Façade Pattern

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

- A facade is a class that provides a simple interface to a complex subsystem which contains lots of moving parts. A facade might provide limited functionality in comparison to working with the subsystem directly. However, it includes only those features that clients really care about. [Source: refactoring.guru]
    - Having a facade is handy when you need to integrate your app with a sophisticated library that has dozens of features, but you just need a tiny bit of its functionality.
- Façade doesn't encapsulate a subsystem. It simply gives you a layer of convenience on a subsystem. You can bypass the façade and go directly to what is underlying.
- A Façade is not hiding what's below. It's simply making it easier to use what's below.
- Example that would benefit from the Façade Pattern
    - A hi-tech mailer subsystem has several classes to specify the destination of mail, routing preferences (shortest, most reliable, most economical, etc.) and delivery options (like urgent, important, confidential, encrypted) and return notification options.
    - While some clients of the subsystem may be interested in these finer classes, other may just be interested in a subset that provides basic functionality.

When to use Façade Pattern?

- When you want to provide simple interface to a complex subsystem.
- Clients that need no customised features do not look beyond Façade.
- You want to decouple subsystems from clients.

- You want to layer the subsystem to reduce dependency between various levels of the subsystem.

Consequences of using Façade

- Shields clients from subsystem components.
- Makes subsystem easier to use.
- Promotes weak coupling between client and subsystem.
- Helps layer subsystem.
- Reduced coupling helps build time for large systems.
- No limit on clients looking beyond Façade.

Façade vs Other Patterns

- Abstract Factory may be used with Façade.
- Interface for creating subsystem objects in subsystem independent way.
- Abstract Factory may be an alternative to Façade to hide platform-specific classes.
- Mediator similar to Façade.
- In Mediator, colleague objects know Mediator.
- In Façade subsystem classes do not see Façade.
- Façades usually implemented as Singletons.
- Façades typically are stateless.