Министерство образования Российской Федерации Пензенский государственный университет

Кафедра «Математическое обеспечение и применение ЭВМ»

**ОТЧЕТ**

по лабораторной работе №5

по курсу «ООП» на тему «Поведенческие потерны»

Выполнили:

студенты группы 23ВП2 Дорофейчук Е.А.

Дорофейчук П.А.

Принял:

к.т.н. Афонин А.Ю.

Пенза 2025

Название

Поведенческие потерны

Цель работы

Научиться использовать поведенческие потерны.

Лабораторное задание

Реализовать паттерн программирования Хранитель (Memento). Предметная область соответствует предметной области первой, второй лабораторной работы.

Листинг

**Файл Form1.cs:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lab5

{

public partial class Form1 : Form

{

public Form1()

{

StartPosition = FormStartPosition.CenterScreen;

\_airportList = new AirportList(new List<Airport> {

new Airport("Белинского", 123, 467, 2, true, "Пенза", 1960),

new Airport("Домодедово", 883, 4967, 159, true, "Москва", 1962)

});

InitializeComponent();

createTable();

}

private void create\_Click(object sender, EventArgs e)

{

try

{

var name = textBoxName.Text;

var countFlight = numericCountFlight.Value;

var countTicket = numericCountTicket.Value;

var area = numericArea.Value;

var isOpen = checkBoxIsOpen.Checked;

var city = textBoxCity.Text;

var year = numericYear.Value;

if (\_airportList.Hubs.FirstOrDefault(x => x.Item1.Name.ToLower().Equals(name.ToLower())).Item1 != null)

{

throw new MyException("Название уже существует");

}

Airport hub;

if (name != string.Empty && city != string.Empty)

{

hub = new Airport(name, Convert.ToInt32(countFlight), Convert.ToInt32(countTicket), Convert.ToInt32(area), isOpen, city, Convert.ToInt16(year));

\_airportList.Add(hub);

addRow();

textBoxName.Text = string.Empty;

numericCountFlight.Value = 0;

numericCountTicket.Value = 0;

numericArea.Value = 0;

checkBoxIsOpen.Checked = false;

textBoxCity.Text = string.Empty;

numericYear.Value = 1900;

}

else

{

showEmpty();

throw new MyException("Заполните все поля");

}

textBoxName.BackColor = Color.White;

textBoxCity.BackColor = Color.White;

}

catch (MyException ex)

{

MessageBox.Show(ex.Message);

}

catch (InvalidCastException ex)

{

MessageBox.Show("Неверный набор параметров");

}

catch (Exception ex)

{

MessageBox.Show("Неизвестная ошибка");

}

}

private void showEmpty()

{

if (textBoxName.Text.Length == 0) textBoxName.BackColor = Color.Red;

if (textBoxCity.Text.Length == 0) textBoxCity.BackColor = Color.Red;

}

private void createTable()

{

var number = 0;

airoportsTable.Rows.Clear();

foreach (var i in \_airportList.Hubs)

{

var airport = i.Item1;

airoportsTable.Rows.Add();

airoportsTable.Rows[number].Cells[0].Value = airport.Name ?? string.Empty;

airoportsTable.Rows[number].Cells[1].Value = airport.CountFlight.ToString() ?? "0";

airoportsTable.Rows[number].Cells[2].Value = airport.CountTicket.ToString() ?? "0";

airoportsTable.Rows[number].Cells[3].Value = airport.Area.ToString() ?? "0";

airoportsTable.Rows[number].Cells[4].Value = airport.IsOpen.ToString() ?? string.Empty;

airoportsTable.Rows[number].Cells[5].Value = airport.City ?? string.Empty;

airoportsTable.Rows[number].Cells[6].Value = airport.YearOfConstruction.ToString() ?? "1900";

airoportsTable.Rows[number].Cells[7].Value = "Назад";

number++;

}

}

private void updateRow(Airport hub)

{

var row = airoportsTable.Rows.Cast<DataGridViewRow>().First(x => x.Cells[0].Value.Equals(hub.Name));

hub.Name = hub.CheckName;

row.Cells[0].Value = hub.Name ?? string.Empty;

row.Cells[1].Value = hub.CountFlight.ToString() ?? "0";

row.Cells[2].Value = hub.CountTicket.ToString() ?? "0";

row.Cells[3].Value = hub.Area.ToString() ?? "0";

row.Cells[4].Value = hub.IsOpen.ToString() ?? string.Empty;

row.Cells[5].Value = hub.City ?? string.Empty;

row.Cells[6].Value = hub.YearOfConstruction.ToString() ?? "1900";

}

private void removeRow(Airport hub)

