



## **AULA 1 – CONCEITOS INTRODUTÓRIOS**

# **PROGRAMAÇÃO ORIENTADA A OBJETOS**

Prof. Ms. Nathan Cirillo e Silva  
Prof. Ms. Peter Jád Júnior  
Prof. Ms. Télvio Orru

**Universidade Paulista UNIP**

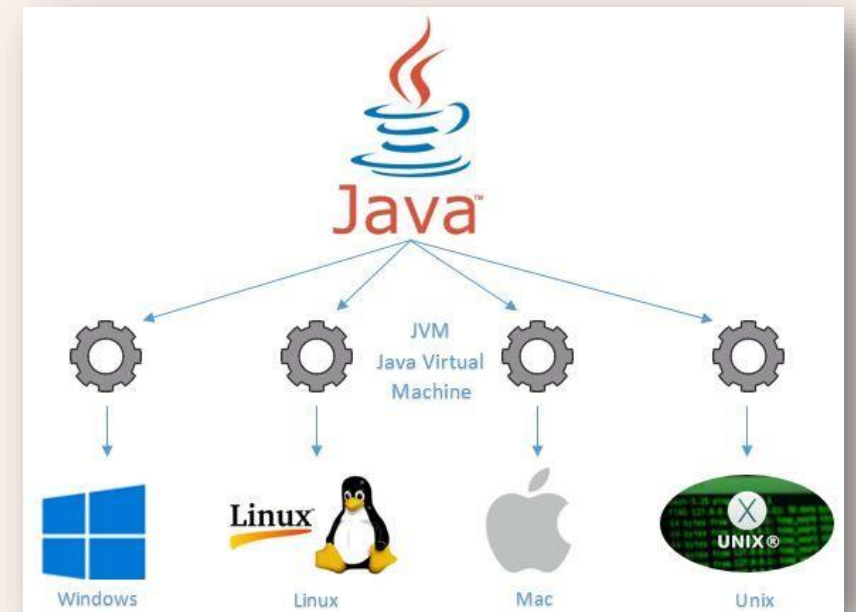
# Características da Linguagem

- Foi concebida com a ideia de escrever uma única aplicação e executá-la em vários dispositivos;
- É uma linguagem robusta, segura, estável e que permite retrocompatibilidade;
- É possível executar atualmente uma aplicação legada com versões mais atuais do Java;
- A comunidade JCP (*Java Community Process*) garante o padrão de estabilidade e compatibilidade.



# Rode em qualquer lugar!!!

- Diferente de outras linguagens, o Java pode ser executado em Windows, Linux e Mac OS;
- Esse tem sido desde então o seu principal atrativo, pois abre muitas oportunidades;
- O seu slogan oficial tem sido o seguinte:  
*Write once, run anywhere;*
- A chave para a portabilidade é a JVM (*Java Virtual Machine*).



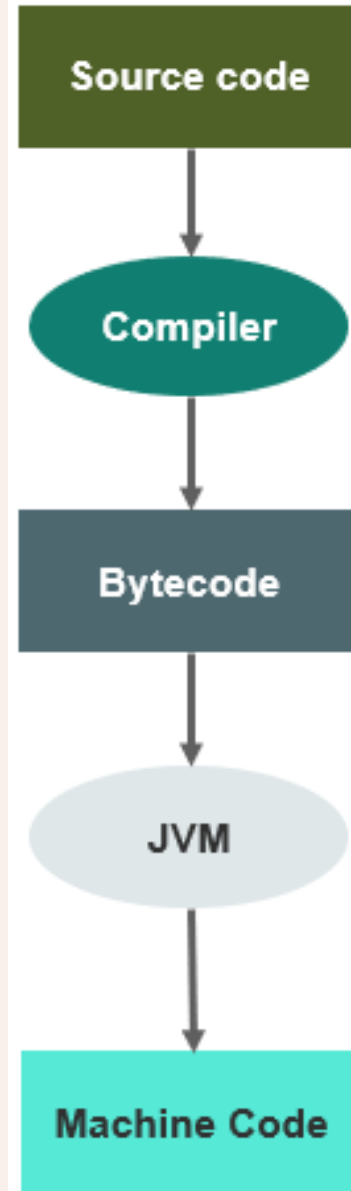
# Então para que o Java é usado?

- Possibilita criar aplicativos corporativos de grande porte;
- Aprimora a funcionalidade dos servidores web;
- Presente na maior parte dos dispositivos populares;
- É a linguagem-chave para criar aplicativos Android.

