

# LINGUAGEM DE PROGRAMAÇÃO ORIENTADA À OBJETOS – LPOO

## Aula 03 – Creating Java Programs

Profa. Thais Rocha- [thais.rocha@docente.unip.br](mailto:thais.rocha@docente.unip.br)



# AULA ANTERIOR - CONCEITOS IMPORTANTES

## Qual a diferença entre JDK, JRE e JVM ?

Uma grande confusão é gerada sobre quem está começando a aprender sobre o mundo Java é **a diferença entre JDK, JRE e JVM.**

Java é muito conhecido por trazer o conceito de **multi-plataforma**. Na verdade esse é o motivo do grande sucesso do Java à mais de vinte anos! O famoso **WORA** (*Write once, run anywhere.*), "Escreva uma vez, execute em qualquer lugar".

## AULA ANTERIOR - CONCEITOS IMPORTANTES (cont.)

**O fluxo é basicamente o seguinte:**

- Você escreve o seu código-fonte (extensão .java).
- Você utiliza o **JDK** para compilar o seu código-fonte e gerar o arquivo *bytecode* (arquivo com a extensão .class).
- Para executar o seu programa, a **JVM** lê o seu arquivo compilado (.class) e as bibliotecas padrões do Java que estão no **JRE**.

Pronto, seu programa está rodando e todo mundo está feliz! :)

## AULA ANTERIOR - CONCEITOS IMPORTANTES (cont.)


Então, de maneira resumida, já deu pra perceber pra quê serve cada um:

- **JDK (Java Development Kit)** - é o Kit de Desenvolvimento Java responsável por compilar código-fonte (.java) em bytecode (.class)
- **JVM (Java Virtual Machine)** - é a Máquina Virtual do Java responsável por executar o bytecode (.class)
- **JRE (Java Runtime Environment)** - Ambiente de Execução do Java que fornece as bibliotecas padrões do Java para o JDK compilar o seu código e para a JVM executar o seu programa.

## AULA ANTERIOR - INTRODUÇÃO A ORIENTAÇÃO A OBJETO

- O principal objetivo da programação orientada a objetos é **facilitar a manutenção das aplicações**.
- Orientação a objetos é uma maneira de programar que ajuda na organização e resolve muitos problemas enfrentados pela programação procedural.

# AULA ANTERIOR - OVERVIEW



**Classes  
(Objetos)**

**Atributos**

**Métodos**

# TÓPICOS DA AULA

- Escrevendo um programa Java (Console Output)
- Interpretar e corrigir erros comuns
- Convenções, comentários e padronização em Java
- Escrevendo um programa Java (GUI Output)

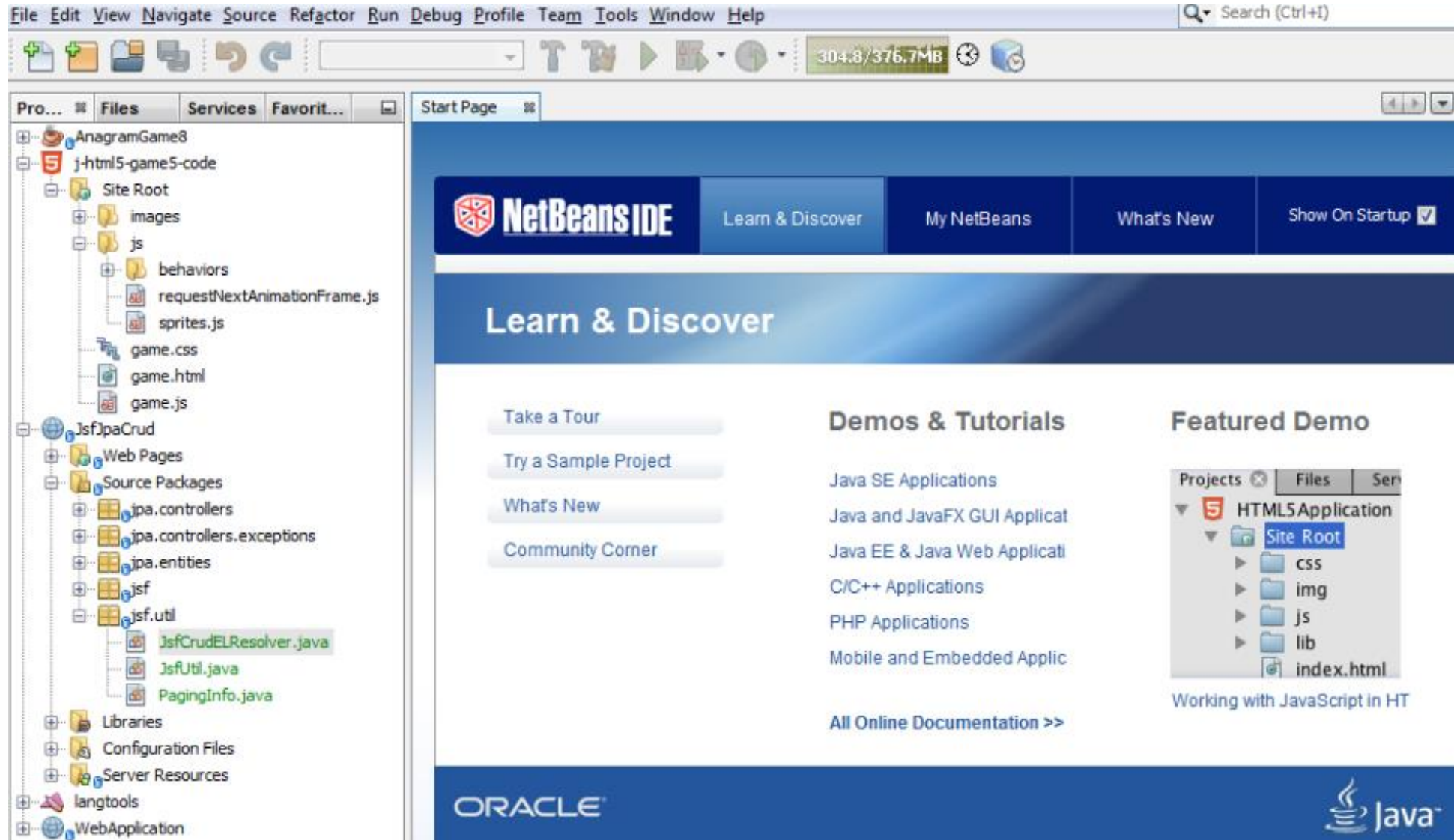
*Livro Java Programming – Joyce Farrell (Chapter 1)*

