

Introdução a OO. (Aula 4)

A dificuldade de programar com pensamento “OO” é certa, mas, estas dificuldades são associadas à linguagem e é comum ler em artigos e textos que Java é difícil, C++ e por aí vai, mas na realidade o que falta entender o “espírito” OO.

```
class Errada
{
    static int Numeroum, Numerodois;
    String Nome;
    public static int CalculaSoma(int Numero1, int Numero2)
    {
        return (Numero1-Numero2);
    }
    public static void SetNumeroUm(int c)
    {
        Numeroum = c;
    }
    public static void main(String a[])
    {
        Numeroum = 115;
        Numerodois = 20;
        System.out.println("A soma é: "+
CalculaSoma (Numeroum, Numerodois));
    }
}
```

Ao compilar a classe você não irá ter nenhum erro de compilação, na realidade ela irá funcionar, pois não há erros de linguagem no exemplo acima, os erros estão todos nos conceitos de O.O.

Agora a pergunta feita é: “Como podemos construir esses conceitos em O.O.?” a resposta é simples, treinando, errando, experimentando e principalmente lendo.

Certamente com prática e muita disciplina de programação com o tempo você será capaz de olhar para o programa acima e perceber todos os seus erros, chegando a me achar um incompetente e certamente será mais fácil para você entender os conceitos de um programa O.O., pois, não conheço nada mais próximo da realidade humana que a programação O.O.

Um programador O.O. termina vendo o mundo com uma visão que somente o universo da O.O. permite, quando colocamos esses óculos temos aquilo que os estudiosos chamam de “*insight*”, como digo dormimos “estruturadamente” e acordamos no mundo O.O.

Mas a dica aqui não é dormir e esperar o mundo O.O. bater em sua porta mostrando assim as maravilhas deste tipo de programação, e sim correr atrás, questionar, programar mesmo que erradamente,

pensar em Objetos ao invés de pensar em estruturas,
pensar em métodos e esquecer as funções,
e numa visão bem radical, mas realista,
o seu programa não deve possuir variáveis, ele possui atributos.

e por aí vamos, numa relação meio inocente de estar sem um destino mas com um objetivo claro, falarmos de O.O. sem medo e num linguajar simples.

POO = Programação Orientada a Objetos.

- 1º objetivo proporcionar um projeto mais simples e melhor.
- 2º objetivo permitir a reusabilidade do código.
- 3º objetivo possibilitar a reusabilidade do projeto.

A POO se fundamenta em 5 mecanismos:

Abstração: capacidade de representação

Identidade: coexistência de múltiplas entidades

Encapsulamento: ocultação de detalhes

Herança: criação de relações hierárquicas

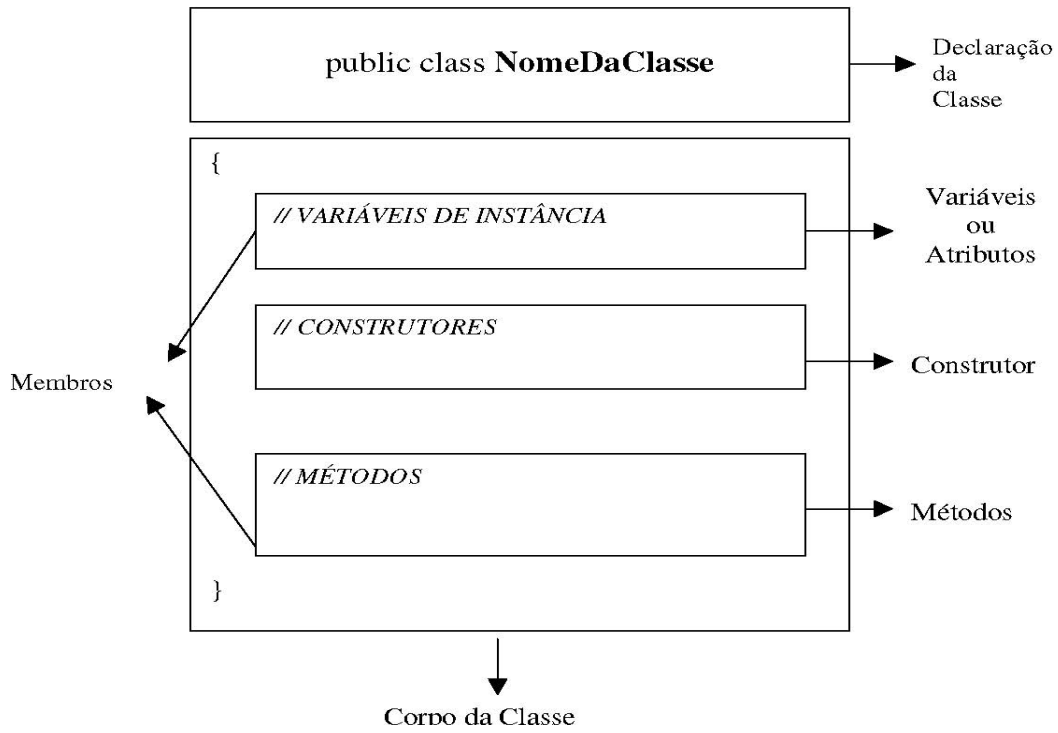
Polimorfismo: admissão de múltiplas formas

Classes

Modelos ou conceitos abstratos para definir famílias de objetos.

Podem ser interligadas estabelecendo hierarquias ou relações.

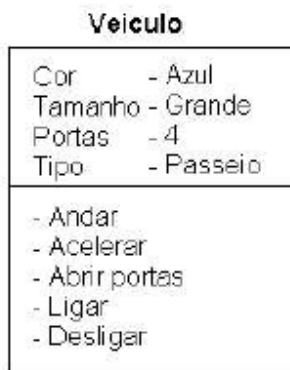
São tipos de dados definidos pelo programador.



O que é um Objeto?

Representantes de uma classe, são entidades concretas.

- Uma bicicleta, um carro, uma pessoa são objetos
- Possuem **características** (cor, tamanho) e comportamentos – **ações** - (andando, acelerando, comendo).



- **Características** são os atributos e as **ações** são os métodos dentro dos objetos.

- Objetos precisam se relacionar : não fazem nada sozinhos (alguém tem que ligar o carro)
- Chamar um método de um objeto

- Qual objeto (Carro)?
- O método (abrirPorta) ?
- Parâmetro do método (Dianteira, Direita)?

- Classe é o padrão a ser seguido pelo objeto: contém atributos e métodos comuns dos objetos (especificação do objeto)

- Exemplo: o objeto Seu Golf é uma instancia da classe Carros
Tendo uma classe criada, você pode instanciar diversas outros objetos a partir da sua classe.

