

Orientação a Objetos

I

Ramon Lummertz



Figura 3.7: Diversas casas com características diferentes

Classes

Relembrando

Se você já está acostumado com algum outro paradigma, esse é o momento de abrir a sua mente.

Nas classes (models)

- * Criamos nossos atributos (características)
- * Criamos nosso métodos (ações)

Criando uma classe Livro

- * observem a diferença entre String e double . por qual motivo isso ocorre?
- * Nomenclatura da classe sempre em maiuscula.
- * Nomenclatura de atributos com letras minusculas

```
3 */  
4 public class Livro {  
5  
6     String nome;  
7     String descricao;  
8     double valor;  
9     String isbn;  
10  
11 }  
12
```

Usando a classe Livro

- * Linha 7 - construindo um objeto livro
- * Linhas 10 a 13 - atribuindo valores
- * Linhas 15 a 18 - saída de valores

```
3  public class Main {  
4  
5      public static void main(String[] args) {  
6          // write your code here  
7          Livro livro = new Livro();  
8  
9  
10         livro.nome = "Java S2";  
11         livro.descricao= "A história de java";  
12         livro.isbn="01-0002-004";  
13         livro.valor=120.00;  
14  
15         System.out.println("Livro "+ livro.nome);  
16         System.out.println("Livro "+ livro.descricao);  
17         System.out.println("Livro "+ livro.isbn);  
18         System.out.println("Livro "+ livro.valor);  
19  
20     }  
21 }  
22  
23 }
```

Então! Classe x Objeto

Uma classe é apenas um molde. Uma especificação que define para a máquina virtual o que um objeto desse tipo deverá ter e como ele deve se comportar. Nossa livraria poderá ter milhares de livros (objetos), mas existirá apenas uma classe Livro (molde). Cada objeto que criarmos do tipo Livro terá seus próprios valores, ou seja, cada livro terá o seu próprio nome, sua descrição, um valor e um número de ISBN.

Um objeto é a instância de uma Classe!

Continuando com a classe livro

Instancie três livros, atribua
valores, e mostre na tela os dados
dos livros na seguinte ordem

nome
isbn
valor
descrição



KEEP
CALM
AND
MÃOS
À OBRA

Esperem!! as regras de negócio mudaram!!

mostre na tela os dados dos livros na seguinte ordem

valor
descrição
isbn
nome

Essa repetição de código sempre tem um efeito colateral desagradável, que é a dificuldade de manutenção. Quer ver? O que acontece se adicionarmos um novo atributo no livro, por exemplo a data de seu lançamento? Para imprimir a data toda vez que criar um livro, teremos que mudar varias partes de nosso código, adicionando o `System.out.println(livro.dataDeLancamento);`

Métodos!!!

No lugar de deixar essa lógica de impressão dos dados do livro toda espalhada, podemos isolar esse comportamento comum entre os livros na classe Livro! Para isso, criamos um método. Uma forma seria:

```
tipoDeRetorno nomeDoComportamento() {  
    // código que será executado  
}
```

Métodos de Livro

* Agora temos um método, que atende todos os objetos livros

```
6  public class Livro {  
7  
8      String nome;  
9      String descricao;  
10     double valor;  
11     String isbn;  
12  
13     void mostraDetalhes(){  
14         System.out.println("Livro "+ this.nome);  
15         System.out.println("Livro "+ this.descricao);  
16         System.out.println("Livro "+ this.isbn);  
17         System.out.println("Livro "+ this.valor);  
18     }  
19  
20 }  
21
```

Nosso código agora tem uma manutenibilidade muito maior! Sempre devemos criar métodos de forma genérica e reaproveitável.

Operador this

Usamos o operador this para mostrar que esse é um atributo da classe, que referencia o próprio objeto que está usando o método ou atributo

Crie um método para ler os dados de um livro.

Não esqueça de instanciar a Classe Scanner



Métodos com parâmetro

- * Imagine que após cadastrar um livro, temos que ajustar o valor para uma possível venda.

```
Livro livro = new Livro();
livro.valor = 59.90;

System.out.println("Valor atual: " + livro.valor);

livro.valor -= livro.valor * 0.1;

System.out.println("Valor com desconto: " + livro.valor);
```

Métodos com parâmetros

- * No código anterior realizamos o desconto de 10% a um livro, não usamos um método e isso como ja sabemos podem nos trazer futuros problemas. Ex:
- * Agora queremos dar 20% de desconto?

