Facade
Facade Intent

Provide a unified interface to a set of interfaces in a subsystem. Façade defines a higher-level interface that makes the subsystem easier to use.

(Structural)
What is a “facade”? 

- From Merriam-Webster Online
  - 1: the front of a building; also: any face of a building given special architectural treatment <a museum's east facade>
  - 2: a false, superficial, or artificial appearance or effect
Clients often need detailed knowledge of a subsystem to obtain the needed services.
A Façade can make it easier for clients to obtain the services that are needed.
There are two characteristics that must be present to have a true Façade.

- **Unified interface**
  - *The clients know of one interface to call for services*
  - *Reduces coupling to the subsystem, but this is probably not the biggest benefit*

- **Higher-level interface**
  - *Each service must provide more than a client can get from a simple call to a subsystem object*
  - *Single-call "pass-through" functions are not a higher-level interface*

```c
facadesFunction() {
    subsystemObject->aFunction()
}
```
Façades tend to quickly become bloated classes.

To eliminate this problem, there can be multiple types of Facades, each dealing with a logically related set of services that the subsystem provides to clients.
There are some interesting questions about the Façade pattern.

- **Public or Private Subsystem**
  - *Should a Façade be the only interface to a subsystem?*
  - *Should the brave (or foolish) be able to call directly into the subsystem?*

- What are the similarities between Façade and Mediator with respect to a subsystem?