

CS456 Homework Assignment 6

Due date: _____

In the real world, your ideas and your work must be communicated effectively to your boss, to your client, and to potential investors. Therefore, **please use clear and well-written English for your answers (as if I was your boss). Incidentally, your boss expects typewritten reports, and so do I.** Drawings should be neatly done with an appropriate software program.

General Instructions: For problems #1 and #2, download and look at the graphics code at <http://academic.regis.edu/dbahr/GeneralPages/ArtificialLife/HomeworkAssignments/HomeworksPage.htm>. In particular, look at the ExampleGraphicsDriver, ALifeWorldView, and StandardWorldView. The ALifeWorldView is a Façade because it provides a simplified interface to the more complex StandardWorldView. (You might argue that it is also an Adapter because it composed of the StandardWorldView; but it doesn't really change or adapt anything. Composition doesn't always mean the Adapter Pattern.)

Problem #1: Using the Façade Pattern, create another class called CritterWorldView that is a façade of the ALifeWorldView. Your façade should *only* allow the user to display (and remove) “critters”. The user should not be able to display walls or food.

Hint: Be sure to run and test your code. How do you run it? Read the code comments, and see the ExampleGraphicsDriver. Does your code really work? It should!

Problem #2: Using the Façade *and* Adapter Patterns, create another class called SoccerWorldView. Your class should be similar to the ALifeWorldView class, but should display critters, food, and soccer balls (instead of walls). It should not display anything else. Like the ALifeWorldView class, the user should not be able to change the images (i.e., default critter, food, and soccer ball images will be used).

Hint: You should adapt the StandardWorldView class. Like the ALifeWorldView class, you should include the ability to “redisplay” and “remove images”, but you should have **no** other functionality (that's the façade part). If you prefer, you may

use two classes, one for the adapter, and a second for the façade. Be sure to run and test your code. How do you run it? Read the code comments, and see the ExampleGraphicsDriver. Does your code really work? It should!

General Instructions: For problems #3 and #4, consider the following business requirement and attached code. Note that the code has already been written, and this is a change in requirements. Therefore, this code has already been tested by QA, and they are not going to be happy if they have to retest the code because of any changes you might make. To minimize company costs (and avoid making enemies in the QA department) it is best satisfy the new requirements without rewriting the pre-existing code. That way, QA only has to test your new code.

Problem #3: Examine the following code for low cohesion. Where is it? Justify your answer.

Problem #4: Write a design document that satisfies the business requirement (don't bother writing the functional specification for this homework). As part of the design, specify any design patterns that are used. Also provide any useful class and interaction diagrams.

As always, do not do anything outside the scope of the business requirement. E.g., do not attempt to fix the low cohesion unless it satisfies the business requirement.

New Interface Requirements Document

As required by the customer Plop, Splash, and Sons, Inc., the Poseidon product will no longer request a user name and password and will permit anyone to use the system at any time. All other features must remain identical.

Scope:

These requirements pertain to version 2.3 of the Poseidon software product.

Detailed Requirements:

This system should be identical to the original system but without a login process. Please remove the login. Note that we expect Plop, Splash, and Sons, Inc. to change their mind next week, but we will cross that bridge when we come to it. Also, our prospective customer, Swish and Co. will probably want the login interface, so please plan on that. To minimize impacts on QA, please minimize changes to existing code. We prefer new code to changes in existing code.

```
public class PoseidonGUIManager
{
    /**
     * Control all GUI functions.
     */
    public String manageGUI()
    {
        //check to see if access permitted
        accessGUI();

        //if get this far, then access permitted
        fishFoodGUI();
    }

    /**
     * Control all GUI functions with the given username and password.
     */
    public String manageGUI(String username, String password)
    {
        //check to see if access permitted
```

```

        accessGUI(username, password);

        //if get this far, then access permitted
        fishFoodGUI();
    }

    /**
     * Run GUI to get username and password and check security.
     */
    private void accessGUI()
    {
        PoseidonSecurity ps = new PoseidonSecurity();
        String username = ps.getUserName();
        String password = ps.getPassword();

        accessGUI(username, password);
    }

    /**
     * Check security with provided username and password..
     */
    private void accessGUI(String username, String password)
    {
        PoseidonSecurity ps = new PoseidonSecurity();
        while(!ps.validUserName(username))
        {
            userName = ps.getUserName();
        }

        while(!ps.validPassword(userName, password))
        {
            password = ps.getPassword();
        }
    }

    /**
     * Primary GUI for Poseidon fish eating product.
     */
    private void fishFoodGUI()
    {
        ... //details are unimportant to this homework
    }
}

```

```
public class PoseidonSecurity
```

```

{
    /**
     * @return A default username that is always valid.
     */
    public String getValidUserName()
    {
        return "phishy";
    }

    /**
     * @return A default password that is always valid.
     */
    public String getValidPassword()
    {
        return "ff266cvbw7";
    }

    /**
     * Get user name from user with GUI components.
     */
    public String getUsername()
    {
        ... //details are unimportant to this homework
    }

    /**
     * Get password from user with GUI components.
     */
    public String getPassword()
    {
        ... //details are unimportant to this homework
    }

    /**
     * @return true if valid user name.
     */
    public boolean validUserName(String name)
    {
        ... //details are unimportant to this homework
    }

    /**
     * @return true if valid password for the given name.
     */
    public boolean validPassword(String name, String pwd)
    {

```

```
} ... //details are unimportant to this homework  
}
```