ФГБОУ ВО

“Дальневосточный государственный университет путей сообщения”

Кафедра «Информационные технологии и системы»

Лабораторная работа № 5

Тема: «Шаблоны программирования»

Выполнил: Шиляев С.П.

Проверил: Жильцов А.В.

Хабаровск

2023

**Цель работы:** научиться работать с шаблоном программирования «Наблюдатель» в среде Visual Studio.

**Задания:**

Задание 1

Замените класс People на три класса: хоббит, орк, энт. Наделите каждого из них особым поведением (выберите любые две особенности для реализации). 1. Хоббиты читают статьи в журнале в обратном порядке, от последней к первой.

2. Орки перед чтением статьи проверяют, упоминаются ли в тексте или заголовке эльфы, и если да, то отписываются от издателя и пропускают эту статью.

3. Энты читают статьи в случайном порядке. И читают их до тех то, пока каждую не прочитают хоть по одному разу.

4. Хоббиты сохраняют прочитанный журнал. И когда получают новый, сперва перечитывают сохраненный ранее.

5. Орки после прочтения статьи проверяют, встречается ли в её тексте "Саурон", и если да, то делают копию статьи, заменив в ней "Саурон" на "Правитель", и отсылают статью издателю под видом новой.

6. Энты после прочтения статьи ищут в её тексте слово "огонь", и если находят, то впадают в депрессию и пропускают следующий номер журнала.

Задание 2

Добавьте ещё одного издателя.

Задание 3

Добавьте возможность управления издателем и подписчиками по ходу выполнения программы. 1. Каждого из "людей" можно подписать/отписать на любого из издателей. 2. Любому издателю можно добавить статью. Сделайте наглядное разделение команд пользователя и описания происходящего ("Маухур прочитал статью А", "Издатель выпустил новый журнал", "Древобород впал в депрессию от прочитанного").  
Задание 4.

Модифицируйте программу так, чтобы события реализовывались специальными средствами C#.

Задание 5

1. Реализуйте в одном из подписчиков два разных метода-обработчика вместо одного. Их поведение должно различаться, к примеру: \* Чтение только четных статей. \* Чтение только статей не длиннее семи слов. \* Чтение статей в случайном порядке. \* Откладывание нового журнала и чтение прошлого выпуска.

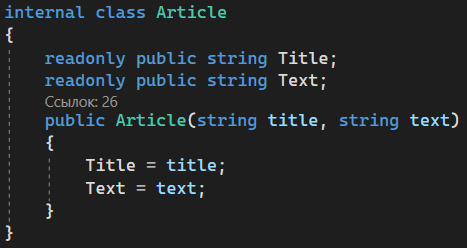
2. Подпишите этого подписчика на одного издателя первым методом, а на другого вторым.

Выполнение:

Для начала, разграничим 3 группы нужные нам для работы:

* Издательство – будет содержать в себе класс самого издательства, класс статьи, класс журнала
* Читатель – будет содержать в себе класс хоббита, орка, энта, и человека
* Интерфейс – будет содержать в себе интерфейс подписчика.

Начнем с описания класса статьи. Он будет содержать в себе два поля: наименование статьи и её текст. Реализация представлена ниже:



readonly public string Title;

readonly public string Text;

public Article(string title, string text)

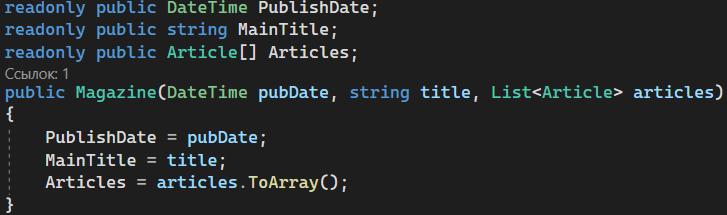
{

Title = title;

Text = text;

}

Далее, опишем класс журнала и класс издательства. Класс журнала будет содержать в себе дату публикации, заголовок и набор статей:



readonly public DateTime PublishDate;

readonly public string MainTitle;

readonly public Article[] Articles;

public Magazine(DateTime pubDate, string title, List<Article> articles)

{

PublishDate = pubDate;

MainTitle = title;

Articles = articles.ToArray();

}

Класс издательства будет иметь методы для подписки/отписки, уведомления читателей, бизнес логику и добавление статей:

int magNumber = 0;

int targetNumberOfArticles = 4;