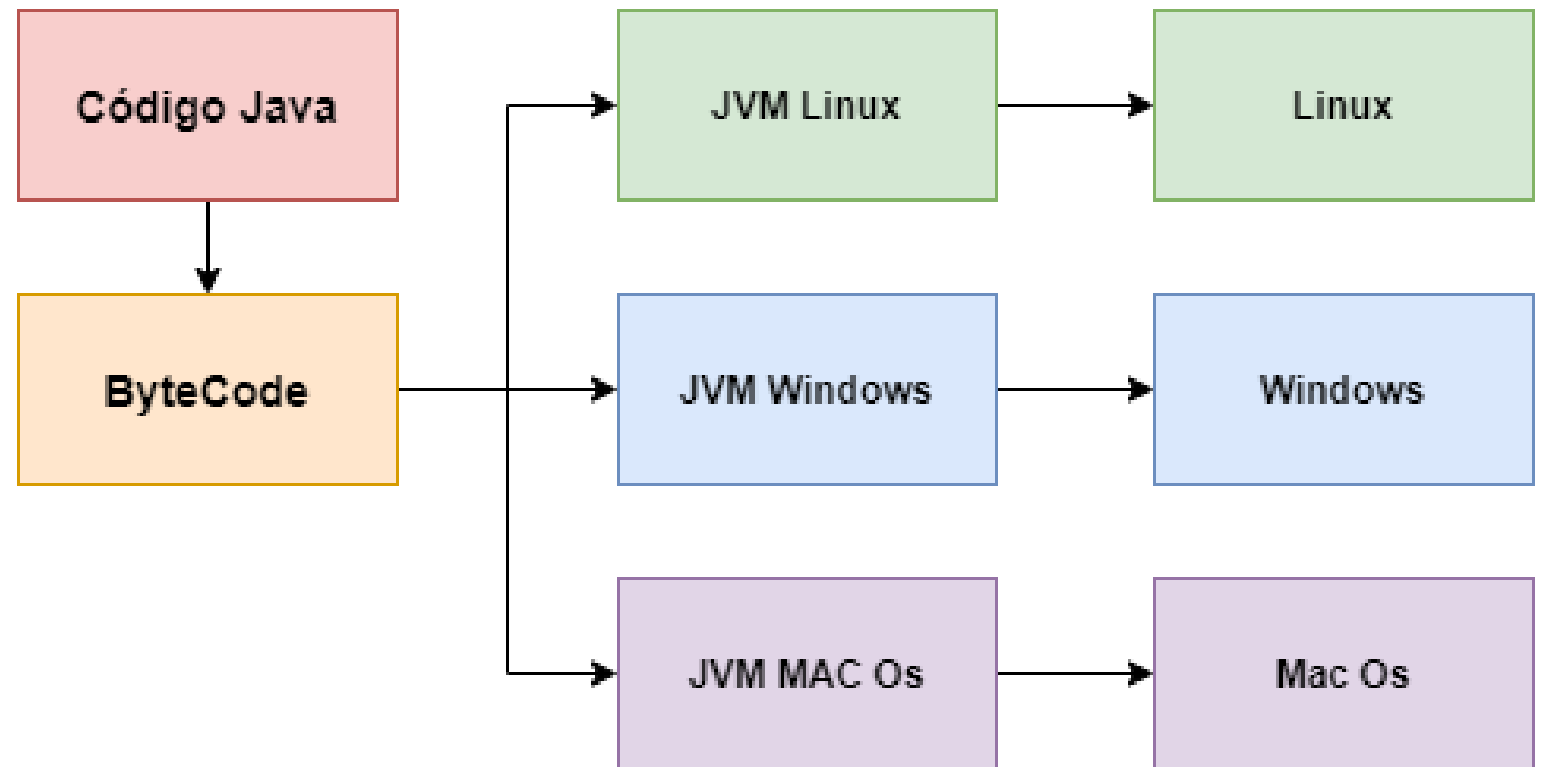


# Java Virtual Machine (JVM)

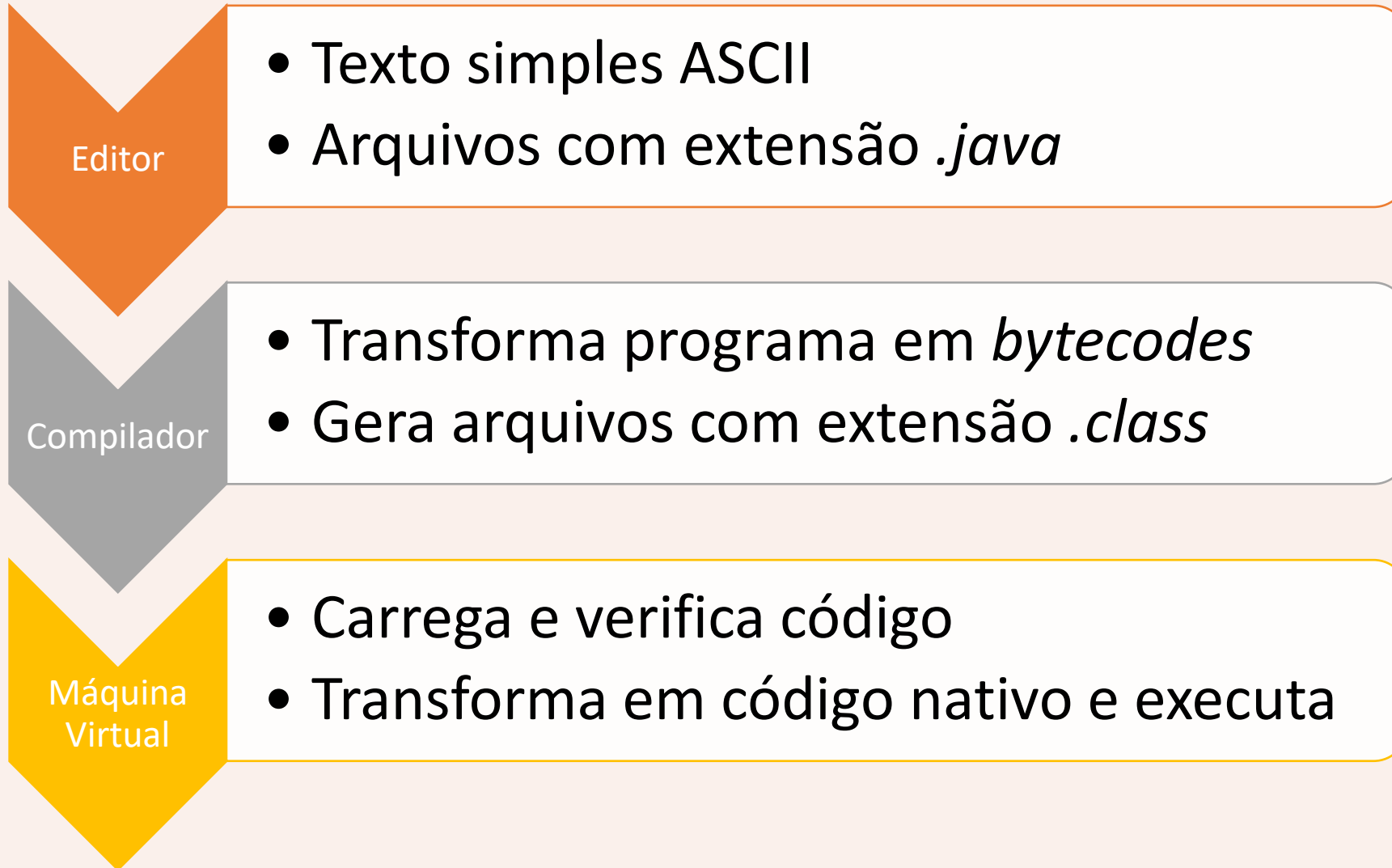
- A ideia por de trás da JVM é fornecer independência do hardware que roda a aplicação Java;
- O código fonte é compilado para um formato intermediário conhecido como bytecode;
- O bytecode independe da arquitetura do sistema em que foi gerado, ou seja, é portátil;
- Dessa forma, ele poderá ser executado em qualquer plataforma que tenha uma JVM instalada.



