

Exercício "A Casa Inteligente"

Contexto: A Casa Inteligente


Imagina que estás a programar uma **casa inteligente** 🏠

Cada dispositivo (como uma lâmpada, um interruptor, um termóstato) é um **objeto**, com **características** (atributos) e **ações** (métodos).

Exercício 1 – Criar a Classe "Lâmpada"

Objetivo: Compreender o conceito de **objeto e classe**.

1. Faz uma lista de 3 objetos do mundo real que existem numa casa inteligente.
(Exemplo: Lâmpada, Televisão, Termóstato...)
2. Para cada um, identifica:
 - 3 **atributos** (características que o definem)
 - 2 **métodos** (ações que ele pode fazer)

 *Exemplo incompleto:*

Objeto: Lâmpada

Atributos: ligada, cor, intensidade

Métodos: ligar(), ...

Exercício 2 – Criar a Classe "Lâmpada"

Objetivo: Escrever a primeira **classe** com atributos e métodos.

1. Cria um novo projeto no IntelliJ chamado CasaInteligente.
2. Cria um ficheiro Lampada.java.

3. Dentro da classe, declara 2 **atributos** que descrevam o estado de uma lâmpada.

(Por exemplo: está ligada? intensidade?)

4. Acrescenta 2 **métodos** que permitam alterar esses estados.

- Um método para "ligar"
- Outro para "desligar"

💡 *Dica:* pensa em como o método vai alterar o valor dos atributos.

📋 **Objetivo de aprendizagem:** perceber que uma classe define o "molde" de um objeto.

💡 Exercício 3 – Criar e Usar Objetos

Objetivo: Aprender a criar **instâncias (objetos)** a partir de uma classe.

1. Cria um novo ficheiro Main.java.
2. No método main(), cria **duas lâmpadas** diferentes.
3. Faz com que uma comece ligada e a outra desligada.
4. Mostra no ecrã o estado de cada uma.

(Podes usar System.out.println() para confirmar o estado.)

💬 *Pergunta:*

O que acontece se alterares o estado de uma das lâmpadas?

A outra também muda?

📋 **Objetivo:** perceber que cada objeto tem o seu **estado próprio**.

🔄 Exercício 4 – Adicionar Comportamento

Objetivo: Fazer com que os objetos **tenham ações mais realistas**.

1. Acrescenta um novo método à classe Lampada que permita **ajustar a intensidade** (de 0 a 100).
2. No main(), testa este novo método.

3. Experimenta alterar a intensidade de uma lâmpada ligada e outra desligada.

💡 *Dica:* o método deve receber um número (por exemplo, `ajustarIntensidade(80)`).

🧠 *Pergunta:*

E se alguém tentar definir uma intensidade de 200?

Como podias evitar esse erro?

📋 **Objetivo:** perceber a importância de **controlar os dados** dentro de uma classe.

🔒 Exercício 5 – Encapsulamento

Objetivo: Aprender a proteger atributos e controlar o acesso.

1. Torna os atributos da `Lampada` privados (`private`).
2. Cria métodos públicos (`get` e `set`) para alterar e consultar os valores.
3. Garante que o método `setIntensidade()` só aceita valores válidos (entre 0 e 100).
4. Testa no `main()` o que acontece se tentares definir uma intensidade inválida.

💬 *Pergunta:*

Porque é que é uma boa prática esconder os atributos dentro da classe?

(Encapsular = proteger dados + controlar acesso)

🏠 Mini-Desafio – A Classe “CasaInteligente”

Objetivo: Fazer classes trabalharem em conjunto.

1. Cria uma nova classe chamada `CasaInteligente`.
2. Esta classe deve ter duas lâmpadas: uma da sala e uma do quarto.
3. Adiciona dois métodos:
 - `ligarTudo()` → liga todas as lâmpadas
 - `desligarTudo()` → desliga todas as lâmpadas
4. Testa no `main()`.

💡 *Dica:* a CasaInteligente não precisa de saber *como* cada lâmpada funciona — apenas precisa de **chamar os métodos** das lâmpadas.

📋 **Objetivo:** perceber o conceito de **composição de objetos**.