

# POO e Herança em Java

## Projeto: Sistema de Folha de Pagamento

Prof. Pedro A.B. Oliveira

FATEC Santana de Parnaíba  
Curso: Análise e Desenvolvimento de Sistemas  
Disciplina: Linguagem de Programação III - Desenvolver aplicações em JAVA

Aula 02 : 19 de Fevereiro de 2026

# Revisão: Arquitetura MVC

Para organizar nosso sistema, continuamos usando o padrão **Model-View-Controller**:

## 1. Model (O Cérebro)

Onde ficam as regras de negócio e os dados.

- Ex: Cálculos de INSS, Salário Líquido.
- *No Java*: Classes Funcionario, Gerente.

## 2. View (A Cara)

A interface com o usuário.

- *No Java*: JFrame, Botões, Caixas de Texto.

## 3. Controller (O Maestro)

Quem liga a tela ao cérebro.

- *No Java*: O método actionPerformed do botão.

# Modelagem: A Importância das Regras

Antes de codificar, precisamos entender o problema.

## Cenário da Folha de Pagamento:

- 1 Todo funcionário tem **Nome**, **CPF** e **Salário Base**.
- 2 Todo funcionário sofre desconto de **INSS** (vamos fixar em 11% para este exemplo).
- 3 O **Salário Líquido** é:  $\text{Base} - \text{INSS}$ .
- 4 **Porém...** existem **Funcionários** e existem **Gerentes**.
- 5 O Gerente tem um **Bônus** exclusivo que soma ao salário.

# Conceitos Fundamentais de POO

## Classe (O Molde)

- É a definição abstrata.
- Define "o que o objeto vai ter".
- Ex: `Arquivo Funcionario.java`

## Instância (O Objeto)

- É o objeto concreto na memória.
- Criado com a palavra `new`.
- Ex: O funcionário "João" criado a partir do molde.

*"A planta da casa (Classe) não é a casa (Objeto)."*

# Encapsulamento: Protegendo os Dados

Como controlar o acesso às variáveis da classe?

## private (Privado)

Só a própria classe enxerga.

- Usamos para **Atributos** (variáveis).
- Evita que alguém faça `func.salario = -5000;`

## public (Público)

Todo o projeto enxerga.

- Usamos para **Métodos** (ações) e Construtores.
- Ex: `calcularINSS()` deve ser público para a tela usar.

**Solução:** Usamos Getters e Setters públicos para acessar variáveis privadas.

# Herança: O Conceito de "É Um"

E se tivermos um **Gerente**?

- O Gerente **também tem** Nome, CPF e Salário.
- Não vamos copiar o código! Vamos **Herdar**.

## Sintaxe Java:

```
private double bonus; // S adiciona o que novo
```

Isso facilita a manutenção. Se mudarmos a regra do CPF no pai, o filho atualiza automaticamente.

# Desafio Prático de Hoje

Vamos implementar no NetBeans 8.2:

- ❶ **Classe Mãe:** Funcionario (Cálculo Básico).
- ❷ **Classe Filha:** Gerente (Cálculo com Bônus).
- ❸ **Tela:** Um formulário para calcular o salário líquido.

**Vamos ao Código!**