# Processo de Compilação



# Ciclo de Compilação



# KIT de Desenvolvimento Java

- O JDK ou JSDK (*Java Software Development Kit*) é o que permite criar as aplicações;
- Fornece um ambiente completo de desenvolvimento Java, ou seja, um ecossistema;



- O JDK é constituído basicamente por: *Bibliotecas, Compilador e Máquina Virtual (JVM)*;
- Portanto, para criar uma aplicação em Java é obrigatório a instalação do JDK.



# Programa Mínimo

Deve haver uma classe com o método principal:

```
public static void main(String[]args)
```

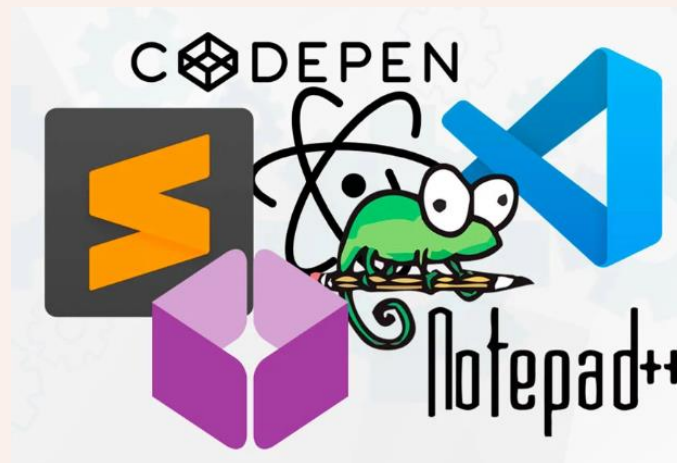


```
public class Exemplo → Declaração da classe
{
    public static void main(String []args) → Método principal
    {
        System.out.println("Olá Pessoal!"); → Código
    }
}
```

# Criar e Editar um Programa

Use qualquer editor que salve arquivos de texto sem formatação. Ideal que seja capaz de:

- tratar nomes longos;
- exibir contagem de linhas;
- fazer destaque de sintaxe; e
- possibilitar a criação de macros.



# Compilação via Terminal

Itens necessários:

kit de desenvolvimento padrão (JDK);

Arquivos fonte (\*.java)

Comandos executados via terminal:

`javac NomeDaClasse.java`

Atenção ao NomeDaClasse e a extensão .java!



```
C:\Users\Nathan Cirillo\Desktop\Programas Aula Java>javac Exemplo.java ← 1

C:\Users\Nathan Cirillo\Desktop\Programas Aula Java>dir
O volume na unidade C não tem nome.
O Número de Série do Volume é B84D-C518

Pasta de C:\Users\Nathan Cirillo\Desktop\Programas Aula Java

08/02/2023  21:10    <DIR>          .
08/02/2023  20:34    <DIR>          ..
08/02/2023  21:10             413 Exemplo.class ← 3
08/02/2023  20:34             107 Exemplo.java ← 2
```

# Execução via Terminal

## Itens necessários:

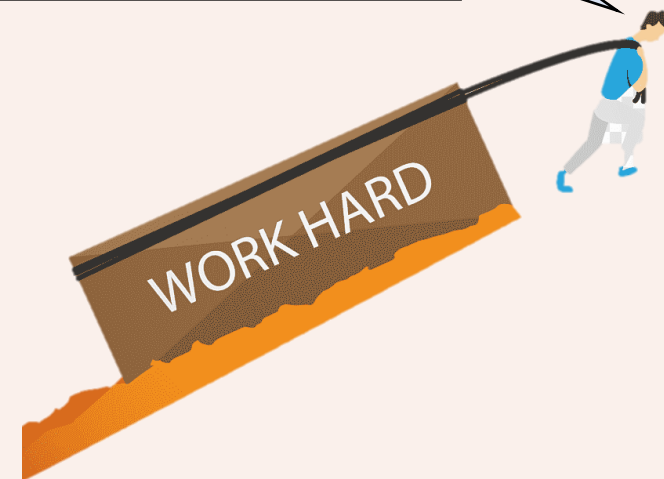
Kit de desenvolvimento padrão (JDK);  
Arquivo compilado (\*.class).