{

var row = airoportsTable.Rows.Cast<DataGridViewRow>().First(x => x.Cells[0].Value.Equals(hub.Name));

airoportsTable.Rows.Remove(row);

}

private void addRow()

{

var number = airoportsTable.RowCount;

airoportsTable.Rows.Add();

airoportsTable.Rows[number].Cells[0].Value = \_airportList[\_airportList.Length - 1].Item1.Name ?? string.Empty;

airoportsTable.Rows[number].Cells[1].Value = \_airportList[\_airportList.Length - 1].Item1.CountFlight.ToString() ?? "0";

airoportsTable.Rows[number].Cells[2].Value = \_airportList[\_airportList.Length - 1].Item1.CountTicket.ToString() ?? "0";

airoportsTable.Rows[number].Cells[3].Value = \_airportList[\_airportList.Length - 1].Item1.Area.ToString() ?? "0";

airoportsTable.Rows[number].Cells[4].Value = \_airportList[\_airportList.Length - 1].Item1.IsOpen.ToString() ?? string.Empty;

airoportsTable.Rows[number].Cells[5].Value = \_airportList[\_airportList.Length - 1].Item1.City ?? string.Empty;

airoportsTable.Rows[number].Cells[6].Value = \_airportList[\_airportList.Length - 1].Item1.YearOfConstruction.ToString() ?? string.Empty;

airoportsTable.Rows[number].Cells[7].Value = "Назад";

}

private void buttonExit\_Click(object sender, EventArgs e)

{

Close();

}

private void deleteButton\_Click(object sender, EventArgs e)

{

var hub = \_airportList.Remove(findNameTextbox.Text);

if (hub != null)

{

removeRow(hub);

findNameTextbox.Text = string.Empty;

return;

}

MessageBox.Show("Не найден");

}

private void findButton\_Click(object sender, EventArgs e)

{

var hub = \_airportList.Hubs.FirstOrDefault(x => x.Item1.Name.ToLower() == findNameTextbox.Text.ToLower());

if (hub.Item1 != null)

{

Form2 form2 = new Form2(hub.Item1, hub.Item2)

{

StartPosition = FormStartPosition.CenterParent

};

form2.ShowDialog();

while (!hub.Item1.Name.Equals(hub.Item1.CheckName) && \_airportList.Hubs.FirstOrDefault(x => x.Item1.Name.ToLower() == hub.Item1.CheckName.ToLower()).Item1 != null)

{

MessageBox.Show("Название уже существует");

form2.ShowDialog();

}

updateRow(hub.Item1);

findNameTextbox.Text = string.Empty;

return;

}

MessageBox.Show("Не найден");

}

private void airoportsTable\_CellEndEdit(object sender, DataGridViewCellEventArgs e)

{

int rowIndex = e.RowIndex;

var name = airoportsTable.Rows[rowIndex].Cells[0].Value.ToString();

var hub = \_airportList.Hubs.First(x => x.Item1.Name == name);

hub.Item2.SaveState(hub.Item1);

hub.Item1.CountFlight = Convert.ToInt32(airoportsTable.Rows[rowIndex].Cells[1].Value);

hub.Item1.CountTicket = Convert.ToInt32(airoportsTable.Rows[rowIndex].Cells[2].Value);

hub.Item1.Area = Convert.ToInt32(airoportsTable.Rows[rowIndex].Cells[3].Value);

hub.Item1.IsOpen = Convert.ToBoolean(airoportsTable.Rows[rowIndex].Cells[4].Value);

hub.Item1.City = airoportsTable.Rows[rowIndex].Cells[5].Value.ToString();

hub.Item1.YearOfConstruction = Convert.ToInt16(airoportsTable.Rows[rowIndex].Cells[6].Value);

}

private void airoportsTable\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

var senderGrid = (DataGridView)sender;

if (senderGrid.Columns[e.ColumnIndex] is DataGridViewButtonColumn && e.RowIndex >= 0)

{

var hub = \_airportList.Hubs[e.RowIndex];

if (hub.Item2.Versions.Count == 0)

{

MessageBox.Show("История пуста");

return;

}

hub.Item2.RestoreState(hub.Item1);

updateRow(hub.Item1);

}

}

}

}

**Файл Form2.cs:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using test;

namespace lab5

{

public partial class Form2 : Form

{

public Form2(Airport airport, History history)

{

\_airport = airport;

\_history = history;

InitializeComponent();

textBoxName.Text = airport.Name ?? string.Empty;

numericCountFlight.Value = airport.CountFlight ?? 0;

numericCountTicket.Value = airport.CountTicket ?? 0;

checkBoxIsOpen.Checked = airport.IsOpen ?? false;

numericArea.Value = Convert.ToInt32(airport.Area ?? 0);

textBoxCity.Text = airport.City ?? string.Empty;

numericYear.Value = airport.YearOfConstruction ?? 1900;

