Curso Programação Orientada a Objetos com Java

Capítulo: Construtores, palavra this, sobrecarga, encapsulamento

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Construtores

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Construtor

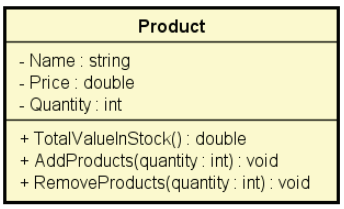
- É uma operação especial da classe, que executa no momento da instanciação do objeto
- Usos comuns:
  - Iniciar valores dos atributos
  - Permitir ou obrigar que o objeto receba dados / dependências no momento de sua instanciação (injeção de dependência)
- Se um construtor customizado não for especificado, a classe disponibiliza o construtor padrão:

```
Product p = new Product();
```

- É possível especificar mais de um construtor na mesma classe (sobrecarga)

Problema exemplo

```
Enter product data:
Name: TV
Price: 900.00
Quantity in stock: 10

Product data: TV, $ 900.00, 10 units, Total: $ 9000.00

Enter the number of products to be added in stock: 5

Updated data: TV, $ 900.00, 15 units, Total: $ 13500.00

Enter the number of products to be removed from stock: 3

Updated data: TV, $ 900.00, 12 units, Total: $ 10800.00
```



```java
package application;

import java.util.Locale;
import java.util.Scanner;

import entities.Product;

public class Program {
	public static void main(String[] args) {
		Locale.setDefault(Locale.US);
		Scanner sc = new Scanner(System.in);

		Product product = new Product();
		System.out.println("Enter product data: ");
		System.out.print("Name: ");
		product.name = sc.nextLine();
		System.out.print("Price: ");
		product.price = sc.nextDouble();
		System.out.print("Quantity in stock: ");
		product.quantity = sc.nextInt();

		System.out.println();
		System.out.println("Product data: " + product);

		System.out.println();
		System.out.print("Enter the number of products to be added in stock: ");
		int quantity = sc.nextInt();
		product.addProducts(quantity);

		System.out.println();
		System.out.println("Updated data: " + product);

		System.out.println();
		System.out.print("Enter the number of products to be removed from stock: ");
		quantity = sc.nextInt();
		product.removeProducts(quantity);

		System.out.println();
		System.out.println("Updated data: " + product);

		sc.close();
	}
}
```

```java
package entities;

public class Product {

	public String name;
	public double price;
	public int quantity;

	public double totalValueInStock() {
		return price * quantity;
	}

	public void addProducts(int quantity) {
		this.quantity += quantity;
	}

	public void removeProducts(int quantity) {
		this.quantity -= quantity;
	}

	public String toString() {
		return name
			+ ", $ "
			+ String.format("%.2f", price)
			+ ", "
			+ quantity
			+ " units, Total: $ "
			+ String.format("%.2f", totalValueInStock());
	}
}
```

Proposta de melhoria

Quando executamos o comando abaixo, instanciamos um produto "**product**" com seus atributos “vazios”:

```
product = new Product();
```

Entretanto, faz sentido um produto que não tem nome? Faz sentido um produto que não tem preço?

Com o intuito de evitar a existência de produtos sem nome e sem preço, é possível fazer com que seja “obrigatória” a iniciação desses valores?

```
package entities;

public class Product {

    public String name;
    public double price;
    public int quantity;

    public Product(String name, double price, int quantity) {
        this.name = name;
        this.price = price;
        this.quantity = quantity;
    }
    (...)
```

```
System.out.println("Enter product data: ");
System.out.print("Name: ");
String name = sc.nextLine();
System.out.print("Price: ");
double price = sc.nextDouble();
System.out.print("Quantity in stock: ");
int quantity = sc.nextInt();
Product product = new Product(name, price, quantity);
```