# 📘 Aula 25/11 UI (SWING)

## 🧠 Introdução: O que é Swing?

**Swing** é uma biblioteca gráfica incluída no Java (está dentro do pacote javax.swing) que permite criar interfaces gráficas simples:

- Janelas (JFrame)
- Botões (JButton)
- Textos (JLabel)
- Caixas de Input (JTextField)
- Pop-ups rápidos (JOptionPane)

### ✔️ Não é preciso instalar nada.

O Swing já vem incluído no **JDK**.

Se já tens Java 17 instalado → Swing funciona automaticamente.

```
import javax.swing.*;

public class App {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Janela VS Code");
        frame.setSize(300,150);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

---

## 🪟 Criar a primeira Janela com Botão

```
import javax.swing.*;

public class JanelaSimples {
    public static void main(String[] args) {

        JFrame frame = new JFrame("Primeira Janela");

        JLabel label = new JLabel("Olá Swing!");
        JButton button = new JButton("Clica em mim");

        button.addActionListener(e ->
            JOptionPane.showMessageDialog(frame, "Botão clicado!")
        );

        JPanel panel = new JPanel();
        panel.add(label);
        panel.add(button);

        frame.setContentPane(panel);
        frame.setSize(300, 150);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

---

# 🧱 Conceitos Fundamentais de Swing

## ✔️ JFrame

A janela principal da aplicação.

## ✔️ JPanel

Um “contentor” onde colocamos botões, texto, inputs.

### ✔️ JButton

Um botão clicável.

### ✔️ JLabel

Um texto.

### ✔️ JTextField

Caixa onde o utilizador escreve.

### ✔️ JOptionPane

Pop-ups de mensagens e inputs.

### ✔️ ActionListener

Um evento que corre quando o utilizador clica num botão.

---

# 🔗 Ligar Swing à Lógica do Programa

Vamos construir:

- uma classe Animal
- um gestor GestorAnimais
- uma UI Swing GestorAnimaisUI

Tudo interligado.

## 🐲 Class Animal

```
public class Animal {
    private String nome;
    private String tipo;
```

```
    // Construtor com validação
    public Animal(String nome, String tipo) {
        // Se nome for inválido → usa valor padrão
        this.nome = (nome == null || nome.isBlank()) ? "Sem nome" : nome;

        // Se tipo for inválido → usa valor padrão
        this.tipo = (tipo == null || tipo.isBlank()) ? "Desconhecido" : tipo;
    }

    public String getNome() {
        return nome;  // devolve nome
    }

    public String getTipo() {
        return tipo;  // devolve tipo
    }

    // Representação em texto
    @Override
    public String toString() {
        return nome + " (" + tipo + ")";
    }
}
```

## 🐒Class GestorAnimais

```
import java.util.ArrayList;
import java.util.List;

public class GestorAnimais {

    // Lista onde guardamos os animais criados
```

```
    private List<Animal> animais = new ArrayList<>();

    // Adiciona um animal à lista
    public void adicionarAnimal(String nome, String tipo) {
        animais.add(new Animal(nome, tipo));
    }

    // Remove um animal pelo nome (retorna true/false)
    public boolean removerAnimal(String nome) {
        return animais.removeIf(a -> a.getNome().equalsIgnoreCase(nome));
    }

    // Devolve todos os animais como texto
    public String listarAnimais() {
        if (animais.isEmpty())
            return "Nenhum animal registado.";

        StringBuilder sb = new StringBuilder();
        for (Animal a : animais) {
            sb.append(a).append("\n"); // usa o método toString()
        }
        return sb.toString();
    }
}
```