# JAVA - OVERVIEW

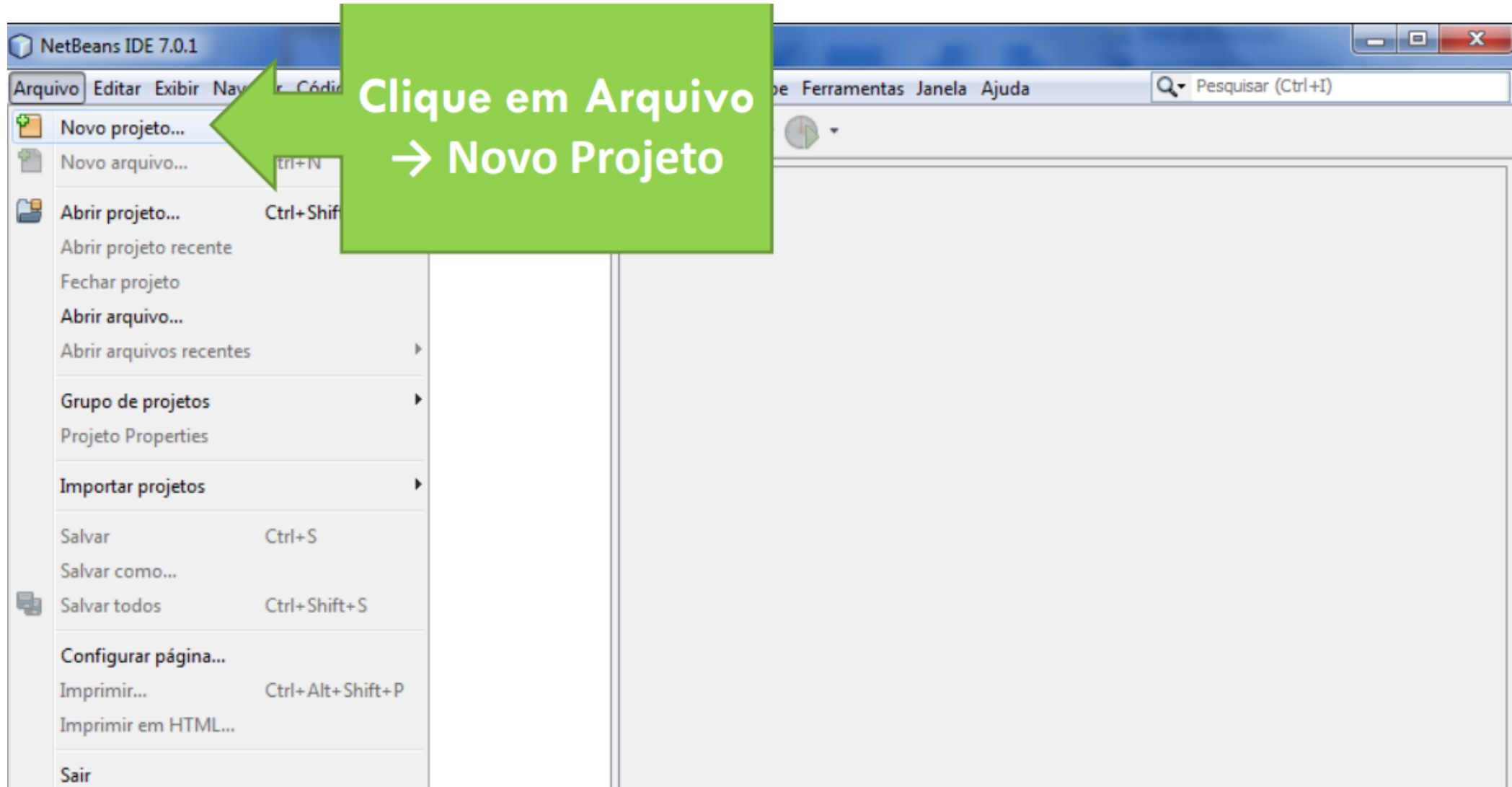
- Cada bloco de instrução é colocado entre chaves { e }.
- Cada linha de comando deve terminar com um ponto-e-vírgula (;).
- Letras maiúsculas e minúsculas NÃO são iguais.
- Para criar os algoritmos utilizando Java, vamos precisar de uma ferramenta chamada Ambiente de Desenvolvimento Integrada (IDE – Integrated Development Environment)



