BUILDER

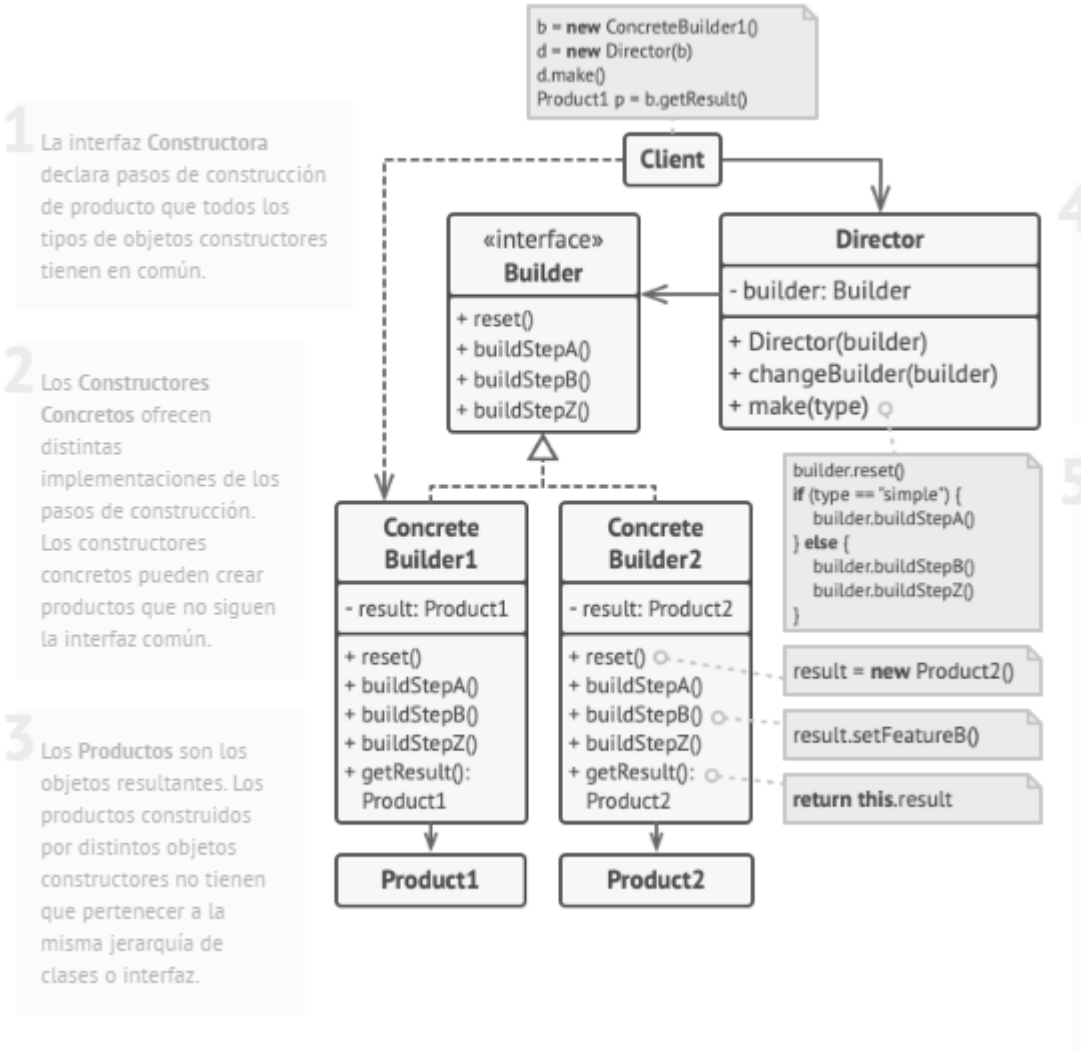
MIGUEL ANGEL RAMIREZ JUAREZ

JAVA ACADEMY

AUGUST/ 2024

Builder Pattern in Java?

The Builder Pattern is a design pattern used to construct complex objects step by step. It is particularly useful when an object needs to be created with various configurations or parts that can be assembled in different ways. This pattern allows for the creation of different representations of an object using the same construction process. By separating the construction of a complex object from its representation, the Builder Pattern enhances code readability and maintainability.



Problem Statement

In a typical scenario, consider a company that manufactures customized cars. Each car can have various options such as engine type, color, transmission, and additional features like sunroof or leather seats. Managing the creation of these complex car configurations directly within a single class would lead to a cumbersome and unmanageable codebase. Different car configurations would require numerous constructors or a multitude of setter methods, resulting in code that is difficult to maintain and extend.

Problem Description

In our scenario, the `Car` class includes fields for engine type, color, transmission, and additional features. We define an interface `CarBuilder` with methods like `setEngineType()`, `setColor()`, `setTransmission()`, and `addFeature()`. Concrete builders like `SportsCarBuilder` and `FamilyCarBuilder` implement this interface and configure the car according to their specific requirements. The `CarDirector` class uses these builders to create different car models.

By applying the Builder Pattern, we achieve a clean separation of concerns, allowing for flexible and manageable construction of complex car objects.