0-to-hero

06/10 Mentors <> 06/10 Sessions

Zeynal Zeynalov

Zetes - Panasonic
Data Technologies
Nivel Solutions - Cready
Azercell
ASBank
Bites

Middle East Technical University – CS (bachelor of science)

linkedin.com/in/zeynal

Software Design Patterns

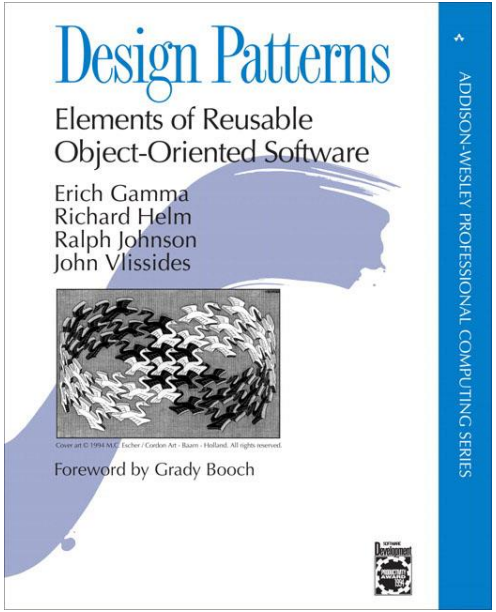
A design pattern is a practical proven solution to a recurring design problem.

In software engineering, a software design pattern is a general, reusable solution to a commonly occurring problem within a given context in software design.[1]

Why should I learn patterns?

- tried and tested solutions
- common language between team members[2]

History - GoF



Gang of Four (GoF)

Gang of Four's pattern catalogue

23 patterns

Pattern categories:

- Creational – object creation methods
- Structural – integration of objects to large structures
- Behavioral – efficient interactions between objects

GoF – Creational patterns

Singleton

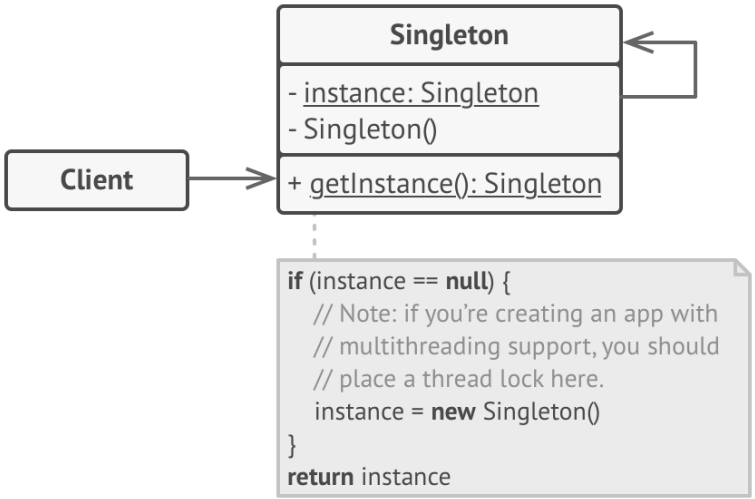
Factory method

Abstract factory

Builder

Prototype

GoF – Creational patterns - Singleton




```java
package com.ab;

public class Singleton {

private static Singleton singletonObject;

    /** A private Constructor prevents any other class from instantiating. */
    private Singleton(){
          //     Optional Code
    }

    public static synchronized Singleton getSingletonObject()
    {
        if (singletonObject == null){
            singletonObject = new Singleton();
        }
        return singletonObject;
    }

    public Object clone()throws CloneNotSupportedException
    {
        throw new CloneNotSupportedException();
    }
}
```

