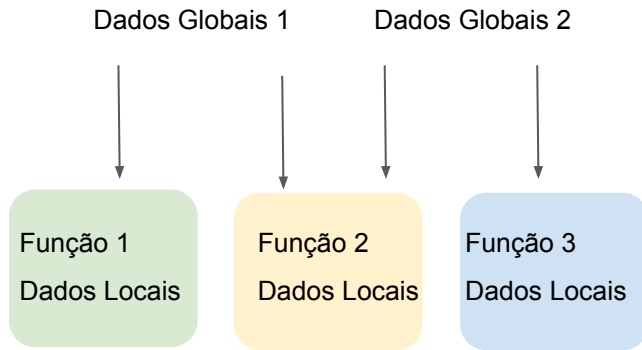


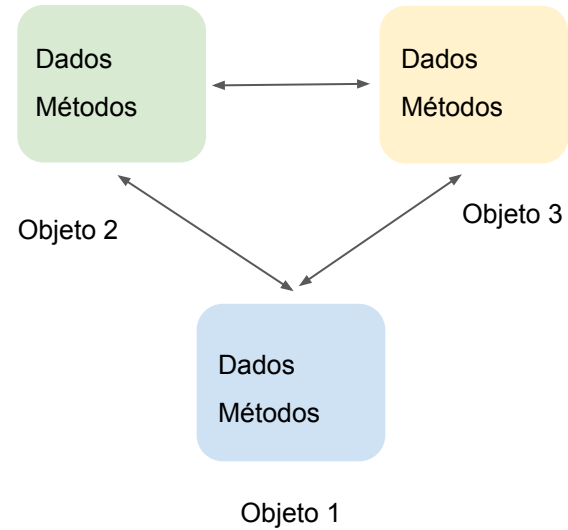
Introdução a Programação Orientada a Objetos

Lilian Passos Scatalon
lpscatalon2@uem.br

Programação Imperativa



Programação Orientada a Objetos



Conceitos de POO

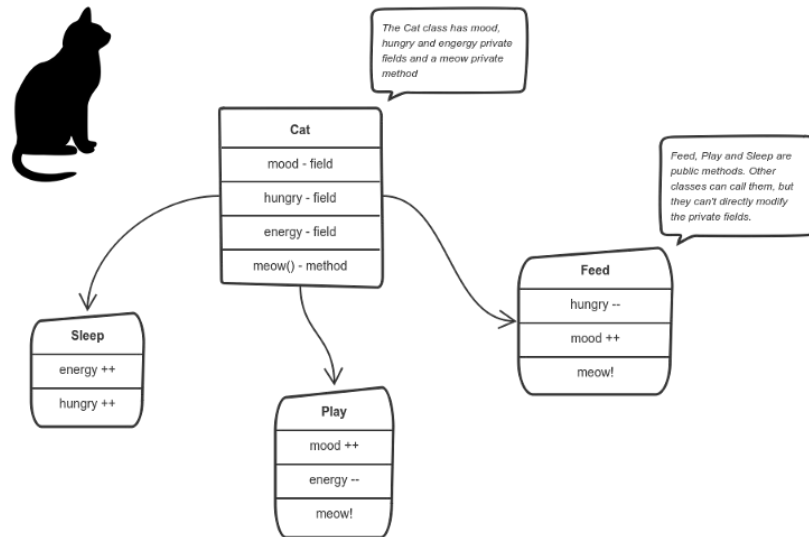
1. Encapsulamento
2. Abstração
3. Herança
4. Polimorfismo