Métodos com parâmetro

Dar desconto
propriedade de um
livro

```
19
20     void aplicaDescontoDe(double porcentagem){
21         this.valor -= this.valor * porcentagem;
22     }
23
24 }
```

Chamando os
métodos no main

```
5
6     public static void main(String[] args) {
7         // write your code here
8         Livro livro = new Livro();
9
10        livro.nome = "Java S2";
11        livro.descricao= "A história de java";
12        livro.isbn="01-0002-004";
13        livro.valor=120.00;
14
15        livro.mostraDetalhes();
16
17        livro.aplicaDescontoDe(0.3);
18        System.out.println("Aplicando desconto");
19        livro.mostraDetalhes();
20
21    }
22
23 }
```

Métodos com retorno

- * Métodos com retorno, podem ou não possuir parâmetros, e ao contrário dos métodos voids, eles retornam via método um valor ao programa que o chamou.

Métodos com retorno

```
24     double retornaValorDoDesconto(double porcentagem){  
25         double desc =this.valor *porcentagem;  
26         return desc;  
27     }
```

```
17  
18     double descontinho = livro.retornaValorDoDesconto(0.3);  
19     livro.aplicaDescontoDe(0.3);  
20     System.out.println("Aplicando desconto "+ descontinho);  
21     livro.mostraDetalhes();  
22
```

Evoluindo nossa código

Composição

Uma classe pode ter outra classe como atributo. já realizamos isso quando usamos a classe String como atributo.
Agora vamos criar nossa próprias classes

Por exemplo, quem escreveu o livro? Qual o e-mail do autor? E quando foi a sua data de publicação? Todas essas informações são relevantes para nossa livraria e também para nossos clientes. Podemos adicionar essas e outras informações na classe Livro:

Adicionando um autor

```
3  /**
4  * Created by ramon on 04/03/15.
5  */
6  public class Livro {
7
8      String nome;
9      String descricao;
10     double valor;
11     String isbn;
12
13     String nomeAutor;
14     String emailAutor;
15     String cpfAutor;
```

ERRADO!!!!

os dados do autor são do autor e não dos livros

Atribuido um autor

```
public class Livro {  
  
    String nome;  
    String descricao;  
    double valor;  
    String isbn;  
    Autor autorLivro;|
```

Atualize seu
código, para
ler e mostrar
os dados do
autor

```
5   public static void main(String[] args) {  
6       // write your code here  
7       Livro livro = new Livro();  
8  
9  
10      livro.nome = "Java S2";  
11      livro.descricao= "A história de java";  
12      livro.isbn="01-0002-004";  
13      livro.valor=120.00;  
14  
15      livro.autorLivro = new Autor();  
16      livro.autorLivro.cpfAutor= "005";  
17      livro.autorLivro.nomeAutor= "ramon";  
18      livro.autorLivro.emailAutor="ramonsl@gmail.com";  
19  
20  
21      Autor outraForma = new Autor();  
22      outraForma.emailAutor ="uma@forma";  
23      outraForma.nomeAutor= "Segudan forma";  
24      outraForma.cpfAutor="9999";  
25      livro.autorLivro = outraForma;  
26  
27
```

Referencia a objetos

É fundamental perceber que, quando instanciamos um novo objeto com a palavra reservada new, um Livro por exemplo, guardamos em sua variável uma referência para esse objeto, e não seus valores. Ou seja, a variável livro não guarda o valor de um nome, isbn e outros atributos da classe Livro, mas sim uma forma de acessar esses atributos do livro em memória. Muito diferente de quando trabalhamos com tipos primitivos que guardam uma cópia do valor.

Referencia a objetos

```
5  public static void main(String[] args) {  
6      // write your code here  
7      Livro livro = new Livro();  
8      livro.nome = "Java S2";  
9      livro.descricao= "A história de java";  
10     livro.isbn="01-0002-004";  
11     livro.valor=120.00;  
12  
13     Livro outroLivro = new Livro();  
14     outroLivro.nome = "Java S2";  
15     outroLivro.descricao= "A história de java";  
16     outroLivro.isbn="01-0002-004";  
17     outroLivro.valor=120.00;  
18  
19  
20     if (livro==outroLivro){  
21         System.out.println("Livros iguais");  
22     }else{  
23         System.out.println("Livros diferentes");  
24     }  
25  
26
```

Referencia a objetos



Referências

- * Desbravando Java e OO. Rodrigo Turini, casa do código.