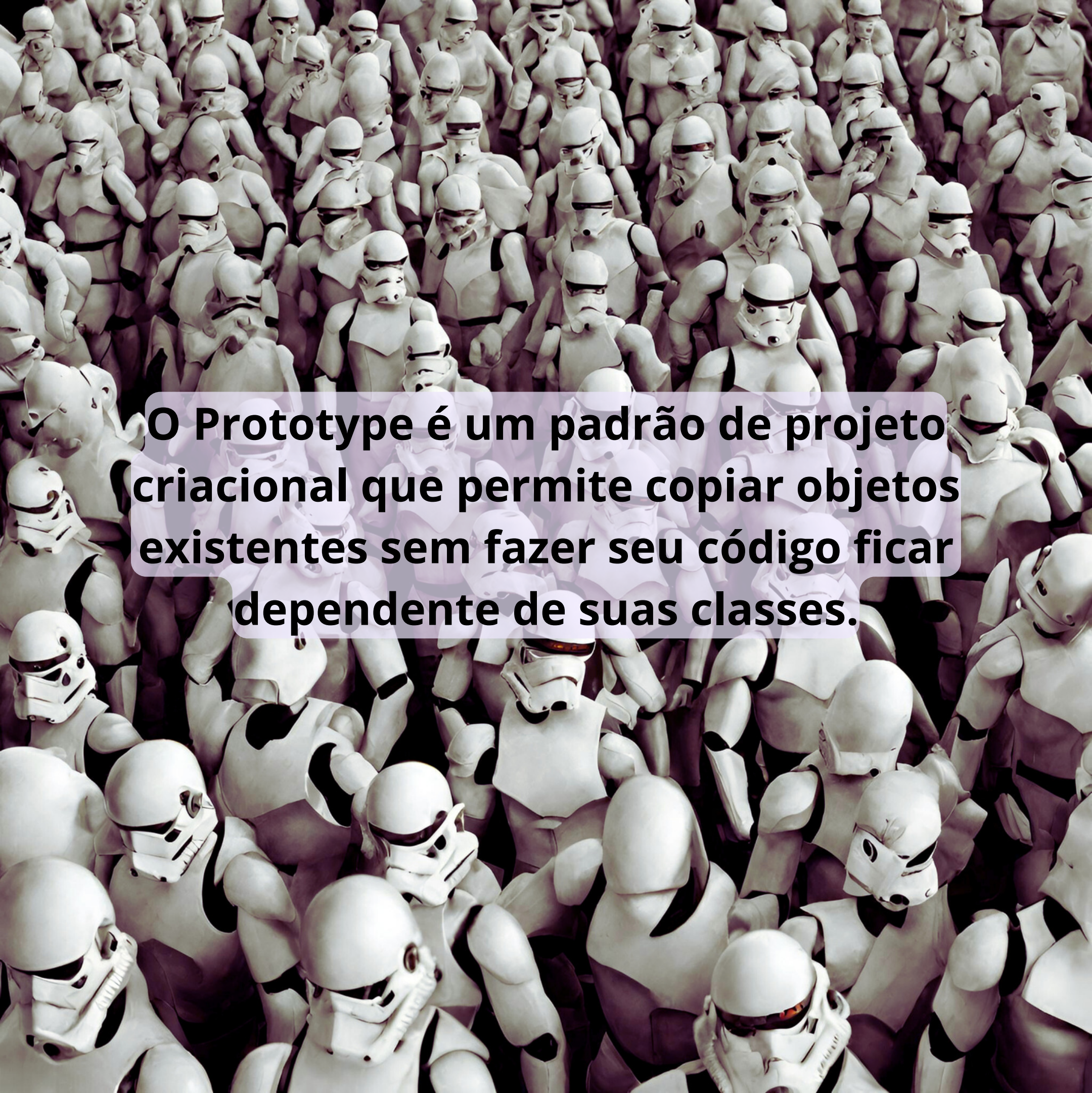