(1) Encapsulamento

Mecanismo que vincula código e os dados manipulados por ele

Uma caixa-preta autocontida é criada: objeto

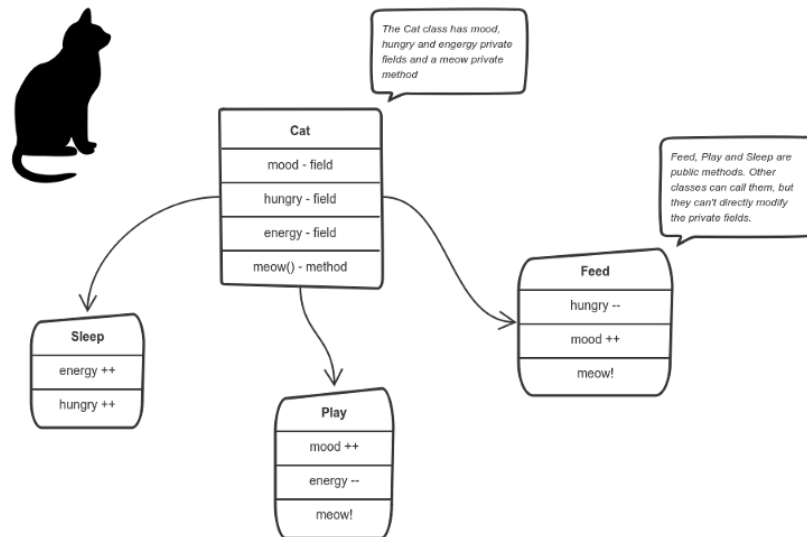
Dentro do objeto, seus membros (código e dados) podem ser privados ou públicos



(1) Encapsulamento (cont.)