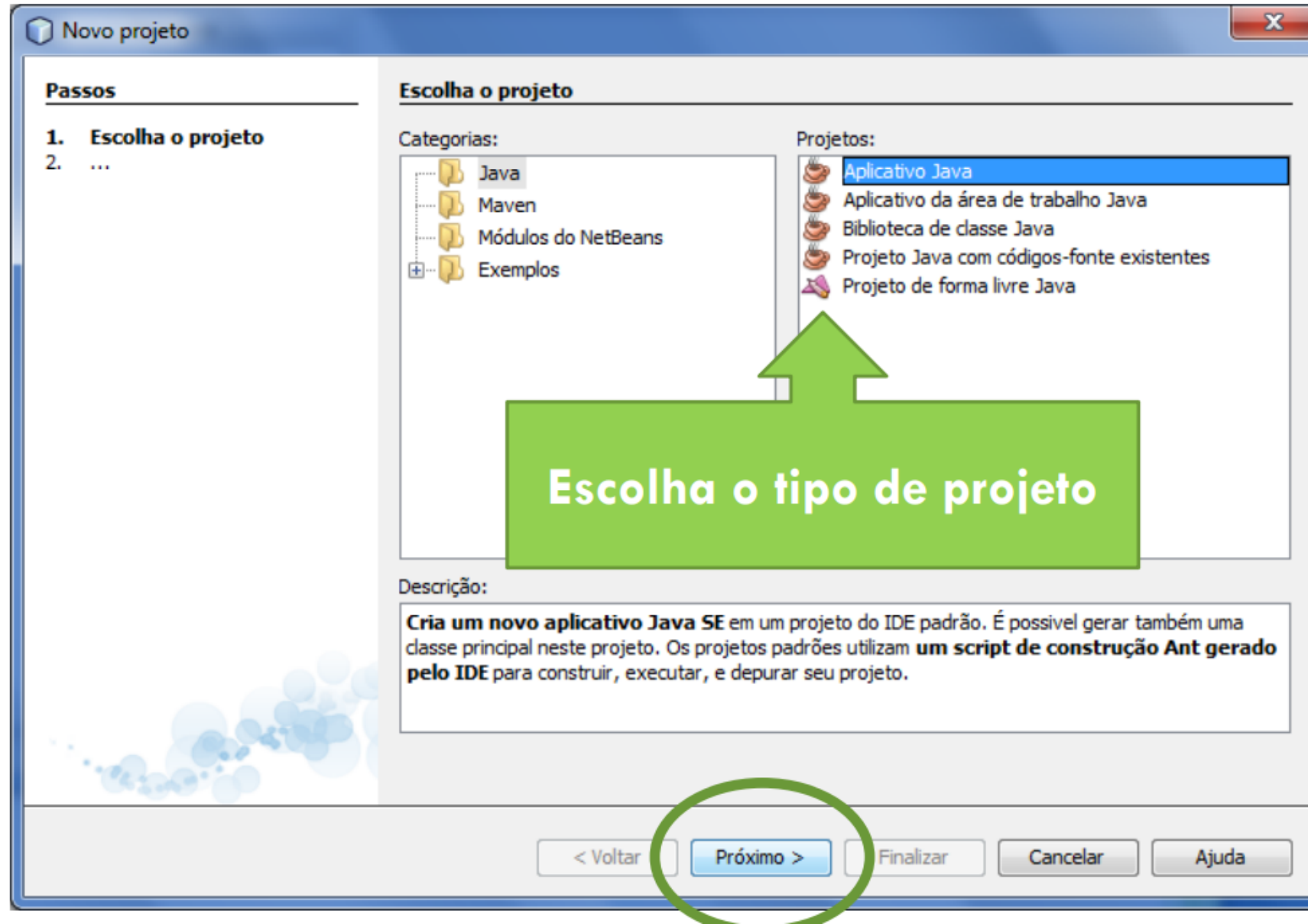
# UTILIZANDO O NETBEANS



# CRIANDO UM PROJETO



# CRIANDO UM PROJETO



# CRIANDO UM PROJETO

**Novo Aplicativo Java**

**Passos**

1. Escolha o projeto
2. **Nome e local**

**Nome e local**

Nome do projeto:

Localização do Projeto:

Pasta do projeto:

☐ Usar Pasta

☐ Criar classe principal

☒ Definir como projeto principal

**Escolha o nome e a localização do projeto**

# TIPOS DE PROGRAMAS EM JAVA

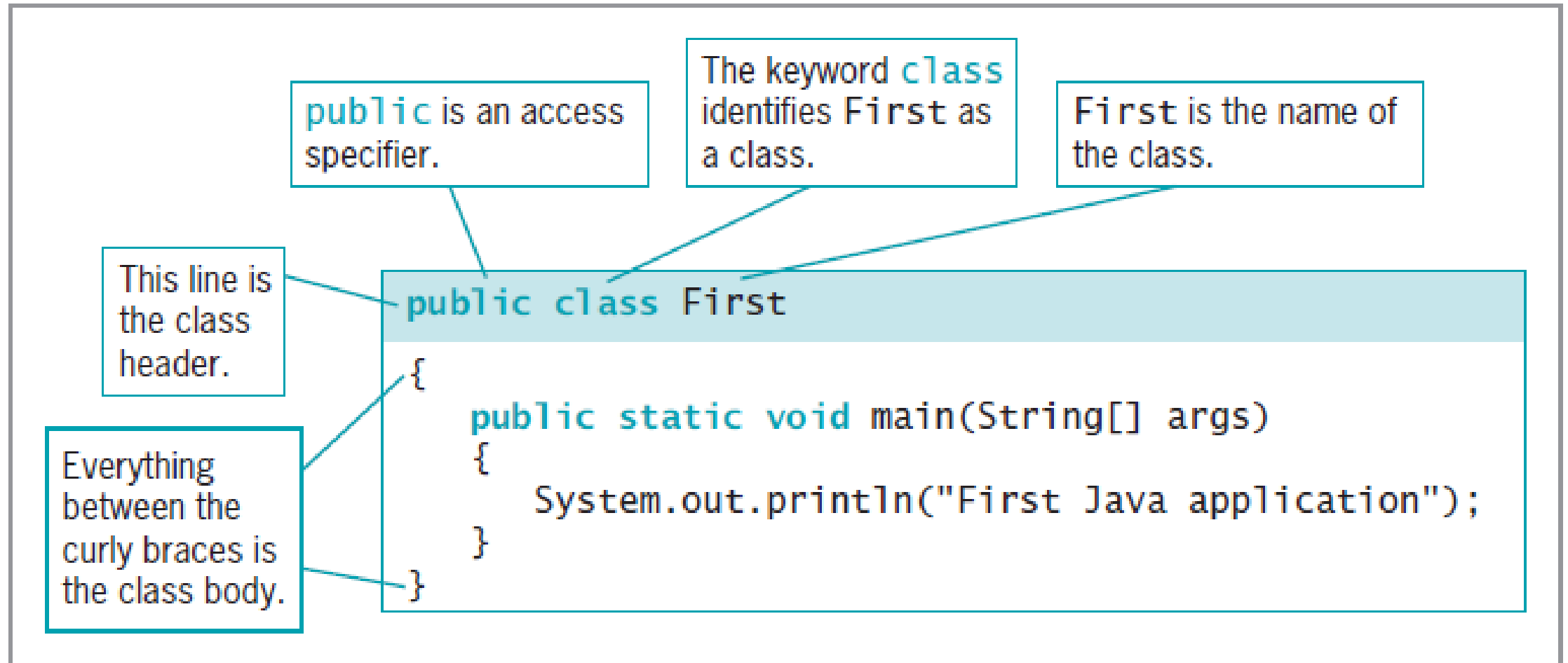
- Os aplicativos Java podem ser subdivididos 3 grupos:
  - **Aplicativos de console**, que suportam saída de caracteres ou texto para uma tela de computador.
  - **Aplicativos em janela (interface gráfica)**, que criam uma GUI com elementos como menus, barras de ferramentas, e caixas de diálogo (Java Swing)
  - **Aplicativos WEB**

