THE UNIVERSITY OF MELBOURNE

SWEN20003 Workshop

Week 10

Demonstrator: Xulin Yang

Workshop tasks

1. About project 1 – no submission people please discuss with me after workshop
2. Go through concept review
3. Divide groups by records
4. Do questions Q1, 2, 3, 4 together
5. Take attendance

Design Pattern - Factory Pattern

Personal understanding: Group similar object's creation logic together as a "factory"

https://www.tutorialspoint.com/design_pattern/factory_pattern.htm

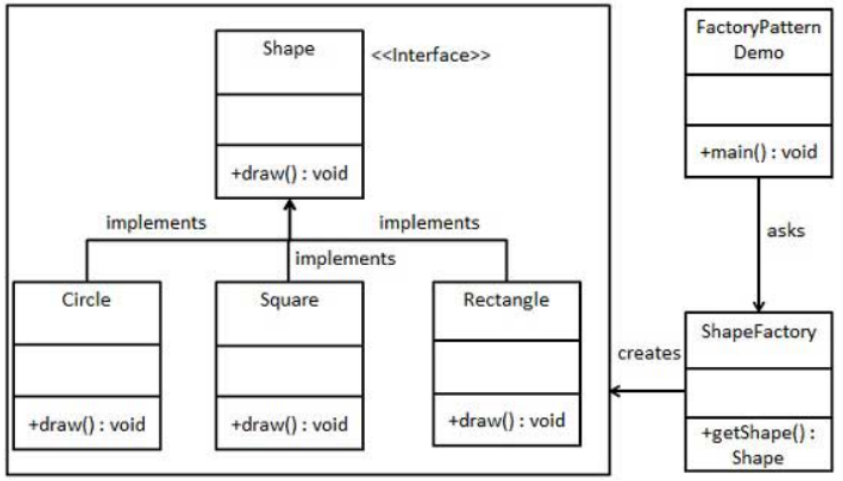
Factory pattern is one of the most used design patterns in Java. This type of design pattern comes under creational pattern as this pattern provides one of the best ways to create an object.

In Factory pattern, we create object without exposing the creation logic to the client and refer to newly created object using a common interface.

Implementation

We're going to create a *Shape* interface and concrete classes implementing the *Shape* interface. A factory class *ShapeFactory* is defined as a next step.

FactoryPatternDemo, our demo class will use *ShapeFactory* to get a *Shape* object. It will pass information (*CIRCLE / RECTANGLE / SQUARE*) to *ShapeFactory* to get the type of object it needs.



Design Pattern – Strategy Pattern

Personal understanding: You have different version of behaviors for your associated object

https://www.tutorialspoint.com/design_pattern/strategy_pattern.htm

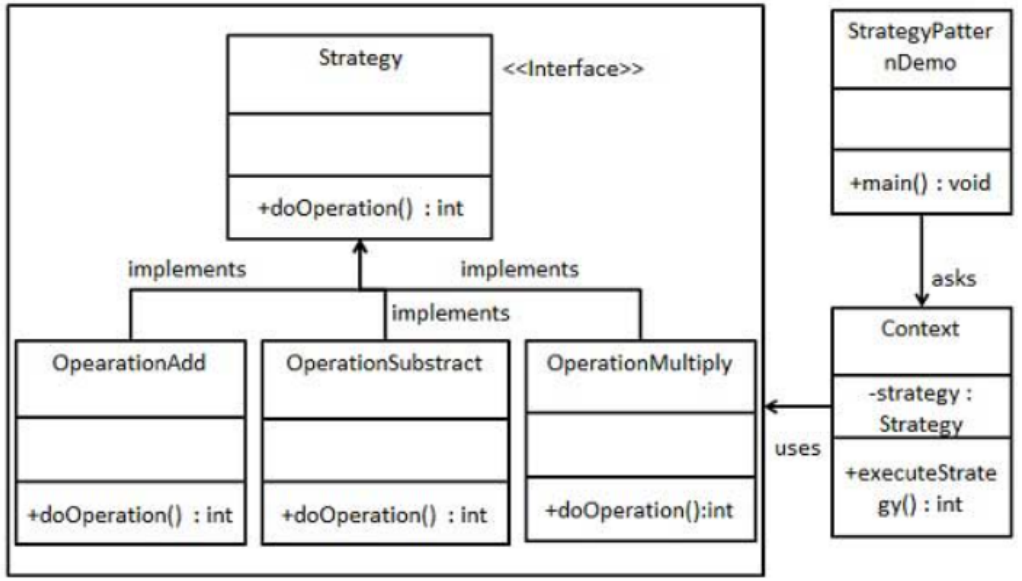
In Strategy pattern, a class behavior or its algorithm can be changed at run time. This type of design pattern comes under behavior pattern.

In Strategy pattern, we create objects which represent various strategies and a context object whose behavior varies as per its strategy object. The strategy object changes the executing algorithm of the context object.

Implementation

We are going to create a *Strategy* interface defining an action and concrete strategy classes implementing the *Strategy* interface. *Context* is a class which uses a Strategy.

StrategyPatternDemo, our demo class, will use *Context* and strategy objects to demonstrate change in Context behaviour based on strategy it deploys or uses.



THE UNIVERSITY OF MELBOURNE

Thank you