A classe é o modelo para criar objetos

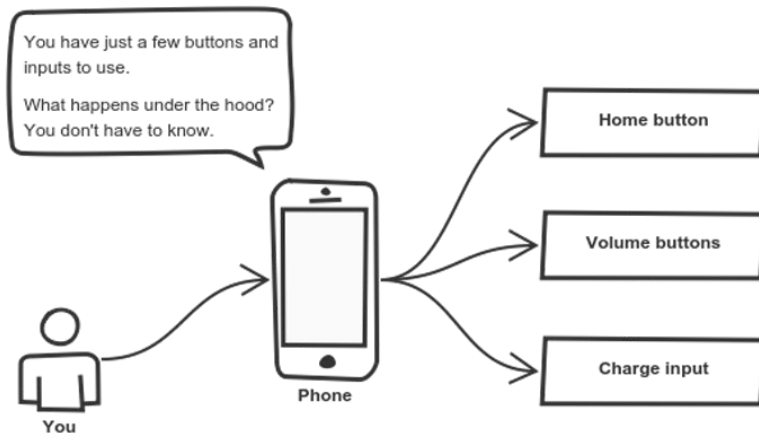
Objetos são instâncias de uma classe



(2) Abstração

Cada objeto deve expor apenas um mecanismo de alto nível para sua utilização

Detalhes internos de implementação ficam ocultos



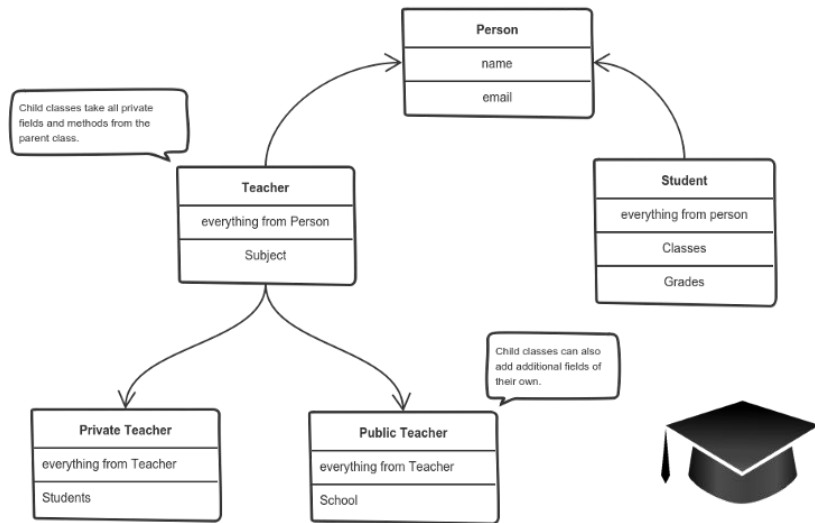
(3) Herança

Mecanismo de reuso

Objetos podem ser similares, mas não exatamente iguais

Uma classe filha pode ser derivada a partir de uma classe mãe

Forma-se uma hierarquia

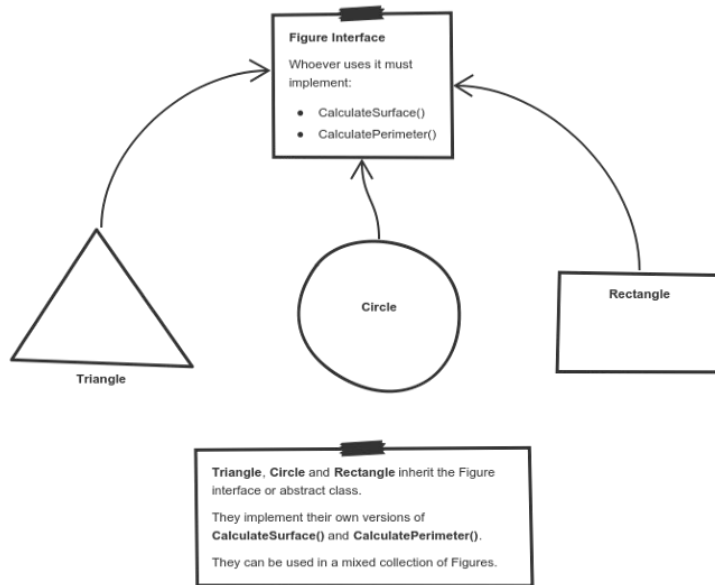


(4) Polimorfismo

“Muitas formas”

Mecanismo que permite usar um objeto da classe filha do mesmo modo como um da classe mãe

Uma interface comum/geral na classe mãe permite o acesso a ações específicas das classes filhas



Suporte a POO

Uma LP Orientada a Objetos provê suporte aos conceitos de encapsulamento, abstração, herança e polimorfismo

Exemplos: C++, [Java](#), Python etc

Plataforma Java

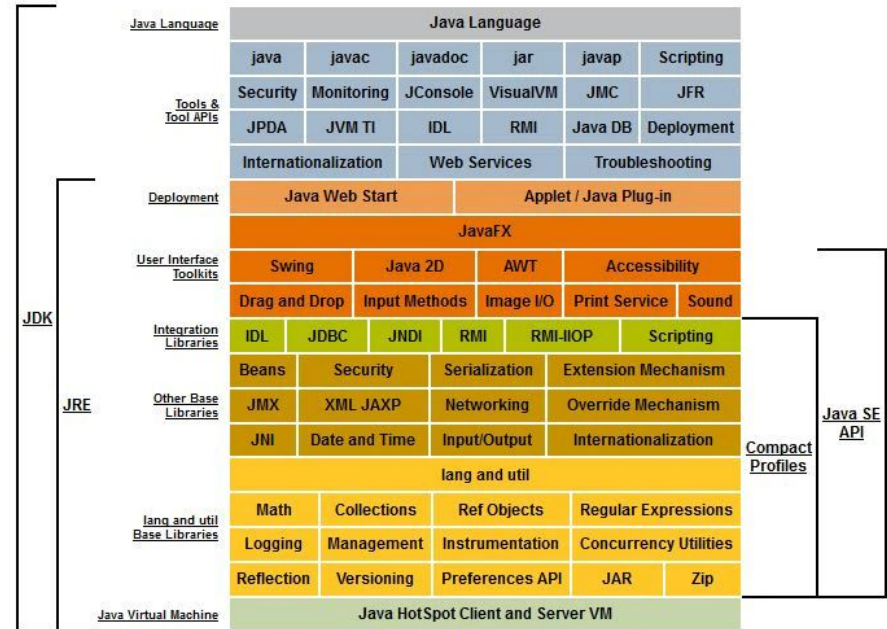
Conjunto de tecnologias

Linguagem Java

Ferramentas

Bibliotecas

Máquina Virtual Java (JVM)