# JAVA APPLICATION - **CONSOLE OUTPUT**

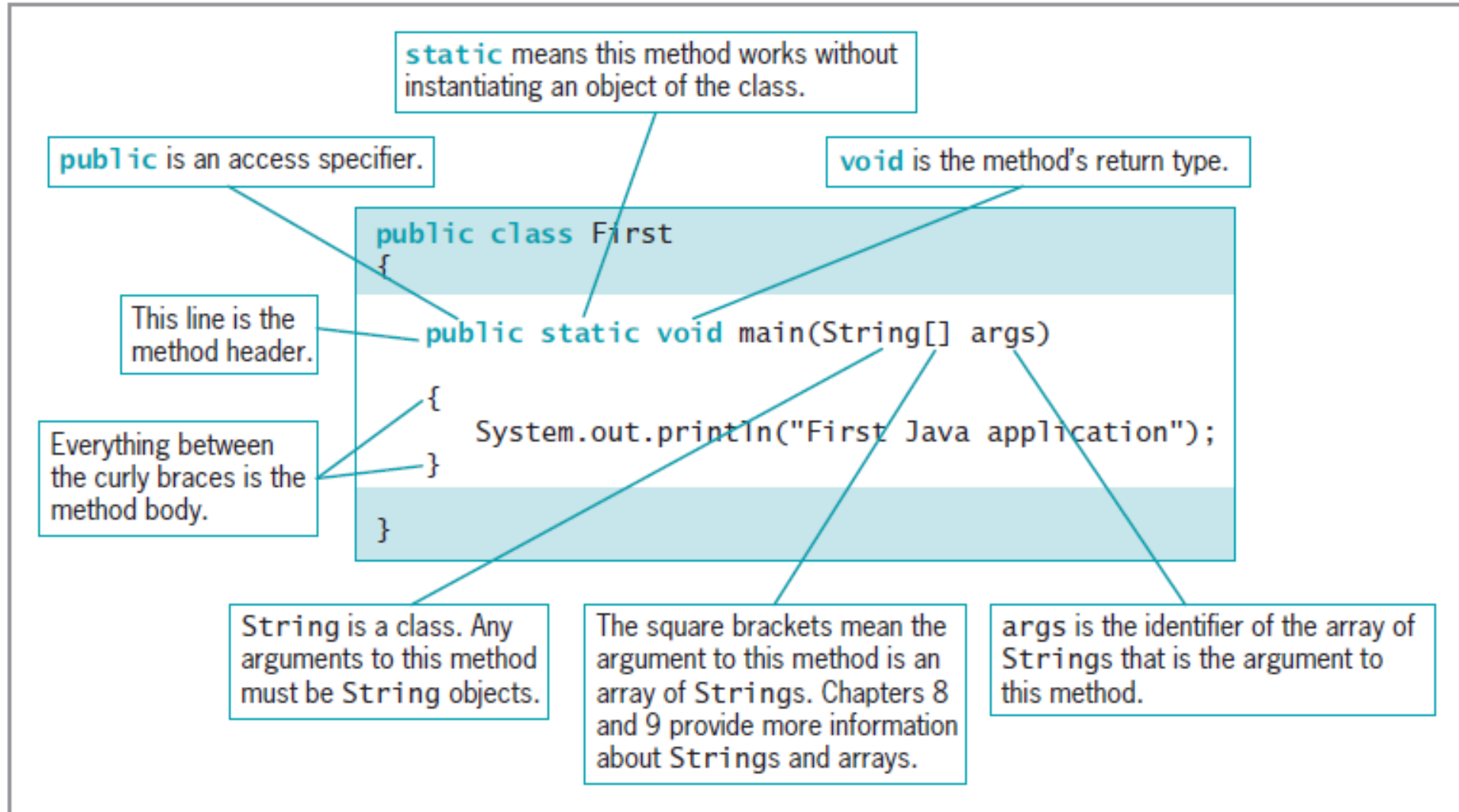
- Vamos agora analisar com detalhe, uma classe Java que produz saída do console

```
public class First
{
    public static void main(String[] args)
    {
        System.out.println("First Java application");
    }
}
```

