PROGRAMAÇÃO ORIENTADA A OBJETOS

Professora Lyrane Bezerra

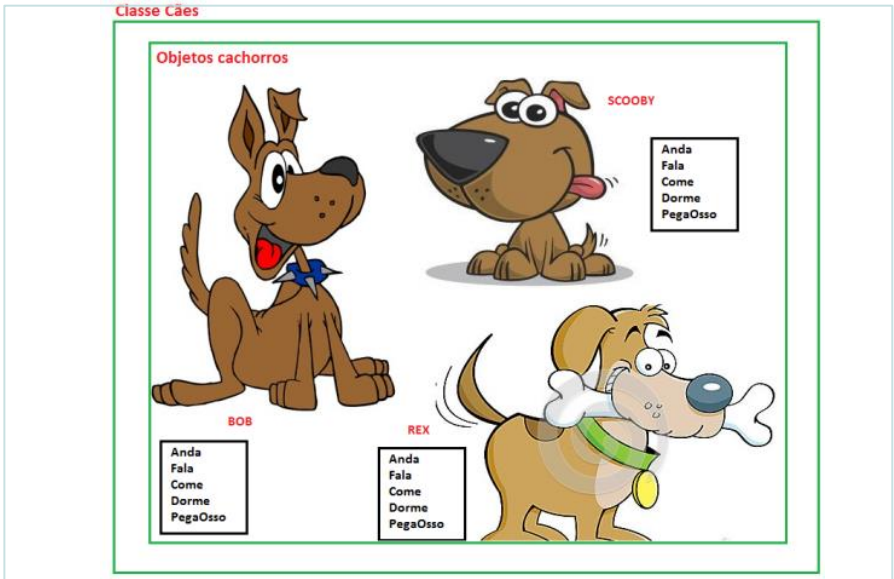
Fundamentos e Definições de Classes e Objetos

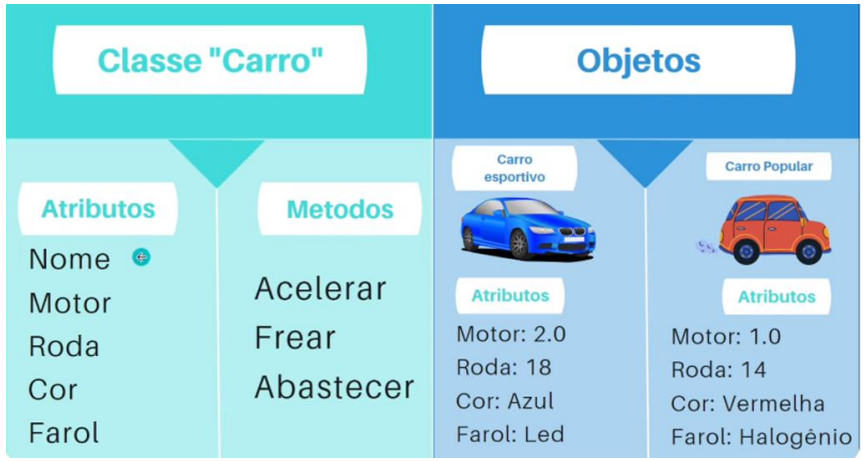
- Em POO pense em uma **classe** como um **molde** ou um **modelo** para criar algo.
- Ela define as características e os comportamentos que o objeto terá.

Atributos
(Características)

Métodos
(Comportamentos)

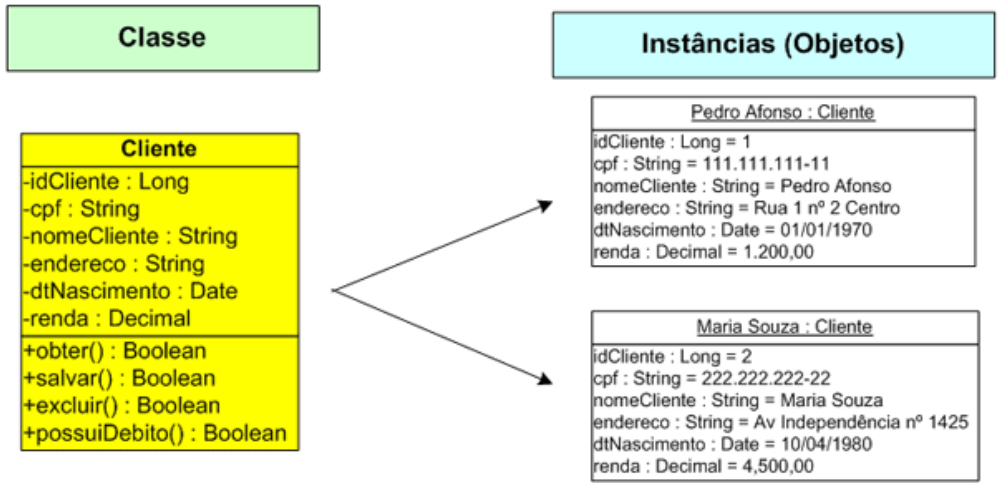
Já um **objeto** é uma **instância** concreta dessa classe, ou seja, algo que foi criado a partir do molde.


