Programação Orientada a Objetos

Aula 02 - Classes, atributos e métodos

Apresentação

Nesta aula, vamos estudar um dos principais conceitos da Programação Orientada a Objetos – a Classe. Vamos aprender também como se representa esse conceito na linguagem Java, além de criar características e comportamentos para uma classe através dos atributos e métodos.

Vídeo 01 - Apresentação

Objetivos

Depois de estudar e praticar o conteúdo desta aula, você será capaz de:

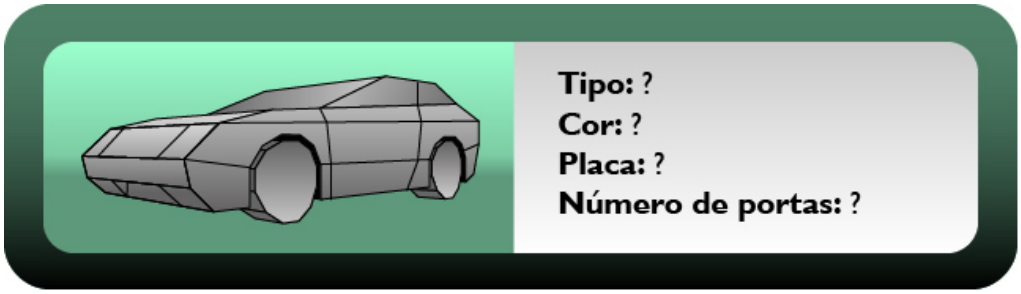
- Entender a teoria sobre Classe;
- Entender em detalhes uma classe codificada usando a linguagem Java;
- Criar atributos e métodos em classes Java.

Criação de Classes

Na primeira aula, falamos que objetos são entidades concretas que possuem características e comportamentos bem definidos, tais objetos podem ser organizados em grupos de características e comportamentos comuns. Esses grupos são chamados de **classe**. Porém, a classe não serve para organizar no sentido de guardar os objetos, ela serve de modelo de construção. Para os nossos exemplos, todos os carros (objetos) são da classe Carro, pois possuem as mesmas características e comportamentos.

- A classe é o modelo ou molde de construção de objetos.
- O modelo define as características e comportamentos que os objetos irão determinar seus valores e desempenhar suas ações, respectivamente.
- A classe é abstrata (não existe concretamente).

Figura 01 - Exemplo da ideia de uma classe Carro



Atenção!

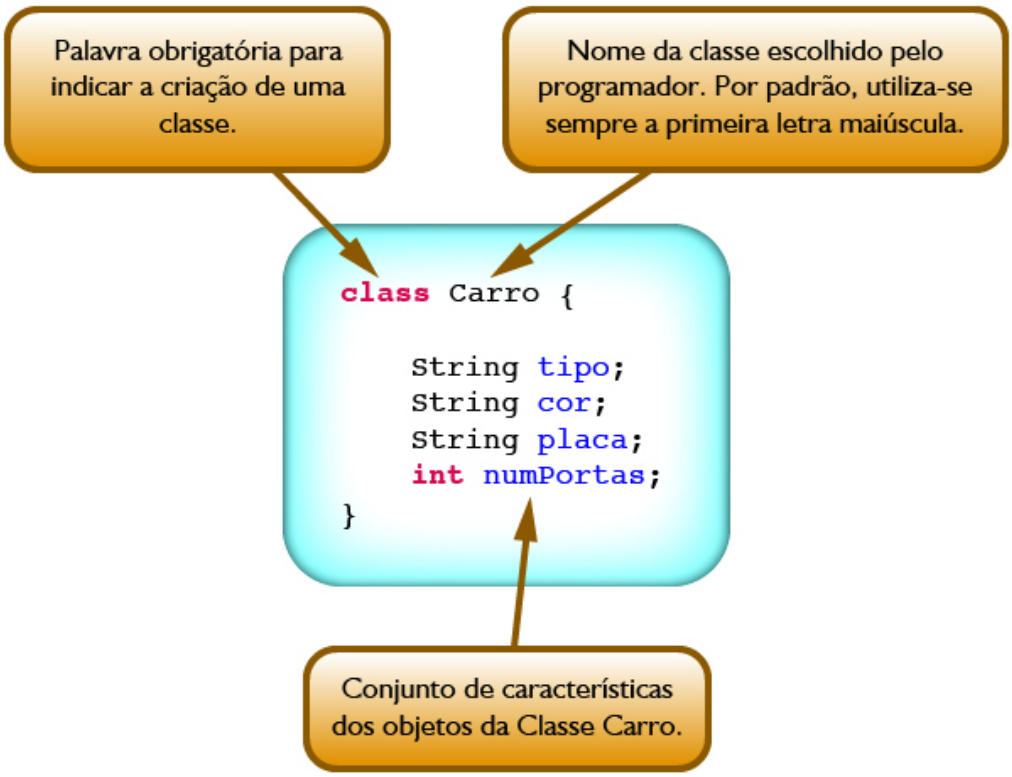
Já que as classes não existem concretamente, elas são abstratas, servem apenas para gerar os objetos.

