CSE 1325: Object-Oriented Programming

From Lecture 21

Patterns Review

Mr. George F. Rice
george.rice@uta.edu

Based on material by Bjarne Stroustrup
www.stroustrup.com/Programming

ERB 402
Office Hours:
Tuesday Thursday 11 - 12
Or by appointment

Applying Patterns

- Our Digital Signal Processing (DSP) application is designed to receive and process data from a QC-107 quadcopter drone.
- However, we could only afford the QC-53½ quadcopter drone, which has a different interface library not supported by our DSP application.

- This problem is a good candidate for the **Adapter** pattern.

Applying Patterns

- Our company is expanding to 10 different sites around the world, each with its own local computing resources.
- When an employee seeks to launch an instance of a tool, we want to ensure that the instance is on computing resources local to that employee.
- This problem is a good candidate for the **Factory** pattern.

Applying Patterns

- Your team is developing a next-generation big data analysis tool that will revolutionize your industry, crush your competition, and possibly result in a pay raise.
- However, the tool's algorithms are dependent on the Goggles data reduction library, which is complex and difficult to learn and use. Only one team member has experience with this library.

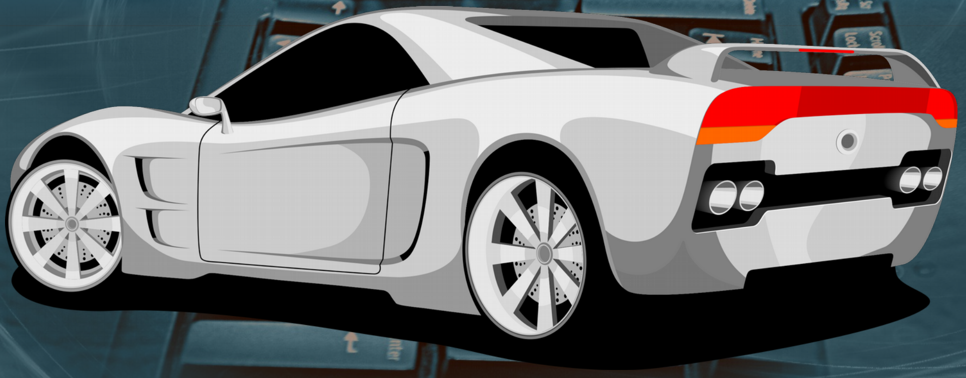
- The tool needs only a small subset of the Goggles library functionality.
- This problem is a good candidate for the **Façade** pattern.

Applying Patterns

- After 42 sprints, your team is ready to deploy your new iPhone app to your eager worldwide fan base.
- To ensure your fan base doesn't become your plaintiffs, you want to collect data about app crashes and operational problems so you can proactively address any issues. One approach is for your data collection server to register to be notified when an installed app detects any anomalies.
- This problem is a good candidate for the **Observer** pattern.

Applying Patterns

- Your new self-driving car software will dominate the coming auto-Uber market by steering each wheel *individually* for optimal maneuvering.
- A central controller program determines the optimal angle of each wheel. However, it's rather important that only one central controller coordinate all 4 wheels, otherwise Bad Things might happen.
- This problem is a good candidate for the **Singleton** pattern.



Applying Patterns

- Your Internet of Things (IoT) application will build an ad hoc network out of the outlets in a building. Your product backlog specifies that your application must try to establish the network via wifi, bluetooth, power line net, infrared, and audio comm technologies, and then use the one that works best in that specific building.
- This problem is a good candidate for the **Strategy** pattern.

Applying Patterns

- Which pattern(s) weren't covered?
- Give an example scenario where each would be appropriate to resolve design issues.
  - **Decorator** – We want to make certain methods in our class available via a web interface. (Flask is an example of the Decorator pattern for this purpose.)
  - **State Design** – We are writing an elevator controller, and want to implement state-based behavior for the floor selection logic.
  - **Model View Controller** – We are writing a Linux tool to backup our data to the cloud that offers both a command line and graphical user interface.