Facade

GoF Structural Pattern
Interface pattern

Facade intent and problem

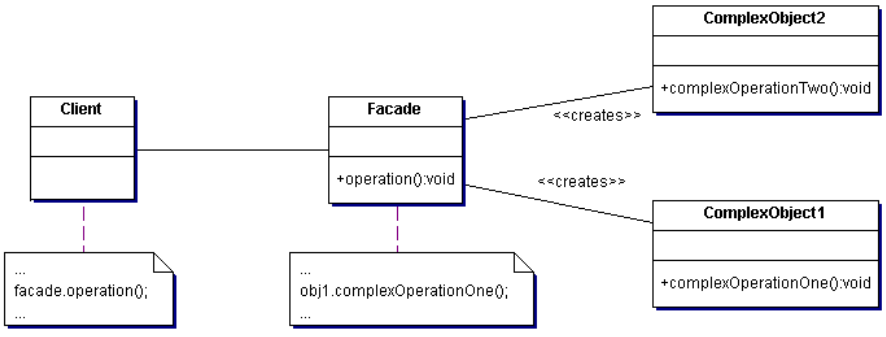
- **Intent**
  - Provide a unified interface to a set of interfaces in a subsystem.
  - Facade defines a higher-level interface that makes the subsystem easier to use.
- **Problem**
  - A segment of the client community needs a simplified interface to the overall functionality of a complex subsystem.
  - There is a subset of the full functionality that is relevant to a segment of the client community.

Motivation, discussion

Discussion

- Facade discusses encapsulating a complex subsystem within a single interface object.
- This reduces the learning curve necessary to successfully leverage the subsystem.
- It also promotes decoupling the subsystem from its potentially many clients.
- On the other hand, if the Facade is the only access point for the subsystem, it will limit the features and flexibility that "power users" may need.
- The Facade object should be a fairly simple advocate or facilitator. It should not become an all-knowing oracle or "god" object.

Structure



Variations of Facade

- Facades can be used not only to create a simpler interface in terms of method calls, but also to reduce then number of objects that a client object must deal with.
- A façade can supplement existing function of the subsystem with new routines.
- The motivation for a façade may be to hide or encapsulate the subsystem, e.g. in order to
  - Track system usage by forcing all acces to the system to go through the Façade.
  - Swap out systems. By making the original subsystem a private member of the façade, it can be changed in the future with minimal effort. Changes are focused only in the façade class.