Atividade 01

Vamos criar a classe Carro em Java? Então, vamos lá! Siga os passos descritos abaixo.

1. Abra um editor de texto qualquer e digite o código mostrado na Listagem 1.

2. Salve com o mesmo nome da classe, adicionando a extensão“.java” ao nome do arquivo.



Listagem 1 - Esquema de criação para classe Carro em Java

Observação 1: Esse arquivo deve ser salvo com o nome Carro.java

Observação 2: Este exemplo apresenta a estrutura básica de uma classe em Java. Mais adiante, veremos que existem vários outros elementos que podem ser adicionados à definição da classe.

Perceba que não inserimos no código nada sobre o comportamento dos carros, como andar para frente ou para trás, virar a esquerda/direita, frear etc.

Devemos ter em mente que a linguagem de programação é uma representação do mundo sob um ponto de vista. Você se lembra do paradigma? Esse ponto de vista ou paradigma é a orientação a objetos, porém, descrito nas **Regras da Linguagem**, que no nosso caso é a linguagem Java.

Na linguagem Java, as **características** são chamadas de **atributos** e são escritas informando o **TIPO** e o **NOME** do atributo, como pode ser visto na Listagem 2.

```
1 class Carro{
2    String tipo;
3    String cor;
4    String placa;
5 }
```

Listagem 2 - Atributos da classe Carro

É necessário escrever um ponto-e-vírgula após a declaração de cada *atributo*, pois é uma exigência da linguagem Java. Além disso, as seguintes regras devem ser obedecidas para escolha do nome do atributo.

- O nome de uma variável deve ser uma sequência de letras Unicode e dígitos, iniciando sempre com uma letra, "$" ou o caractere de sublinhado "_".
- Caracteres subsequentes podem ser letras, dígitos, $ ou "_".
- Espaços não são permitidos em nomes de variáveis.
- Palavras reservadas da linguagem, tais como, class, int, float, for, while etc. não podem ser usadas como nomes de variáveis.
- Nomes de variáveis são sensíveis a letras minúsculas e maiúsculas. Em outras palavras, dois atributos com nomes "placa" e "Placa", por exemplo, são considerados dois atributos diferentes.

Dica!

Use nomes completos, ao invés de abreviações na escolha dos nomes de atributos, isso tornará seu código mais legível.

Antes de prosseguirmos, é necessário instalarmos o Netbeans, isto é, o ambiente de desenvolvimento que iremos programar em Java. Embora a aula esteja direcionada para a versão 7.3.1, este link possui o arquivo de instalação do Netbeans e JDK (Kit de Desenvolvimento Java) necessários para rodar a versão 8.2 do Netbeans (a mais atual) no Windows 10 x64. Poderão surgir algumas pequenas diferenças entre estas versões, mas você certamente tem a habilidade de diferenciá-las e de sanar possíveis dificuldades.

Para baixar outras versões do Netbeans, acesse https://netbeans.org/downloads/.

Vídeo 02 - Atributos

Tipo de Atributo

Todo atributo deve possuir um tipo, os quais podem ser um dos descritos a seguir.

