Facade

Provide a unified interface to a set of interfaces in a sub-system. Facade defines a higher-level interface that makes the sub-system easier to use

Facade

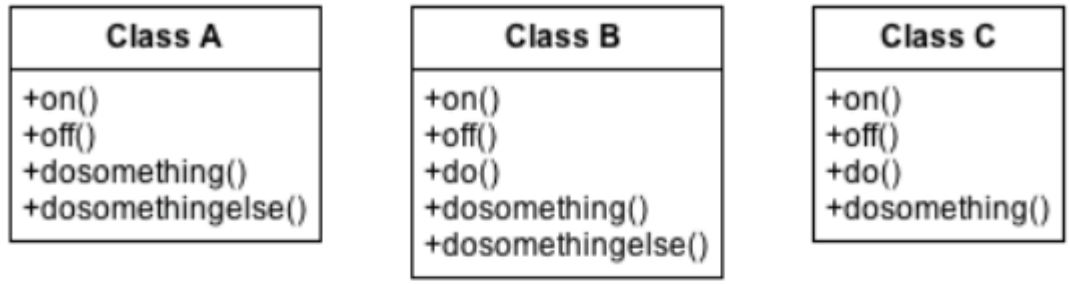
- ★ Structural Patterns
  - » strategy
  - » adapter
  - » Façade
- » Behavioral Patterns
  - » observer
  - » decorator
  - » command

- ★ Creational Patterns
  - » factory method
  - » abstract factory
  - » singleton

Problem

Lots of different interfaces and too much low level complexity

Example



turn system on

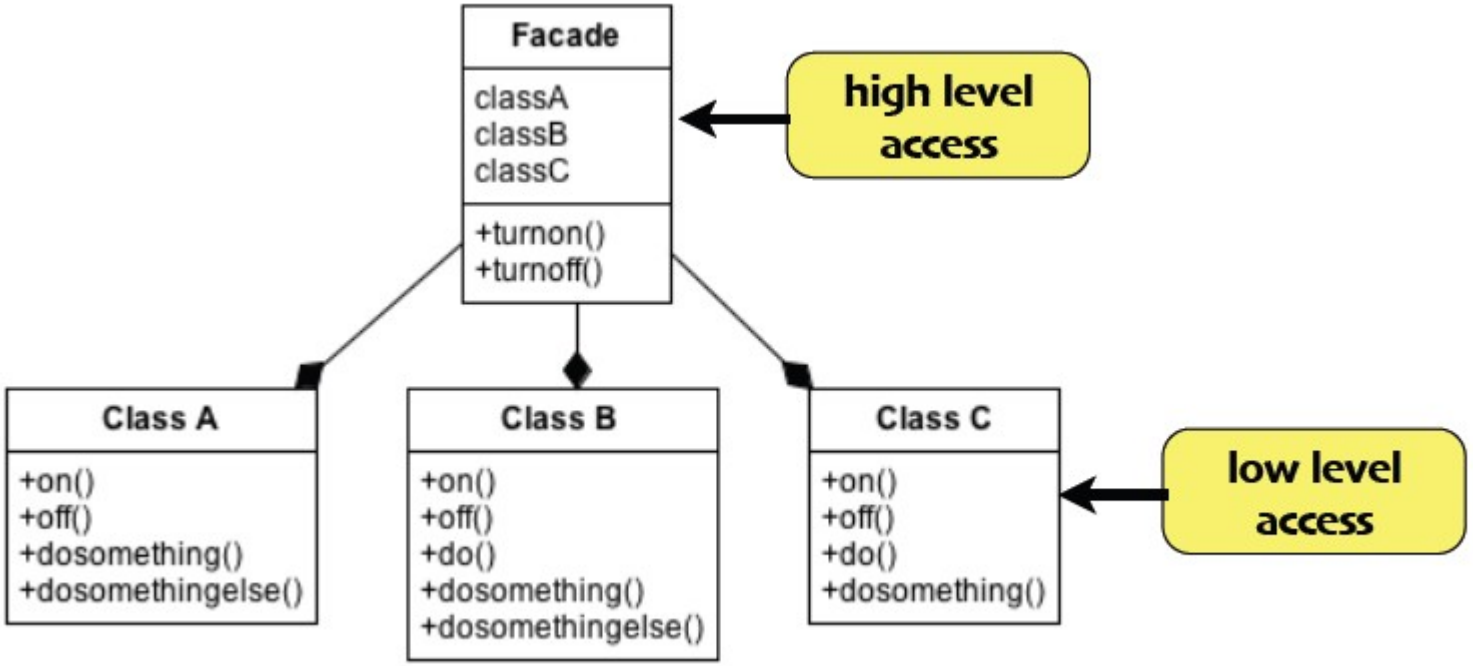
```
ClassA.on();
ClassA.dosomething();
ClassA.dosomethingElse();
ClassB.on();
ClassB.do();
ClassB.dosomething();
ClassC.on();
ClassC.do();
```