## Acionar arquivo principal da aplicação:

```
java NomeDaClasse
```

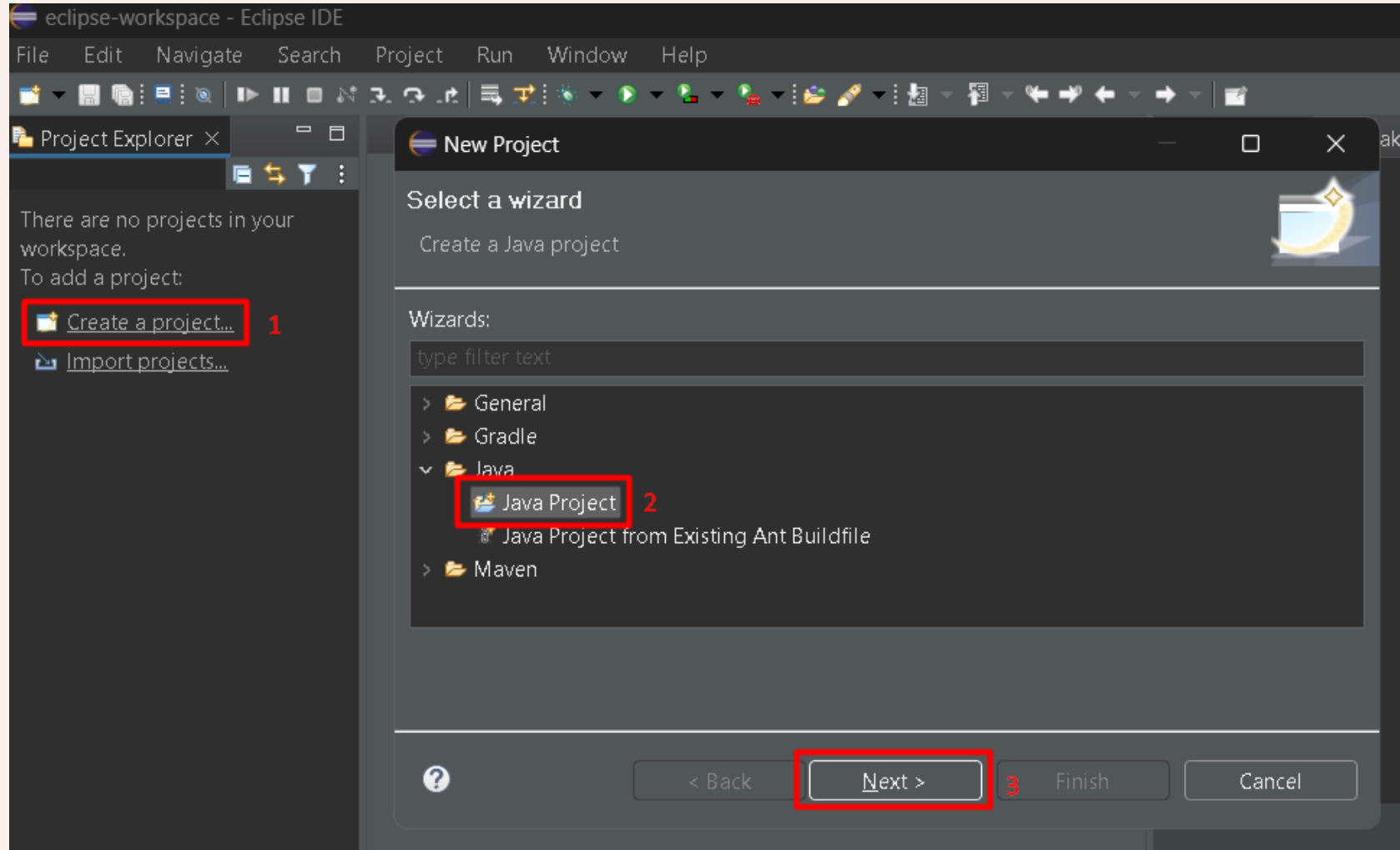
Na execução não se fornece a extensão .class!

Melhor é usar um IDE (*Integrated Development Environment*) para Java, como o Eclipse!

