ONE PAGE SUMMARY

IT & Software › Design Patterns › Behavioral Patterns

Introduction to Behavioral Patterns

Mastering Behavioral Patterns Enhances Software Interaction and Flexibility

DEFINITION

> Behavioral patterns are design patterns that focus on how objects interact and communicate. They help define clear communication and responsibilities among software components.

KEY CONCEPTS

Observer Pattern

Allows a subject to notify multiple observers about changes in its state, promoting loose coupling.

Strategy Pattern

Enables selecting an algorithm's behavior at runtime, allowing for flexible and interchangeable strategies.

Command Pattern

Encapsulates a request as an object, allowing for parameterization of clients with queues, requests, and operations.

State Pattern

Allows an object to change its behavior when its internal state changes, making state management easier.

Chain of Responsibility

Passes a request along a chain of handlers, allowing multiple objects to handle the request without tight coupling.

EXAMPLES

→ Using the Observer Pattern in a stock market application to update clients on price changes.

→ Implementing the Strategy Pattern in a payment system to switch between credit card and PayPal options.

→ Applying the Command Pattern to create undo functionality in a text editor.

MEMORY TIPS

★ Remember 'O' for Observer as 'O' for 'Notify' - they notify each other.

★ Think of Strategy as a 'Game Plan' that can change based on the situation.

★ Visualize Command as a 'Remote Control' that sends commands to different devices.

COMMON MISTAKES

✗ Confusing the Observer Pattern with the Mediator Pattern, which serves a different purpose.

✗ Overusing the Strategy Pattern when a simple conditional statement would suffice.

✗ Neglecting to manage state transitions properly in the State Pattern.

QUICK RECAP

Behavioral patterns focus on object interactions and responsibilities. Key patterns include Observer, Strategy, Command, State, and Chain of Responsibility. Understanding these patterns enhances software flexibility and maintainability.