Basics

* Abstraction, Encapsulation, Polymorphism, Inheritance

Patterns

Strategy

* Defines a family of algorithms, encapsulate each one, and makes them interchangeable.
* Let’s the algorithm vary independently from clients that use it.

Observer

* Defines a one-to-many dependency between objects, so…
* When one object changes state, all its dependents are notified and updated automatically.

Singleton

* Ensure a class has only one instance and provide global point of access to it.

Principles

* Encapsulate what varies.
* Favor composition over inheritance.
* Program to interfaces, not implementations.
* Strive for loosely coupled designs between objects that interact.
* Classes should be open for extension but closed for modification.
* Depend on abstractions. Do not depend on concrete classes.