



EducaCiência FastCode

Fala Galera,

Neste artigo, abordaremos um tema muito interessante.

- Artigo: 13/2020 Data: Março/2020
- Público Alvo: Desenvolvedores – Iniciantes ao Avançado
- Tecnologia: Java
- Tema: Orientação à Objeto – parte 3
- Link: <https://github.com/perucello/DevFP>

Desta vez , escolhi um tema onde amedronta e muito os Desenvolvedores principalmente os iniciantes que é Orientação à Objeto.

Optei em dividir o tema em três partes com intuito de facilitar o entendimento.

Vamos abordar nesse artigo, a “**parte 3**”, lembrando que a “**parte 1**” está disponível no [artigo 11](#) e que a “**parte 2**” está disponível no [artigo 12](#) .

Primeiramente, vamos recapitular um pouco o Termo Orientação à Objeto !

"O termo orientação a objetos significa organizar o mundo real como uma coleção de objetos que incorporam estrutura de dados e um conjunto de operações que manipulam estes dados."

“Programação Orientada a Objetos (também conhecida pela sua sigla POO) é um modelo de análise, projeto e programação de software baseado na composição e interação entre diversas unidades chamadas de 'objetos'. A POO é um dos 4 principais paradigmas de programação (as outras são programação imperativa, funcional e lógica). Os objetos são operados com o conceito de 'this', de forma que seus métodos (muitas vezes) modifiquem os dados da própria instância. Os programas são arquitetados através de objetos que interagem entre si. Dentre as várias abordagens da POO, as baseadas em classes são as mais comuns: objetos são instâncias de classes, o que em geral também define o tipo do objeto. Cada classe determina o comportamento (definido nos métodos) e estados possíveis (atributos) de seus objetos, assim como o relacionamento com outros objetos...”

https://pt.wikipedia.org/wiki/Orienta%C3%A7%C3%A3o_a_objetos

Bom, feito uma breve review da explicação, vamos ao que interessa, vamos aos códigos!





Lembro que não trabalharemos com demonstração em “desenhos”, vou tentar explicar este conceito de “**parte 3**” puramente em código, que é o mais importante.

Cenário 3:

A ideia é termos um Objeto Chamado IDE e teremos dois atributos sendo “nome” que representa o nome da nossa IDE e outro atributo chamado “Fabricante” que informaremos o fabricante da nossa IDE.

Neste cenário, não abordaremos Banco de Dados e sim script em modo root para facilitar o entendimento.

Neste cenário também, criaremos uma Estrutura de Decisão “if/else” em nosso método Status onde de acordo com nosso código criado, traremos um comentário sobre o Fabricante que escolhemos.

Manipularemos o OO dentro de um mesmo Package neste cenário.

Descrição:

- a) Teremos nosso Projeto chamado **OrientadoObjetoParte2**
- b) Teremos apenas agora **três Package** (pacote neste exemplo)

Package Main

⇒ **A Classe que teremos nosso método Main**

- **MetodoMain.java** => Neste momento instanciaremos nosso Método Main com o Objeto e populando suas características/atributos e será nesta classe que traremos nosso retorno SOUT, não mais dentro do método Status como anteriormente.

Package IDE

⇒ **A Classe que criará nosso Objeto**

- **ObjetoIDE.java** => nesta Classe, criamos nosso ObjetoIDE e seus atributos/características.

Package Funcionário

⇒ **A Classe que criará nosso Objeto**

- **ObjetoFuncionario.java** => nesta Classe, criamos nosso ObjetoFuncionario e seus atributos/características que se estenderá a classe Setor que criaremos.

Package Setor

⇒ **A Classe que criará nosso Objeto**

- **ObjetoSetor.java** => nesta Classe, criamos nosso ObjetoSetor e seus atributos/características.

Após criarmos nossas duas classes como acima definidas, vamos aos códigos!