## 🖥️ Interface Gráfica (Swing)

```
import javax.swing.*;

public class GestorAnimaisUI {
    public static void main(String[] args) {

        // Instância do gestor (a lógica da aplicação)
```

```
GestorAnimais gestor = new GestorAnimais();

// Criação da janela
JFrame frame = new JFrame("Gestor de Animais");

// Campo e label do nome
JLabel labelNome = new JLabel("Nome:");
JTextField fieldNome = new JTextField(10);

// Campo e label do tipo
JLabel labelTipo = new JLabel("Tipo:");
JTextField fieldTipo = new JTextField(10);

// Botões
JButton btnAdicionar = new JButton("Adicionar");
JButton btnListar = new JButton("Listar");
JButton btnRemover = new JButton("Remover");

// ------------------------------------------------------------
// Evento: adicionar animal
// ------------------------------------------------------------
btnAdicionar.addActionListener(e -> {
    String nome = fieldNome.getText();  // lê texto da caixa
    String tipo = fieldTipo.getText();

    gestor.adicionarAnimal(nome, tipo); // chama lógica

    JOptionPane.showMessageDialog(frame, "Animal registado!");

    // Limpar caixas após adicionar
    fieldNome.setText("");
    fieldTipo.setText("");
});

// ------------------------------------------------------------
// Evento: listar animais
```

```
        // ------------------------------------------------------------
        btnListar.addActionListener(e -> {
            String lista = gestor.listarAnimais();  // resultado da lógica
            JOptionPane.showMessageDialog(frame, lista);
        });

        // ------------------------------------------------------------
        // Evento: remover animal
        // ------------------------------------------------------------
        btnRemover.addActionListener(e -> {
            String nome = JOptionPane.showInputDialog(frame, "Nome a remove
r:");
            boolean removido = gestor.removerAnimal(nome);

            if (removido)
                JOptionPane.showMessageDialog(frame, "Animal removido!");
            else
                JOptionPane.showMessageDialog(frame, "Animal não encontrad
o.");
        });

        // ------------------------------------------------------------
        // Criar painel e adicionar componentes
        // ------------------------------------------------------------
        JPanel panel = new JPanel();
        panel.add(labelNome);
        panel.add(fieldNome);
        panel.add(labelTipo);
        panel.add(fieldTipo);
        panel.add(btnAdicionar);
        panel.add(btnListar);
        panel.add(btnRemover);

        // Colocar painel na janela
        frame.setContentPane(panel);
```

```
        // Tamanho da janela
        frame.setSize(450, 180);

        // Quando clicar no X → fecha programa
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Mostrar no ecrã
        frame.setVisible(true);
    }
}
```

---

## 🦸 Mini Simulador de Herói

Este exemplo mostra:

- Uma **janela com informação do herói** (nome, nível, XP, energia)
- 3 ações: **Treinar**, **Descansar**, **Missão**
- Regras:
  - Treinar → gasta energia, dá XP
  - Descansar → recupera energia
  - Missão → gasta mais energia, dá mais XP, mas só se tiver energia suficiente
  - Ao chegar a 100 XP → sobe de nível, XP volta a 0
  - Energia nunca passa de 100, nem vai abaixo de 0

---

## 🦸 Class Heroi

```
class Heroi {
    private String nome;
    private int nivel;
```

```
    private int xp;
    private int energia;

    public Heroi(String nome) {
        this.nome = nome;
        this.nivel = 1;
        this.xp = 0;
        this.energia = 100;
    }

    public String getNome() { return nome; }
    public int getNivel() { return nivel; }
    public int getXp() { return xp; }
    public int getEnergia() { return energia; }

    public String treinar() {
        if (energia < 10) {
            return "Energia insuficiente para treinar!";
        }
        energia -= 10;
        ganharXp(15);
        return "Treino concluído! Ganhou 15 XP e gastou 10 de energia.";
    }

    public String descansar() {
        if (energia >= 100) {
            return "O herói já está com energia máxima!";
        }
        energia += 20;
        if (energia > 100) energia = 100;
        return "Descanso feito! Recuperou 20 de energia.";
    }

    public String fazerMissao() {
        if (energia < 40) {
            return "Energia insuficiente para ir numa missão!";
```

```
        }
        energia -= 40;
        ganharXp(40);
        return "Missão concluída! Ganhou 40 XP e gastou 40 de energia.";
    }

    private void ganharXp(int quantidade) {
        xp += quantidade;
        while (xp >= 100) {
            xp -= 100;
            nivel++;
        }
    }
}
```

