Gebze Technical University

Computer Engineering

CSE443 - Object Oriented Analysis and Design

Homework 1 - Part 2 Report

Student:
Nevra GÜRSES
161044071

1 INTRODUCTION

1.1 Project Definition

Every people has their own favorite website but nobody wants to check them often if website have been updated with new content.Our project goal is if subscriber subscribe to own favorite website, the websites should notify the subscribers of new content.
The important point in the project is; a subscriber might be interested in only new text updates, or photograph updates or audio updates or a combination thereof. There is no need to disturb them if the update is not of the desired type. We have to design solution that will support this. And also, if users or websites demand your software to support a fourth type of content it have to be easy to modify.This software also must has maximum flexibility, loose coupling and minimum cost of maintenance.

2 METHOD

2.1 Selected Design Pattern and Why That is Selected

I select Observer Design Pattern and I apply this pattern for solving given project.Now, I tell the reasons why I select this pattern for the project.

The observer design pattern is a software design pattern in which an object, called the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods.This is actually the goal of our project also. The Observer Pattern defines a one-to-many dependency between objects so that when one object changes state, all of its dependents are notified and updated automatically. When the state of one object changes, all of its dependents are notified.It ensure that when one object changes state an open-ended number of dependent objects are updated automatically. It is possible that one object can notify an open-ended number of other objects.

If I need to talk about loose coupling, minimum cost and maximum flexibility of the Observer Design Pattern:

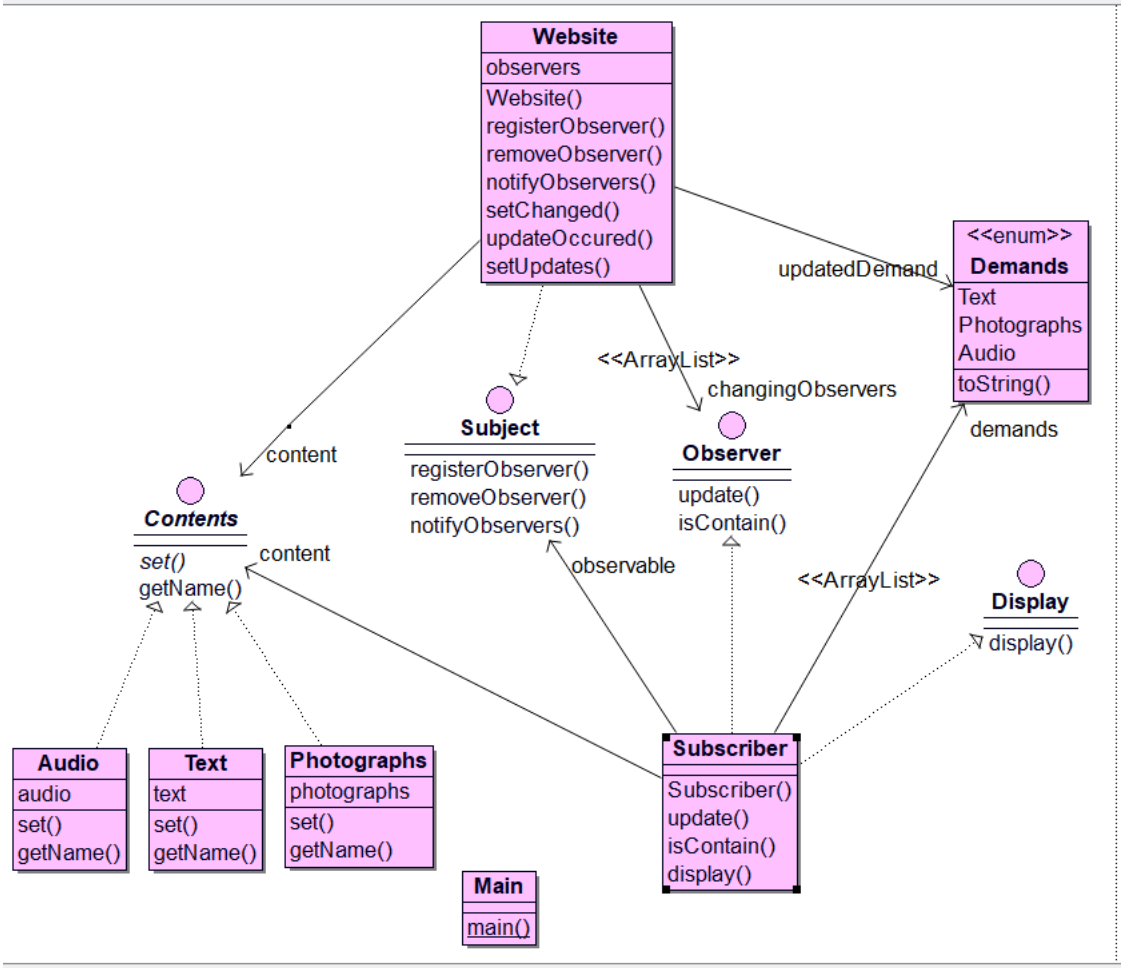
Loose Coupling means that two objects can interact with each other, but they have very little knowledge of each other.The Observer Pattern provides an object design where subjects and observers are loosely coupled. Because the only thing that the subject knows about an observer is that it implements a certain interface that is the Observer interface. It doesn't need to know the concrete class of the observer, what it does, or anything else about it.