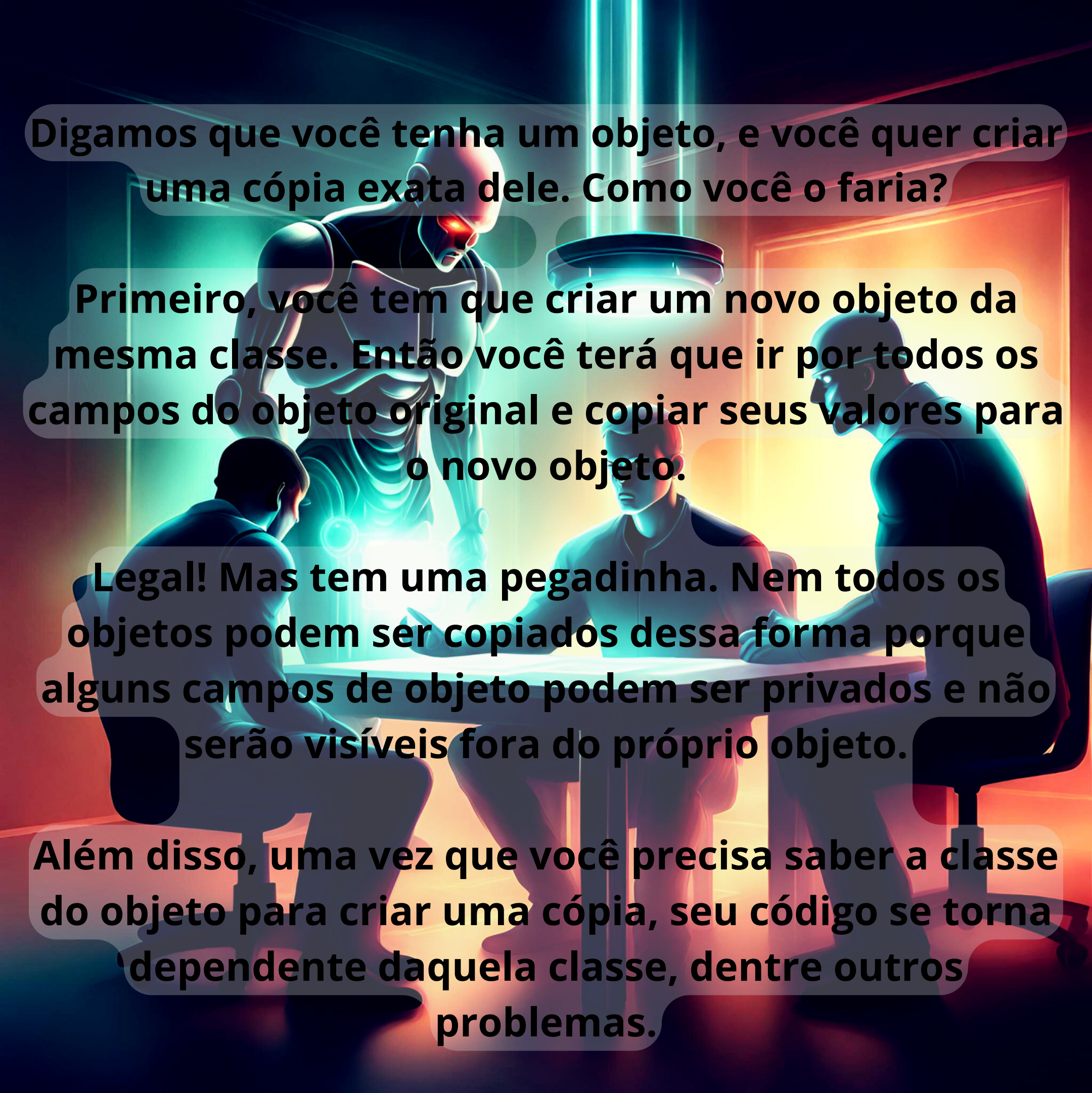


