**LEARN IN ORDER - BSC**

<https://refactoring.guru/design-patterns>

<https://stackabuse.com/design-patterns-in-java/>

<https://refactoring.guru/design-patterns/chain-of-responsibility/java/example>

1. **INTERPRETOR DESIGN PATTERN**

The Interpreter pattern is used anytime we need to evaluate any kind of language grammar or expressions. A good example of this pattern would be [Google Translate](https://translate.google.com/), which interprets the input, and shows us the output in another language.

Example : converting and interpreting which mathematical operation to do based on “add” or “multiply” in the string .

1. **TEMPLATE DESIGN PATTERN**

**Template Method** is a behavioral design pattern that defines the skeleton of an algorithm in the superclass but lets subclasses override specific steps of the algorithm without changing its structure.

**Example : We can template office life steps , such as work , take a pause , work , get paid as a template both for managers & developers employee . Also login and logout steps of web-applications**

**Can also be thought of as template way of doing things .**

1. **CHAIN OF RESPONSIBILITY PATTERN**

**Chain of Responsibility** is behavioral design pattern that allows passing request along the chain of potential handlers until one of them handles request.

1. **COMMAND PATTERN**

**Command** is a behavioral design pattern that turns a request into a stand-alone object that contains all information about the request. This transformation lets you parameterize methods with different requests, delay or queue a request’s execution, and support undoable operations.

1. **ITERATOR**

**Iterator** is a behavioral design pattern that lets you traverse elements of a collection without exposing its underlying representation (list, stack, tree, etc.).

**6 . MEDIATOR**

**Mediator** is a behavioral design pattern that reduces coupling between components of a program by making them communicate indirectly, through a special mediator object.

1. **. MEMENTO**

**Memento**

The Memento pattern is concerned with previous states of the object. This means that the pattern is used when we want to save some state of an object, in the case we might want to restore the object to that state later on.

This pattern relies on the work of three classes, also known as actor classes. The Memento object contains a state that we wish to save for later use. The Originator object creates and stores states in the Memento objects, while the CareTaker object takes care of the restoration process.

1. . OBSERVER PATTERN

**State** is a behavioral design pattern that allows an object to change the behavior when its internal state changes.

**COMPOSITE DESIGN PATTERN**

* **9 .** component – is the base interface for all the objects in the composition. It should be either an interface or an abstract class with the common methods to manage the child composites.
* leaf – implements the default behavior of the base component. It doesn't contain a reference to the other objects.
* composite – has leaf elements. It implements the base component methods and defines the child-related operations.
* client – has access to the composition elements by using the base component object.

<https://www.baeldung.com/java-composite-pattern>