Classe Cães
Objetos cachorros
SCOOBY
Anda
Fala
Come
Dorme
PegaOsso
BOB
Anda
Fala
Come
Dorme
PegaOsso
REX
Anda
Fala
Come
Dorme
PegaOsso


Classe "Carro"
Objetos
Atributos
Nome
Motor
Roda
Cor
Farol
Metodos
Acelerar
Frear
Abastecer
Carro esportivo
Atributos
Motor: 2.0
Roda: 18
Cor: Azul
Farol: Led
Carro Popular
Atributos
Motor: 1.0
Roda: 14
Cor: Vermelha
Farol: Halogênio

Exemplo

Exemplo



Objetos (Instâncias) da classe Cliente

Classes

As classes de programação são projetos de um objeto, aonde têm características e comportamentos, ou seja, permite armazenar propriedades e métodos dentro dela.

Para construir uma classe é preciso utilizar o pilar da abstração. Uma classe geralmente representa um substantivo, por exemplo: uma pessoa, um lugar, algo que seja “abstrato”.

Agrupar vários objetos com as mesmas características e métodos em uma classe é um processo chamado de **generalização**.

Características de uma Classe

- Toda classe possui um nome;
- Possuem visibilidade, exemplo: public, private, protected;
- Possuem membros como: Características e Ações;
- Para criar uma classe basta declarar a visibilidade + digitar a palavra reservada class + NomeDaClasse + abrir e fechar chaves { }.

- Exemplo da sintaxe:

```
Public class Teste{
    //Atributos;
    //Métodos;
}
```

Exemplo

```java
public class Caes {

    public String nome;
    public int peso;
    public String corOlhos;
    public int quantPatas;

    public void falar(){
        //MÉTODO FALAR
    }

    public void andar(){
        //MÉTODO ANDAR
    }

    public void comer(){
        //MÉTODO COMER
    }

    public void dormir(){
        //MÉTODO DORMIR
    }
}
```

Observação:
Os métodos, geralmente, aparecem com o parêntese vazio no final.

Objetos

Um **objeto** é uma **instância** concreta da classe, ou seja, algo que foi criado a partir do molde.

Os objetos são características definidas pelas classes. Neles é permitido instanciar objetos da classe para inicializar os atributos e invocar os métodos.

Um **objeto** é uma extensão do conceito de objeto do mundo real, em que se podem ter coisas **tangíveis**, um **incidente** (evento ou ocorrência) ou uma **interação** (transação ou contrato).

Tipos de Objetos:

Os **objetos tangíveis** representam coisas concretas, que existem no mundo real e que têm um estado e um ciclo de vida. São as "coisas" sobre as quais o sistema precisa saber. Eles são a espinha dorsal de muitas aplicações, pois encapsulam os dados mais importantes. **Características:** Possuem **atributos** (dados) que definem seu estado, e geralmente têm métodos para acessar e modificar esses dados. São o que normalmente chamamos de "entidades" ou "modelos".

Os **objetos incidentes** representam eventos ou ocorrências que acontecem em um momento específico no tempo. Eles são geralmente imutáveis, ou seja, uma vez que são criados, seu estado não muda. Eles capturam a informação sobre algo que "aconteceu". **Características:** Descrevem um evento e suas circunstâncias. Frequentemente, têm atributos como data, hora e os objetos tangíveis envolvidos.

