

APPLYING GOF DESIGN PATTERNS

In 1994, four authors Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides published a book titled Design Patterns - Elements of Reusable Object-Oriented Software which initiated the concept of Design Pattern in Software development.

These authors are collectively known as Gang of Four (GOF). According to these authors design patterns are primarily based on the following principles of object orientated design.

- Program to an interface not an implementation
- Favor object composition over inheritance

Pattern& Description


There are 23 design patterns which can be classified in three categories:

Creational Patterns : These design patterns provide a way to create objects while hiding the creation logic, rather than instantiating objects directly using new operator. This gives program more flexibility in deciding which objects need to be created for a given use case.

Structural Patterns : These design patterns concern class and object composition. Concept of inheritance is used to compose interfaces and define ways to compose objects to obtain new functionalities.

Behavioral Patterns : These design patterns are specifically concerned with communication between objects.

The 23 design patterns are listed below:

		Purpose		
		Creational	Structural	Behavioral
	Class	Factory Method	Adapter	Interpreter Template Method
	Object	Abstract Factory Builder Prototype Singleton	Adapter Bridge Composite Decorator Façade Flyweight Proxy	Chain of Responsibility Command Iterator Mediator Memento Observer State Strategy Visitor

Creational Patterns

- Make the system independent of how objects are created composed and represented
 - Abstract the instantiation process
 - o Hide how instances of these classes are created and assembled
 - o Hide references to concrete classes used in the system
 - Govern the what, when, who, how object creation
1. **Abstract Factory:** Creates an instance of several families of classes. Provide an interface for creating families of related or dependent objects without specifying their concrete classes.
 2. **Builder:** Separates object construction from its representation. Separate the construction of a complex object from its representation so that the same construction processes can create different representations.
 3. **Factory Method:** Creates an instance of several derived classes. Define an interface for creating an object, but let subclasses decide which class to instantiate. Factory Method lets a class defer instantiation to subclasses.
 4. **Prototype:** A fully initialized instance to be copied or cloned. Specify the kinds of objects to create using a prototypical instance, and create new objects by copying this prototype.
 5. **Singleton:** A class of which only a single instance can exist. Ensure a class only has one instance, and provide a global point of access to it.