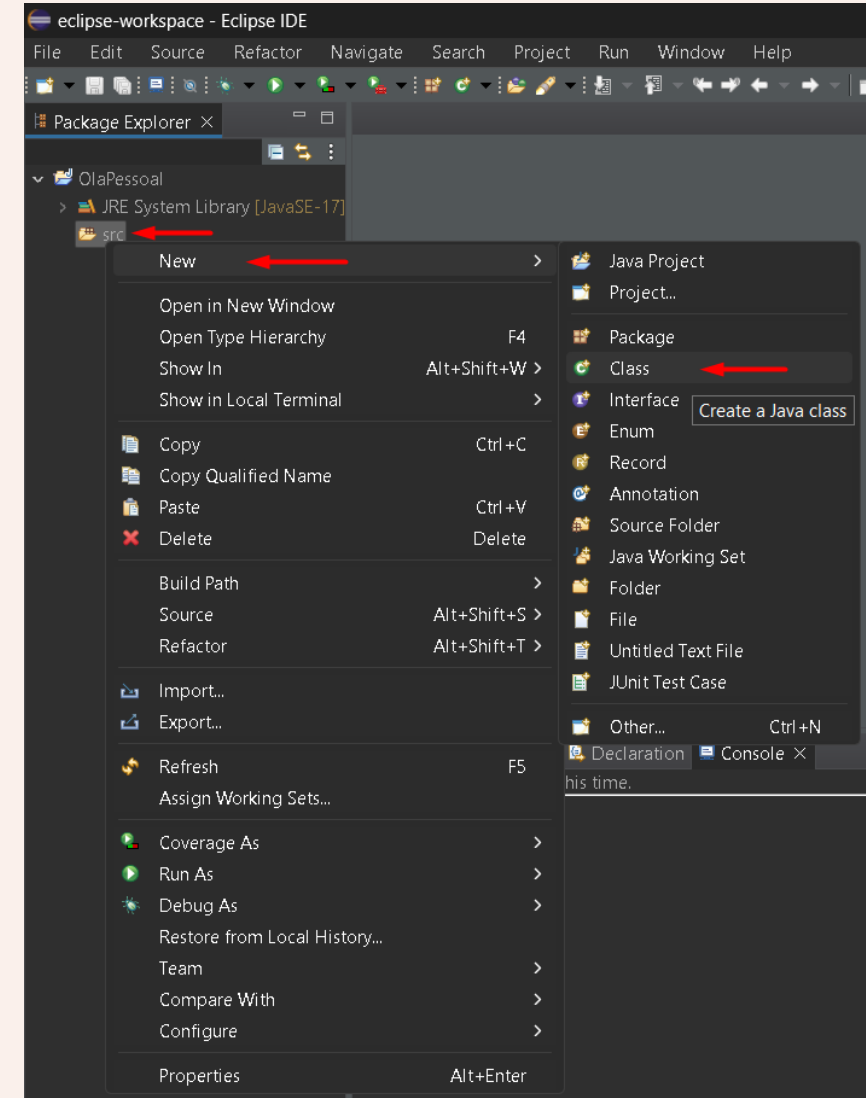
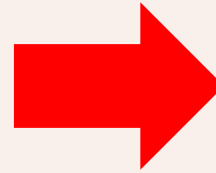
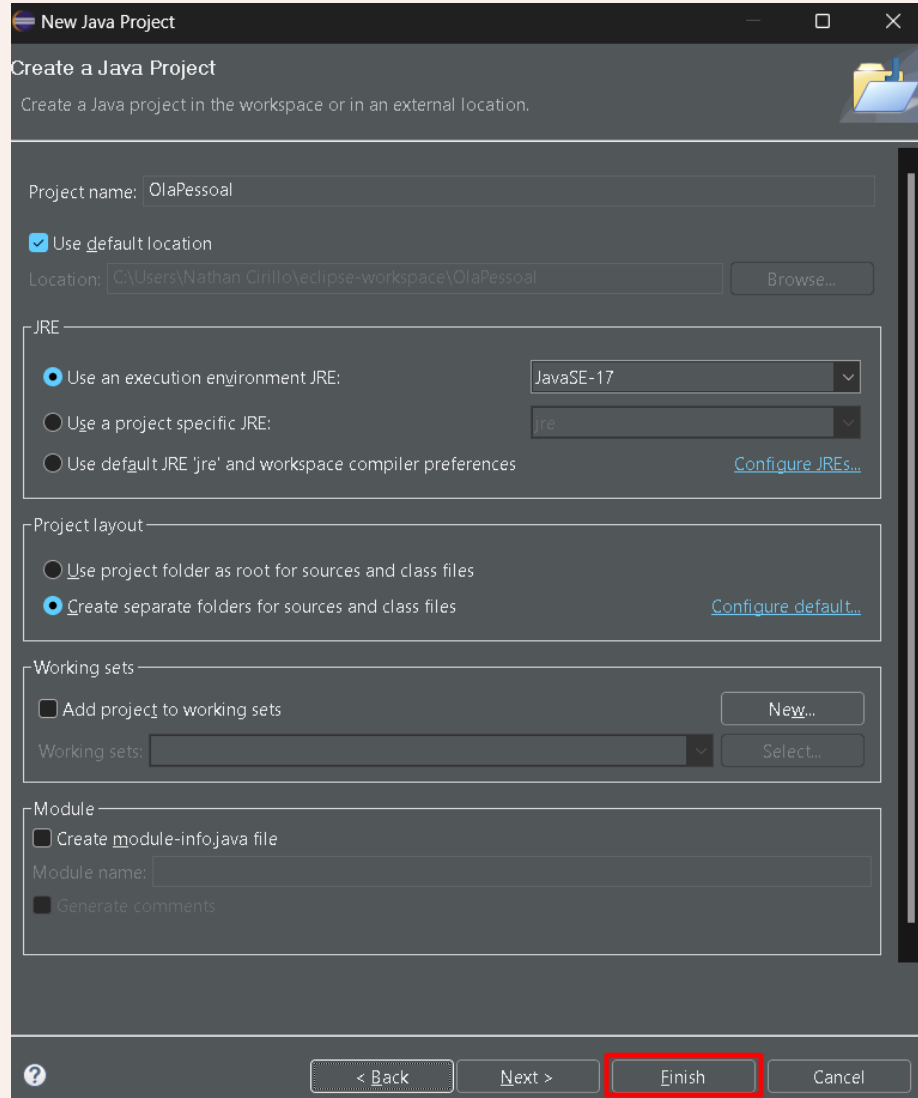


```
C:\Users\Nathan Cirillo\Desktop\Programas Aula Java>java Exemplo 1  
Olá Pessoal! 2
```