Fonte: <https://www.oracle.com/java/technologies/platform-glance.html>

História do Java

Desenvolvido na Sun Microsystems no início da década de 1990

Time liderado por James Gosling

Inicialmente disseminado para executar pequenas aplicações no navegador Web

A Sun foi adquirida posteriormente em 2009 pela Oracle

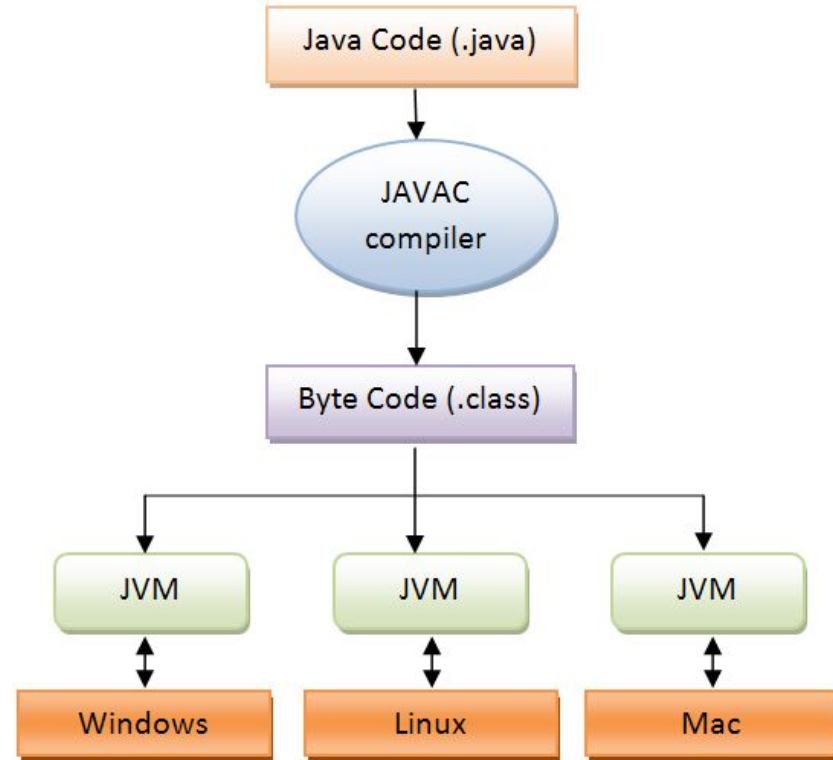
<https://www.oracle.com/java/>

Execução de Programas em Java

Implementação híbrida, envolvendo compilação e interpretação

O código-fonte (arquivo .java) é compilado para um código intermediário, o bytecode (arquivo .class)

Compilador javac



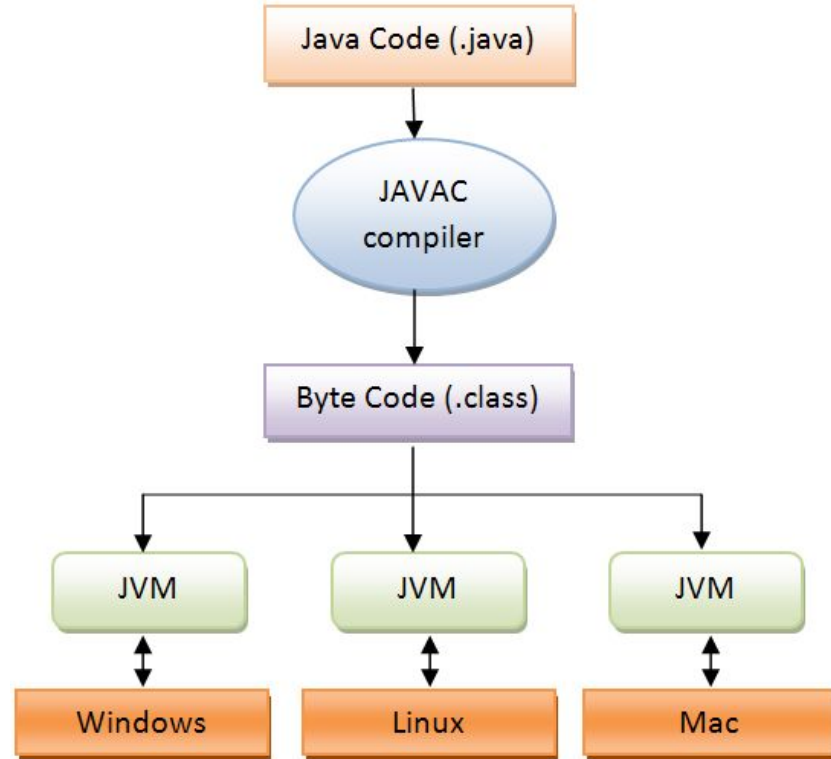
Execução de Programas em Java (cont.)

