



Introdução a Programação Orientada a Objetos

O que iremos aprender nessa lição:

- O que é uma classe
- Como criar uma classe
- Como criar um objeto ou instância de uma classe
- O que são atributos
- Como definir e visualizar valores de atributos
- Como definir métodos com parâmetros e retorno
- Como usar a referência this para acessar um atributo

1) O que é um programa de computador?

Segundo SANTOS (2003) os programas de computadores são conjuntos de regras que um programador deve conhecer para poder manipular os recursos de um computador. Programas são escritos usando linguagem de programação, por exemplo java, que definem regras específicas e um conjunto de operadores e comandos que podem ser usados. O conteúdo dos programas, escrito por programadores de forma que outros programadores possam ler e entendê-los também é chamado de código fonte ou código. Para que o código seja executado por um computador, ele deve ser traduzido da linguagem de programação (alto nível) para uma linguagem que possa ser compreendida pelo computador através de um compilador.

2) O que é programação orientada a objetos?

Programa orientada a objetos ou, abreviadamente, POO, é um paradigma de programação de computadores onde se usam classes e objetos.

3) Objetos

Conforme o dicionário Aurélio, temos as seguintes definições para a palavra objeto:

1. Tudo que é manipulável e/ou manufaturável;
 2. Tudo que é perceptível por qualquer dos sentidos;
 3. Coisa, peça, artigo de compra e venda;
 4. Matéria, assunto;
 5. Agente, motivo, causa;
 6. Mira, fim propósito, intento, objetivo;
- etc...



Pelas definições, o termo objeto pode ter muitos significados diferentes. Podemos entender um objeto como uma “coisa” física: uma pedra, uma jaca ou um sapo, por exemplo. Também podemos considerar uma equação ou uma conta bancária como um objeto, nesse caso puramente mental, pois não existe uma “coisa” física que possa impressionar nossos sentidos para que estes o percebam como um objeto físico.

Pois bem, nos Objetos Computacionais (objetos que se encontram dentro de sistemas de computador), procuramos reproduzir as mesmas características dos objetos do mundo real, programando comportamentos semelhantes àqueles que encontramos nos objetos do mundo real.

Um programador pode interagir com esses objetos ativando os comportamentos programados, sem necessidade de entender o funcionamento interno desses comportamentos. Ou seja, para interagir com os objetos, precisamos apenas conhecer o que fazer para usá-los, nada mais.

A proposta da Programação Orientada a Objetos (POO) é representar o mais fielmente possível as situações do mundo real nos Sistemas Computacionais.

Nós entendemos o mundo como um todo, composto por vários objetos que interagem uns com os outros. Da mesma maneira, a POO consiste em considerar os Sistemas Computacionais não como uma coleção estruturada de processos, mas, sim, como uma coleção de objetos que interagem entre si.

Por exemplo, vamos considerar um cachorro como nosso “objeto” de estudo. O cachorro possui característica, como: cor do pelo, peso, idade e outros, e ainda o cachorro possui ações, como por exemplo: latir, correr, comer e outros. A seguir é descritos algumas características e ações do cachorro.

Características (Um cachorro possui)?

- Nome;
- Idade;
- Cor dos pelos;
- Cor dos olhos;
- Peso;

Ações que ele executa?

- Latir;
- Babar;
- Correr em círculos;
- Sentar;
- Comer;
- Dormir;

As características chamamos de Atributos e as ações de Métodos. Portanto, os objetos são compostos de Atributos e Métodos.



Os atributos dos objetos são “variáveis” ou “campos” que armazenam os diferentes valores que as características dos objetos podem conter. O cachorro do nosso exemplo poderia ser representado pelos seguintes atributos:

Cachorro	
Nome	Pluto
Idade	3 anos
Cor pelos	Laranja
Cor olhos	Castanhos
Peso	8 kg

Outros cachorros teriam valores diferentes para os mesmos atributos:

Cachorro	
Nome	Pluto
Idade	3 anos
Cor pelos	Laranja
Cor olhos	Castanhos
Peso	8 kg

Cachorro	
Nome	Scooby doo
Idade	7 anos
Cor pelos	Marrom
Cor olhos	Negros
Peso	10 kg

Os métodos são procedimentos ou funções que realizam as ações próprias do objeto:

Cachorro	
Comer	
Dormir	
Correr	
Pular	

Outros cachorros teriam os mesmos métodos.