# Criando um Projeto no Eclipse



# Criando um Projeto no Eclipse



# Criando um Projeto no Eclipse

New Java Class

Java Class  
Create a new Java class.

Source folder: OlaPessoal/src Browse...

Package: exemplos Browse...

☐ Enclosing type: Browse...

Name: OlaPessoal

Modifiers: ☒ public ☐ package ☐ private ☐ protected  
☐ abstract ☐ final ☐ static  
☒ none ☐ sealed ☐ non-sealed ☐ final

Superclass: java.lang.Object Browse...

Interfaces: Add...

Which method stubs would you like to create?

☒ public static void main(String[] args)

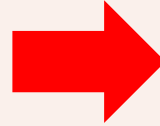
☐ Constructors from superclass

☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

Finish Cancel



eclipse-workspace - OlaPessoal/src/exemplos/OlaPessoal.java - Eclipse IDE

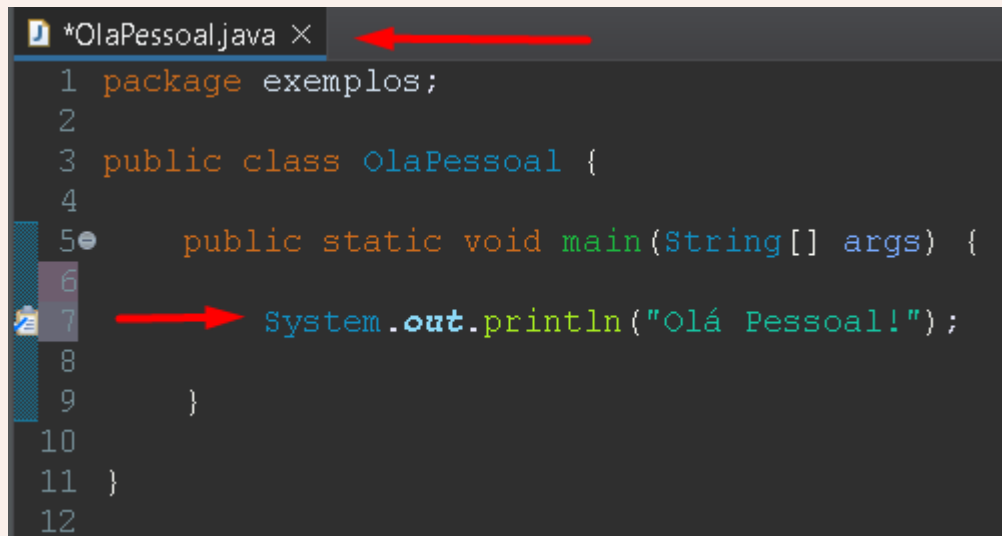
File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer X OlaPessoal.java X

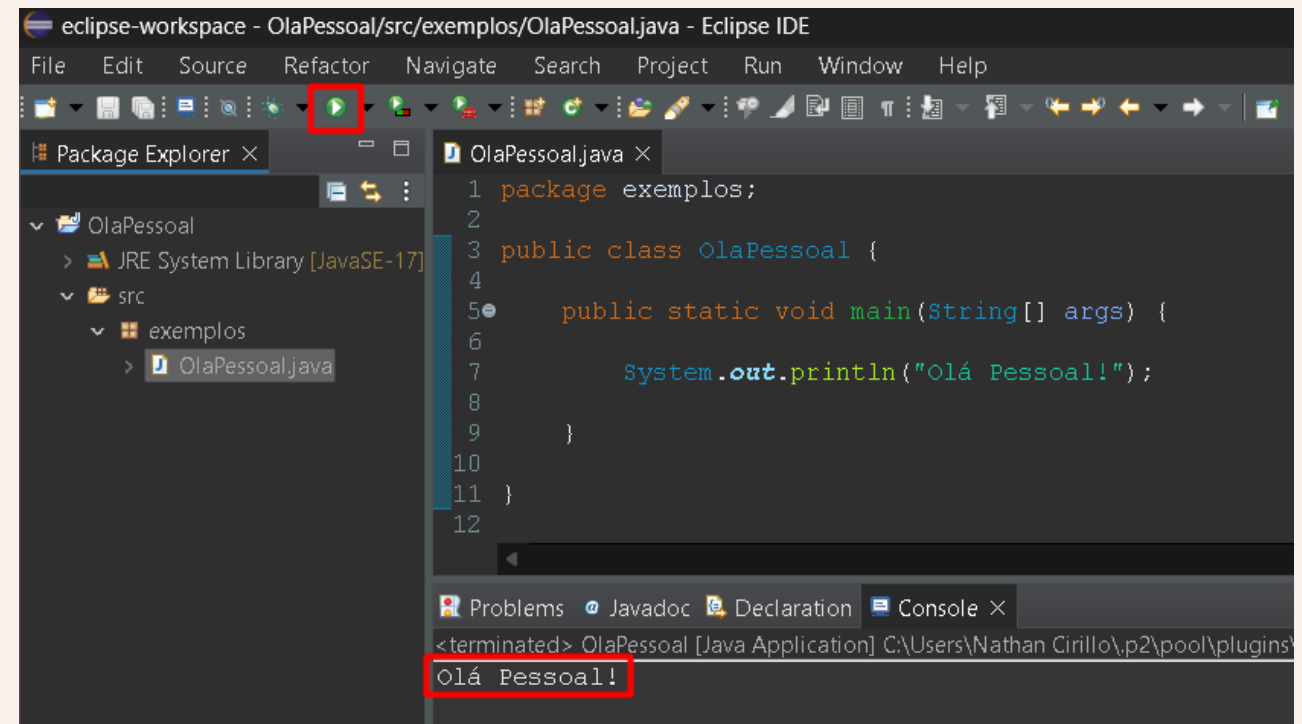
```
1 package exemplos;
2
3 public class OlaPessoal {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7     }
8
9
10 }
11
```



