Designing Interactive Systems II

Assignment 3 "Window System (Part 2/3):
Now You See Me, Now You Don't"

Submitted by: group 10 (Vitalii Isaenko 383303, Ahmed Soliman 350055, Anastasiia Belova 383305)

3. Testing Your Understanding

1. WindowSystem class contains a collection of SimpleWindows (WindowComponent in our case - decorated SimpleWindows). It also contains a WindowManager to use - for adding behaviour to windows and visual decorating. WindowSystem has responsibilities to provide mechanisms to handle input by handling events, handle output by drawing any graphics and manage windows by adding, removing, assigning them IDs. This way, WindowSystem provides essential functionality which it should and does not contain any behaviour that WindowManager is responsible for.
2. Window manager contains a reference to WindowSystem that it is used for. It uses its functionality to work with windows. It has a method to add simple windows that are to be decorated by WindowManager. It will add all additional behaviors and dressing to windows. Any particular window manager would change this behaviour by implementing its own version of adding windows. We created interface IWindowManager to exchange easily with another WindowManager for WindowSystem. This why we can use different WindowManagers and change it any time. Window manager itself does not perform adding elements (like close button) inside its methods, but instead simply appends decorators to simple window that have behaviour to draw

4. Expert Question

3. themselves, react to mouse clicks, and drag&drop events. For this we used decorator design pattern where each component associated with behaviour is a separate object deriving from specific decorator abstract class (for example, we have TitleBarDecorator and a specific TitleBar that our window manager appends during window creating. From an extra features we chose to implement Minimizing a window to an icon and Window reordering.
   Implementation of new features was relatively easy to achieve using builded architecture. We managed to add minimization by adding new decorator -

MinimizeDecorator and concrete Minimize class deriving from it. This class has behaviour as any other WindowComponent to draw itself and react to mouse events. So we implemented drawing of button on the window and drawing minimized icon by overriding draw() method and added behavoiur to actually minimize window by reacting to mouse click event in react(Point clickedPoint) method.

Implementation of making a window active also fitted out design. We overrided react method for handling mouse click in existing TitleBar class to reorder window by clicking at the bar.