DESIGN PATTERNS

1. Singleton is a creational design pattern that lets you ensure that a class has only one instance while providing a global access point to this instance.
2. Prototype is a creational design pattern that lets you copy existing objects without making your code dependent on their classes.
3. Iterator is a behavioral design pattern that lets you traverse elements of a collection without exposing its underlying representation (list, stack, tree, etc.).
4. Proxy is a structural design pattern that lets you provide a substitute or placeholder for another object. A proxy controls access to the original object, allowing you to perform something either before or after the request gets through to the original object.
5. Dependency Injection design pattern allows us to remove the hard-coded dependencies and make our application loosely coupled, extendable and maintainable. We can implement dependency injection in java to move the dependency resolution from compile-time to runtime.
6. 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.
7. Model-View-Controller Pattern is used to separate the application's concerns.

* Model - Model represents an object or JAVA POJO carrying data. It can also have logic to update the controller if its data changes.
* View - View represents the visualization of the data that the model contains.
* Controller - Controller acts on both model and view. It controls the data flow into the model object and updates the view whenever data changes. It keeps the view and model separate.

1. Data Access Object Pattern or DAO pattern is used to separate low-level data accessing API or operations from high-level business services. Following are the participants in Data Access Object Pattern.

* Data Access Object Interface - This interface defines the standard operations to be performed on a model object(s).
* Data Access Object concrete class - This class implements the above interface. This class is responsible to get data from a data source which can be a database / XML or any other storage mechanism.
* Model Object or Value Object - This object is a simple POJO containing get/set methods to store data retrieved using DAO class.

1. In the Null Object pattern, a null object replaces the check of the NULL object instance. Instead of putting the check for a null value, the Null Object reflects a do-nothing relationship. Such a Null object can also be used to provide default behavior in case data is not available.

In the Null Object pattern, we create an abstract class specifying various operations to be done, concrete classes extending this class and a null object class providing do-nothing implementation of this class and will be used seamlessly when we need to check the null value.

1. Façade is a structural design pattern that provides a simplified interface to a library, a framework, or any other complex set of classes.
2. Builder is a creational design pattern that lets you construct complex objects step by step. The pattern allows you to produce different types and representations of an object using the same construction code.
3. Factory Method is a creational design pattern that provides an interface for creating objects in a superclass but allows subclasses to alter the type of objects that will be created.
4. State Pattern says that "the class behavior changes based on its state". In State Pattern, we create objects which represent various states and a context object whose behavior varies as its state object changes.

The State Pattern is also known as Objects for States.

ДИЗАЙН МОДЕЛИ

1. Singleton е модел на творческо проектиране, който ви позволява да гарантирате, че даден клас има само един екземпляр, като същевременно предоставя глобална точка за достъп до този екземпляр.

2. Prototype е модел на творческо проектиране, който ви позволява да копирате съществуващи обекти, без да правите кода си зависим от техните класове.

3. Итераторът е модел на поведенчески дизайн, който ви позволява да трансверсирате елементи от колекция, без да излагате основното му представяне (списък, стека, дърво и т.н.).

4. Прокси е структурен дизайн модел, който ви позволява да предоставите заместител или контейнер за друг обект. Проксито контролира достъпа до оригиналния обект, което ви позволява да извършите нещо или преди, или след като заявката премине към оригиналния обект.

5. Зависимост Инжектиране дизайн модел ни позволява да премахнете твърдо кодира зависимости и направи нашето приложение хлабаво съчетани, разширяеми и поддържаеми. Можем да внедрим зависимост инжекция в Java, за да преместите разделителната способност на зависимостта от compile-time до runtime.

6. Шаблонният метод е модел на поведенчески дизайн, който определя скелета на алгоритъм в суперкласа, но позволява на подкласовете да отменят конкретни стъпки на алгоритъма, без да променят структурата му.

7. Модел-View-Controller Pattern се използва за разделяне на опасенията на заявлението.

        Модел - Модел представлява обект или JAVA POJO носещи данни. Също така може да има логика да актуализира администратора, ако данните му се променят.

        Изглед - Изглед представлява визуализацията на данните, които моделът съдържа.

        Контролер - Контролер действа както по модел, така и по изглед. Той контролира потока от данни в обекта на модела и актуализира изгледа винаги, когато данните се променят. Тя поддържа изгледа и модела отделно.

8. Обектен модел за достъп до данни или МОДЕЛ DAO се използва за отделяне на данни от ниско ниво, достъп до API или операции от бизнес услуги на високо ниво. Следват участниците в Обектен модел за достъп до данни.

        Обектен интерфейс за достъп до данни - Този интерфейс определя стандартните операции, които трябва да се извършват на обект(и) модел.

        Data Access Object бетон клас - Този клас внедрява горния интерфейс. Този клас е отговорен да получи данни от източник на данни, който може да бъде база данни / XML или всеки друг механизъм за съхранение.

        Обект на модел или обект на стойност - Този обект е прост POJO, съдържащ методи за получаване/задаване за съхраняване на данни, извлечени с помощта на DAO клас.

9. В Null Object модел null обект замества проверката на nuLL обект екземпляр. Вместо да поставите чека за нищожна стойност, Null Object отразява релация do-нищо. Такъв Null обект може да се използва и за осигуряване на поведение по подразбиране в случай, че не са налични данни.

В модела Null Object създаваме абстрактен клас, уточняващ различни операции, които трябва да се направят, конкретни класове, разширяващи този клас и клас null обект, осигуряващ do-нищо изпълнение на този клас и ще се използва безпроблемно, когато трябва да проверим нищожната стойност.

10. Фасадата е модел на структурен дизайн, който осигурява опростен интерфейс към библиотека, рамка или друг сложен набор от класове.

11. Builder е модел на творческо проектиране, който ви позволява да конструирате сложни обекти стъпка по стъпка. Моделът ви позволява да произвеждате различни видове и представяния на обект с помощта на един и същ строителен код.

12. Factory Method е модел на творческо проектиране, който осигурява интерфейс за създаване на обекти в суперклас, но позволява на подкласовете да променят вида на обектите, които ще бъдат създадени.

13. State Pattern казва, че "поведението на класа се променя въз основа на състоянието си". В State Pattern създаваме обекти, които представляват различни държави и контекстен обект, чието поведение варира с промяната на нейния държавен обект.

Държавният модел е известен и като Обекти за държави.