1. Um dos oito tipos de dados primitivos.
    1. Inteiros: byte (1 byte), short (2 bytes), int (4 bytes), long (8 bytes);
    2. Reais: float (4 bytes), double (8 bytes)
    3. Booleanos: boolean (1 byte)
    4. Caracter: char (2 bytes)

2. O nome de uma classe ou interface.

3. Uma coleção.

Constantes

Para declarar uma constante, use a palavra chave final antes da declaração do atributo e inclua um valor inicial para esse último. O valor de uma constante nunca poderá ser modificado. A Listagem 3 exemplifica a declaração de algumas constantes.

```
1 class Matematica {
2    final int DEZENA = 10;
3    final double E = 2.71828;
4 }
```

Listagem 3 - Constantes da classe Matemática

Outra exigência da linguagem são os blocos, nos quais definimos o comportamento da classe e de seus métodos. Um bloco é definido por um ({}) e contém um grupo de outros blocos. Quando um novo bloco é criado, um novo escopo local é aberto e permite a definição de variáveis locais. As variáveis definidas dentro de um bloco só podem ser vistas internamente a esse, e são terminadas e extintas no final da execução do bloco (}).

```
1  void testBlock() {
2     int x = 10,w=1;
3     if (x>w)
4     { //inicio do bloco
5        int y=50;
6        System.out.println("dentro do bloco");
7        System.out.println("x:"+x);
8        System.out.println("y:"+y);
9     } //fim do bloco
10    System.out.printl("x:"+x);
11    System.out.printl("y:"+y); //erro variável y não conhecida
12 }
```

Listagem 4 - Bloco de código

Programar é traduzir uma solução de um problema para uma linguagem de programação. Logo, é imprescindível conhecer as regras da linguagem. O programador é como um intérprete ou tradutor de um idioma (linguagem no

mundo real) para outro (linguagem entendida pelo computador).

Vamos a outro exemplo apresentado na Listagem 5.

```
1 class Pessoa {
2    String nome;
3    String corDoCabelo;
4    String biotipo;
5    int idade;
6 }
```

Listagem 5 - Classe Pessoa em Java

Cada atributo é definido com o par TIPO e NOME. *String* significa tipo de texto. **int** significa tipo numérico inteiro.

Veja que uma pessoa, assim como um carro, possui milhares de características, porém, utilizando o princípio da abstração, iremos deixar de fora tudo que não nos interessa, ou que não seja importante ou relevante para o sistema que estamos desenvolvendo. Ora, se formos programar um sistema para uma locadora de veículos, precisamos saber algumas informações sobre a pessoa que irá alugar o veículo, como o seu nome, habilitação de condutor, idade e o que mais acharmos relevante. Porém, é muito pouco provável que iremos exigir dados escolares das pessoas, como notas de uma disciplina etc. Para os carros, também há essa observação, pois um sistema de oficina mecânica deverá conhecer mais detalhes "internos" do veículo, como numeração do motor, período das revisões, marcas de componentes, como amortecedores, dentre outros.

Mas, como saber o que é relevante? Isso é determinado no estudo do problema que pretendemos resolver e comumente chamamos esse contexto do problema de domínio.

Vídeo 03 - Variáveis

Atividade 02

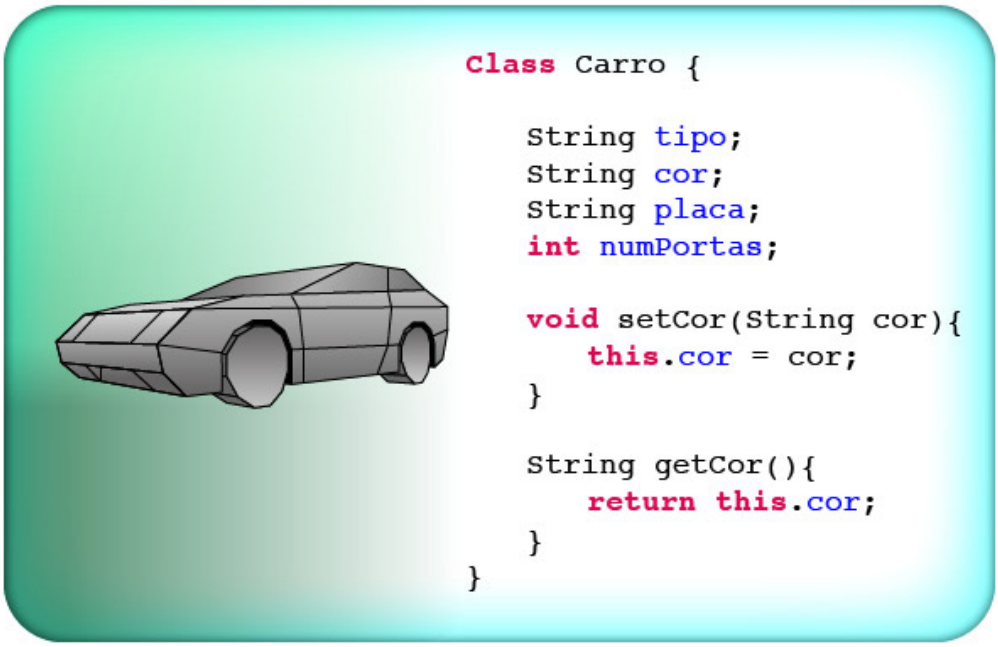
Na aula anterior, fizemos um exercício que pedia para observar os objetos a sua volta, escolher um deles e descrever de 3 a 5 características (que agora chamamos de atributos). Faça o mesmo para dois outros objetos e identifique seus atributos.

