Programação Orientação a Objeto em C#
Aula 1

Introdução à Programação Orientada a Objetos

A programação orientada a objetos (POO) é um paradigma de programação que se baseia na utilização de objetos e suas interações para criar programas. Algumas das características fundamentais da POO incluem:

Abstração: a POO permite a criação de objetos que representam entidades reais ou conceituais do mundo. Esses objetos podem ter atributos (dados) e métodos (ações) que os definem e os tornam únicos.

Encapsulamento: a POO fornece uma forma de ocultar o comportamento interno dos objetos, expondo apenas o que é necessário para sua utilização. Isso ajuda a prevenir que outros objetos interfiram com seu estado interno, aumentando a segurança e a robustez do sistema.

Herança: a POO permite que as classes herdem características e comportamentos de outras classes. Isso facilita a reutilização do código, já que as classes filhas podem usar os métodos e atributos de suas classes pais sem precisar reescrevê-los.

Polimorfismo: a POO permite que objetos de diferentes classes sejam tratados de forma uniforme, desde que implementem métodos com a mesma interface. Isso significa que um mesmo método pode ser usado com diferentes tipos de objetos, aumentando a flexibilidade e a reusabilidade do código.

Classes: na POO, as classes são o meio fundamental de definição de objetos. Uma classe é um modelo para criar objetos que compartilham um conjunto de atributos e métodos.

Objetos: os objetos são as instâncias das classes e representam entidades individuais que possuem características específicas.

No geral, a POO fornece uma abordagem poderosa para a criação de software flexível, modular e extensível. Ao modelar objetos e suas interações, a POO ajuda os programadores a construir sistemas mais facilmente compreensíveis e fáceis de manter.

Classe, Objetos e a Funcionalidade de POO

Classe: é um modelo que define a estrutura e o comportamento de um objeto. Uma classe contém os atributos e métodos que definem a aparência e o comportamento do objeto.

Objeto: é uma instância de uma classe com valores específicos. Ele é criado a partir da classe e possui os mesmos atributos e métodos definidos pela classe, mas com valores específicos definidos pelo programador.

Por exemplo, uma classe "Carro" pode ter atributos como marca, modelo, cor, número de portas, entre outros, e métodos como acelerar, frear, virar, etc. Um objeto específico criado a partir dessa classe, pode ter valores como marca = "Toyota", modelo = "Corolla", cor = "prata", número de portas = 4, entre outros.

A funcionalidade da POO é modelar conceitos do mundo real em objetos e permitir a interação entre eles através de classes e métodos, proporcionando organização, reutilização de código e facilidade de manutenção em projetos complexos.

Vantagens da POO:

- reutilização de código;
- organização;
- modularidade;
- flexibilidade;
- segurança;
- manutenção;
- evolução facilitadas.

Desvantagens da POO:

- aumento de complexidade;
- sobrecarga de memória e processamento;
- dificuldades de otimização;
- curva de aprendizagem íngreme.

Declaração de Classes e Instanciar Objetos em C#

A declaração de uma classe em C# segue o seguinte padrão:

```
public class NomeDaClasse {
    // Definição dos atributos da classe
    // ...
    // Definição dos construtores da classe
    // ...
    // Definição dos métodos da classe
    // ...
}
public class Carro {
      // Atributos
      public string Marca;
      public string Modelo;
      public int Ano;

      // Construtores
      public Carro() {}
      public Carro(string marca, string modelo, int ano)
      {
      this.Marca = marca;
       this.Modelo = modelo;
      this.Ano = ano; }
      // Métodos
      public void Acelerar()
      {
      Console.WriteLine("Acelerando o carro...");
      }
      public void Frear() {
      Console.WriteLine("Freando o carro...");
```

```
    }
}
```

Para instanciar (criar um objeto) da classe "Carro" no exemplo que foi dado anteriormente, é necessário utilizar a palavra-chave "new", seguida do nome da classe e, se houver, dos parâmetros do construtor desejado. Por exemplo:

```
// Instanciando um objeto da classe Carro com o construtor padrão
Carro meuCarro = new Carro();
//Instanciando um objeto da classe Carro com o construtor que recebe parâmetros
Carro outroCarro = new Carro("Fiat", "Palio", 2010);
```

Modificadores de Acesso(Visibilidade), Atributos e Métodos

Modificadores de Acesso ou visibilidade

A visibilidade de um atributo determina onde e como ele pode ser acessado no código. Em C#, existem três níveis de visibilidade:

- "**public**": o atributo é visível para todos os códigos e pode ser acessado de qualquer lugar;
- "**private**": o atributo é visível somente dentro da própria classe e não pode ser acessado fora dela;
- "**protected**": o atributo é visível dentro da própria classe e também em subclasses que herdam dela.

Atributos

Atributos em POO são variáveis que armazenam os dados dos objetos criados a partir de uma classe.

Para definir um atributo em C#, usamos a seguinte sintaxe:

```
visibilidade tipo nomeDoAtributo;
```

Por exemplo:

```csharp
public class Pessoa {
    public string nome;       // atributo público
    private int idade;        // atributo privado
    protected string endereco;  // atributo protegido
}
```

Métodos

Métodos são blocos de código que realizam uma tarefa específica e podem ser chamados em diferentes partes do programa. Eles podem receber argumentos, realizar cálculos, manipular dados e retornar valores.

A sintaxe para definir um método em C# é a seguinte:

```
visibilidade tipoDeRetorno nomeDoMetodo(parâmetros) {
    // corpo do método
}
```

Por exemplo:

```
public class Calculadora {
    public int Soma(int a, int b) {
        int resultado = a + b;
        return resultado;
    }
}
```