# ENTENDENDO O CÓDIGO-FONTE – VISÃO GERAL



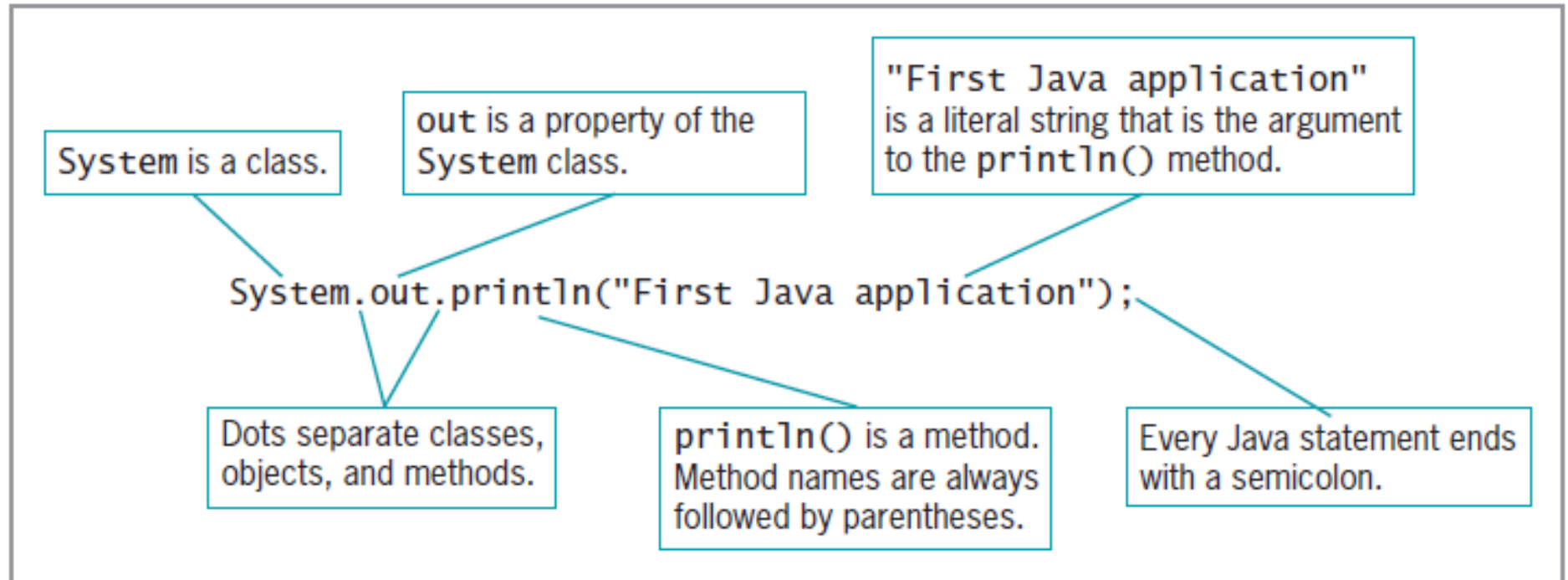
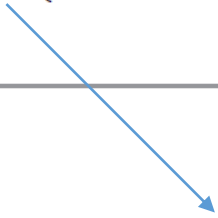
# ENTENDENDO O CÓDIGO-FONTE – Método `main()`





# ENTENDENDO O CÓDIGO-FONTE – Imprimindo valores no console

```
public class First
{
    public static void main(String[] args)
    {
        System.out.println("First Java application");
    }
}
```



# YOU DO IT !

## *Your First Application*

*Agora que você entende os fundamentos de um programa escrito em Java, você está pronto para insira seu próprio aplicativo Java.*

*É uma tradição entre os programadores que o primeiro programa que você escreve em qualquer linguagem produz “Hello, world!” como seu saída.*

*Você criará tal programa agora. Você pode usar qualquer IDE.*

*Salve o aplicativo como **Hello.java**. O nome da classe e o nome do arquivo **devem corresponder exatamente**, e você deve usar a extensão .java.*

# BOAS PRÁTICAS NA PROGRAMAÇÃO - CONVENÇÕES JAVA

- É um padrão Java, embora não seja um requisito, começar identificadores de classe com **letras maiúsculas** e use outras letras maiúsculas conforme necessário para **melhorar a legibilidade**.

**Você deve seguir as convenções estabelecidas para Java para seus programas sejam mais fáceis de serem interpretados e entendidos por outros programadores.**

## Classes:

- Nomes de classes devem começar com letra maiúscula.
- Use substantivos para nomear classes.
- Se o nome da classe consistir de várias palavras, utilize a notação CamelCase, começando cada palavra subsequente com letra maiúscula (por exemplo, MyClass, ShoppingCart, CustomerOrder).

## Métodos:

- Nomes de métodos devem começar com letra minúscula.
- Use verbos ou frases verbais para nomear métodos.
- Se o nome do método consistir de várias palavras, utilize a notação camelCase, começando a primeira palavra com letra minúscula e as subsequentes com letra maiúscula (por exemplo, `calculatePrice()`, `displayTotal()`, `getOrderDetails()`).

# COMENTÁRIOS EM JAVA – IMPORTANTE

- Existem 3 tipos de comentários em Java:  
*Linha, blocos e Javadoc*
- **Line comments** start with two forward slashes ( // ) and continue to the end of the current line. A line comment can appear on a line by itself or at the end (and to the right) of a line following executable code. Line comments do not require an ending symbol.
  - **Block comments** start with a forward slash and an asterisk ( /\* ) and end with an asterisk and a forward slash ( \*/ ). A block comment can appear on a line by itself, on a line before executable code, or on a line after executable code. Block comments also can extend across as many lines as needed.
- **Javadoc** comments are a special case of block comments called **documentation comments** because they are used to automatically generate nicely formatted program documentation with a program named javadoc. Javadoc comments begin with a forward slash and two asterisks ( /\*\* ) and end with an asterisk and a forward slash ( \*/ ). Appendix E teaches you how to create javadoc comments.

## COMENTÁRIOS EM JAVA – EXEMPLOS

```
// Demonstrating comments
/* This shows
   that these comments
   don't matter */
System.out.println("Hello"); // This line executes
// up to where the comment started
/* Everything but the println()
   is a comment */
```

# JAVA APPLICATION - GUI OUTPUT

- Além de permitir que você use a classe `System` para produzir saída de janela de comando, Java fornece classes integradas que produzem saída GUI.
- Java contém uma classe chamada **JOptionPane** que permite a você produzir **caixas de diálogo**.
- Uma caixa de diálogo é um objeto GUI semelhante a uma janela na qual você pode colocar as mensagens que deseja exibir.