Em seguida, escreva o código das classes que representam tais objetos na linguagem Java, conforme mostrado nas Listagens 1 e 2.

Adicionando Comportamento

Os objetos das classes que criamos, Carro e Pessoa, precisam não só de suas características, mas também de comportamentos que expressem as ações possíveis de serem executadas, pois um carro precisa oferecer funções para que as pessoas os manobrem. E pessoas, por sua vez, precisam desempenhar suas atividades, como andar, correr, estudar etc. Percebam que falamos da capacidade dinâmica do objeto através dos métodos que possuem.

Vamos dar ações aos nossos carros! Veja a Listagem 6:



Listagem 6 - Classe Carro com métodos

Observe que adicionamos novas linhas ao código, agora com dois métodos *setCor(String cor)* e *getCor()*. Esses métodos servem, respectivamente, para definir e recuperar o valor do atributo "cor". Situação onde levaríamos o nosso carro a uma equipadora para modificar a sua cor. Atente para o fato de que os atributos são alterados por métodos da sua classe.

Veja que adicionamos a palavra **this** antes do atributo cor. Essa palavra-chave representa a instância da classe Carro e particularmente neste caso está sendo usada para corrigir o problema de ambiguidade gerado pela definição de um parâmetro com o mesmo nome do atributo da classe no método *setCor(String cor)*. Ao se usar a palavra **cor** dentro deste método, como saberíamos se estamos referenciando o atributo da classe ou o parâmetro do método? Para solucionar esse problema, usa-se a palavra reservada **this** antes do nome **cor**, informando ao compilador que estamos referenciando o atributo da classe e não o parâmetro do método.

Curiosidade!

Dentro de um método, quando há ambiguidade de nome de variáveis (entre o atributo da classe e o parâmetro do método), o compilador sempre dá prioridade aos parâmetros. Em outras palavras, se não tivéssemos usado o **this**, o compilador iria achar que estávamos referenciando ao parâmetro do método e assim o atributo da classe nunca seria atualizado. Cuidado com isso! Essa é uma fonte muito comum de erros na programação em Java. Uma boa prática de codificação em Java é sempre usar a palavra **this** antes de atributos da classe, mesmo quando não há ambiguidade.

