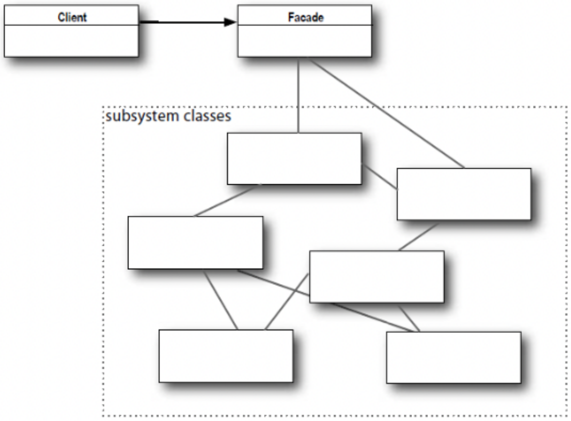
Lecture 6 - Design Patterns

- What are Design Patterns
  - Descriptions of communicating objects and classes that are customized to **solve** a general design **problem** in a particular **context**
  - Gamma et al. described 23 design patterns divided into three categories:
    1. Creational patterns
    2. Structural patterns
    3. Behavioral patterns
- Creational Patterns
  - Concern the process of object creation
  - Six creational patterns
    1. Factory Method
    2. Abstract Factory
    3. Singleton
    4. Prototype
    5. Builder
    6. Object Pool
- Structural Patterns
  - Deal with the composition of classes or objects
  - Seven structural patterns
    1. Adapter
    2. Bridge
    3. Composite
    4. Decorator
    5. Facade
    6. Flyweight
    7. Proxy
- Behavioral Patterns
  - Characterize the ways in which classes or objects interact and distribute responsibility
  - Ten Behavioral patterns
    1. Chain of Responsibility
    2. Command
    3. Interpreter
    4. Iterator
    5. Mediator
    6. Memento
    7. Observer
    8. State
    9. Strategy
    10. Template
- Singleton (Creational)
  - Intent: Ensure a class has only one instance, and provide a global point of access to it

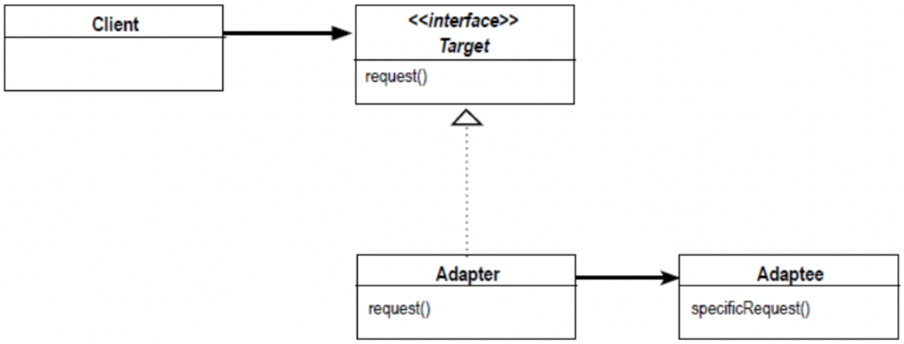
Singleton
−instance: Singleton
−Singleton() +getInstance(): Singleton . . .

(− means private, + means public)

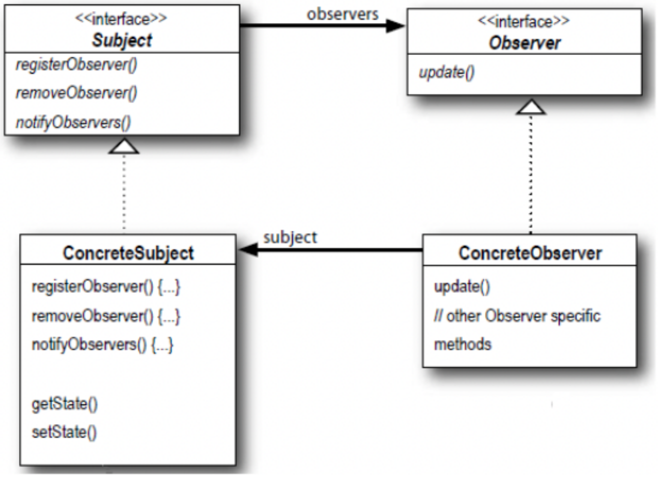
- Facade (Structural)
  - Intent: Hide complexities and provide a unified interface to a set of interfaces in a subsystem



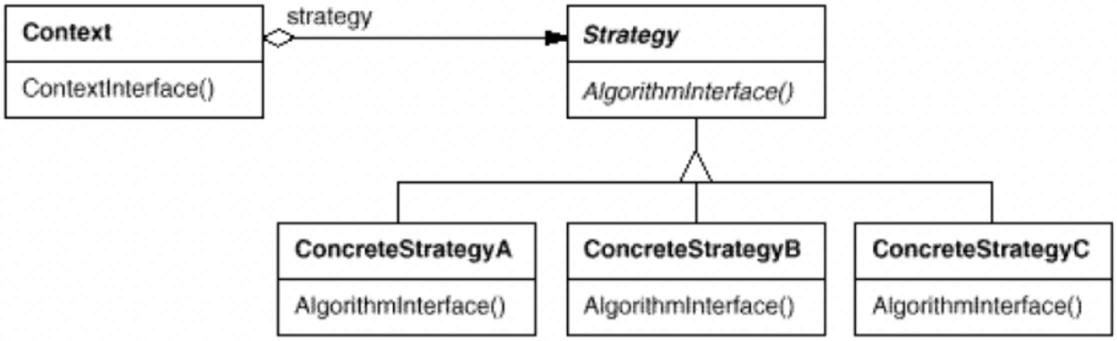
- Adapter (Structural)
  - Intent: Let classes work together that couldn't otherwise because of incompatible interfaces



- Observer (Behavioral)
  - Intent: Define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically



- Strategy (Behavioral)
  - Intent: Define a family of algorithms, encapsulate each one, and make them interchangeable


Context
ContextInterface()
strategy
Strategy
AlgorithmInterface()
ConcreteStrategyA
AlgorithmInterface()
ConcreteStrategyB
AlgorithmInterface()
ConcreteStrategyC
AlgorithmInterface()