# JOptionPane – showMessageDialog ()

```
import javax.swing.JOptionPane;

public class FirstDialog {

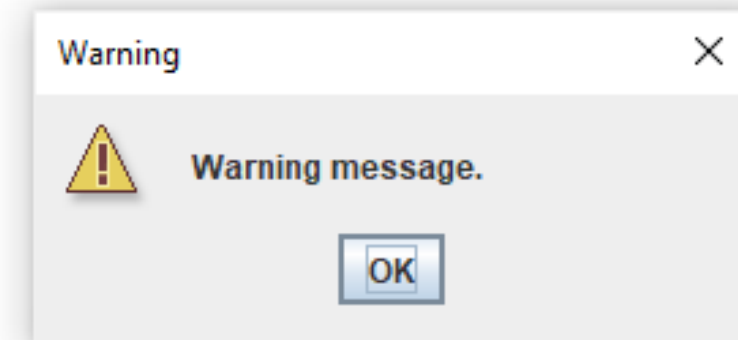
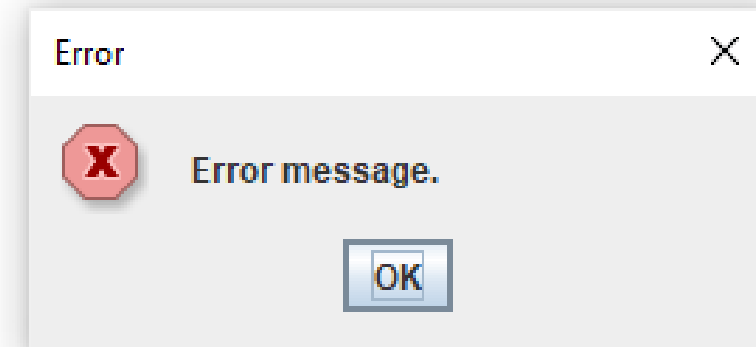
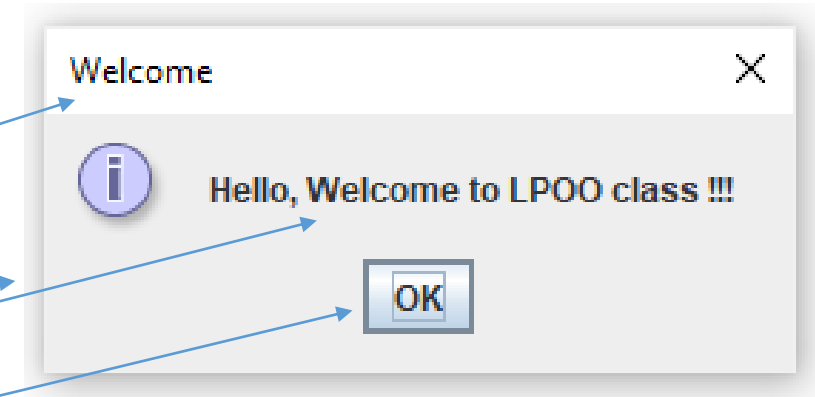
    public static void main(String[] args) {

        JOptionPane.showMessageDialog(null,
            "Hello, Welcome to LPOO class !!! ",
            "Welcome", JOptionPane.INFORMATION_MESSAGE);

        JOptionPane.showMessageDialog(null, "Error message.",
            "Error", JOptionPane.ERROR_MESSAGE);

        JOptionPane.showMessageDialog(null, "Warning message.",
            "Warning", JOptionPane.WARNING_MESSAGE);

    }
}
```



# JOptionPane – showOptionDialog ()

```
import javax.swing.JOptionPane;

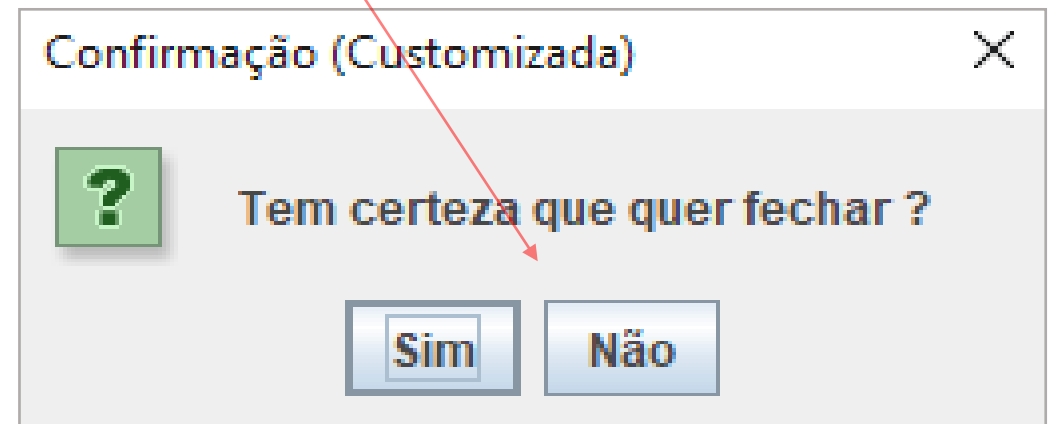
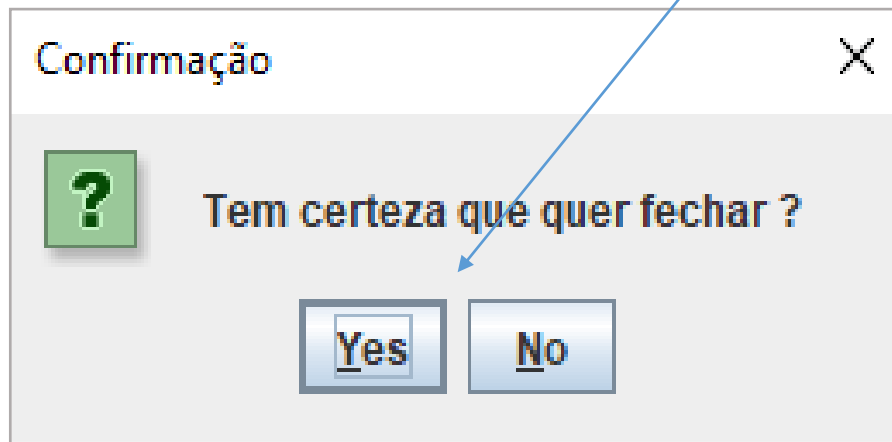
public class YesNoDialog {

    public static void main(String[] args) {

        int valor = JOptionPane.showConfirmDialog(null, "Tem certeza que quer fechar ?",
            "Confirmação", JOptionPane.YES_NO_OPTION);

        Object[] options = { "Sim", "Não" };
        int valorConf = JOptionPane.showOptionDialog(null, "Tem certeza que quer fechar ?",
            "Confirmação (Customizada)", JOptionPane.YES_NO_OPTION,
            JOptionPane.QUESTION_MESSAGE, null, options, options[0]);

    }
}
```