Vídeo 04 - Métodos

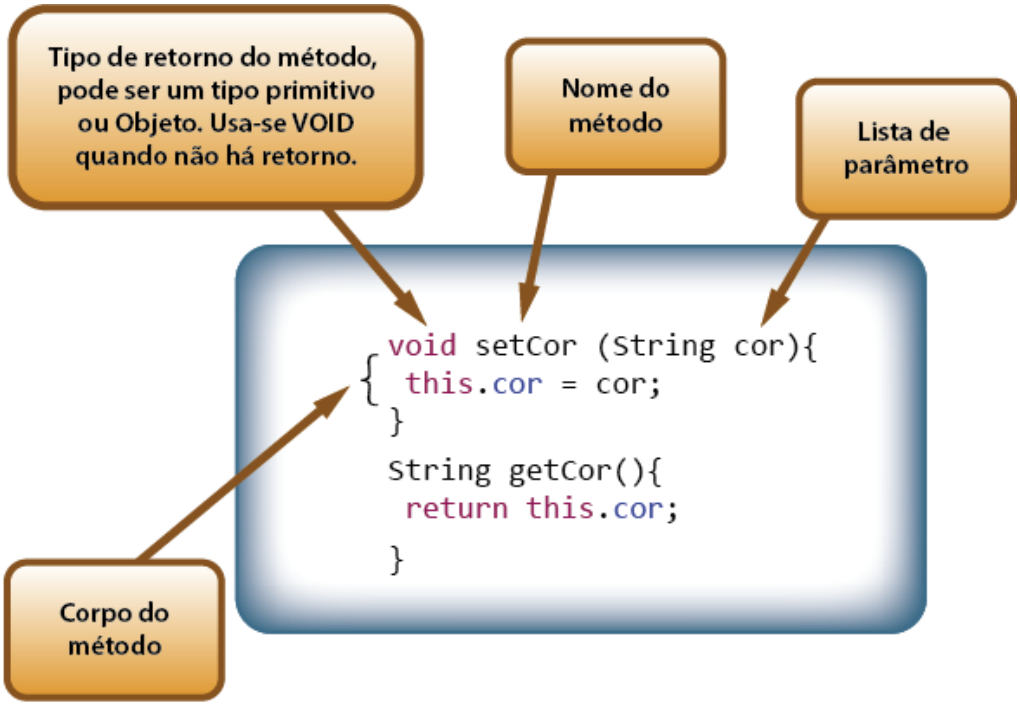
Outra maneira de implementar os métodos sem usar a palavra chave **this** será apresentado na Listagem 7. Nesse caso, temos que mudar o nome do parâmetro para “c” evitando a ambiguidade.

```
1  class Carro {
2     String tipo;
3     String cor;
4     String placa;
5     int numPortas;
6
7     void setCor(String c){
8        cor = c;
9     }
10
11    String getCor(){
12       return cor;
13    }
14 }
```

Listagem 7 - Classe Carro com métodos sem a palavra chave *this*

A definição dos métodos tem cinco partes básicas:

- tipo de acesso do método, (public, private, protected);
- nome do método;
- o tipo de retorno, o qual pode ser um objeto ou tipo primitivo;
- um ou mais parâmetros;
- finalmente, o corpo do método.



Listagem 8 - Exemplos de métodos

Logo, senão construímos outros métodos que acessem e modifiquem os valores dos demais atributos – do tipo placa e num Portas– esses não poderão ser vistos fora do objeto nem alterados.

Com isso, o nosso código ficará um pouco mais longo, veja a **Listagem 9**.

```
class Carro {

   String tipo;
   String cor;
   String placa;
   int numPortas;

   void setCor(String c){
      cor = c;
   }

   String getCor(){
      return cor;
   }

   String getTipo(){
      return tipo;
   }

   void setTipo(String tipo){
      this.tipo = tipo;
   }

   String getPlaca(){
      return placa;
   }

   void setPlaca(String placa){
      this.placa = placa;
   }

   String getNumPortas(){
      return numPortas;
   }
   void setNumPortas(String numPortas){
      this.numPortas = numPortas;
   }
}
```

Listagem 9 - Classe Carro completa

Atividade 03

1. Dando continuidade a Atividade 2, vamos agora acrescentar os comportamentos dos objetos nas classes, escrevendo o código de seus métodos.

2. De acordo com os exemplos vistos, crie uma classe chamada Livro e especifique os seus atributos e métodos.

Resumo

Nesta aula, você conheceu a concepção de uma classe, como também o conceito sobre classes. Você entendeu a criação de uma classe na linguagem Java. Viu também como representar as características e comportamentos, agora chamados, respectivamente, de atributos e métodos em código Java. Você aprendeu ainda que a classe organiza as características e comportamentos comuns dos objetos e que essas características (atributos) podem sofrer alterações em suas informações através dos comportamentos (métodos).

Autoavaliação

1. O que é uma classe e para que serve?
2. Como se escreve uma classe em Java?
3. Como se representam as características em uma classe?
4. Quais tipos de atributos você conhece?
5. Qual a regra básica de criação de um método de uma classe?

Referências

THE JAVA: tutorials. **[Tutorial Oficial Java]**. Disponível em: <http://java.sun.com/docs/books/tutorial/>. Acesso em: 13 maio 2010.

_____. **About the Java Technology**. Disponível em: <http://java.sun.com/docs/books/tutorial/getStarted/intro/definition.html>. Acesso em: 13 maio 2010.

WIKIPÉDIA. **Máquina virtual Java**. Disponível em: <http://pt.wikipedia.org/wiki/Máquina_virtual_Java>. Acesso em: 13 maio 2010.