# Executando o Projeto



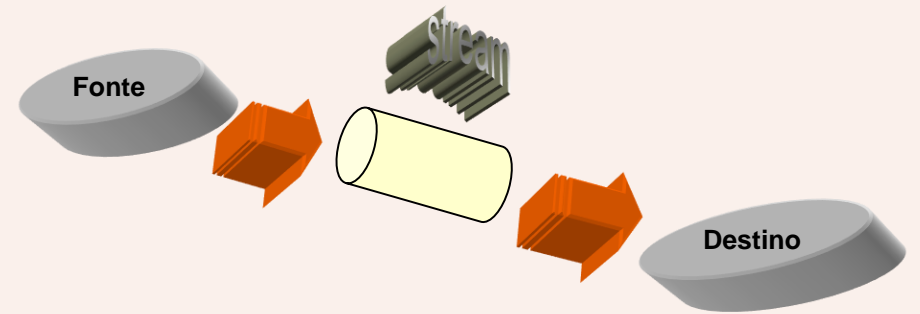
```
*OlaPessoal.java X
1 package exemplos;
2
3 public class OlaPessoal {
4
5     public static void main(String[] args) {
6
7         System.out.println("Olá Pessoal!");
8
9     }
10
11 }
12
```





# Saída de Dados

- A saída padrão de dados (default) é o próprio console;
- É acessado via classe `java.lang.System` que oferece o objeto `out`;
- `out` é uma stream de dados que leva dados da aplicação para o console (i.e. “imprime na tela”).



# System.out

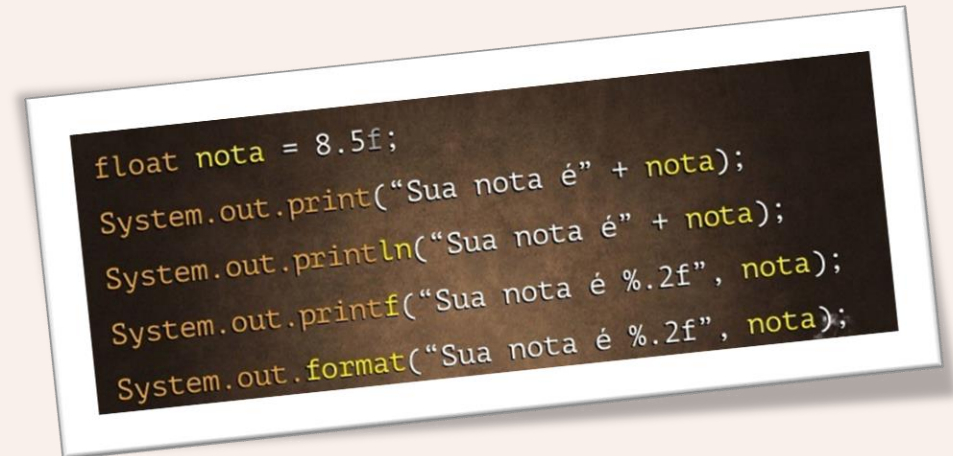
Objeto do tipo `java.io.PrintStream`;

Métodos importantes:

`print(argumento);`

`println(argumento);`

`printf("formato", argumentos);` // ling C



Onde argumento pode ser: *inteiro, String, real, char etc.*

# Orientação a Objetos



- Eleva o desenvolvimento de sistemas para um novo patamar de organização;
- A estrutura dos programas passa a ser baseada em conceitos do mundo real (**classes**);
- As partes do programa (**objetos**) são separadas por responsabilidades, facilitando a manutenção;
- Os objetos, se comunicam entre si, por meio de um **mecanismo de troca mensagens**.

# P00 vs Estruturada

- Na programação estruturada o sistema é decomposto em partes menores (funções);
- Passa a existir um emaranhado de inúmeras funções que chamam umas às outras;
- Não há separação de responsabilidades, o sistema passa a ter enormes dependências;
- Os código ficam duplicados, não havendo reaproveitamento, dificultando as manutenções.