MetodoMain.java

```
File Edit Selection View Go Run Terminal Help
MetodoMain.java - OrientadoObjetoPart3 - Visual Studio Code

EXPLORER
> MYSQL
> JAVA DEPENDENCIES
> ORIENTADOOBJETO(PART3)
  > Funcionario
    > ObjetoFuncionario...
  > IDE
    > ObjetoIDE.java
  > Main
    > MetodoMain.java
  > Setor
    > ObjetoSetor.java

OPEN EDITORS
X MetodoMain.java ...
OUTLINE
MAVEN PROJECTS

Main > MetodoMain.java > ...
1 package Main;
2
3 import Funcionario.ObjetoFuncionario;
4 import IDE.ObjetoIDE;
5 import Setor.ObjetoSetor;
6
7 public class MetodoMain {
8     public static void main(String[] args) {
9
10         ObjetoIDE ide = new ObjetoIDE();
11         ide.setNome("VSCode");
12         ide.setFabricante("Microsoft");
13
14         ObjetoIDE idel = new ObjetoIDE();
15         idel.setNome("Netbeans 11");
16         idel.setFabricante("SunMicrosystems");
17
18         ObjetoFuncionario func = new ObjetoFuncionario();
19         func.setNome("Voliano de Tal");
20         func.setTecnologia("Java");
21
22         ObjetoSetor setor = new ObjetoSetor();
23         setor.setFuncao("FullStack");
24
25         System.out.println("-----");
26         System.out.println("O funcionario " + func.getNome() + " " + "trabalha com a tecnologia " + func.getTecnologia());
27         System.out.println("Utiliza da IDE " + ide.getNome() + " e seu fabricante é " + ide.getFabricante());
28         System.out.println("O funcionario " + func.getNome() + " nao trabalha com a IDE " + idel.getNome());
29         System.out.println("Funcionario " + func.getNome() + " tem como função na empresa " + setor.getFuncao());
30     }
31 }
32
33
```

ObjetoIDE.java

```
File Edit Selection View Go Run Terminal Help
ObjetoIDE.java - OrientadoObjetoPart3 - Visual Studio Code

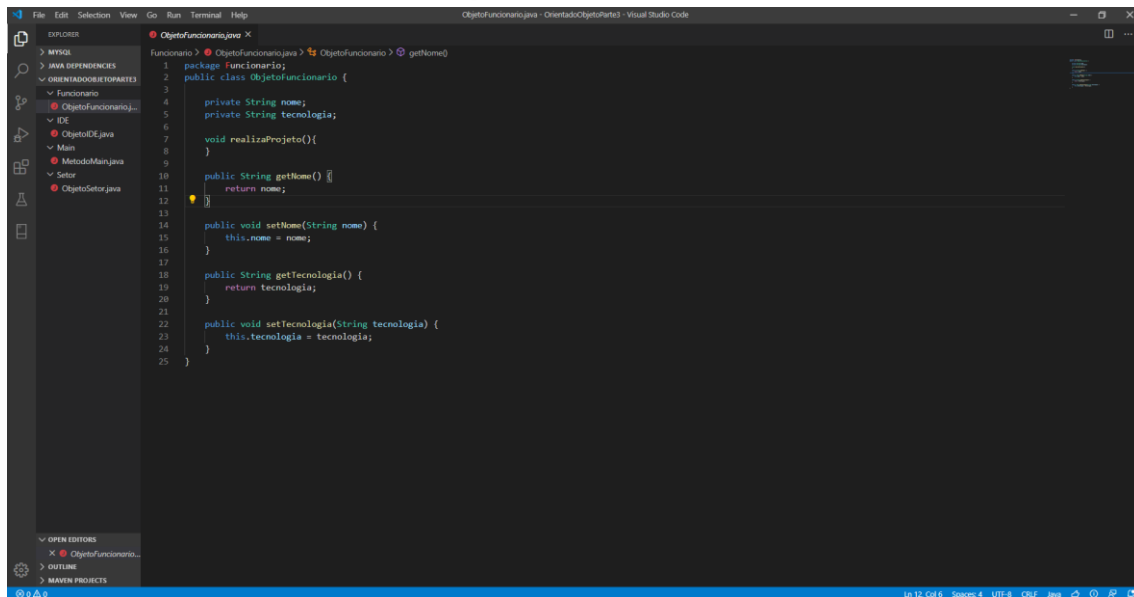
EXPLORER
> MYSQL
> JAVA DEPENDENCIES
> ORIENTADOOBJETO(PART3)
  > Funcionario
    > ObjetoFuncionario...
  > IDE
    > ObjetoIDE.java
  > Main
    > MetodoMain.java
  > Setor
    > ObjetoSetor.java

OPEN EDITORS
X ObjetoIDE.java IDE
OUTLINE
MAVEN PROJECTS

IDE > ObjetoIDE.java > ObjetoIDE > setNome(String)
1 package IDE;
2
3 public class ObjetoIDE {
4
5     private String nome; //criado nosso atributo ou caracteristica do Objeto - neste caso chamado NOME
6     public String fabricante; //criado nosso atributo ou caracteristica do Objeto - neste caso chamado FABRICANTE
7
8     public String getNome() {
9         return nome;
10    }
11
12    public void setNome(String nome) {
13        this.nome = nome;
14    }
15
16    public String getFabricante() {
17        return fabricante;
18    }
19
20    public void setFabricante(String fabricante) {
21        this.fabricante = fabricante;
22    }
23 }
24
```