O bytecode é então interpretado pela JVM, instrução a instrução

A JVM “não conhece” a linguagem Java

Há uma versão da JVM para cada Sistema Operacional

“Write once, run everywhere”



Java Development Kit (JDK)

Disponível em <https://www.oracle.com/java/technologies/downloads/>

Composto pela JVM, bibliotecas e ferramentas necessárias para o desenvolvimento de programas em Java

Máquina Virtual Java

- Especificação da JVM
 - Documento que determina como o bytecode deve ser interpretado
 -
- Implementação da JVM
 - JVM implementada de acordo com a especificação
 - Oracle JVM (inclusa no JDK), OpenJDK, IBM J9 etc

The Java® Virtual
Machine Specification
Java SE 18 Edition

Tim Lindholm
Frank Yellin
Gilad Bracha
Alex Buckley
Daniel Smith

2022-02-23

HelloWorld.java

Para escrever e executar o primeiro programa, é necessário:

- O JDK (Java Development Kit)
- Um editor de texto

HelloWorld.java

Para criar a primeira aplicação:

1. Criar um arquivo fonte .java
2. Compilar o arquivo fonte .java em um arquivo .class
3. Executar o programa

(1) Arquivo fonte

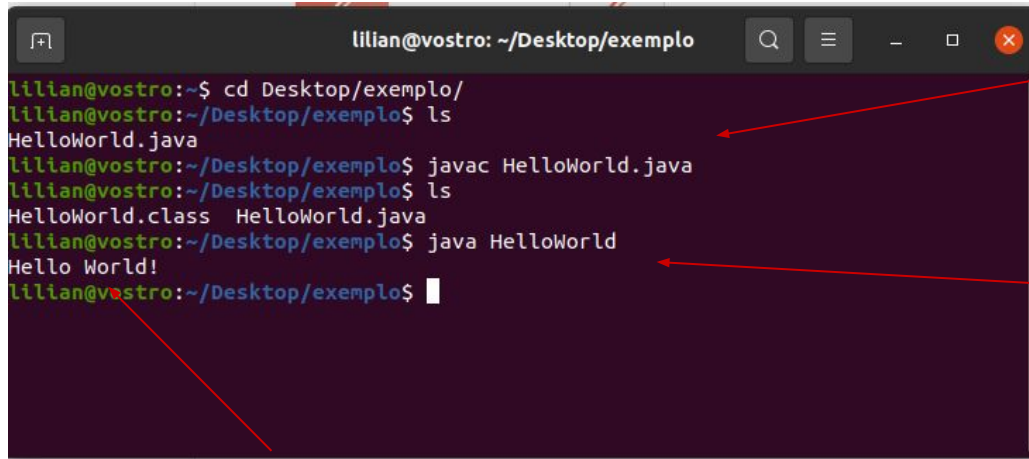


The image shows a code editor window with a dark theme. The title bar at the top reads 'HelloWorld.java' and shows the file path '~/.Desktop/exemplo'. The editor contains the following Java code:

```
1 |
2 public class HelloWorld {
3     public static void main(String[] args) {
4         System.out.println("Hello World!");
5     }
6 }
```

The status bar at the bottom indicates the file is a 'Java' file, the 'Tab Width' is 8, the cursor is at 'Ln 1, Col 1', and the input mode is 'INS'.