Cachorro	
Comer	
Dormir	
Correr	
Pular	



Cachorro	
Comer	
Dormir	
Correr	
Pular	



Olhando os dois cães do exemplo anterior, vemos que ambos possuem exatamente o mesmo conjunto de atributos. Isso acontece porque se trata de dois objetos da mesma classe ou categoria, o que significa que os dois possuem exatamente o mesmo conjunto de atributos e métodos, embora cada objeto possa ter valores diferentes para seus atributos.

A seguir é apresentado o conceito de classe.

4) Classes

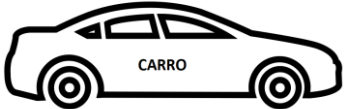



A classe é um modelo e todos os seus objetos têm os mesmos atributos (embora esses atributos possam ter valores diferentes) e os mesmos métodos.

Uma classe nada mais é do que um projeto que abstrai um conjunto de objetos com características similares. Uma classe define o comportamento de seus objetos através de métodos e os estados possíveis destes objetos através de atributos. Em outros termos, uma classe descreve os serviços providos por seus objetos e quais informações eles podem armazenar.

Note que uma Classe não tem vida, é só um conceito. Mas os Objetos possuem vida.

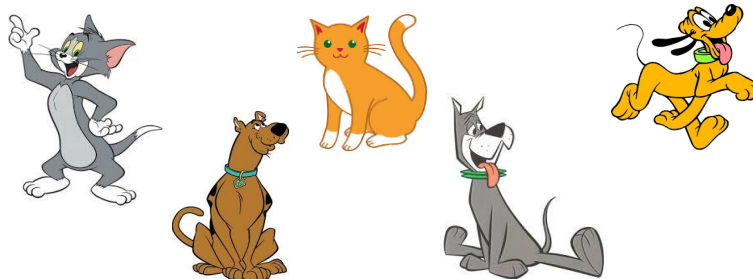
Classes não são diretamente suportadas em todas as linguagens, e são necessárias para que uma linguagem seja orientada a objetos. Classes são os elementos primordiais de um diagrama de classes.

Exemplo de classe e objeto:

CLASSE	OBJETO
	
	
	



Para diferenciar o conceito de classes de objetos observe a figura a seguir.



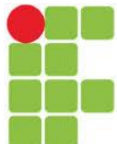
Observando a figura podemos distinguir duas categorias diferentes de bichos: cães e gatos. Existem cinco objetos, de duas classes diferentes. Essa é a distinção entre classe e objeto.

5) Instanciação

A instanciação acontece quando a classe produz um objeto como se ela fosse uma espécie de modelo ou gabarito para a criação de objetos. Conforme a teoria da Orientação a Objetos, dizemos que um objeto é, nada mais, nada menos que a instância de uma classe.

As classes são definições de como os objetos devem ser e não existem na realidade. Somente os objetos tem existência. Usando o exemplo dos animais, quando vamos mostrar nosso cachorro a um amigo, não dizemos “este é um cão”, e sim “este é o Rex”, ou “Bob” etc.. O que nosso amigo vê não é uma classe de seres, mas um objeto específico, um cachorro.

Ainda, antes de implementarmos o conceito de POO em Java, temos outro exemplo, vamos considerar a classe clientes. Cada cliente que for armazenado é um novo objeto, uma nova instância da classe clientes. A classe serve de modelo para a criação de novos objetos.



6) Implementação de Classes e Métodos em Java

Implementando a classe cachorro:

```
public class Cachorro{  
  
    public String nome;  
    public int peso;  
    public String corOlhos;  
    public int idade;  
}
```

A classe acima mostra as características de um cachorro, mas os valores que são guardados nos atributos (nome, peso, corOlhos e idade) são diferentes variando de cachorro para cachorro.

A seguir é apresentado a instanciação da classe cachorro, sendo instanciada três vezes, mostrando que cada cachorro instanciado tem características diferentes.

```
public class TestaCachorro {  
  
    public static void main(String[] args) {  
  
        Cachorro cachorro1 = new Cachorro();  
        cachorro1.nome = "Pluto";  
        cachorro1.corOlhos = "preto";  
        cachorro1.peso = 35;  
        cachorro1.idade = 4;  
  
        Cachorro cachorro2 = new Cachorro();  
        cachorro2.nome = "Rex";  
        cachorro2.corOlhos = "amarelo";  
        cachorro2.peso = 27;  
        cachorro2.idade = 3;  
  
        Cachorro cachorro3 = new Cachorro();  
        cachorro3.nome = "Scooby";  
        cachorro3.corOlhos = "marrom";  
        cachorro3.peso = 20;  
        cachorro3.idade = 3;  
  
    }  
}
```



Implementando métodos na classe cachorro:

```
public class Cachorro{

    public String nome;
    public int peso;
    public String corOlhos;
    public int idade;
    public int tamanho; //novo atributo

    void latir(){
        if(tamanho > 60)
            System.out.println("latido forte!");
        else if(tamanho > 25)
            System.out.println("latido médio!");
        else
            System.out.println("não assusta!");
    }
}
```

Na classe cachorro foi criado um novo atributo chamado “tamanho” e um método chamado “latir”. No método latir tem uma condição de acordo com o valor informado no atributo “tamanho”.

