Design Patterns

A design pattern is a tried-and-true method for resolving the particular issues and tasks. Design patterns are solutions for resolving common object-oriented design issues that are independent of programming languages. In other words, a design pattern does not reflect a specific implementation but rather an idea.

You may improve the flexibility, reuse, and maintainability of your code by utilizing design patterns. That is crucial since Java internally adheres to design patterns. You must be familiar with at least a few common solutions to coding issues (such as design patterns) in order to work as a professional software developer.

Advantages of design pattern:

- Design patterns have the benefit of being reusable in numerous projects.
- They offer the answers that contribute to defining the system architecture.
- They document the experiences with software engineering.
- They make an application's design more transparent.

These have been based on the expertise and experience of seasoned software engineers, making them well-proven and tested solutions. Design patterns might not always provide the perfect answer to a problem. They make the system architecture more understandable and open the door to creating a more effective system.

When should we use the design patterns?

Throughout the SDLC's analysis and requirement phase, we must employ design patterns (Software Development Life Cycle). Design patterns facilitate the SDLC's analysis and requirement phase by offering knowledge based on practical experience.

Categorization of design patterns: Basically, design patterns are categorized into two parts: In core java, there are mainly three types of design patterns, which are further divided into their sub-parts:

1. **Creational Design Patterns include:** Factory Pattern, Abstract Factory Pattern, Singleton Pattern, Prototype Pattern, Builder Pattern.
2. **Structural Design Patterns include:** Adapter Pattern, Bridge Pattern, Composite Pattern, Decorator Pattern, Facade Pattern, Flyweight Pattern, Proxy Pattern
3. **Behavioral Design Patterns include the following:** Chain Of Responsibility Pattern, Command Pattern, Interpreter Pattern, Iterator Pattern, Mediator Pattern, Memento Pattern, Observer Pattern, State Pattern, Strategy Pattern, Template Pattern, Visitor Pattern