Os **objetos de interação** (ou objetos de serviço) representam as ações, os processos ou as lógicas de negócio que conectam os objetos tangíveis e incidentes. Eles não representam "coisas" ou "eventos" por si só, mas sim a lógica de como as coisas se comportam ou interagem entre si. **Características:** Geralmente não têm um estado próprio significativo (ou têm um estado temporário durante a execução de um método). Sua principal responsabilidade é coordenar operações, conter a lógica de negócio e servir como um ponto de acesso para funcionalidades.

Exemplos de Tipos de Objetos:

Os **objetos tangíveis:**

Em um sistema de e-commerce, Produto, Cliente e Pedido são objetos tangíveis. Um Produto tem um nome, preço e estoque. Um Cliente tem um nome, endereço e histórico de compras.

Em um sistema escolar, Aluno, Professor e Disciplina são objetos tangíveis. Um Aluno tem um nome, matrícula e notas.

Em um jogo, Personagem, Arma e Cenário são objetos tangíveis.

Os **objetos incidentes**

Em um sistema bancário, um Saque ou um Depósito são incidentes. Eles registram a data, o valor e a conta afetada.

Em uma aplicação de redes sociais, um Post ou um Like são incidentes. Eles capturam a ação do usuário em um determinado momento.

Em um sistema de rastreamento de entregas, um EventoEntrega (como "Saiu para entrega" ou "Entregue") é um incidente..

Os **objetos de interação:**

Em um sistema bancário, um GerenciadorContas seria um objeto de interação. Ele contém a lógica para transferir dinheiro entre duas contas, criando Saque e Depósito (os incidentes) no processo.

Em uma aplicação de agendamento, um AgendadorConsultas seria um objeto de interação, responsável por validar e criar uma nova consulta entre um paciente (objeto tangível) e um médico (objeto tangível), registrando a data e hora do evento.

Atributos

- Os **atributos** são as **características** ou os **dados** que um objeto possui. Eles representam o estado do objeto. Usando a classe **Carro** como exemplo, os atributos seriam **cor, marca e velocidade_maxima.** Cada objeto criado a partir dessa classe terá seus próprios valores para esses atributos (por exemplo, um carro pode ser "vermelho" e outro "preto").

- Os atributos são as propriedades de um objeto, também são conhecidos como variáveis ou campos. Essas propriedades definem o estado de um objeto, fazendo com que esses valores possam sofrer alterações.

Exemplo de atributos:

```java
public class Cachorro{

        public String nome;
        public int peso;
        public String corOlhos;
        public int quantPatas;
    }
```

```java
public class TestaCaes {

    public static void main(String[] args) {
        Cachorro cachorro1 = new Cachorro();
        cachorro1.nome = "Pluto";
        cachorro1.corOlhos = "azuis";
        cachorro1.peso = 53;
        cachorro1.quantPatas = 4;

        Cachorro cachorro2 = new Cachorro();
        cachorro2.nome = "Rex";
        cachorro2.corOlhos = "amarelo";
        cachorro2.peso = 22;
        cachorro2.quantPatas = 3;

        Cachorro cachorro3 = new Cachorro();
        cachorro3.nome = "Bob";
        cachorro3.corOlhos = "marrom";
        cachorro3.peso = 13;
        cachorro3.quantPatas = 4;

    }

}
```

Métodos

Os métodos são ações ou procedimentos, onde podem interagir e se comunicarem com outros objetos. A execução dessas ações se dá através de mensagens, tendo como função o envio de uma solicitação ao objeto para que seja efetuada a rotina desejada.
Como boas práticas, é indicado sempre usar o nome dos métodos declarados como verbos, para que quando for efetuada alguma manutenção seja de fácil entendimento.

Os **métodos** são as **ações** ou os **comportamentos** que um objeto pode realizar. Eles definem o que o objeto faz. Para a classe Carro, os métodos poderiam ser acelerar(), frear() e ligar_motor(). Quando você chama o método acelerar() em um objeto meu_carro, ele executa a ação de aumentar a velocidade do carro.

Exemplo de Métodos

```java
class Cachorro{
    int tamanho;
    String nome;


    void latir(){
        if(tamanho > 60)
            System.out.println("Wooof, Wooof!");
        else if(tamanho > 14)
            System.out.println("Ruff!, Ruff!");
        else
            System.out.println("Yip!, Yip!");
    }
}
```

```java
public class Testa_Cachorro {

    public static void main(String[] args) {

        Cachorro bob = new Cachorro();
        bob.tamanho = 70;
        Cachorro rex = new Cachorro();
        rex.tamanho = 8;
        Cachorro scooby = new Cachorro();
        scooby.tamanho = 35;

        bob.latir();
        rex.latir();
        scooby.latir();

    }
}
```

Construtores

- Um **construtor** é um método especial que é executado **automaticamente** no momento em que um objeto é criado.