Prototype

Alysson Fernandes
Hilton Luan Barros



O Prototype é um padrão de projeto criacional que permite copiar objetos existentes sem fazer seu código ficar dependente de suas classes.

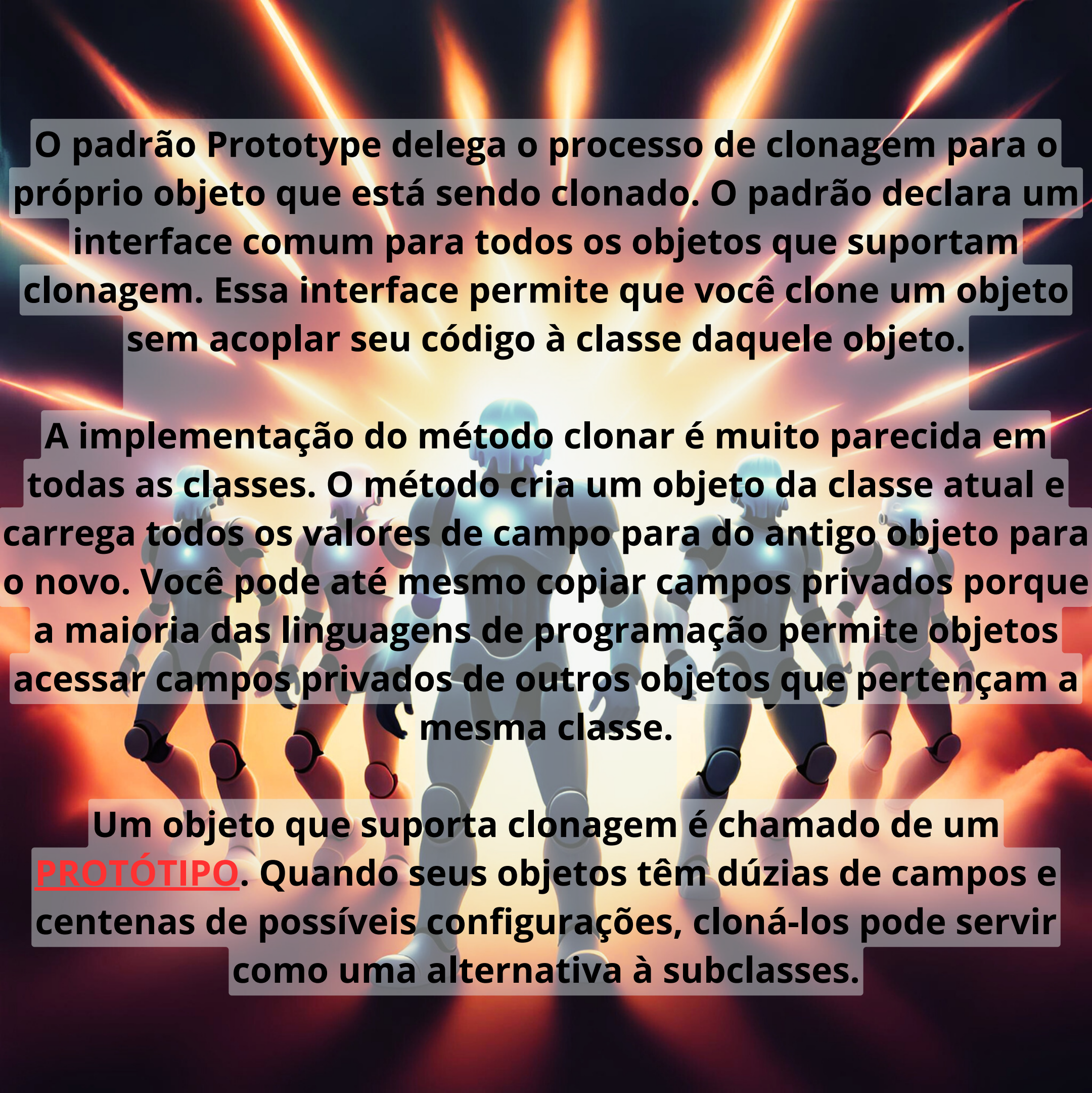


Digamos que você tenha um objeto, e você quer criar uma cópia exata dele. Como você o faria?

Primeiro, você tem que criar um novo objeto da mesma classe. Então você terá que ir por todos os campos do objeto original e copiar seus valores para o novo objeto.

Legal! Mas tem uma pegadinha. Nem todos os objetos podem ser copiados dessa forma porque alguns campos de objeto podem ser privados e não serão visíveis fora do próprio objeto.