ObjetoFuncionario.java

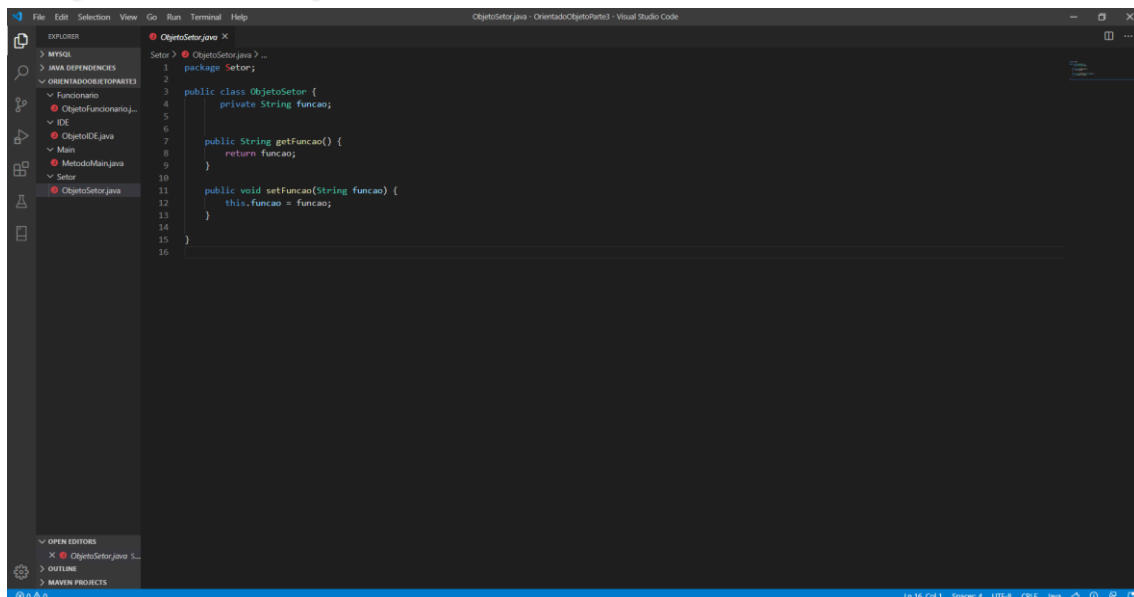


```

1 package funcionario;
2 public class ObjetoFuncionario {
3
4     private String nome;
5     private String tecnologia;
6
7     void realizaProjeto(){
8     }
9
10    public String getNome() {
11        return nome;
12    }
13
14    public void setNome(String nome) {
15        this.nome = nome;
16    }
17
18    public String getTecnologia() {
19        return tecnologia;
20    }
21
22    public void setTecnologia(String tecnologia) {
23        this.tecnologia = tecnologia;
24    }
25 }

```

ObjetoSetor.java



```

1 package Setor;
2
3 public class ObjetoSetor {
4     private String funcao;
5
6     public String getFuncao() {
7         return funcao;
8     }
9
10    public void setFuncao(String funcao) {
11        this.funcao = funcao;
12    }
13
14
15 }
16