(2) Compilação e (3) Execução



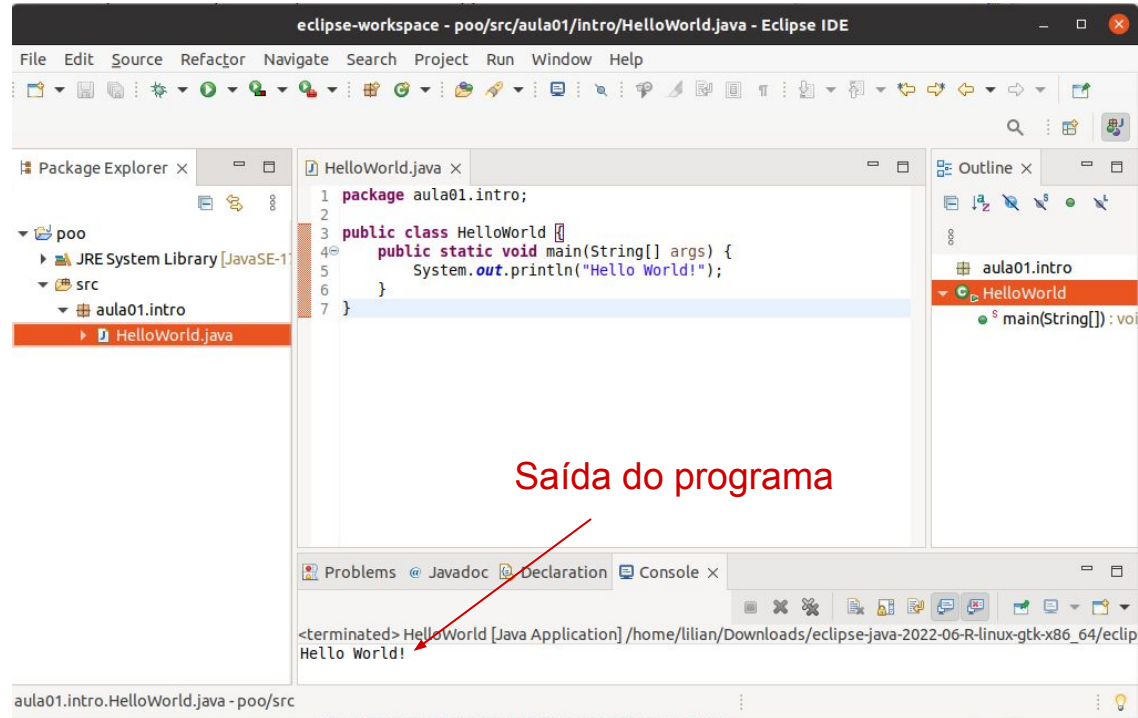
```
lilian@vostro: ~/Desktop/exemplo
lilian@vostro:~$ cd Desktop/exemplo/
lilian@vostro:~/Desktop/exemplo$ ls
HelloWorld.java
lilian@vostro:~/Desktop/exemplo$ javac HelloWorld.java
lilian@vostro:~/Desktop/exemplo$ ls
HelloWorld.class HelloWorld.java
lilian@vostro:~/Desktop/exemplo$ java HelloWorld
Hello World!
lilian@vostro:~/Desktop/exemplo$
```

Arquivo fonte

Arquivo class

Saída do programa

IDE Eclipse



<https://wiki.eclipse.org/Eclipse/Installation>
<https://www.eclipse.org/downloads/packages/installer>

HelloWorld.java

```
1 package aula01.intro;  
2  
3 public class HelloWorld {  
4     public static void main(String[] args) {  
5         System.out.println("Hello World!");  
6     }  
7 }
```

HelloWorld é uma classe

Em Java, todo o código é definido dentro de classes

HelloWorld.java

```
1 package aula01.intro;
2
3 public class HelloWorld {
4     public static void main(String[] args) {
5         System.out.println("Hello World!");
6     }
7 }
```

main é um método, ou seja, uma função declarada dentro de uma classe

O método main é o primeiro a ser chamado quando o programa executa

HelloWorld.java

```
1 package aula01.intro;  
2  
3 public class HelloWorld {  
4     public static void main(String[] args) {  
5         System.out.println("Hello World!");  
6     }  
7 }
```

A classe HelloWorld e o método main estão declarados como public

Há outros controles de acesso, como private

HelloWorld.java

```
1 package aula01.intro;
2
3 public class HelloWorld {
4     public static void main(String[] args) {
5         System.out.println("Hello World!");
6     }
7 }
```

Um pacote é um conjunto de classes relacionadas

Ajuda a evitar conflitos quando há múltiplas classes com o mesmo nome

HelloWorld.java

```
1 package aula01.intro;
2
3 public class HelloWorld {
4     public static void main(String[] args) {
5         System.out.println("Hello World!");
6     }
7 }
```

No corpo do método main há um único comando

System.out é um objeto que representa a saída padrão do programa Java

println() é um método que recebe uma string como parâmetro e a imprime

Chamada de métodos

```
System.out.println("Hello World!");
```

`System.out` é um objeto, instância da classe `PrintStream`

A classe `PrintStream` tem métodos `println()`, `print()`, etc

Para invocar um método em um objeto, usa-se o ponto

objeto.nomeMetodo(argumentos)

Nesse caso, há apenas um argumento, a string “Hello, World!”

Bibliografia

H. Schildt. **Java: A Beginner's Guide**. McGraw-Hill Education, 8th edition, 2018.

C. S. Horstmann. **Core Java SE 9 for the Impatient**. Addison-Wesley, 2nd Edition, 2018.

Caelum. Apostila **Java e Orientação a Objetos**. Disponível em <https://www.alura.com.br/apostila-java-orientacao-objetos>