List<Article> newArticles = new List<Article>();

Magazine currentMagazine;

public delegate void Update(Publisher publisher, Magazine magazine);

public event Update issueOfTheMagazine;

public List<Subscriber> subscribers = new List<Subscriber>();

List<Subscriber> unsubs = new List<Subscriber>();

public void MainBuisnessLogic()

{

// Поиск материала

// Написание статей

if (newArticles.Count >= targetNumberOfArticles) // Выпуск журнала

{

Console.WriteLine();

Console.WriteLine("Издатель выпустил новый журнал!");

magNumber++;

currentMagazine = new Magazine(DateTime.Now, "Mag №" + magNumber, newArticles);

NotifySubScribers();

newArticles = new List<Article>();

}

}

public void NotifySubScribers()

{

foreach (Subscriber s in unsubs)

if (subscribers.Contains(s))

subscribers.Remove(s);

foreach (Subscriber s in subscribers)

s.Update(this, currentMagazine);

foreach (Subscriber s in unsubs)

if (subscribers.Contains(s))

subscribers.Remove(s);

}

public bool Subscribe(Subscriber newSubscriber)

{

if (subscribers.Contains(newSubscriber) == false)

{

subscribers.Add(newSubscriber);

return true;

}

else return false;

}

public bool Unsubscribe(Subscriber remSubscriber)

{

if (subscribers.Contains(remSubscriber) == true)

{

unsubs.Add(remSubscriber);

return true;

}

else return false;

}

public void AddArticle(Article article)

{

newArticles.Add(article);

MainBuisnessLogic();

}

Далее, опишем классы читателей. В большинстве своем они будут похожи, за исключением метода оповещения. Классы хоббит, орк и энт представлены ниже:

public string Name { get; }

public Ent(string name)

{

Name = name;

}

public void Update(Publisher sender, object context)

{

Magazine magazine = (Magazine)context;

Random rnd = new Random();

int[] readed = new int[magazine.Articles.Length];

for (int i = 0; i < magazine.Articles.Length; i++)

{

int numb;

do

{

numb = rnd.Next(1, magazine.Articles.Length + 1);

}

while (readed.Contains(numb));

Read(magazine.Articles[numb - 1]);

readed[i] = numb;

}

}

private void Read(Article article)

{

Console.WriteLine($"{Name} читает статью \"{article.Title}\"");

}

public string Name { get; }

public Hobbit(string name)

{

Name = name;

}

public void Update(Publisher sender, object context)

{

Magazine magazine = (Magazine)context;

for (int i = magazine.Articles.Length - 1; i > -1; i--)

{

Read(magazine.Articles[i]);

}

}

private void Read(Article article)

{

Console.WriteLine($"{Name} читает статью \"{article.Title}\"");

}

public string Name { get; }

public Orc(string name)

{

Name = name;

}

public void Update(Publisher sender, object context)

{

Magazine magazine = (Magazine)context;

foreach (Article article in magazine.Articles)

Read(article, sender);

}

private void Read(Article article, Publisher sender)

{

if (article.Title.Contains("Эльф") || article.Title.Contains("эльф") || article.Text.Contains("Эльф") || article.Text.Contains("эльф") )

sender.Unsubscribe(this);

else

Console.WriteLine($"{Name} читает статью \"{article.Title}\"");

}

В интерфейсе напишем метод оповещения подписчиков и поле Name, содержащее имя читателя.

interface Subscriber

{

void Update(Publisher sender, object context);

public string Name { get; }

}

После, в классе program реализуем командный интерфейс и опишем использование вышеупомянутых классов:

List<Article> articles = new List<Article>()

{

new Article("Кольца Власти", "Магические Кольца сделают Средиземье лучше."),

new Article("Три Кольца", "Келебримбор завершил создание Колец Огня, Воды и Воздуха. Даже без помощи Саурона."),

new Article("Единое Кольцо", "Саурон всех обманул и сделал кольцо для подчинения владельцев остальных колец!"),

new Article("Восстание", "Келебримбор спрятал кольца эльфов от Саурона. Начинается восстание против зла."),

new Article("Враг наступает", "Неприятель знает, где кольца, его армия продвигается вперед."),

new Article("Враг разбит!", "Властелин Тьмы повержен. Большую роль в победе сыграли нуменорцы"),

new Article("Жить становится страшно", "Лиходейские твари рядом. В Мглистых Горах множатся орки и нападают на гномов."),

new Article("Мории не будет", "Балин предпринял всё возможное, чтобы вернуть Морию. Но ничего не получилось."),

new Article("Множится мощь Мордора", "Мордор объявил призыв. Саруман стал предателем."),

new Article("Рубежи падают...", "Саурон напал на Осгилиат."),

new Article("Черные Всадники", "Черные Всадники рыскают в Хоббитландии."),

new Article("Совет у Эльронда", "Принято решение отнести сами-знаете-что в Роковую Гору для уничтожения."),