## 👻 Class SimuladorHeroi

```
import javax.swing.*;
import java.awt.*;

public class SimuladorHeroi { // Helper: carregar e redimensionar imagem par
a um tamanho fixo
    private static ImageIcon loadAndScale(String path, int width, int height) {
        ImageIcon icon = new ImageIcon(path);
        Image img = icon.getImage();
        Image scaled = img.getScaledInstance(width, height, Image.SCALE_SMO
OTH);
        return new ImageIcon(scaled);
    }

    public static void main(String[] args) {

        // Instância da lógica
```

```
        Heroi heroi = new Heroi("Nova Estrela");

        // ------------------------
        // CARREGAR IMAGENS DO HERÓI
        // ------------------------
        int HERO_WIDTH = 260;
        int HERO_HEIGHT = 180;

        // Coloca as imagens numa pasta "img" ao lado do src
        ImageIcon iconNormal = loadAndScale("img/heroi_normal.png", HERO_W
IDTH, HERO_HEIGHT);
        ImageIcon iconCansado = loadAndScale("img/heroi_cansado.png", HER
O_WIDTH, HERO_HEIGHT);
        ImageIcon iconPower = loadAndScale("img/heroi_power.png", HERO_WI
DTH, HERO_HEIGHT);

        // ------------------------
        // JANELA PRINCIPAL
        // ------------------------
        JFrame frame = new JFrame("Simulador de Herói – com Imagens");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(650, 360);
        frame.setLayout(new BorderLayout());

        // Cor de fundo da janela (um cinza ligeiramente azulado)
        frame.getContentPane().setBackground(new Color(235, 240, 245));

        // ------------------------
        // TOPO – NOME DO HERÓI
        // ------------------------
        JLabel labelNome = new JLabel("Herói: " + heroi.getNome(), SwingConst
ants.CENTER);
        labelNome.setFont(new Font("Arial", Font.BOLD, 26));
        labelNome.setForeground(new Color(30, 30, 60));

        JPanel painelTopo = new JPanel();
```

```
        painelTopo.setBackground(new Color(215, 225, 245)); // barra superior c
olorida
        painelTopo.add(labelNome);

        // ------------------------
        // CENTRO ESQUERDO – IMAGEM
        // ------------------------
        JLabel labelImagem = new JLabel(iconNormal);
        labelImagem.setHorizontalAlignment(SwingConstants.CENTER);
        labelImagem.setVerticalAlignment(SwingConstants.CENTER);

        JPanel painelImagem = new JPanel(new BorderLayout());
        painelImagem.setBackground(new Color(245, 245, 245)); // fundo claro a
trás da imagem
        painelImagem.add(labelImagem, BorderLayout.CENTER);
        painelImagem.setPreferredSize(new Dimension(300, 220));

        // ------------------------
        // CENTRO DIREITO – ESTADO
        // ------------------------
        JLabel labelNivel = new JLabel();
        JLabel labelXp = new JLabel();
        JLabel labelEnergia = new JLabel();

        labelNivel.setFont(new Font("Arial", Font.PLAIN, 18));
        labelXp.setFont(new Font("Arial", Font.PLAIN, 18));
        labelEnergia.setFont(new Font("Arial", Font.BOLD, 18));

        JPanel painelEstado = new JPanel(new GridLayout(3, 1, 0, 10));
        painelEstado.setBorder(BorderFactory.createEmptyBorder(20, 10, 20, 1
0));
        painelEstado.setBackground(new Color(235, 240, 245));
        painelEstado.add(labelNivel);
        painelEstado.add(labelXp);
        painelEstado.add(labelEnergia);
```

```
        // Junta imagem + estado lado a lado
        JPanel painelCentro = new JPanel(new GridLayout(1, 2));
        painelCentro.setBackground(new Color(235, 240, 245));
        painelCentro.add(painelImagem);