GoF – Structural patterns

Adapter

Composite

Decorator

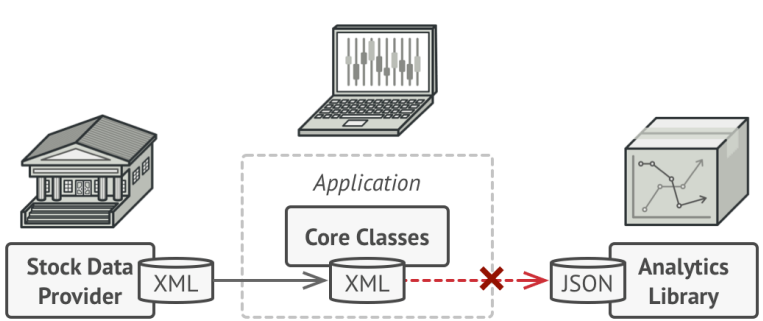
Facade

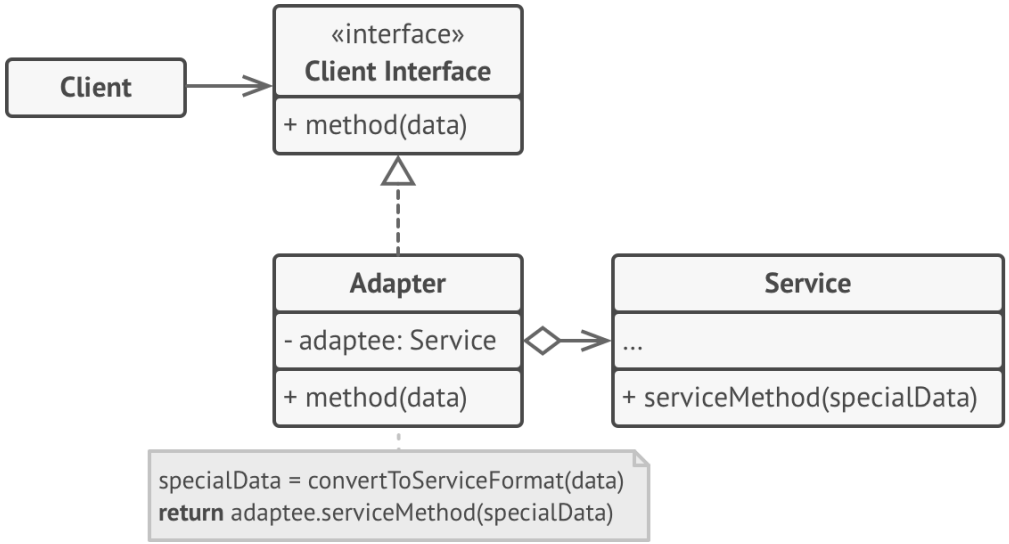
Bridge

Proxy

Flyweight

GoF – Structural patterns - Adapter





GoF – Behavioral patterns

Chain of responsibility

Command

Iterator

Mediator

Memento

Observer

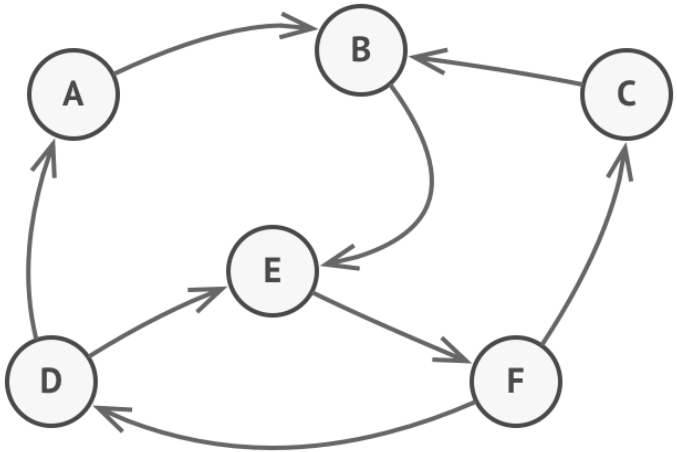
State

Strategy

Template method

Visitor

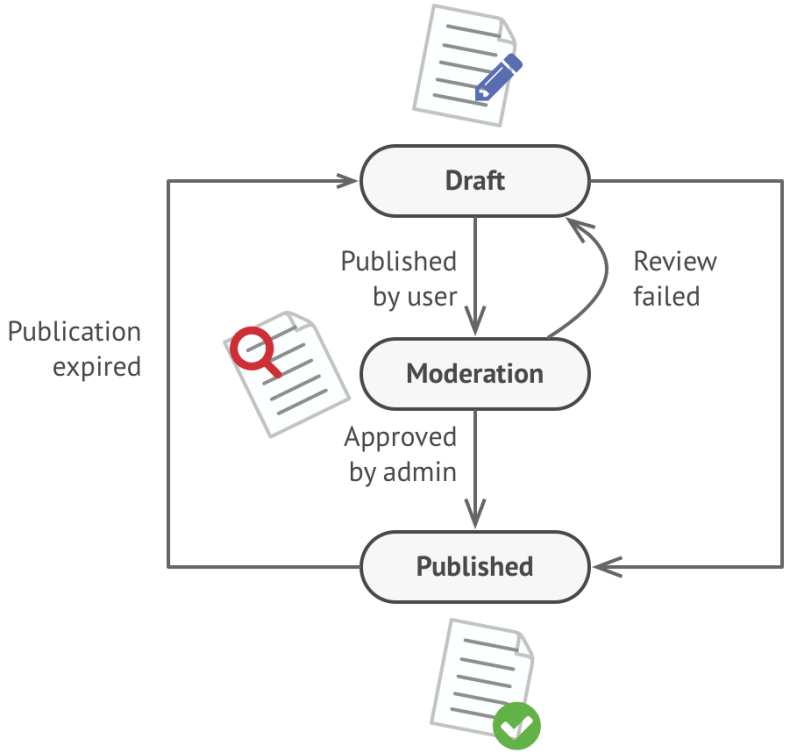
GoF – Behavioral patterns - State

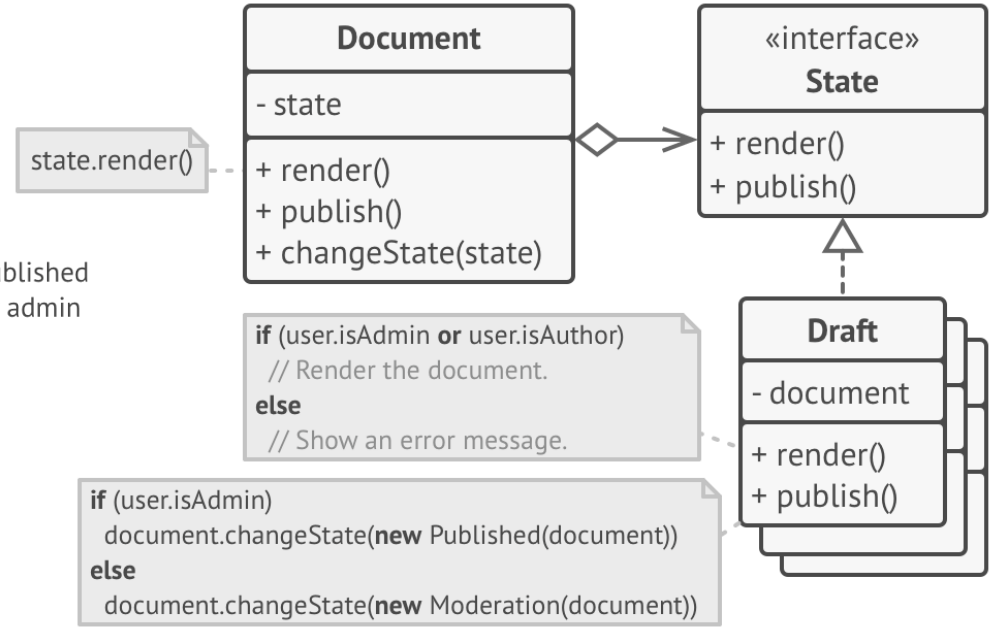


GoF – Behavioral patterns - State

```
class Document is
    field state: string
    // ...
    method publish() is
        switch (state)
            "draft":
                state = "moderation"
                break
            "moderation":
                if (currentUser.role == 'admin')
                    state = "published"
                break
            "published":
                // Do nothing.
                break
    // ...
```

GoF – Behavioral patterns - State





Thanks!

– – –

References:

[1] https://en.wikipedia.org/wiki/Software_design_pattern
[2] https://refactoring.guru/design-patterns/why-learn-patterns
https://refactoring.guru/design-patterns/singleton
https://springframework.guru/gang-of-four-design-patterns/
https://www.flickr.com/photos/10591680@N07/1499817187