```

Após concluirmos nosso Projeto , com os Package criados e seus objetos junto de seus atributos, no método Main traremos nosso SOUT para que receba por parâmetro nossos métodos Getters cujos quais foram criados nas Classes dos respectivos Objetos.

```

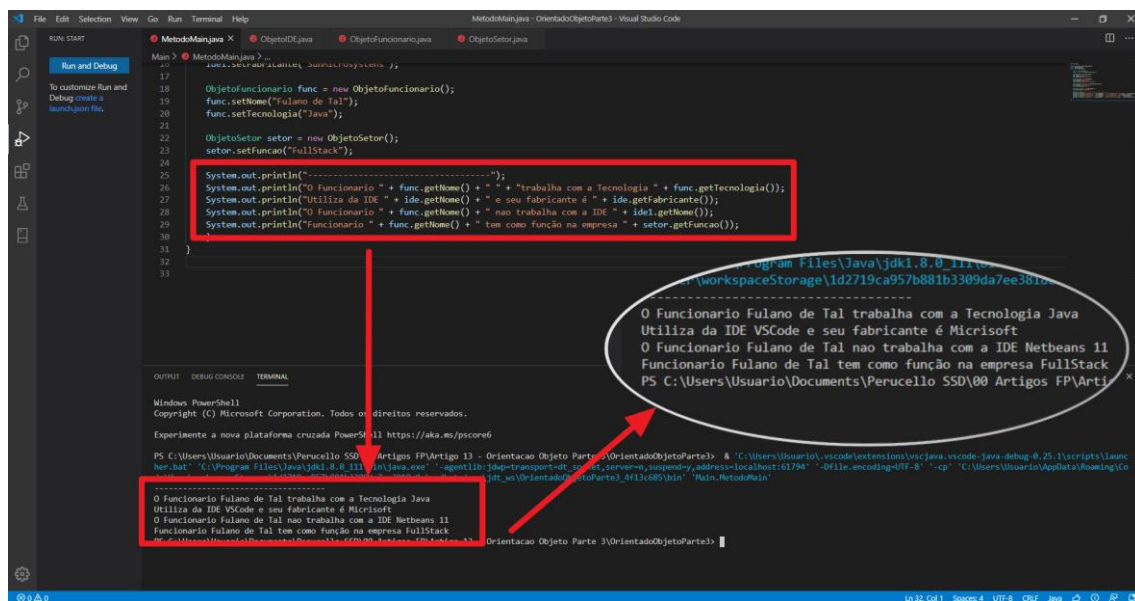
System.out.println("-----");
System.out.println("O Funcionario " + func.getNome() + " " + "trabalha com a Tecnologia " + func.getTecnologia());
System.out.println("Utiliza da IDE " + ide.getNome() + " e seu fabricante é " + ide.getFabricante());
System.out.println("O Funcionario " + func.getNome() + " nao trabalha com a IDE " + ide1.getNome());
System.out.println("Funcionario " + func.getNome() + " tem como função na empresa " + setor.getFuncao());

```

As características dos objetos também foram criados no próprio Main onde passamos via Setters.

```
10 ObjetoIDE ide = new ObjetoIDE();
11 ide.setNome("VSCode");
12 ide.setFabricante("Micrisoft");
13
14 ObjetoIDE ide1 = new ObjetoIDE();
15 ide1.setNome("Netbeans 11");
16 ide1.setFabricante("SunMicrosystems");
17
18 ObjetoFuncionario func = new ObjetoFuncionario();
19 func.setNome("Fulano de Tal");
20 func.setTecnologia("Java");
21
22 ObjetoSetor setor = new ObjetoSetor();
23 setor.setFuncao("FullStack");
```

Agora, nos resta executarmos nosso código para vermos se obtivemos êxito em nossa atividade.



```
System.out.println("O Funcionario " + func.getNome() + " trabalha com a Tecnologia " + func.getTecnologia());
System.out.println("Utiliza da IDE " + ide.getNome() + " e seu fabricante é " + ide.getFabricante());
System.out.println("O Funcionario " + func.getNome() + " nao trabalha com a IDE " + ide1.getNome());
System.out.println("Funcionario " + func.getNome() + " tem como função na empresa " + setor.getFuncao());
```

O Funcionario Fulano de Tal trabalha com a Tecnologia Java
Utiliza da IDE VSCode e seu fabricante é Micrisoft
O Funcionario Fulano de Tal nao trabalha com a IDE Netbeans 11
Funcionario Fulano de Tal tem como função na empresa FullStack

Os códigos estarão disponíveis no Git.

Até mais !

Espero ter ajudado !