**Design Pattern in Java**

**Agenda:**

* Why we need Design pattern?
* What is Design Pattern?
* Structure of Design pattern.
* Types of Design pattern.
* Overview of Design pattern.

**Why we need Design Pattern:**

Design patterns are a way of solving commonly occurring problems in software design. They provide a proven and tested solution to these problems, which can help to improve the quality, reliability, maintainability, and scalability of software systems.

* **Reusability:** Design patterns are reusable solutions to common problems in software design. By using design patterns, developers can avoid reinventing the wheel and instead use proven solutions to problems.
* **Maintainability**: Design patterns can make software systems more maintainable by making them easier to understand and modify. Since design patterns are well-known and documented, other developers can easily understand and modify the code.
* **Scalability**: Design patterns can help to make software systems more scalable by providing a structured approach to software design. By using design patterns, developers can ensure that their software systems are well-structured and can handle increased loads and changing requirements.
* **Quality:** Design patterns can help to improve the quality of software systems by ensuring that they are designed to be reliable, efficient, and secure.

**What is Design pattern?**

A design pattern is a general repeatable solution to a commonly occurring problem in software design. It represents a proven and tested way to approach a specific type of problem and can be applied to various contexts. A design pattern can provide a template or blueprint for creating software that is scalable, maintainable, and flexible.

Design patterns can be classified into three categories: creational, structural, and behavioral.

**Structure of Design Pattern:**

Design patterns have a common structure consisting of four elements:

1. **Pattern Name**: A descriptive name that identifies the pattern.
2. **Problem:** A statement describing the problem and its context. This section explains when and where the pattern can be applied.
3. **Solution:** A description of the components that make up the design pattern and how they interact to solve the problem. This section provides a general reusable solution to the problem.
4. **Consequences**: A discussion of the advantages and disadvantages of using the pattern, including its impact on system qualities such as flexibility, extensibility, and maintainability.

**Note**: - Some design patterns may also include additional elements.

**Types of Design Pattern:**

Design patterns can be classified into three main categories:

1: **Creational Design Patterns**: - Creational Design pattern are concerned with the method of creating objects.

The creational Design patterns are further classified as follows:

* Factory Design Pattern
* Abstract Design Pattern
* Singleton Pattern
* Prototype Pattern
* Builder Pattern
* Object Pool Pattern

2: **Structural Design Pattern: -** Structural Design Pattern deal with the composition of classes and objects which form larger structures.

The Structural Design Pattern are further classified as follows:

* Façade Pattern
* Bridge Pattern
* Composite Pattern
* Decorator Pattern
* Adapter Pattern
* Flyweight Pattern
* Proxy Pattern
* Filter Pattern

**3: Behavioral Design Pattern:** Behavioral Design Pattern are concerned with the responsibility and interaction between the objects.

The Behavioral Design Pattern are further classified as follows:

* Strategy Pattern
* Iterator Pattern
* Mediator Pattern
* Command Pattern
* State Pattern
* Observer Pattern
* Template Pattern
* Visitor Pattern