}

private void changeButton\_Click(object sender, EventArgs e)

{

try

{

\_history.SaveState(\_airport);

\_airport.CheckName = textBoxName.Text;

\_airport.CountFlight = Convert.ToInt32(numericCountFlight.Value);

\_airport.CountTicket = Convert.ToInt32(numericCountTicket.Value);

\_airport.IsOpen = checkBoxIsOpen.Checked;

\_airport.Area = Convert.ToInt32(numericArea.Value);

\_airport.City = textBoxCity.Text;

\_airport.YearOfConstruction = Convert.ToInt16(numericYear.Value);

this.Close();

}

catch (MyException ex)

{

MessageBox.Show(ex.Message);

}

catch (Exception ex)

{

MessageBox.Show("Неизвестная ошибка");

}

}

private void backButton\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

**Файл Airport.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;

using test;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.TextBox;

namespace lab5

{

public class Airport

{

public string CheckName { get; set; }

public string Name { get; set; }

public int? CountFlight { get; set; }

public int? CountTicket { get; set; }

public int? Area { get; set; }

public bool? IsOpen { get; set; }

public string City { get; set; }

public short? YearOfConstruction { get; set; }

public Airport()

{

Name = string.Empty;

CountFlight = 0;

CountTicket = 0;

Area = 0;

IsOpen = false;

City = string.Empty;

YearOfConstruction = 1900;

CheckName = string.Empty;

}

public Airport(string name, int countFlight, int countTicket, int area, bool isOpen, string city, short yearOfConsuption)

{

Name = name;

CountFlight = countFlight;

CountTicket = countTicket;

Area = area;

IsOpen = isOpen;

City = city;

YearOfConstruction = yearOfConsuption;

CheckName = name;

}

public Airport(string name)

{

Name = name;

CountFlight = 0;

CountTicket = 0;

Area = 0;

IsOpen = false;

City = name;

YearOfConstruction = 1900;

CheckName = name;

}

public Airport(string name, string city)

{

Name = name;

CountFlight = 0;

CountTicket = 0;

Area = 0;

IsOpen = false;

City = city;

YearOfConstruction = 1900;

CheckName = name;

}

public Memento Save()

{

return new Memento(this);

}

}

}

**Файл AirportList.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using test;

namespace lab5

{

public class AirportList

{

public List<(Airport, History)> Hubs { get; set; } = new List<(Airport, History)>();

public int Length { get; set; } = 0;

public delegate void MessageListener(string message);

public event MessageListener Listener = message => MessageBox.Show(message);

public AirportList()

{

Hubs = new List<(Airport, History)>();

Length = 0;

}

public AirportList(List<Airport> hub)

{

foreach (var i in hub)

{

Hubs.Add((i, new History()));

}

Length = hub.Count;

}

public (Airport, History) this[int index]

{

get => Hubs[index];

set => Hubs[index] = value;

}

public void Add(Airport hub)

{

Hubs.Add((hub, new History()));

Listener?.Invoke($"Добавлен: {hub.Name}");

Length++;

}

public Airport Remove(string hubName)

{

var hub = Hubs.FirstOrDefault(x => x.Item1.Name.ToLower() == hubName.ToLower());

if (hub.Item1 != null)

{

Hubs.Remove(hub);

Listener?.Invoke($"Удалён: {hub.Item1.Name}");

Length--;

return hub.Item1;

}

return null;

}

}

}

**Файл Memento.cs:**

using lab5;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace test

{

public class Memento

{

public readonly string CheckName;

public readonly string Name;

public readonly int? CountFlight;

public readonly int? CountTicket;

public readonly int? Area;

public readonly bool? IsOpen;

public readonly string City;

public readonly short? YearOfConstruction;

public Memento(Airport airport)

{

CheckName = airport.CheckName;

Name = airport.Name;

CountFlight = airport.CountFlight;

CountTicket = airport.CountTicket;

Area = airport.Area;

IsOpen = airport.IsOpen;

City = airport.City;

YearOfConstruction = airport.YearOfConstruction;

}

}

}

**Файл History.cs:**

using lab5;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace test

{

public class History

{

public Stack<Memento> Versions { get; set; } = new Stack<Memento>();

public void SaveState(Airport airport)

{

Versions.Push(airport.Save());

}

public void RestoreState(Airport airport)

{

var memento = Versions.Pop();

airport.CheckName = memento.CheckName;

airport.Name = memento.Name;

airport.CountFlight = memento.CountFlight;

airport.CountTicket = memento.CountTicket;

airport.Area = memento.Area;

airport.City = memento.City;

airport.IsOpen = memento.IsOpen;

airport.YearOfConstruction = memento.YearOfConstruction;

}

}

}

Пояснительный текст к программе

Мы создали класс Memento, который хранит в себе прошлые значения объекта типа Airport, а именно: Name (Название), City (Город) и YearOfConstruction (Год основания). Также был добавлен класс History, который является контейнерным классом содержащий в себе объекты типа Memento.