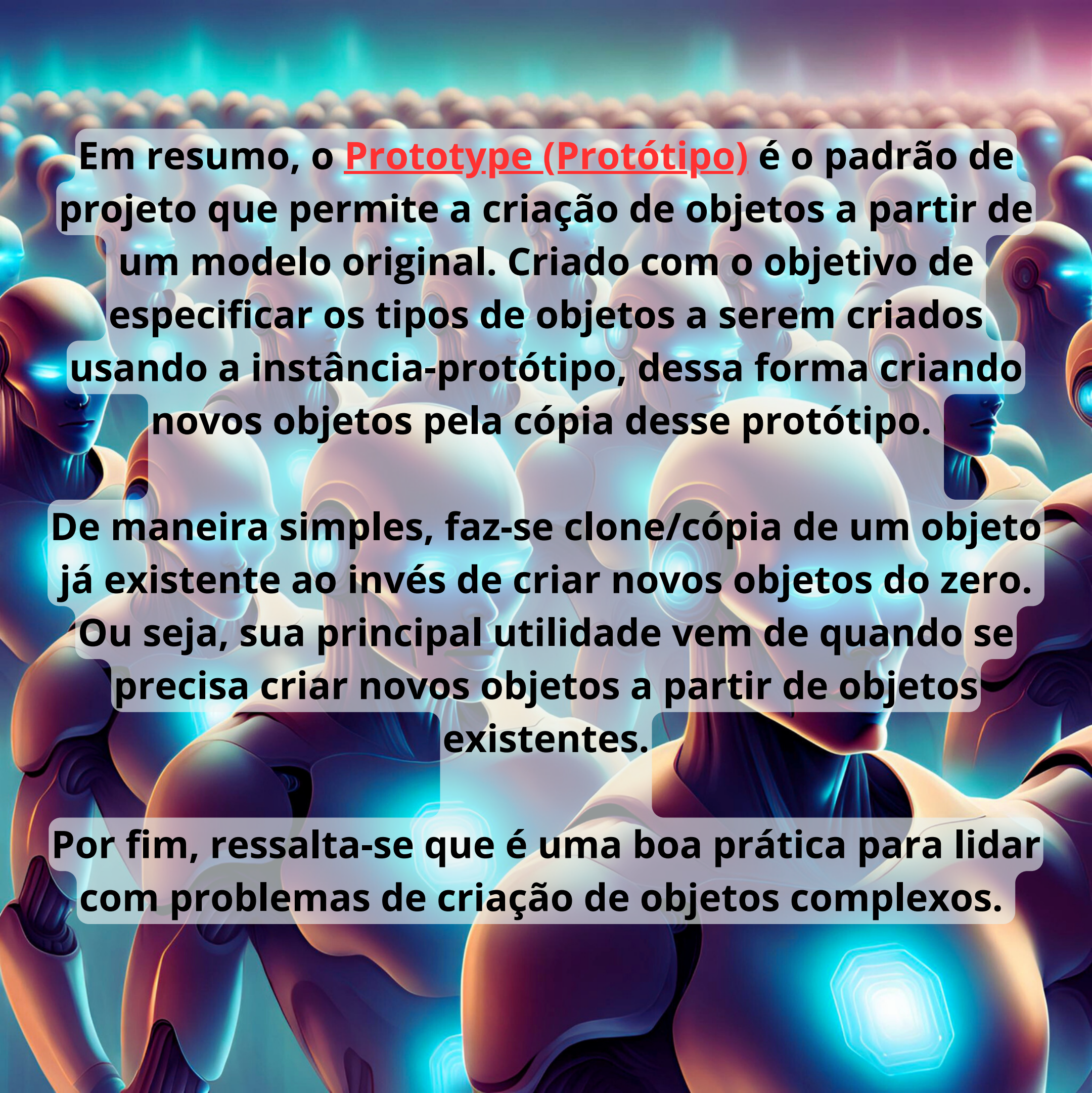
Além disso, uma vez que você precisa saber a classe do objeto para criar uma cópia, seu código se torna dependente daquela classe, dentre outros problemas.



O padrão Prototype delega o processo de clonagem para o próprio objeto que está sendo clonado. O padrão declara um interface comum para todos os objetos que suportam clonagem. Essa interface permite que você clone um objeto sem acoplar seu código à classe daquele objeto.

A implementação do método clonar é muito parecida em todas as classes. O método cria um objeto da classe atual e carrega todos os valores de campo para do antigo objeto para o novo. Você pode até mesmo copiar campos privados porque a maioria das linguagens de programação permite objetos acessar campos privados de outros objetos que pertençam a mesma classe.

Um objeto que suporta clonagem é chamado de um **PROTÓTIPO**. Quando seus objetos têm dúzias de campos e centenas de possíveis configurações, cloná-los pode servir como uma alternativa à subclasses.



Em resumo, o **Prototype (Protótipo)** é o padrão de projeto que permite a criação de objetos a partir de um modelo original. Criado com o objetivo de especificar os tipos de objetos a serem criados usando a instância-protótipo, dessa forma criando novos objetos pela cópia desse protótipo.

De maneira simples, faz-se clone/cópia de um objeto já existente ao invés de criar novos objetos do zero. Ou seja, sua principal utilidade vem de quando se precisa criar novos objetos a partir de objetos existentes.

Por fim, ressalta-se que é uma boa prática para lidar com problemas de criação de objetos complexos.

**Boas férias,
professor!**