        painelCentro.add(painelEstado);

        // ------------------------
        // BAIXO – MENSAGEM + BOTÕES
        // ------------------------
        JLabel labelMensagem = new JLabel("Pronto para começar o treino!", S
wingConstants.CENTER);
        labelMensagem.setFont(new Font("Arial", Font.ITALIC, 14));
        labelMensagem.setForeground(new Color(60, 60, 60));

        JButton btnTreinar = new JButton("Treinar");
        JButton btnDescansar = new JButton("Descansar");
        JButton btnMissao = new JButton("Missão");

        // Um pouco de estilo nos botões
        Color btnBg = new Color(230, 230, 240);
        btnTreinar.setBackground(btnBg);
        btnDescansar.setBackground(btnBg);
        btnMissao.setBackground(btnBg);

        JPanel painelBotoes = new JPanel(new FlowLayout());
        painelBotoes.setBackground(new Color(235, 240, 245));
        painelBotoes.add(btnTreinar);
        painelBotoes.add(btnDescansar);
        painelBotoes.add(btnMissao);

        JPanel painelInferior = new JPanel(new BorderLayout());
        painelInferior.setBackground(new Color(235, 240, 245));
        painelInferior.setBorder(BorderFactory.createEmptyBorder(5, 10, 10, 10));
        painelInferior.add(labelMensagem, BorderLayout.CENTER);
        painelInferior.add(painelBotoes, BorderLayout.SOUTH);
```

```
        // ------------------------
        // FUNÇÃO: atualizar estado visual
        // ------------------------
        Runnable atualizarEstadoVisual = () -> {
            labelNivel.setText("Nível: " + heroi.getNivel());
            labelXp.setText("XP: " + heroi.getXp() + " / 100");
            labelEnergia.setText("Energia: " + heroi.getEnergia() + " / 100");

            int energia = heroi.getEnergia();

            // Cor da energia
            if (energia > 60) {
                labelEnergia.setForeground(new Color(0, 150, 0)); // verde
            } else if (energia > 30) {
                labelEnergia.setForeground(new Color(210, 140, 0)); // laranja
            } else {
                labelEnergia.setForeground(new Color(200, 0, 0)); // vermelho
            }

            // Imagem do herói consoante a energia
            if (energia <= 30) {
                labelImagem.setIcon(iconCansado);
            } else if (energia >= 80) {
                labelImagem.setIcon(iconPower);
            } else {
                labelImagem.setIcon(iconNormal);
            }
        };

        // Atualizar pela primeira vez
        atualizarEstadoVisual.run();

        // ------------------------
        // EVENTOS DOS BOTÕES
        // ------------------------
        btnTreinar.addActionListener(e -> {
```

```
            String msg = heroi.treinar();
            labelMensagem.setText(msg);
            atualizarEstadoVisual.run();
        });

        btnDescansar.addActionListener(e -> {
            String msg = heroi.descansar();
            labelMensagem.setText(msg);
            atualizarEstadoVisual.run();
        });

        btnMissao.addActionListener(e -> {
            String msg = heroi.fazerMissao();
            labelMensagem.setText(msg);
            atualizarEstadoVisual.run();
        });

        // -------------------------
        // ADICIONAR TUDO À JANELA
        // -------------------------
        frame.add(painelTopo, BorderLayout.NORTH);
        frame.add(painelCentro, BorderLayout.CENTER);
        frame.add(painelInferior, BorderLayout.SOUTH);

        frame.setLocationRelativeTo(null); // centra a janela no ecrã
        frame.setVisible(true);
    }
}
```