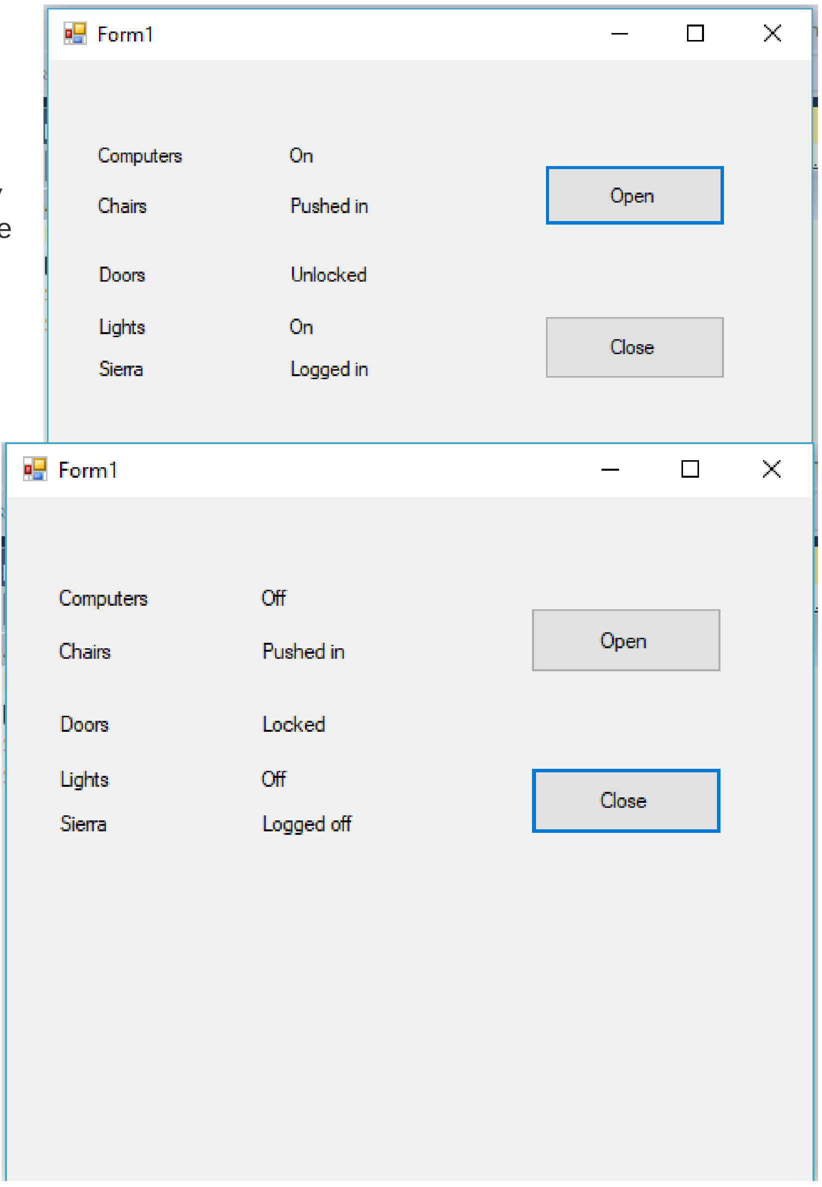
Annalise Dake Design Patterns 9/17/17

# Façade Pattern

Introduction: I was asked to write a demo application that implemented the façade design pattern. The façade design pattern provides a unified interface to a set of interfaces in a subsystem. Façade defines a higher-level interface that makes the subsystem easier to use.

Narrative: For my demo, I wrote a small application that is sort of a ‘remote’ for opening and closing the library, where I work. Assuming that all of these things could eventually be done remotely with the use of smart home technology and the such, my app tells what things should be done when opening the library such as, unlocking the doors and turning on the computers. My façade class contains two methods, one called open that calls all of the things necessary to open the library, and one called close that calls all of the methods necessary to close the library. These methods are controlled with a boolean, since the library is always either open (true) or closed (false).



| | |
|---|---|
| Computers | On |
| Chairs | Pushed in |
| Doors | Unlocked |
| Lights | On |
| Sierra | Logged in |

| | |
|---|---|
| Computers | Off |
| Chairs | Pushed in |
| Doors | Locked |
| Lights | Off |
| Sierra | Logged off |

```
        bool open;

        public void Open()
        {
            open = true;
            getLights(open);
            getChairs(open);
            doorsLocked(open);
            getDoorsOpen(open);
            sierraOn(open);
            computersOn(open);

        }

        public void Close()
        {
            open = false;
            getLights(open);
            getChairs(open);
            doorsLocked(open);
            getDoorsOpen(open);
            sierraOn(open);
            computersOn(open);

        }
```

Conclusion: I think that this design pattern was very conceptually similar to what I’ve been learning since we started OOD in Programming 2. The concept of making methods that call multiple other methods in the same order because you’re going to need to call them in that order multiple times is pretty cool and it really simplifies things.