Head First Design Patterns

Chapter One - Introduction to Design Patterns

When designing software, bear in mind some of the following:

- Identify what stays the same and what varies
  - Encapsulate whatever stays the same
- Program to an interface
  - Define common methods in superclass and then implement these individually within the class
- Favour composition over inheritance
  - Understand when to use a HAS-A relationship instead of an IS-A relationship
  - Construct objects from different classes over getting them to inherit

Also, remember The Strategy Pattern Design Principle which is where a family of algorithms is encapsulated and can be used interchangeably

Chapter Two - The Observer Pattern

The observer pattern is analogous to people subscribing to a newsletter - we have a series of objects keeping tabs on any changes for a particular subject

- We call the publisher of any new information the **subject**
- We call the subscriber of any new information the **observer**
- A more formal definition of the observer pattern is a **one-to-many dependency between objects so that when one object changes state, all dependents are also notified**

Java has a built-in API for creating observers/observables which has some drawbacks:

- Observable has many crucial methods protected so you need to always subclass

- Observable is a class so its very hard to add more methods underneath