};

List<Article> articles2 = new List<Article>()

{

new Article("Кольца Пасти", "Магические Кольца сделают Средиземье лучше."),

new Article("Три Кольца два конца", "Келебримбор завершил создание Колец Огня, Воды и Воздуха. Даже без помощи Саурона."),

new Article("Единое Олимпийское Кольцо", "Саурон всех обманул и сделал кольцо для подчинения владельцев остальных колец!"),

new Article("Восстание ягнят", "Келебримбор спрятал кольца эльфов от Саурона. Начинается восстание против зла."),

new Article("Враг наступает на...", "Неприятель знает, где кольца, его армия продвигается вперед."),

new Article("Враг разбит толком!", "Властелин Тьмы повержен. Большую роль в победе сыграли нуменорцы"),

new Article("Жить становится страшно весело", "Лиходейские твари рядом. В Мглистых Горах множатся орки и нападают на гномов."),

new Article("Моря не будет", "Балин предпринял всё возможное, чтобы вернуть Морию. Но ничего не получилось."),

new Article("Множится мощь Дамблдора", "Мордор объявил призыв. Саруман стал предателем."),

new Article("Рубежи падают на...", "Саурон напал на Осгилиат."),

new Article("Черные Всадники", "Черные Всадники рыскают в Хоббитландии."),

new Article("Совет у Агента Смита", "Принято решение отнести сами-знаете-что в Роковую Гору для уничтожения."),

};

Subscriber[] subscribers = new Subscriber[]

{

new Hobbit("Сэм"), new Orc("Маухур"), new Ent("Древобород")

};

Subscriber[] subs2 = new Subscriber[]

{

new Hobbit("Федор"), new Hobbit("Бульба"),

};

Publisher LOTRMag = new Publisher();

Publisher LOTR = new Publisher();

Publisher[] pblsrs = { LOTRMag, LOTR };

string[] command;

foreach (Subscriber someone in subscribers)

LOTRMag.Subscribe(someone);

foreach (Subscriber someone in subs2)

LOTR.Subscribe(someone);

Console.ForegroundColor = ConsoleColor.White;

Info();

while (true)

{

do

{

Console.Write(">");

command = Console.ReadLine().Split();

int numb;

switch (command[0])

{

case "unSub":

{

if (command.Length < 3)

break;

if (Convert.ToInt16(command[2]) - 1 < pblsrs.Length)

{

numb = Convert.ToInt16(command[2]) - 1;

if (pblsrs[numb].Unsubscribe(pblsrs[numb].subscribers.Find(s => s.Name == command[1])))

{

MessageGood($"{command[1]} отписался от издателя {command[2]}");

break;

}

else

{

MessageBad("Данный субъект не подписан на данное издание, посмотреть все субъекты и издания - printAll");

}

}

else

MessageBad("Неправильный номер издателя");

break;

}

case "printAll":

{

Console.WriteLine($"Подписота 1 издательства:");

foreach (var s in pblsrs[0].subscribers)

Console.WriteLine(s.Name);

Console.WriteLine($"Подписота 2 издательства:");

foreach (var s in pblsrs[1].subscribers)

Console.WriteLine(s.Name);

break;

}

case "sub":

{

Console.WriteLine("Введите номер издания");

numb = Convert.ToInt16(Console.ReadLine());

Console.WriteLine("Введите имя читателя");

string name = Console.ReadLine();

Console.WriteLine("Введите расу читателя 1 - хоббит, 2 - орк, 3 - энт");

switch (Convert.ToInt16(Console.ReadLine()))

{

case 1:

if(pblsrs[numb - 1].Subscribe(new Hobbit(name)))

MessageGood("Подписота добавлена");

else MessageBad("Что-то не так");

break;

case 2:

if(pblsrs[numb - 1].Subscribe(new Orc(name)))

