Armen Melkumyan • 1st
Technical / Solutions Architect
1yr • Edited •

🚀 Diving Into Design Patterns: The Builder Pattern with a Real-World Twist! 🏗️

Ever wondered how to construct complex objects step by step without sinking into the chaos of constructors? 🤔 Let's explore the Builder Design Pattern through the lens of building a house in C#. 🏠 💻

🔍 Complexity Level: Intermediate ⚙️
The Builder pattern is perfect for cases where you need to assemble various parts of a product to produce different representations. Imagine constructing a house 🏡. You wouldn't start with the roof, right? This pattern allows us to specify the type and details of the house we want to build, step by step.

Pros: 👍
1) Flexibility in Construction: Just like choosing the blueprint for your dream house, the Builder pattern lets you construct objects piece by piece.
2) Reusable: Once you have your builders set up, you can produce myriad variations of your product (houses, in our case) without redoing the entire construction process.
3) Improved Code Readability: Say goodbye to telescoping constructor antipattern. Your codebase becomes cleaner and more understandable.

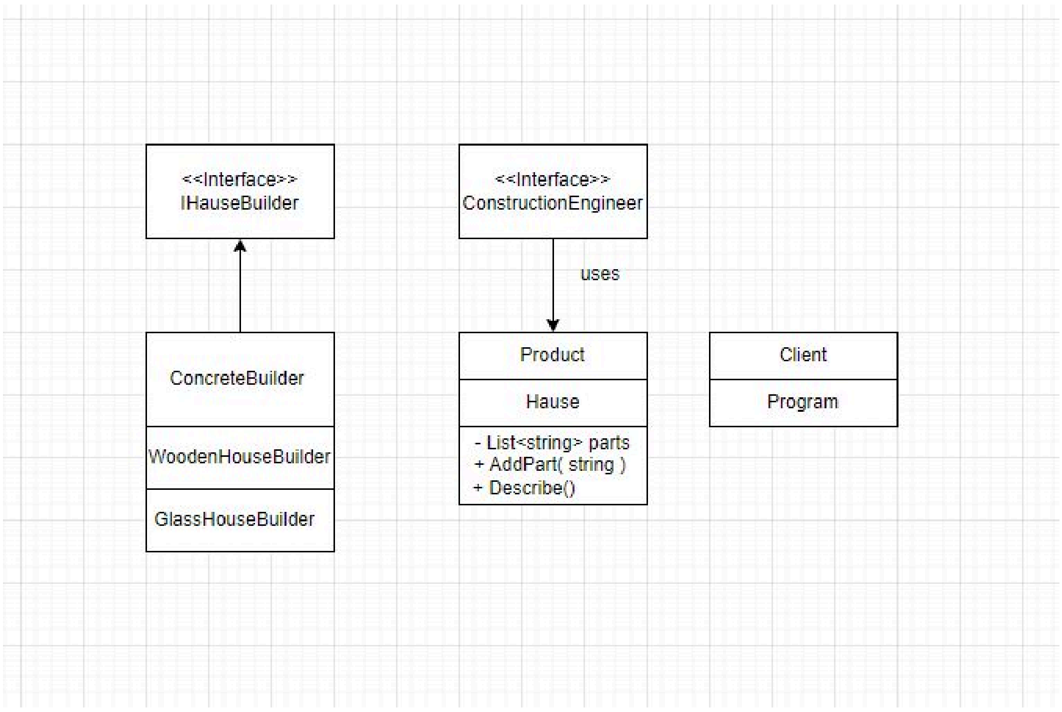
Cons: 👎
1) Complexity: Introducing the Builder pattern can complicate your codebase if your project is simple. It's like using a sledgehammer to crack a nut.
2) Proliferation of Classes: Every new product requires a new concrete builder, leading to an increase in the number of classes.

Real-World Example: 🌍🛠️
Let's construct a house. We start with a HouseBuilder interface defining methods like BuildWalls(), BuildDoors(), BuildRoof(), and BuildWindows(). Concrete builders (e.g., WoodenHouseBuilder, GlassHouseBuilder) implement this interface to offer different types of houses. The ConstractionEngineer

class guides the construction process, and the client interacts with it to construct the house step by step, finally calling BuildHouse() to get the finished house.

Github code: https://lnkd.in/dgyiZubF

#DesignPatterns #CSharp #SoftwareDevelopment #CodingLife



29