Study Material: 10 Design Patterns in 10 Minutes

1. Introduction

Purpose: Learning design patterns helps you structure and scale your code-regardless of your programming language.

Goal of the Video: To briefly outline 10 essential design patterns from the "Gang of Four" catalogue, highlighting their purpose and typical use cases.

2. List of Design Patterns

Creational Patterns

- Singleton: Ensures one instance of a class.
- Factory Method: Creates objects based on input conditions.
- Builder: Constructs complex objects step-by-step.
- Prototype: Copies existing objects instead of creating new ones.

Structural Patterns

- Adapter: Transforms one interface into another.
- Facade: Provides a simplified interface to a complex system.
- Composite: Lets you treat objects and collections uniformly.
- Proxy: Controls access to another object.

Behavioral Patterns

- Observer: Updates observers automatically when something changes.
- Strategy: Encapsulates interchangeable behaviors or algorithms.

3. Key Insights

- Pattern Thinking: Understanding patterns shapes your approach-don't just memorize them, internalize the problem-solving mindset.
- Language Agnostic: Patterns transcend language specifics. Once you grasp them, you can apply the concepts in Java, JavaScript, Python-you name it.
- Reusability & Clarity: Patterns help make code modular, extendable, and easier to maintain-key skills as your codebases grow.

4. How to Use This Guide

- Review Each Pattern: Recall its definition and standard use case.
- Apply It: Try coding a minimal example yourself. For instance, implement Observer in Java or Strategy in Python.
- Reflect: Ask when you've seen similar problems in your projects and consider whether a pattern could simplify the solution.