## JOptionPane – **showInputDialog ()**

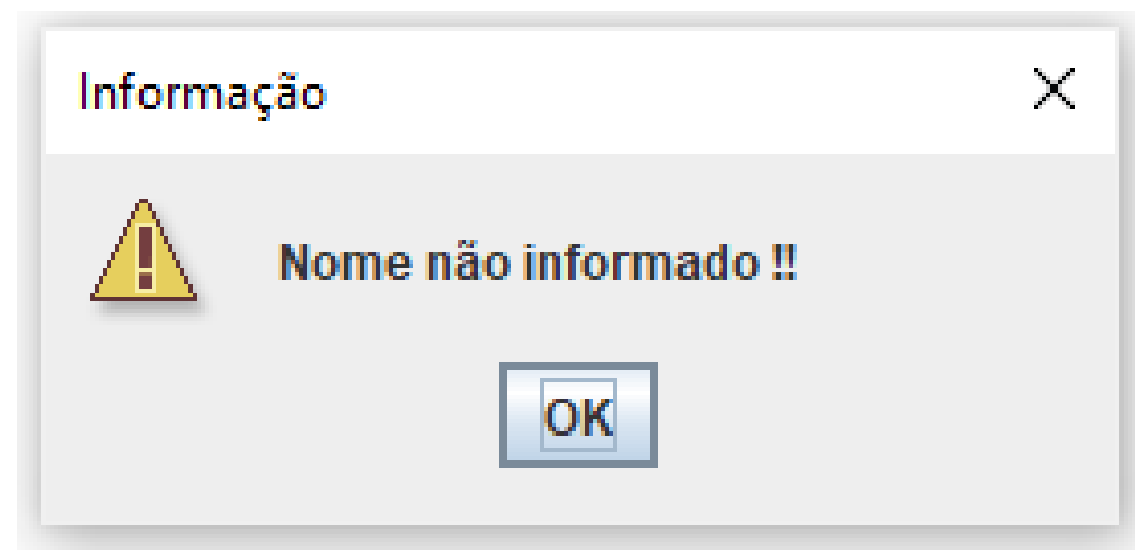
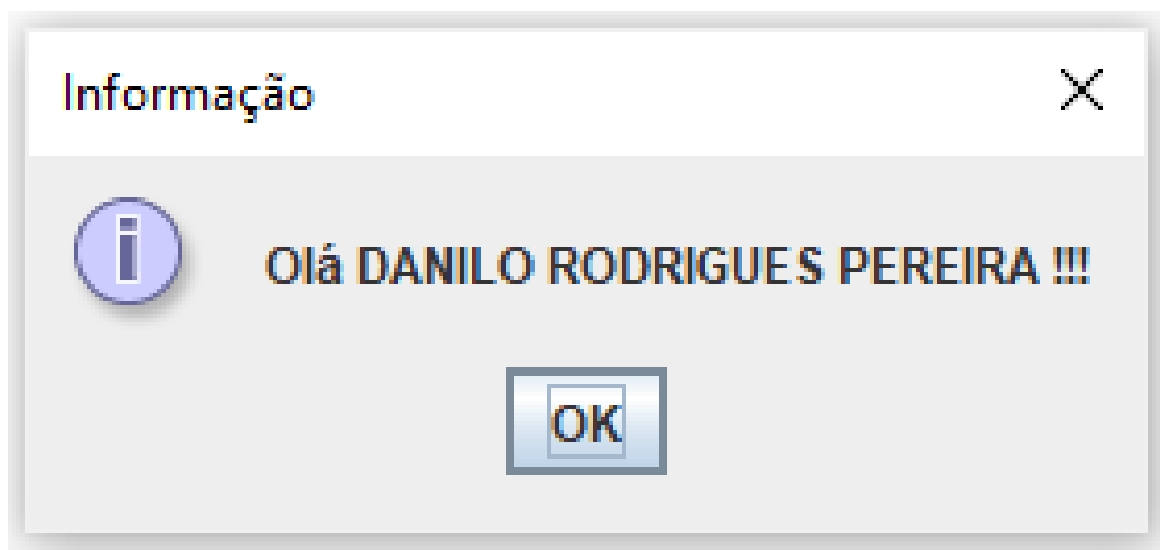
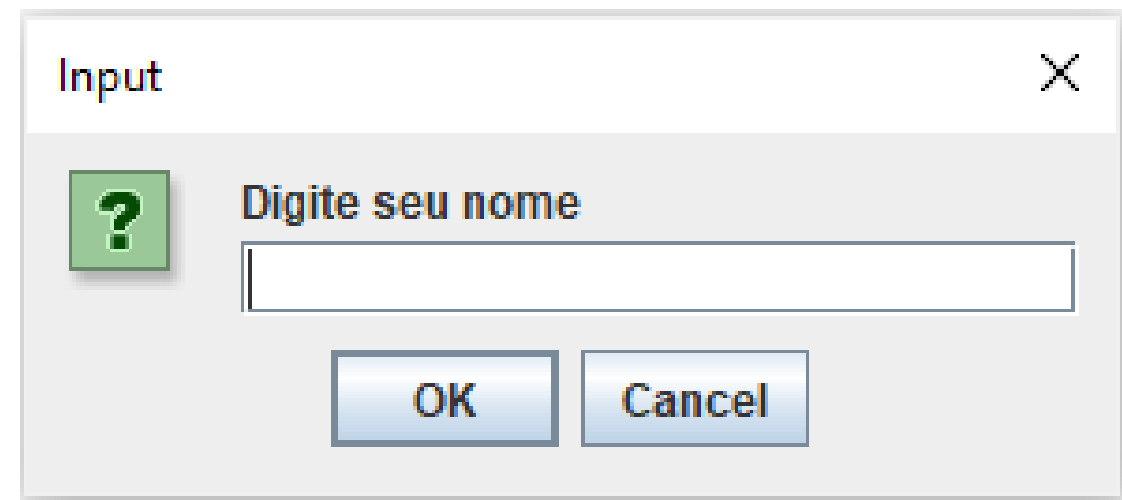
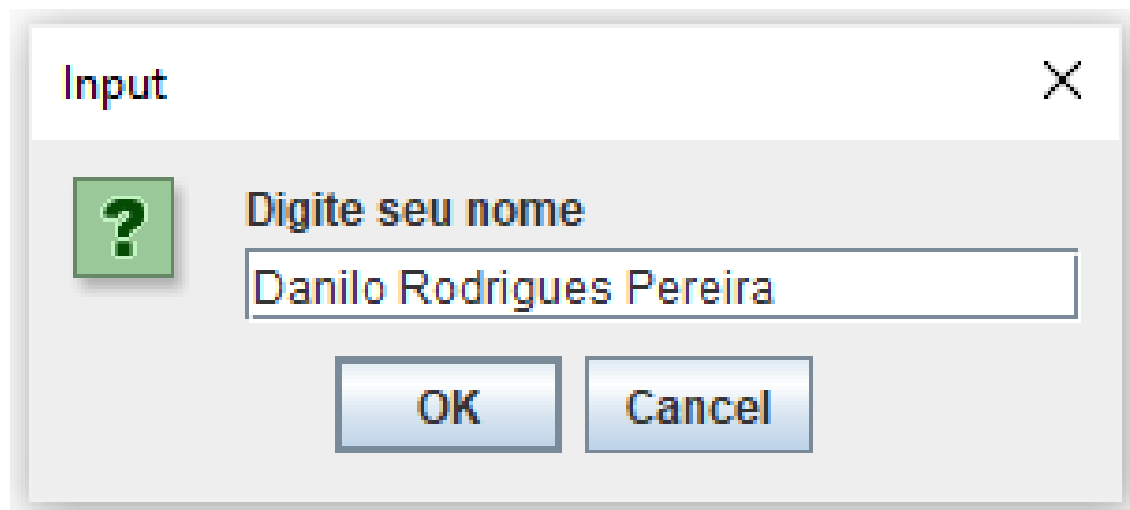
```
import javax.swing.JOptionPane;

public class InputDialog {

    public static void main(String[] args) {
        String nome = JOptionPane.showInputDialog("Digite seu nome");

        if (nome != null && !nome.isBlank()) {
            JOptionPane.showMessageDialog(null, "Olá " + nome.toUpperCase() + " !!! ",
                "Informação", JOptionPane.INFORMATION_MESSAGE);
        } else {
            JOptionPane.showMessageDialog(null, "Nome não informado !!",
                "Informação", JOptionPane.WARNING_MESSAGE);
        }
    }
}
```