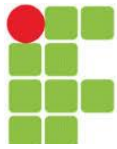
A seguir o código para testar o método criado.

```
public class TestaCachorro {

    public static void main(String[] args) {

        Cachorro bob = new Cachorro();
        bob.tamanho = 65;
        Cachorro rex = new Cachorro();
        rex.tamanho = 8;
        Cachorro scooby = new Cachorro();
        scooby.tamanho = 25;

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



7) Métodos Construtores

O construtor de um objeto é um método especial, pois inicializa seus atributos toda vez que é instanciado (inicializado).

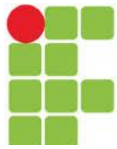
Toda vez que é digitada a palavra reservada `new`, o objeto solicita para a memória do sistema armazená-lo, onde chama o construtor da classe para inicializar o objeto. A identificação de um construtor em uma classe é sempre o mesmo nome da classe.

Abaixo é apresentado um código com o método construtor que recebe como parâmetro uma variável do tipo `String`, essa variável será um argumento de entrada na classe.

```
public class Cachorro {  
    private String nome;  
  
    public Cachorro(String nome)  
    {  
        this.nome = nome;  
    }  
  
    public String getNome()  
    {  
        return "Nome do Cachorro "+nome;  
    }  
}
```

A palavra chave `this` serve para mostrar que esse é um atributo da classe local (classe cachorro), e não a variável passada por parâmetro. A seguir é testada a classe cachorro com o método construtor.

```
public class TestaConstrutor {  
  
    public static void main(String[] args) {  
  
        Cachorro dog = new Cachorro("Pingo");  
        System.out.println(dog.getNome());  
  
    }  
}
```

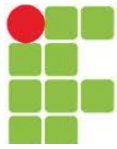
8) Exemplo Conta Bancária

Classe Cliente:

```
public class Cliente{  
    String nome;  
    String sobrenome;  
    String cpf  
}
```

Classe Conta:

```
public class Conta{  
  
    double saldo;  
    int agencia;  
    int numero;  
    Cliente titular;  
  
    //método com retorno  
    boolean saca(double valor) {  
        if (this.saldo < valor) {  
            return false;  
        }  
        else {  
            this.saldo = this.saldo - valor;  
            return true;  
        }  
    }  
  
    void deposita(double quantidade) {  
        this.saldo += quantidade;  
    }  
  
    boolean transfere(Conta destino, double valor) {  
        boolean retirou = this.saca(valor);  
        if (retirou == false) {  
            // sem saldo para transferir  
            return false;  
        }  
        else {  
            destino.deposita(valor);  
            return true;  
        }  
    }  
}
```



Classe criaConta:

```
public class CriaConta {  
  
    public static void main(String[] args) {  
  
        Conta conta1 = new Conta();  
        Cliente cliente1 = new Cliente();  
        cliente1.nome = "Maria";  
        conta1.titular = cliente1;  
        conta1.saldo = 1500.0;  
        boolean sacou = conta1.saca(1000);  
        if (sacou) {  
            System.out.println("Saque realizado com  
sucesso!");  
        } else {  
            System.out.println("Saldo insuficiente");  
        }  
  
        Conta conta2 = new Conta();  
        Cliente cliente2 = new Cliente();  
        cliente2.nome = "Joao";  
        conta2.titular = cliente2;  
        conta2.saldo = 1000.0;  
        boolean transferiu = conta2.transfere(conta1,  
500.0);  
        if (transferiu) {  
            System.out.println("Transferência realizado com  
sucesso!");  
        } else {  
            System.out.println("Saldo insuficiente para  
transferência");  
        }  
  
        System.out.println("Saldo do cliente  
"+conta1.titular.nome+ " = R$ "+conta1.saldo);  
        System.out.println("Saldo do cliente  
"+conta2.titular.nome+ " = R$ "+conta2.saldo);  
  
    }  
}
```

REFERÊNCIA

SANTOS, Rafael. Introdução a programação orientada a objetos usando Java - Rio de Janeiro : Elsevier, 2003.