The Observer Design Pattern has maximum flexibility because we can add new observers at any time. The only thing the subject depends on is a list of objects that implement the Observer interface, we can add new observers whenever we want.

The Observer Design Pattern has minimum cost because we don't have to need to modify the subject to add new types of observers.We can reuse subjects or observers independently of each other. Changes to either the subject or an observer will not affect the other.Because the two are loosely coupled, we are free to make changes.

I chose the observer design pattern for the project for these reasons, but I made some additions to the Observer design Pattern due to the additional requirement that is there is no need to disturb subscribers if the update is not of the desired type a required in the project. Observer design pattern notify all the observers if any update was occurred.For changing this feature, I made some control conditions and with this conditions I notify subscribers if updated content is desired type for subscriber. I will show how I made this via Class diagram of project.

2.2 Class Diagram and Explanation of Selected Design Pattern with Diagram



- **Demands Class:** This class is an enum class. It keeps content names as enum. This class is used to avoid typos while creating the requested contents.

- **Contents Interface:** This class is an interface. It keeps contents depending on common location.It has 2 methods that are set and getName method. Every concrete content classes implements this methods.Set method provides setting an updating in content when update

occurred.getName method returns content name as enum for example if content is Audio, it returns Audio enum with using Demans enum class.

- **Audio - Text- Photographs Classes:** These classes are concrete Contents classes. These classes implements Contents interface. They implement set and getName methods.Set method provides setting an updating in content when update occurred.getName method returns content name as enum for example if content is Audio, it returns Audio enum with using Demans enum class.

- **Display Interface:** This is an interface.It has a method that is display. Every subscriber implements this method for display update setting results.

- **Subject Interface:** This interface is very important for Observer Design Pattern.Objects use this interface to register as observers and also to remove themselves from being observers.notifyObservers method is using for notifying specific observers according to updated content. 3 method is implementing in concrete Subject class that is name Website.

- **Website Class:** This class is a concrete subject class that always implements the Subject interface. In addition to the register and remove methods, the concrete subject implements a notifyObservers() method that is used to update specific observers according to updated content.Now I will explain how I provide notifying specific observers and don't disturb observers if the updated content is not demanding content.

  - **setUpdates Method:** This method takes 2 parameter that are Contents class type and new values of it as Object type.In Website class I provide Contents interface association with instance variable that is name content.In setUpdates method, I assign content type in parameter in instance content variable. After that I call set method of Contents class with dynamic binding with new value of content in parameter that is Object type, so updating is performing.After that I call updateOccured method.

- **updateOccured Method:** This method only calls 2 another methods in Website class that are setChanged and notifyObservers method.So I will explain these methods.
- **setChanged Method:** This is important method.In this method, I controlling all observers with loop and if updated content type is desired type for current observer, I keep that observer in Arraylist that is name changingObservers.I use isContain method of Observer class for controlling observers.After that, in notifyObservers method, I send notifies for these observers.
- **notifyObservers Method:** In this method, I call update methods of specific observers that are keeping in arraylist with content type and new values of it, so updating information is reaching that observers.

- **Observer Interface:** This is an interface.All potential observers need to implement the Observer interface. This interface just has 2 method, one of them that is name update, that gets called when the Subject's state changes.And isContain method that is using for controlling whether a current updated content is desiring for observer or not.

- **Subscriber Class:** This is concrete observer class that implements the Observer interface and Display interface. Each observer registers with a concrete subject to receive updates. In Subscriber class there is a association with Demands enum class for keeping desired content names for observer in arraylist. isContain method is implementing Subscriber Class for controlling whether a current updated content is desiring for observer or not.Update method using for setting updating for content. And display method is implementing for seeing new values of updated content.

- **Main Class:** For testing and working project.

  NOTE: According to my design, if there will be fourth type of content, modifying will be very easy.Only thing that I will make, I create new content class that implements Contents interface.And also adding content name in Demands enum class.

3 SAMPLE RUNNING RESULT:

Explanation of this output: There are 2 subscribers; one of them desiring text and photograph contents, other one desiring audio and photograph contents.Firstly, an update coming for text content and after this updating, subject class notifies only first subscriber that is desiring text content and so update function of first subscriber is working.Secondly, an update coming for photograph content and after this updating, subject class is notifying two subscribers because two of them desiring photograph content and so update function of two subscriber is working.Thirdly, an update coming for audio content and after this updating, subject class notifies only second subscriber that is desiring audio content and so update function of second subscriber is working. So, there is no disturbing for subscriber if updating content is not desired type for that subscriber.