***Лабараторна робота №6***

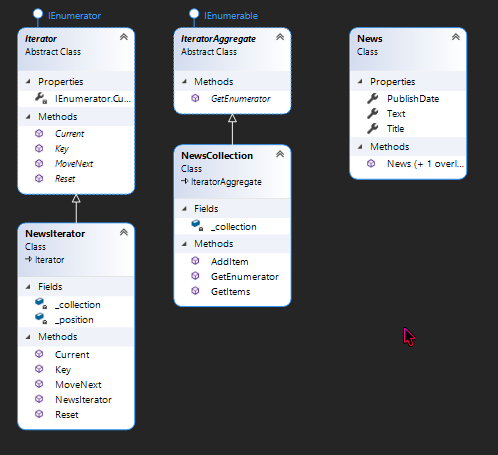
***Варіант №14***

**Завдання №1**

**14.** Предположим у нас есть новостной блок, на который выводится максимум 10 новостей. Каждый новостной блок имеет дату создания. Предположим что метод getLatestNews используется во многих частях системы, и все клиенты этого кода ожидают в результате массив объектов новостей из заданого новостного блока по введенной дате. Применить паттерн Итератор вывода найденного блока новостей, общего количество новостей в блоке и их описание.

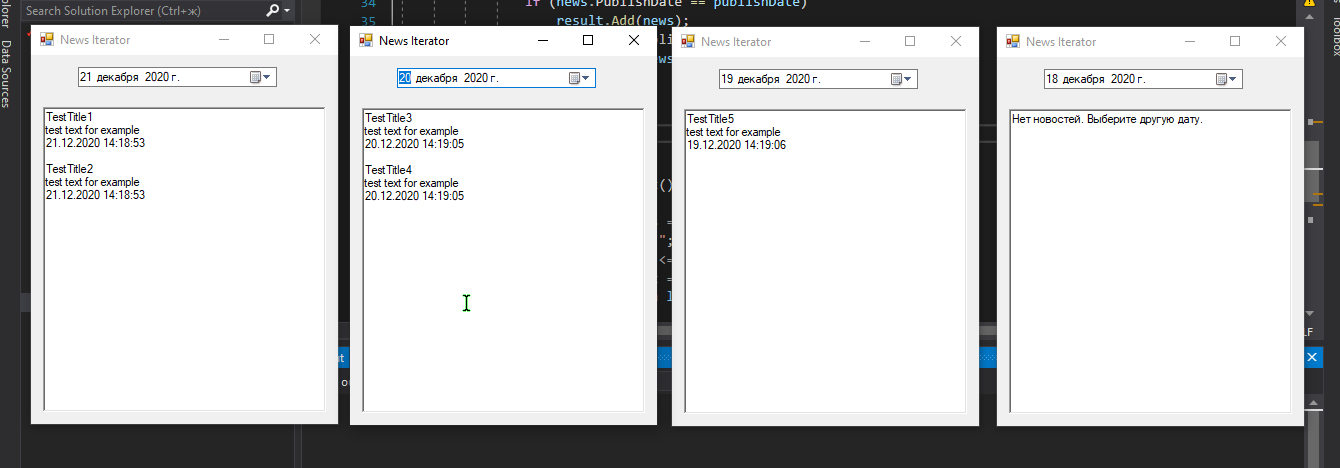
***Результати виконання завдання 1:***

*Діаграма класів додатку (рис.1):*



**Рис. 1 Діаграма класів додатку**

*Результати роботи додатку (рис.2)*



**Рис. 2 Результат виконаня додатку**

*Лістинг програмного коду:*

*\*\* MainForm.cs \*\**

using System;

using System.Collections.Generic;

using System.Windows.Forms;

namespace Lab\_6\_1

{

public partial class MainForm : Form

{

private readonly NewsCollection \_newsCollection = new NewsCollection();

public MainForm()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

DateTime yesterday = DateTime.Now.Subtract(TimeSpan.FromDays(1));

DateTime dayBeforeYesterday = yesterday.Subtract(TimeSpan.FromDays(1));

\_newsCollection.AddItem(new News("TestTitle1", "test text for example"));

\_newsCollection.AddItem(new News("TestTitle2", "test text for example"));

\_newsCollection.AddItem(new News("TestTitle3", "test text for example", yesterday));

\_newsCollection.AddItem(new News("TestTitle4", "test text for example", yesterday));

\_newsCollection.AddItem(new News("TestTitle5", "test text for example", dayBeforeYesterday));

UpdateResult();

}

public List<News> getLatestNews(DateTime publishDate)

{

List<News> result = new List<News>();

foreach (News news in \_newsCollection)

{

if (news.PublishDate == publishDate)

result.Add(news);

else if (news.PublishDate.Date == publishDate.Date)

result.Add(news);

}

return result;

}

private void UpdateResult()

{

List<News> latestNews = getLatestNews(dateTimePicker1.Value);

richTextBox1.Text = "";

if (latestNews.Count <= 0)

richTextBox1.Text = "Нет новостей. Выберите другую дату.";

foreach (News news in latestNews)

{

richTextBox1.Text += $"{news.Title}\n{news.Text}\n{news.PublishDate}\n\n";

}

}

private void dateTimePicker1\_ValueChanged(object sender, EventArgs e)

{

UpdateResult();

}

}

}

*\*\* NewsIterator.cs \*\**

using System;

using System.Collections;

using System.Collections.Generic;

namespace Lab\_6\_1

{

public class News

{

public string Title { get; }

public string Text { get; }

public DateTime PublishDate { get; }

public News(string title, string text)

{

Title = title;

Text = text;

PublishDate = DateTime.Now;

}

public News(string title, string text, DateTime publishDate)

{

Title = title;

Text = text;

PublishDate = publishDate;

}

}

abstract class Iterator : IEnumerator

{

object IEnumerator.Current => Current();

public abstract int Key();

public abstract object Current();

public abstract bool MoveNext();

public abstract void Reset();

}

abstract class IteratorAggregate : IEnumerable

{

public abstract IEnumerator GetEnumerator();

}

class NewsCollection : IteratorAggregate

{

List<News> \_collection = new List<News>();

public List<News> GetItems()

{

return \_collection;

}

public void AddItem(News item)

{

\_collection.Add(item);

}

public override IEnumerator GetEnumerator()

{

return new NewsIterator(this);

}

}

class NewsIterator : Iterator

{

private NewsCollection \_collection;

private int \_position = -1;

public NewsIterator(NewsCollection collection)

{

\_collection = collection;

}

public override object Current()

{

return \_collection.GetItems()[\_position];

}

public override int Key()

{

return \_position;

}

public override bool MoveNext()

{

int updatedPosition = \_position + 1;

if (updatedPosition >= 0 && updatedPosition < \_collection.GetItems().Count)

{

\_position = updatedPosition;

return true;

}

return false;

}

public override void Reset()

{

\_position = 0;

}

}

}