Facade: Exposing several components through a single interface

1. Balancing complexity and presentation/usability
2. Example:Typical home
   1. Many subsystems (electrical, sanitation)
   2. Complex internal structure (e.g., floor layers)
   3. End user is not exposed to internals
3. Same with software
   1. Many systems work to provide flexibility, but…
   2. API consumers want it to “just work”
4. Facade: Provides a simple, easy to understand/user interface over a large and sophisticated body of code.

Facade Design Pattern

1. User wants a simple apı to use.
2. Example: We have a console class that takes a viewPort class in constructor or something. And viewport takes Buffer class in its constructor. To create console you need to first create buffer and than viewport.
3. Instead of making the user do this have a static method that does this for the user in the console class. (public static Console newConsole(...))

Summary

1. Build a Facade to provide a simplified API over a set of classes
2. May wish to (optionally) expose internals through the facade – to give the user more control over the library for example.
3. May allow users to escalate to use more complex APIs if they need to.