MessageGood("Подписота добавлена");

else MessageBad("Что-то не так");

break;

case 3:

if(pblsrs[numb - 1].Subscribe(new Ent(name)))

MessageGood("Подписота добавлена");

else MessageBad("Что-то не так");

break;

}

break;

}

case "addArt":

{

Console.WriteLine("Введите название статьи:");

string name = Console.ReadLine();

Console.WriteLine("Введите содержание статьи:");

string topic = Console.ReadLine();

Console.WriteLine("Введите номер издателя");

numb = Convert.ToInt16(Console.ReadLine());

if (numb - 1 < pblsrs.Length) { Console.WriteLine("Всего 2 издателя"); break; }

if (numb - 1 == 0)

{

articles.Add(new Article(name, topic));

MessageGood("Статья успешно добавлена");

break;

}

if (numb - 1 == 1)

{

articles2.Add(new Article(name, topic));

MessageGood("Статья успешно добавлена");

break;

}

MessageBad("Что-то не так");

break;

}

case "exit": Environment.Exit(0); break;

case "help":

Info(); break;

default:

Console.WriteLine("Команда не найдена, help - для помощи"); break;

}

} while (command[0] != "start");

foreach (Article article in articles2)

LOTRMag.AddArticle(article);

foreach (Article article in articles)

LOTR.AddArticle(article);

}

}

static void MessageGood(string message)

{

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine(message);

Console.ForegroundColor = ConsoleColor.White;

}

static void MessageBad(string message)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine(message);

Console.ForegroundColor = ConsoleColor.White;

}

static void Info()

{

Console.WriteLine();

Console.WriteLine("unSub арг1 арг2 - Отписать\_читателя Имя номер\_издателя ");

Console.WriteLine("sub арг1 арг2 - Подписать\_читателя Имя номер\_издателя ");

Console.WriteLine("addArt - Добавить\_статью");

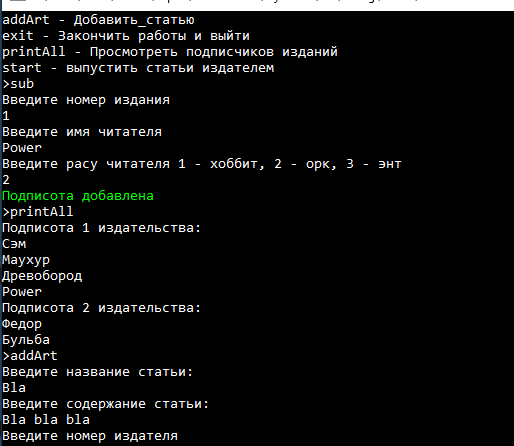
Console.WriteLine("exit - Закончить работы и выйти");

Console.WriteLine("printAll - Просмотреть подписчиков изданий");

Console.WriteLine("start - выпустить статьи издателем");

}

Результат работы приложения на данном этапе выглядит так:



Для задания 4, удалим интерфейс и все связанные с ним методы и поля. Вместо него в классе издателя создадим делегат обновления:

public delegate void Update(Publisher publisher, Magazine magazine);

И на его основе событие оповещения:

public event Update issueOfTheMagazine;

Теперь для добавления читателя в подписчики нужно просто подписать его на событие, например вот так:

LOTR.issueOfTheMagazine += Bulba.Update;

Также обновим методы для добавления и удаления подписчиков:

Console.WriteLine("Введите номер издания");

numb = Convert.ToInt16(Console.ReadLine());

Console.WriteLine("Введите имя читателя");

string name = Console.ReadLine();

Console.WriteLine("Введите расу читателя 1 - хоббит, 2 - орк, 3 - энт");

switch (Convert.ToInt16(Console.ReadLine()))

{

case 1:

pblsrs[numb - 1].issueOfTheMagazine += new Hobbit(name).Update;

MessageGood("Подписота добавлена");

break;

case 2:

pblsrs[numb - 1].issueOfTheMagazine += new Orc(name).Update;

break;

case 3:

pblsrs[numb - 1].issueOfTheMagazine += new Ent(name).Update;

break;

}

break;

Для выполнения 5 задания классу хоббита добавим новый метод, чтения случайной статьи из журнала, а также подпишем его на событие обновление:

public void Updated(Publisher sender, object context)

{

Random rnd = new Random();

Magazine magazine = (Magazine)context;

Read(magazine.Articles[rnd.Next(0, magazine.Articles.Length)]);

}

Результат работы программы:

