




Design Pattern

By Ramtin Shakeri



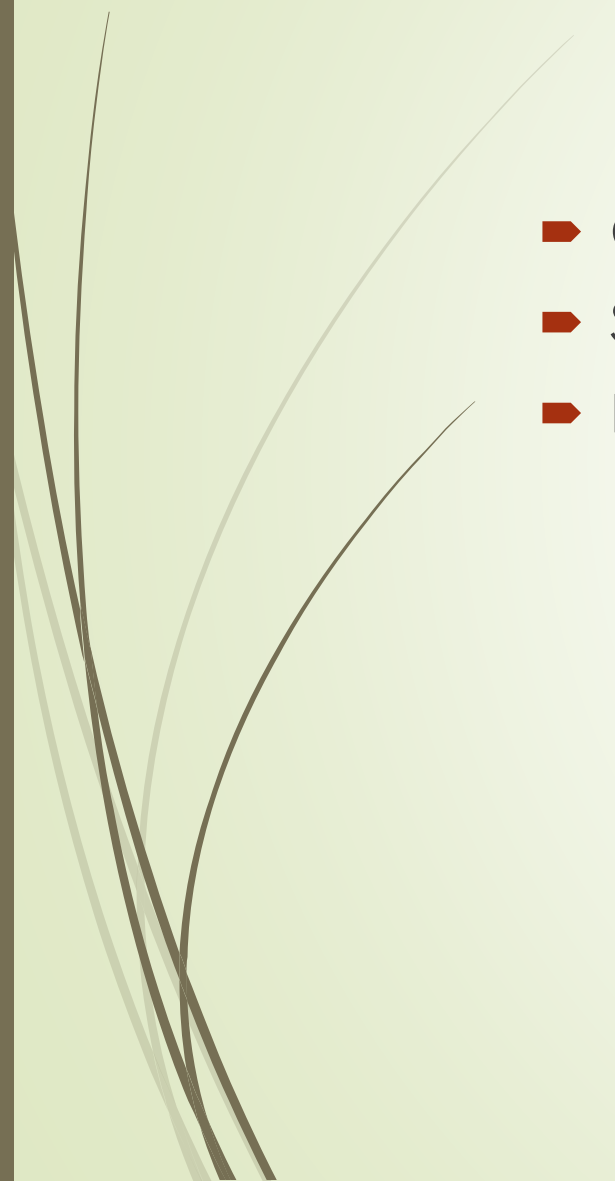
Design Pattern



In software engineering, a software design pattern is a general, reusable solution to a commonly occurring problem within a given context in software design. It is not a finished design that can be transformed directly into source or machine code. It is a description or template for how to solve a problem that can be used in many different situations. Design patterns are formalized best practice that the programmer can use to solve common problems when designing an application or system.




Design Pattern

- Creational Patterns
 - Structural Patterns
 - Behavioral Patterns
- 



Creational Patterns



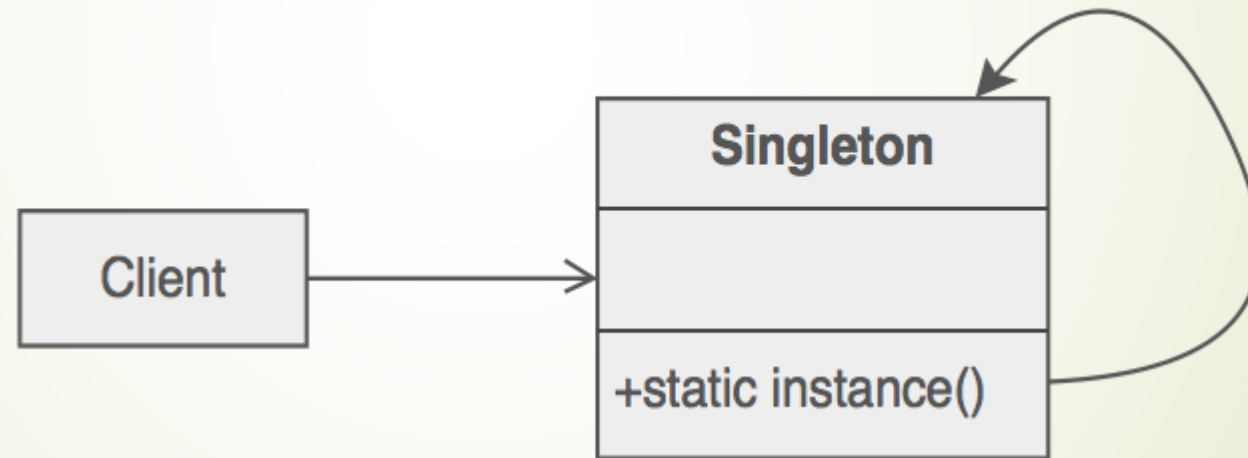
In software engineering, creational design patterns are design patterns that deal with object creation mechanisms, trying to create objects in a manner suitable to the situation. The basic form of object creation could result in design problems or added complexity to the design. Creational design patterns solve this problem by somehow controlling this object creation.



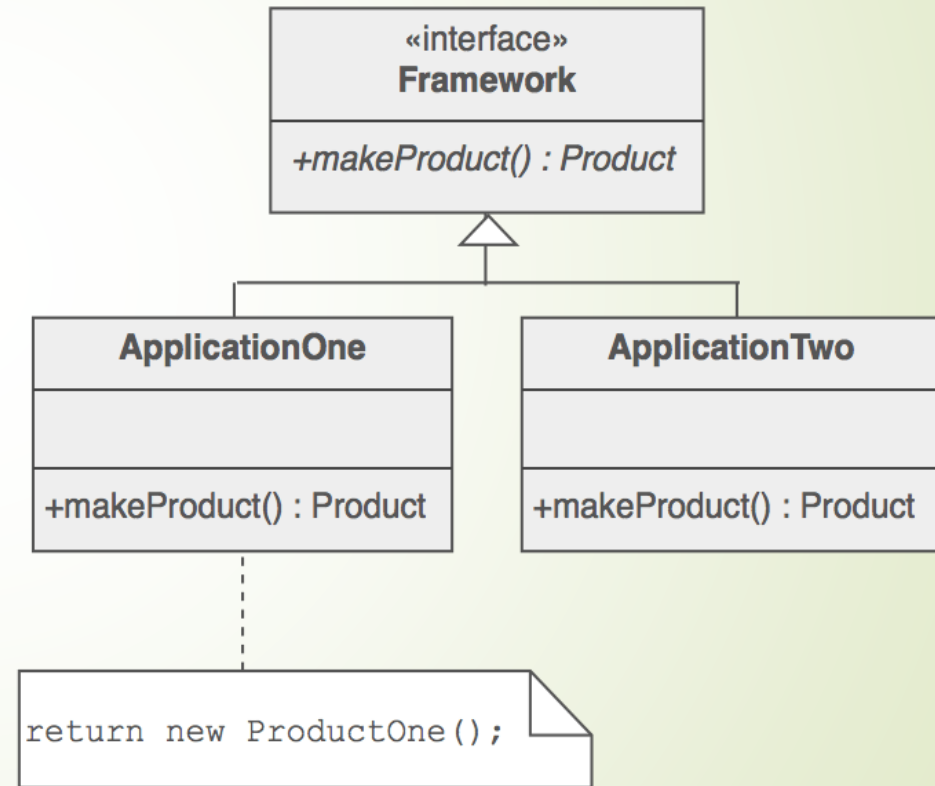
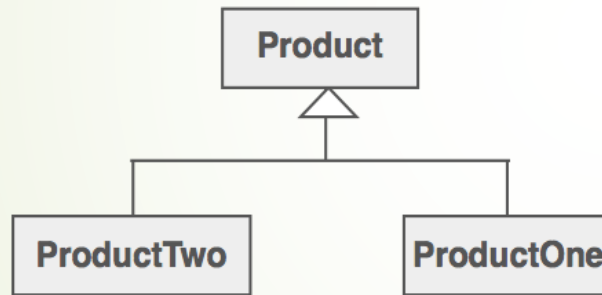
Creational Patterns

- Singleton
 - Factory Method
 - Abstract Factory
- 

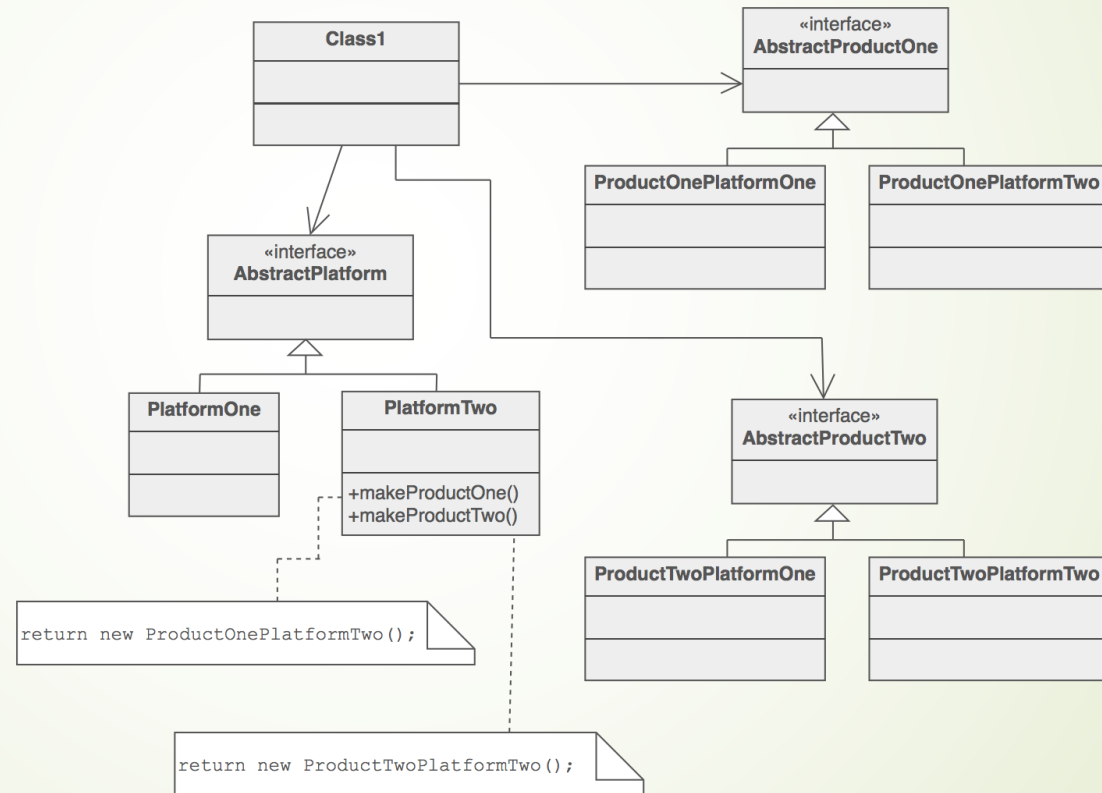
Singleton Pattern



Factory Method Pattern



Abstract Factory Pattern





Structural Patterns

In Software Engineering, Structural Design Patterns are Design Patterns that ease the design by identifying a simple way to realize relationships between entities.





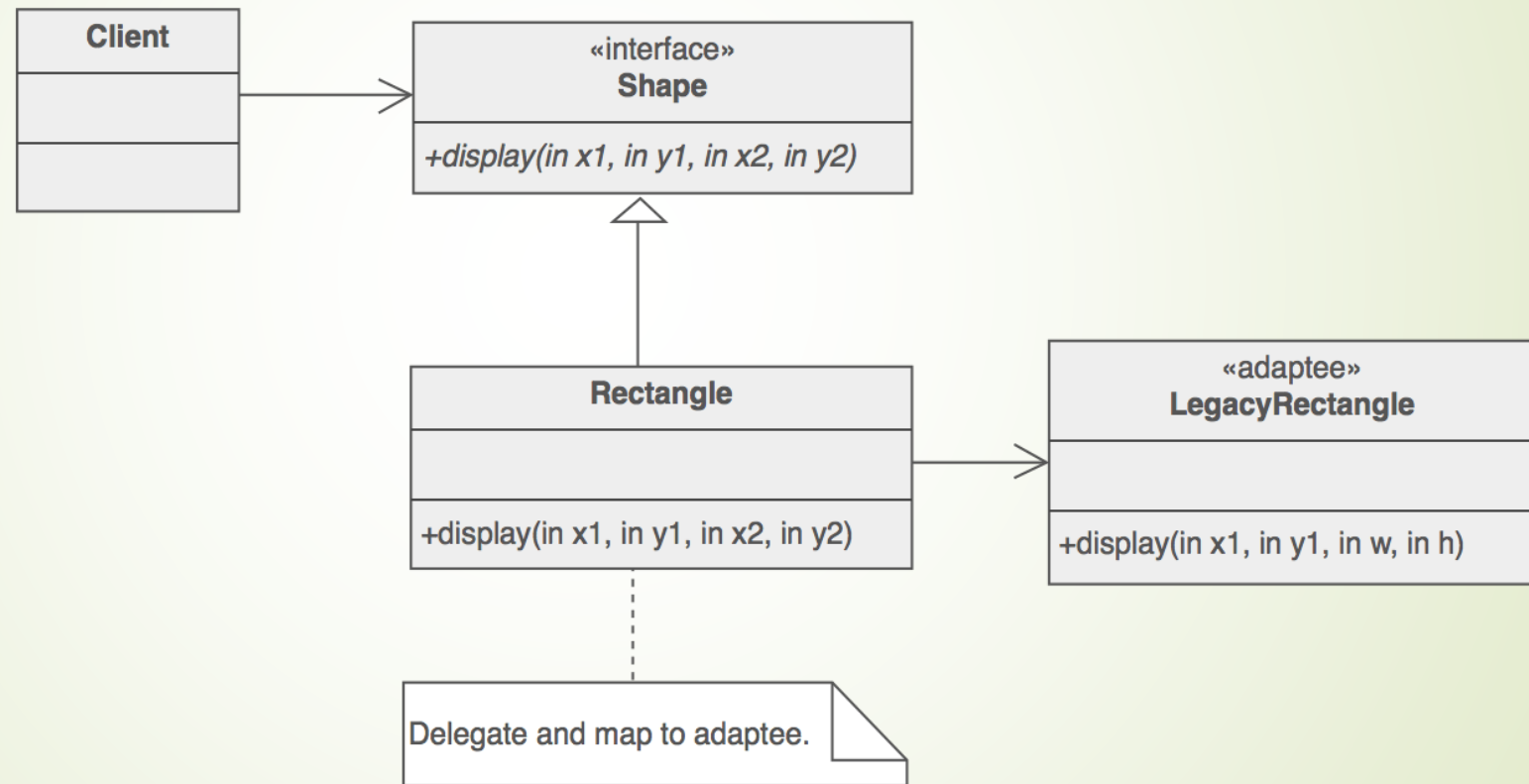
Structural Patterns



Adapter



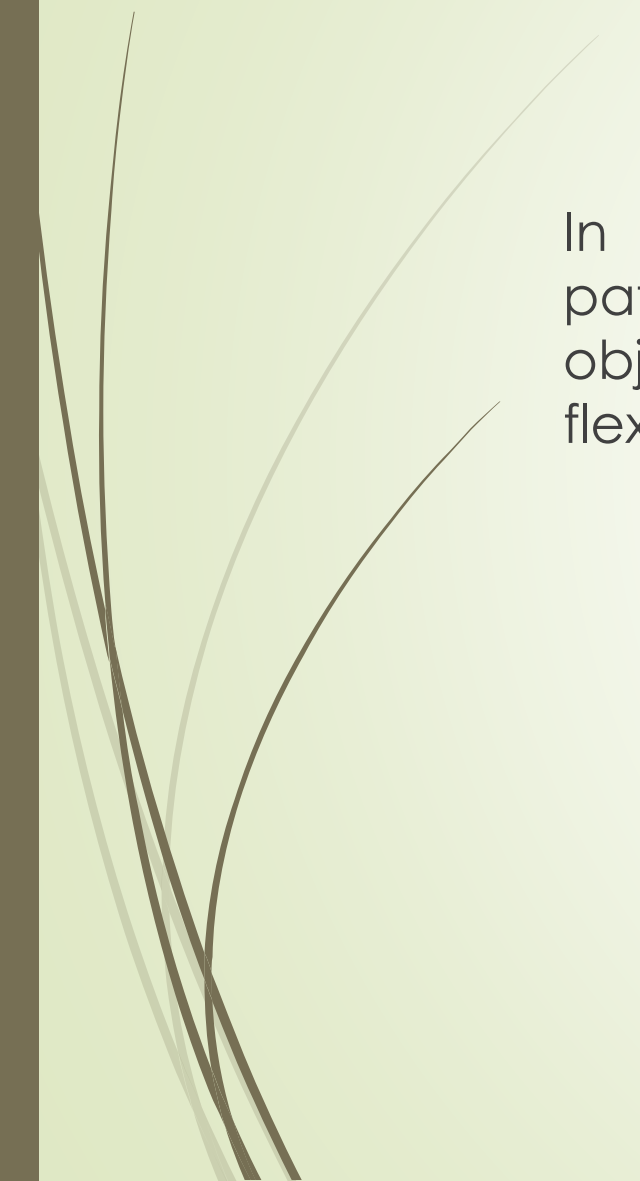
Adapter Pattern





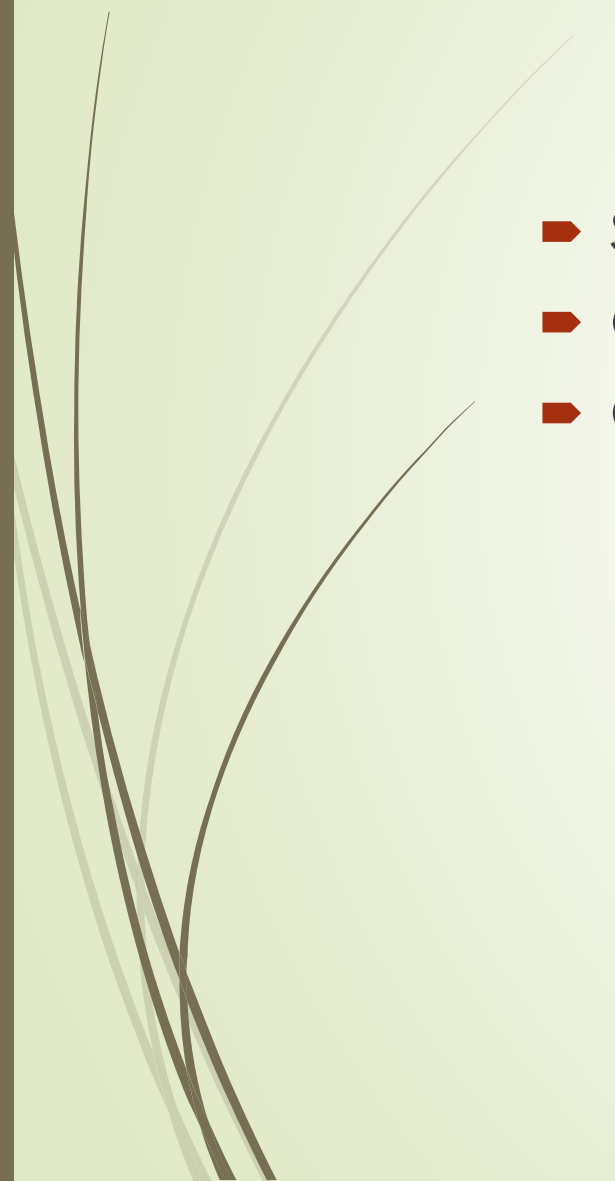
Behavioral Patterns

In software engineering, behavioral design patterns are design patterns that identify common communication patterns between objects and realize these patterns. By doing so, these patterns increase flexibility in carrying out this communication.

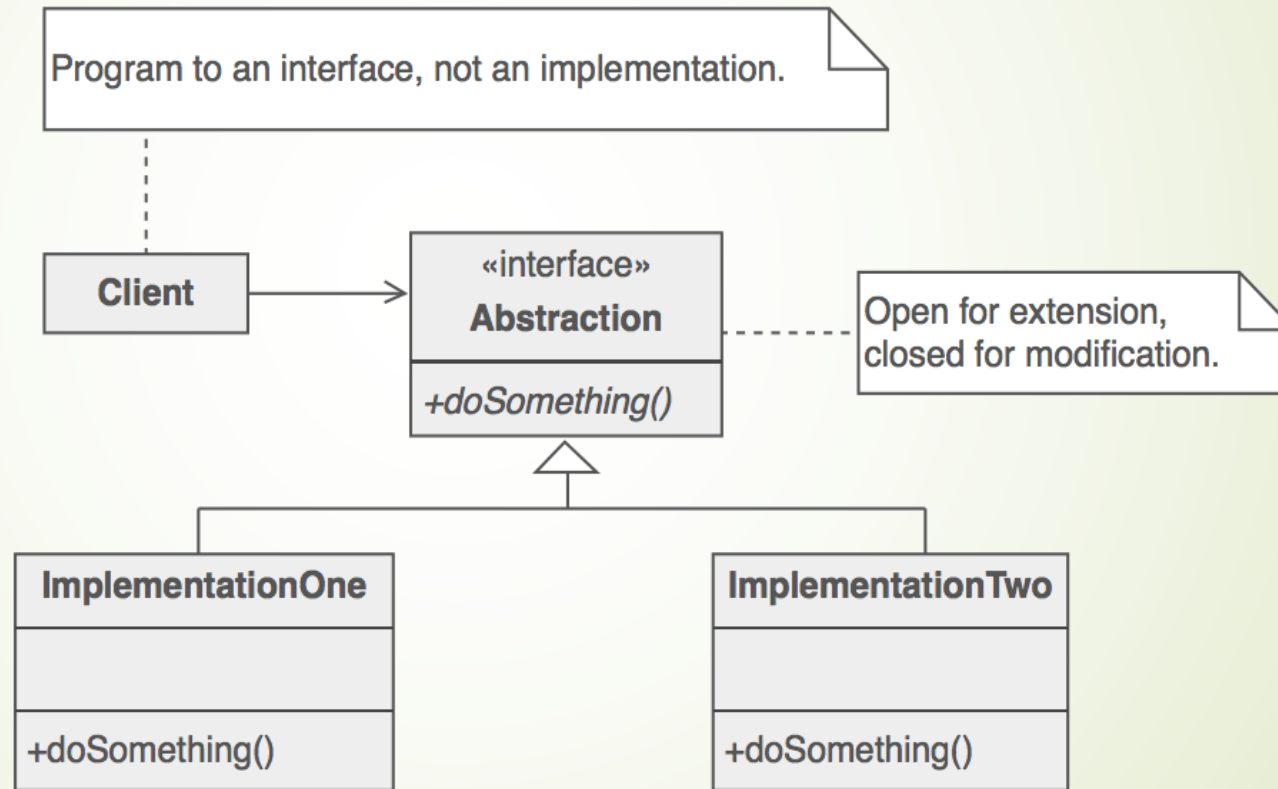




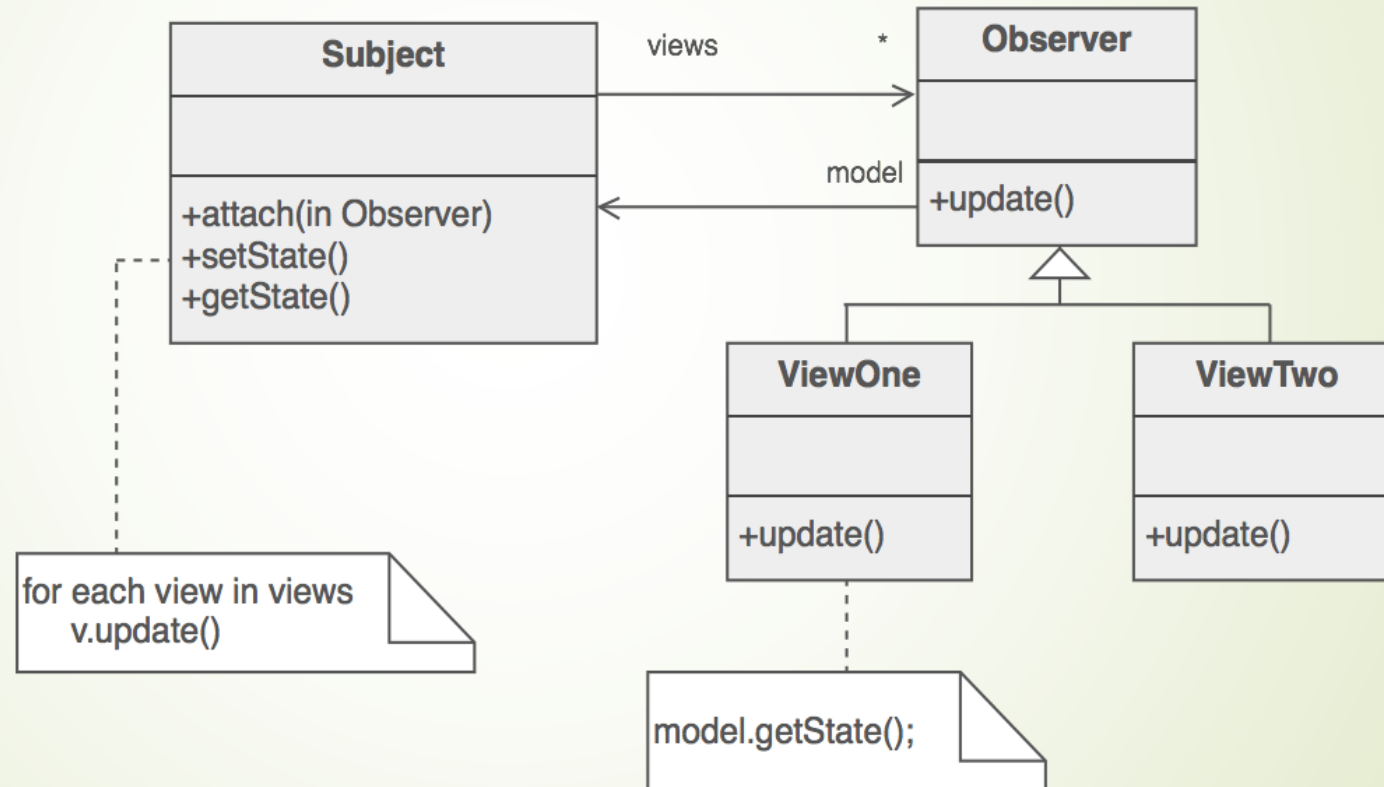
Behavioral Patterns

- Strategy
 - Observer
 - Command
- 

Strategy Pattern



Observer Pattern



Command Pattern