# JOptionPane – **showInputDialog ()**



# EXERCÍCIOS NO PROMPT

**javac -> compila e gera o arquivo .class**

**java [nome da classe “sem o .class”] -> executa a classe**

```
c:\java\aula3>javac FavoriteSongConsole.java

c:\java\aula3>java FavoriteSongConsole
Título: Bohemian Rhapsody
Letra:
Is this the real life?
Is this just fantasy?
Caught in a landslide,
No escape from reality
Open your eyes,
Look up to the skies and see
Autor: Queen
Ano: 1975
```

# EXERCÍCIOS DE FIXAÇÃO

## I) Exercício I: Nomes de Classes

Dado o seguinte conjunto de nomes de classes, marque aqueles que seguem a convenção Java:

- a) CarroVermelho ( )
- b) Cliente\_de\_rede ( )
- c) RetornaDetalhes ( )
- d) CalculadoraFinanceira ( )
- e) Arquivo\_txt ( )
- f) JogadorDeFutebol ( )
- g) produto\_eletrônico ( )
- h) Cadastro\_Clientes ( )

# EXERCÍCIOS DE FIXAÇÃO

## 2) Nomes de Métodos

Dado o seguinte conjunto de nomes de métodos, identifique aqueles que seguem a convenção Java:

- a) calcular\_total ( )
- b) RetornaDadosCliente ( )
- c) obterDetalhes ( )
- d) MostraOpcoes ( )
- e) calcularMedia ( )
- f) GetNomeCompleto ( )
- g) validar\_entrada ( )
- h) listarItensEstoque ( )
- i) BuscarUsuario ( )
- j) calcularImpostoDeRenda ( )

# EXERCÍCIOS DE FIXAÇÃO

**3)** Escreva, compile e teste a classe que exibe no console da IDE (Eclipse ou NetBeans) as seguintes informações (uma informação em cada linha)

- Título de uma música favorita
- Pelo menos 5 linhas desse musica
- Autor
- Ano

Salve a classe com o seguinte nome: **FavoriteSongConsole.java**

**4)** Escreva, compile e teste a classe que existe uma janela (utilize a classe JoptionPane) as seguintes informações (uma informação em cada linha)

- Título de uma música favorita
- Pelo menos 5 linhas desse musica
- Autor
- Ano

Salve a classe com o seguinte nome: **FavoriteSongWindow.java**



# EXERCÍCIOS DE FIXAÇÃO

5) Escreva, compile e teste uma classe que apresenta o seguinte padrão na tela.  
Salve a classe como as **TableAndChairs.java**.

```

X                               X
X                               X
X      XXXXXXXXXXXX           X
XXXXX  X                X  XXXXX
X   X  X                X  X   X
X   X  X                X  X   X

```

6) Escreva, compile e teste uma classe que use a janela de comando para exibir a seguinte declaração sobre comentários:

**"Comentários de programa são instruções não executáveis que você adiciona a um arquivo para fins de documentação."**

Inclua também a mesma declaração em três comentários diferentes na classe; cada comentário deve usar um dos três métodos diferentes de inclusão de comentários em uma classe Java

Salve a classe como **Comments.java**.

# EXERCÍCIOS PARA ENTREGAR

**7)** De 1925 a 1963, os cartazes publicitários da Burma Shave apareceram junto às auto-estradas de todos os Estados Unidos. Havia sempre quatro ou cinco placas seguidas contendo partes de uma rima, seguidas de uma placa final que dizia "Burma Shave".

Por exemplo, um conjunto de sinais que foi preservado pelo Smithsonian Institution diz o seguinte:

Pincéis de barbear  
Em breve os verás  
Numa prateleira  
Nalgum museu

Burma Shave

Escreva, compile e teste uma classe que produza uma série de quatro caixas de diálogo para que cada uma exiba uma linha do slogan de Burma Shave. Guarde a classe como **BurmaShave.java**.

# PRÓXIMA AULA

- *Aprender a declarar e usar variáveis e constantes*
- *Data types: Integer, Double, Float, Char e String*
- *Classe Math e Operações aritméticas*
- *Entrada de dados através da classe Scanner e JOptionPane*



