МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Полоцкий государственный университет»

Факультет информационных технологий

Кафедра технологий программирования

**Лабораторная работа №5**

по дисциплине: «Объектно-ориентированные технологии

программирования и стандарты проектирования»

на тему: «Разработка приложения с применением паттерна Наблюдатель»

Вариант №1 «Журнал регистрации корреспонденции»

|  |  |
| --- | --- |
| Выполнил | Студент группы 21-ИТ-1  Чиникайло А.П. |
|  |  |
| Проверил | Преподаватель  Хирьянов И.Д. |

Полоцк 2023

**Цель работы:** Разработать диаграмму классов по паттерну Наблюдатель и соответствующее приложение согласно варианту.

**Ход работы**

**Задание:**

Тема проекта: Проектирование приложения «Журнал регистрации корреспонденции».

Применить паттерн Наблюдатель для информирования пользователей об изменении журнала регистрации. Дополнительно к ранее разработанным классам добавить класс. Пользователь с методом получить Корреспонденцию(Параметры), который выводит на экран имя пользователя и название полученной корреспонденции. Пользователи должны иметь возможность подписываться и отписываться от получения информации о корреспонденции.

**Результат работы**

Листинг 1.1 - реализация класса Correspondence:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOTPiSP\_4

{

public class Correspondence

{

public Correspondence(int IdCorrespondence, string Content)

{

this.IdCorrespondence = IdCorrespondence;

this.Content = Content;

}

public int IdCorrespondence { get; set; }

public string Content { get; set; }

}

}

Листинг 1.2 - реализация интерфейса IObservable:

public interface IObservable

{

void RegisterObserver(IObserver o);

void RemoveObserver(IObserver o);

void NotifyObservers(string changes);

string GetItemAt(int indexItem);

}

Листинг 1.3 - реализация интерфейса IObserver::

public interface IObserver

{

void Update(string changes);

void ShowItemAt(int indexItem);

}

Листинг 1.4 - реализация класса Magazine:

public class Magazine : IObservable

{

List<IObserver> observers;

List<Correspondence> correspondence;

public Magazine()

{

observers = new List<IObserver>();

correspondence = new List<Correspondence>

{

new Correspondence(0, "Correspondence 0"),

new Correspondence(1, "Correspondence 1"),

new Correspondence(2, "Correspondence 2"),

new Correspondence(3, "Correspondence 3"),

new Correspondence(4, "Correspondence 4"),

};

}

public void RegisterObserver(IObserver o)

{

observers.Add(o);

}

public void RemoveObserver(IObserver o)

{

observers.Remove(o);

}

public void NotifyObservers(string changes)

{

foreach (IObserver o in observers)

{

o.Update(changes);

}

}

public string GetItemAt(int indexItem)

{

return correspondence[indexItem].Content;

}

}

Листинг 1.5 - реализация класса Student:

public class Student : IObserver

{

public string Name { get; set; }

IObservable magazine;

public Student(string name, IObservable obs)

{

this.Name = name;

Subscribe(obs);

}

public void Update(string changes)

{

Console.Write(changes);

Console.WriteLine($"\tfor {Name}\n");

}

public void Unsubscribe()

{

if (magazine == null)

return;

magazine.RemoveObserver(this);

magazine.NotifyObservers($"{Name} unsubscribed");

magazine = null;

}

public void Subscribe(IObservable obs)

{

if (magazine != null)

return;

magazine = obs;

magazine.RegisterObserver(this);

magazine.NotifyObservers($"{Name} signed up");

}

public void ShowItemAt(int indexItem)

{

if (magazine == null)

return;

Console.WriteLine($"{Name} receive: {magazine.GetItemAt(indexItem)}\n");

}

}

Листинг 1.6 - реализация класса Teacher:

public class Teacher : IObserver

{

public string Name { get; set; }

IObservable magazine;

public Teacher(string name, IObservable obs)

{

this.Name = name;

Subscribe(obs);

}

public void Update(string changes)

{

Console.Write(changes);

Console.WriteLine($"\tfor {Name}\n");

}

public void Unsubscribe()

{

if (magazine == null)

return;

magazine.RemoveObserver(this);

magazine.NotifyObservers($"{Name} unsubscribed");

magazine = null;

}

public void Subscribe(IObservable obs)

{

if (magazine != null)

return;

magazine = obs;

magazine.RegisterObserver(this);

magazine.NotifyObservers($"{Name} signed up");

}

public void ShowItemAt(int indexItem)

{

if (magazine == null)

return;

Console.WriteLine($"{Name} receive: {magazine.GetItemAt(indexItem)}\n");

}

}

Листинг 1.7 - реализация класса Program:

namespace OOTPiSP\_5;

class Program

{

static void Main(string[] args)

{

Magazine magazine = new Magazine();

Teacher teacher = new Teacher("Mark", magazine);

Student student = new Student("Lesha", magazine);

teacher.ShowItemAt(0);

student.ShowItemAt(1);

student.Unsubscribe();

student.ShowItemAt(2);

student.Subscribe(magazine);

Console.Read();

}

}

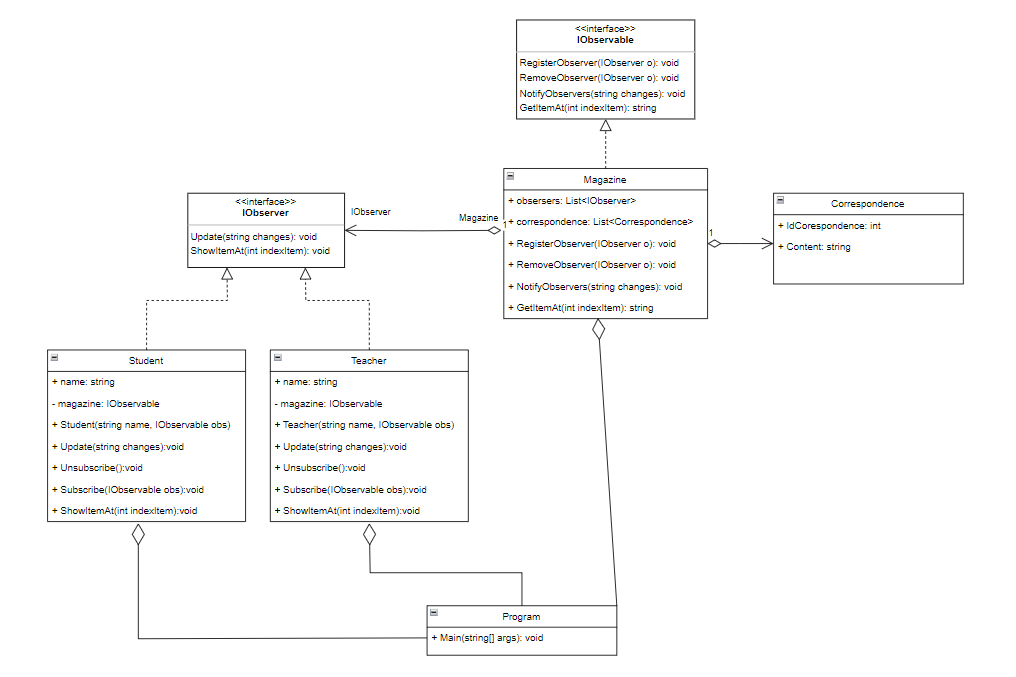
****

Рисунок 1. Диаграмма классов по паттерну Наблюдатель.

**Вывод:** В ходе данный лабораторной работе было разработано диаграмму классов по паттерну Наблюдатель и соответствующее приложение согласно варианту.