# Vantagens da P00

- Aumento de produtividade;
- Reuso de código;
- Redução de linhas de código;
- Separação de responsabilidades;
- Maior flexibilidade do sistema;
- Escalabilidade.



# O que são Objetos?



- Tudo ao nosso redor é um objeto, seja ele algo físico ou abstrato: *mesa, cadeira, livro*;
- Todo objeto tem duas características essenciais: o estado e o comportamento;
- Estado: são as suas características físicas, por exemplo: *cor, tamanho, peso, largura, altura*;
- Comportamento: são as suas capacidades, o que ele é capaz de fazer: *andar, cantar, imprimir*.

# Modelando os Objetos

- Os objetos modelados via programação também possuem estado e comportamento;
- O estado é mantido por variáveis tipadas, conhecidas como atributos ou propriedades;
- Os atributos de um dado objeto são responsáveis por determinar as suas características;
- Através dos seus respectivos valores é possível distinguir objetos que são semelhantes.



# Comportamentos de um Objeto



- **Métodos**

- Falar
- Andar
- Comer

➤ O comportamento de um objeto é definido pelo uso de uma ou mais funções;

➤ No paradigma OO, as funções são chamadas de métodos ou procedimentos;

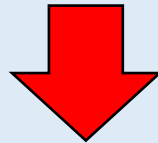
➤ Os métodos de um objeto são os responsáveis por determinar o seu comportamento;

➤ Ao serem executados, realizam tarefas, como alterar o estado das variáveis de um objeto.



# Exemplos de Objetos

## Objeto Real: Veículo



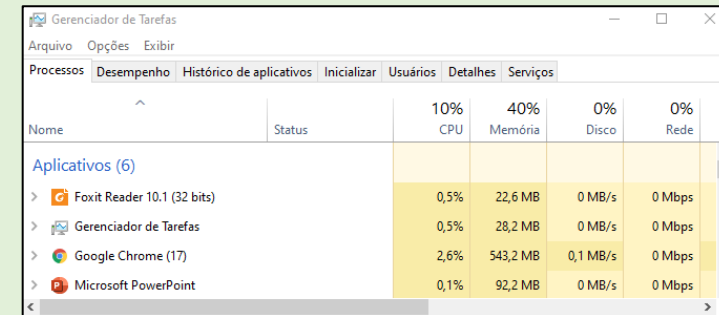
### Propriedades (estado):

- Velocidade;
- Número de portas;
- Limite de passageiros.

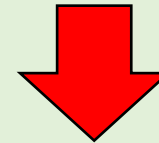
### Procedimentos (ações):

- Ligar e Desligar;
- Acelerar;
- Parar.

## Objeto Abstrato: Processo



Nome	Status	10% CPU	40% Memória	0% Disco	0% Rede
<b>Aplicativos (6)</b>					
> Foxit Reader 10.1 (32 bits)		0,5%	22,6 MB	0 MB/s	0 Mbps
> Gerenciador de Tarefas		0,5%	28,2 MB	0 MB/s	0 Mbps
> Google Chrome (17)		2,6%	543,2 MB	0,1 MB/s	0 Mbps
> Microsoft PowerPoint		0,1%	92,2 MB	0 MB/s	0 Mbps



### Propriedades (estado):

- Identificação;
- Prioridade;
- Privilégios.

### Procedimentos (ações):

- Criar;
- Eliminar;
- Alterar.

# Utilizando o Conhecimento Comum



Use sempre duas perguntas bastante simples:

O que uma conta tem que é importante?	O que uma conta faz que é importante?
Número da Conta	Sacar
Saldo	Depositar
Nome do Titular	Consultar Saldo

*Poderiam haver outros! Isso vai depender da necessidade de representação identificada pelo programador!*

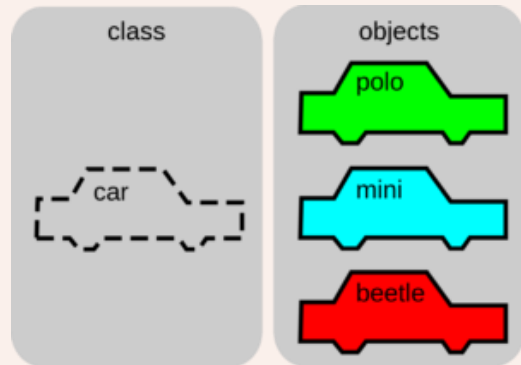
# Tempo de Vida dos Objetos

- Geralmente, os objetos são criados e eliminados em função da execução do programa;
- Um objeto pode ser criado (*instanciado*) por um certo período de tempo e depois eliminado;



- Ao eliminar o objeto, o espaço em memória é liberado, através do processo *garbage collector*;
- É possível criar objetos persistentes através da utilização de sistemas de banco de dados.

# O que são Classes?



- Uma classe funciona como uma espécie de molde para a definição (criação) de objetos;
- Possui atributos e métodos para representar as suas características e comportamentos;
- Através da classe é possível criar tipos de dados por referência e definir variáveis desse tipo;
- Um objeto nada mais é do que a instância de uma classe, a classe é estática e o objeto dinâmico.

# Agora é a sua vez!

---

Para cada um dos tipos abaixo, determine atributos e métodos:

- ☐ Avião;
- ☐ Televisão;
- ☐ Telefone;
- ☐ Aluno;
- ☐ Pedido de Compra;
- ☐ Pessoa.



Dúvidas?

ANY  
QUESTIONS

