atributo e o acesso ao mesmo, tornando o controle e modificações mais práticas e limpas, sem, contudo, precisar alterar assinatura do método usado para acesso ao atributo.

* Set = envia uma informação para um atributo
* Get = obtém a informação enviada

**O que é this?**

É usado para referenciar um atributo da classe.

**O que é herança?**

A herança é um princípio próprio à programação orientada a objetos (POO) que permite criar uma nova classe a partir de uma já existente. ... A principal vantagem da herança é a capacidade para definir novos atributos e métodos para a subclasse, que se somam aos atributos e métodos herdados.

**O que é polimorfismo?**

O Polimorfismo é um mecanismo por meio do qual selecionamos as funcionalidades utilizadas de forma dinâmica por um programa no decorrer de sua execução. Com o Polimorfismo, os mesmos atributos e objetos podem ser utilizados em objetos distintos, porém, com implementações lógicas diferentes.

**O que é abstração?**

A abstração é o ponto de partida para a criação de programas utilizando POO. Trata-se da capacidade de extrair dos personagens ou dos itens presentes no contexto, suas principais características, criando, dessa forma, objetos.