if this sequence is the same all the time?

turn system off

```
ClassA.off();
ClassB.dosomethingelse();
ClassB.off();
ClassC.dosomething();
ClassC.off();
```

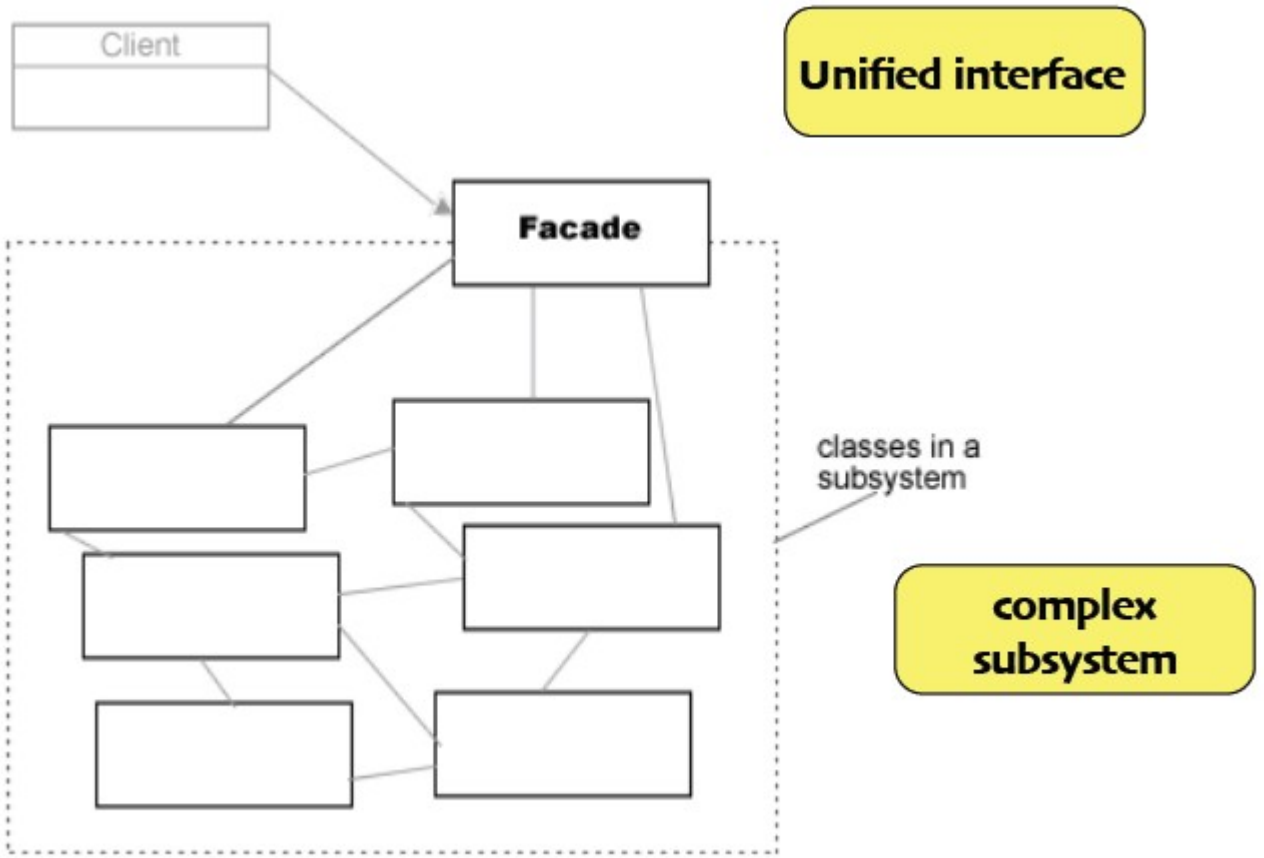
Build a simpler interface



Code

```
public class Facade {
    ClassA a;
    ClassB b;
    ClassC c;
    public Facade (ClassA a, ClassB b, ClassC c) {
        this.a=a;
        this.b=b;
        this.c=c;
    }
    public void TurnOn() {
        a.on();
        a.dosomething();
        a.dosomethingElse();
        b.on();
        b.do();
        b.dosomething();
        c.on();
        c.do();
    }
}
```