- Sua principal função é **inicializar** os atributos do objeto, garantindo que ele comece a vida em um estado válido. Na maioria das linguagens de programação, o construtor tem o mesmo nome da classe. Por exemplo, em Java ou C++, o construtor da classe Carro se chamaria Carro().

- Você pode passar valores para o construtor para definir os atributos iniciais do objeto, como a cor e a marca do carro no momento da sua criação.

Construtores

- O construtor de um objeto é um **método especial**, pois inicializa seus atributos toda vez que é instanciado (inicializado). Toda vez que é digitada a palavra reservada **new**, o objeto solicita para a memória do sistema armazená-lo, onde chama o construtor da classe para inicializar o objeto.

- A identificação de um construtor em uma classe é sempre o mesmo nome da classe.

Construtores

```java
class ConstrutorProg{
    private String nomeCurso;


    public ConstrutorProg(String nome)
    {
        nomeCurso = nome;
    }

    public String getNome()
    {
        return "Nome do Curso retornado "+nomeCurso;
    }

}
```

O construtor recebe um parâmetro de uma String que será um argumento de entrada na classe testadora.

A classe “ConstrutorProg” é inicializada, passando apenas um argumento no parâmetro que foi definido, se fosse apenas inicializado sem nenhum argumento estaria ocorrendo um erro de sintaxe pois já foi definido que no método construtor iria haver uma entrada de um parâmetro.

```java
public class Construtor {

    public static void main(String[] args) {

        ConstrutorProg cp = new ConstrutorProg("DevMedia - Java");
        System.out.println(cp.getNome());

    }
}
```

Sempre uma classe terá um construtor padrão, mesmo não sendo declarado o compilador irá fornecer um. Esse construtor não recebe argumentos e existe para possibilitar a criação de objetos para uma classe.

Destrutor

- Um **destrutor** (disponível em algumas linguagens, como C++) é o oposto do construtor. Ele é chamado automaticamente quando um objeto é **destruído** ou sai de escopo.

- A função do destrutor é **liberar recursos** que o objeto possa ter alocado, como memória, conexões de banco de dados ou arquivos abertos.

- Em linguagens com gerenciamento automático de memória, como Python e Java, o destrutor não é tão comum, pois a coleta de lixo (garbage collection) se encarrega de liberar a memória.

Modificadores de Acesso

- Os **modificadores de acesso** controlam a visibilidade dos atributos e métodos de uma classe. Eles determinam de onde um membro pode ser acessado. Os mais comuns são:
  - **Público (public):** Membros públicos podem ser acessados de **qualquer lugar**. Qualquer código pode ler ou modificar um atributo público ou chamar um método público.
  - **Privado (private):** Membros privados são acessíveis **apenas de dentro da própria classe**. Nenhuma outra classe pode acessá-los diretamente. Isso é útil para encapsular dados e proteger o estado interno do objeto.
  - **Protegido (protected):** Membros protegidos são acessíveis de dentro da própria classe e também por **classes que herdam** dela. Eles não podem ser acessados por outras classes que não sejam herdeiras.

- A utilização de modificadores de acesso é fundamental para o **encapsulamento**, um dos pilares da POO, que consiste em esconder a complexidade interna do objeto, expondo apenas o que é necessário.

Exemplo:

```
public class TiposDeDadosExemplo {

    public static void main(String[] args) {

        // Tipos Primitivos

        byte idade = 30;

        int populacao = 150000;

        long distancia = 9876543210L; // O 'L' indica que é um long

        double salario = 2500.50;

        float peso = 68.5f; // O 'f' indica que é um float

        char primeiraLetra = 'A';

        boolean estaAtivo = true;

 // Tipo de Referência (String é uma classe)
        String nome = "Maria";

        System.out.println("Idade: " + idade);
        System.out.println("População: " + populacao);
        System.out.println("Salário: " + salario);
        System.out.println("Nome: " + nome);
        System.out.println("Está ativo? " + estaAtivo);
    }
}
```