Design Patterns cont.

To date...
- Singleton
- Facade
- Bridge
- Adapter
- Composite

Today
- Strategy
- State
- Command

Problem
I am building a physics engine that performs collision detection between a sphere and some triangles. My physics engine class contains the following method:

```
cCollision cPhysicsEngine::detectCollision(cPath p, cTriangles t)
```
Later I may want to optimize my collision detection algorithm so that I can avoid testing the sphere against every triangle if the scene is large. Come up with a design that will allow for future changes in my collision detection algorithm.

Strategy Design Pattern

What difference (if any) is there between the bridge and the strategy design pattern?
**Design Principle**

Encapsulate variability

**Problem**

- I am building a drawing program. The user enters keystrokes to change modes (Add, Delete, Move) and mouse input that is interpreted based on the current mode.
- Currently I use some global variables to record state information and long switch statements in my mouse and keyboard functions to process input. What is wrong with this picture?
- Come up with a better design.

**Design Principles**

Encapsulate variability.

Information expert: Responsibilities should be assigned to the class that has the information to handle the responsibility.

**Problem continued**

- I also want to support “Undo”
- Help!