Class Diagram



Facade Intent

Simplify the interface the client needs to work with to communicate with complex sub-systems

Design Principle

Principle of Least Knowledge: Talk only to your immediate friends
>Law of Demeter<

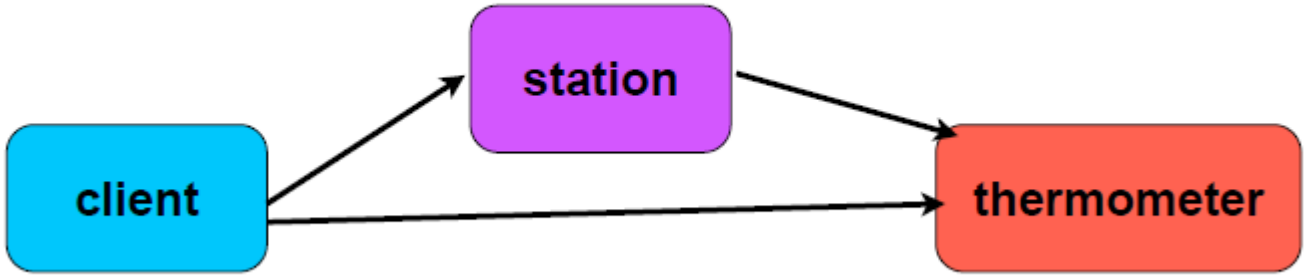
Law of Demeter

From Wikipedia:

The Law of Demeter (LoD) or Principle of Least Knowledge is a design guideline for developing software, particularly object-oriented programs. In its general form, the LoD is a specific case of loose coupling. The guideline was invented at Northeastern University towards the end of 1987, and can be succinctly summarized in one of the following ways:
--Each unit should have only limited knowledge about other units: only units "closely" related to the current unit.
--Each unit should only talk to its friends; don't talk to strangers.
--Only talk to your immediate friends.

Breaks Law of Demeter

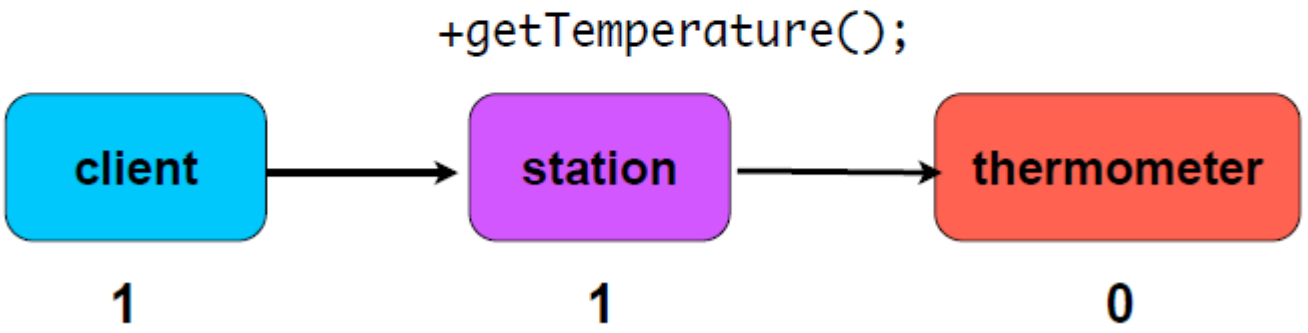
```
public float getTemp() {
    return station.getThermometer().getTemperature();
}
```



2

Follows Law of Demeter

```
public float getTemp() {
    return station.getTemperature();
}
```



Facade

- Helps us avoid spaghetti code
- Allows us to program to an interface, not an implementation
- Keeps client blissfully unaware of the underlying 'nastiness'
- To make it work, the underlying sub-system needs to contain some commonality so those behaviors can be abstracted