Результаты работы программы показаны на рисунках 1 – 3.

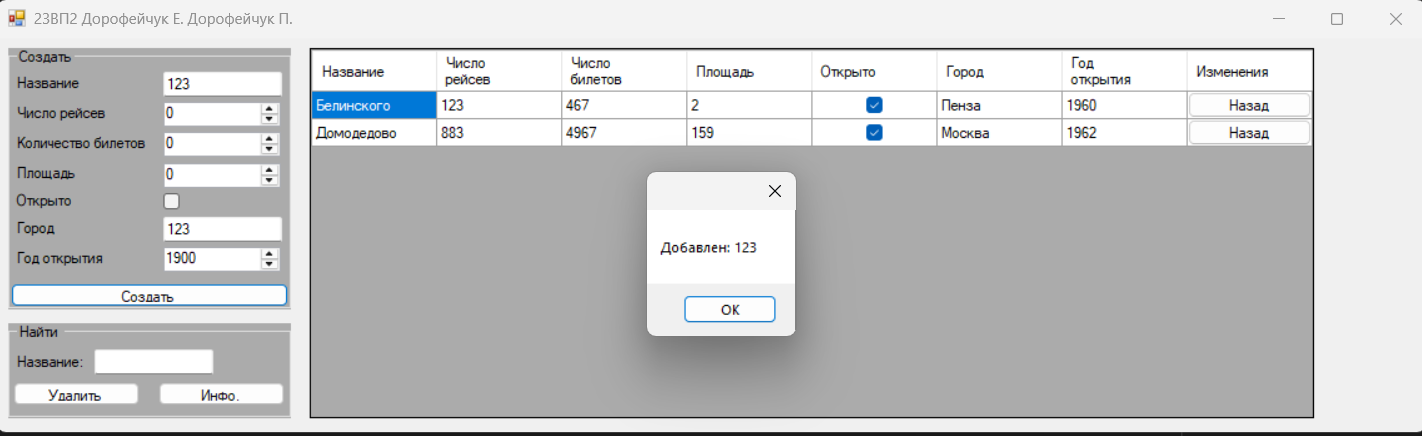


Рисунок 1 – Создание аэропорта

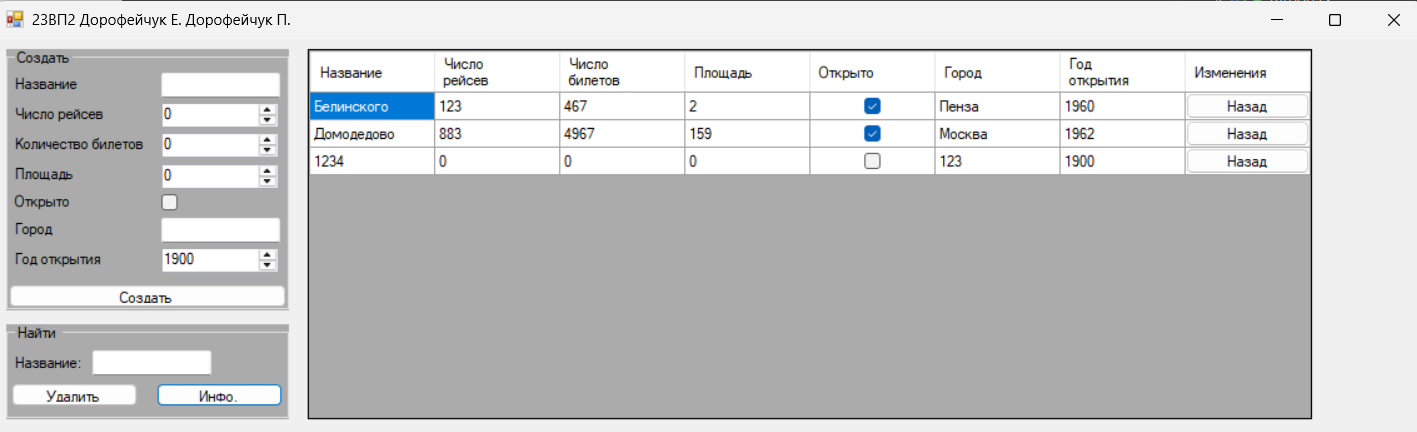


Рисунок 2 – Изменение аэропорта

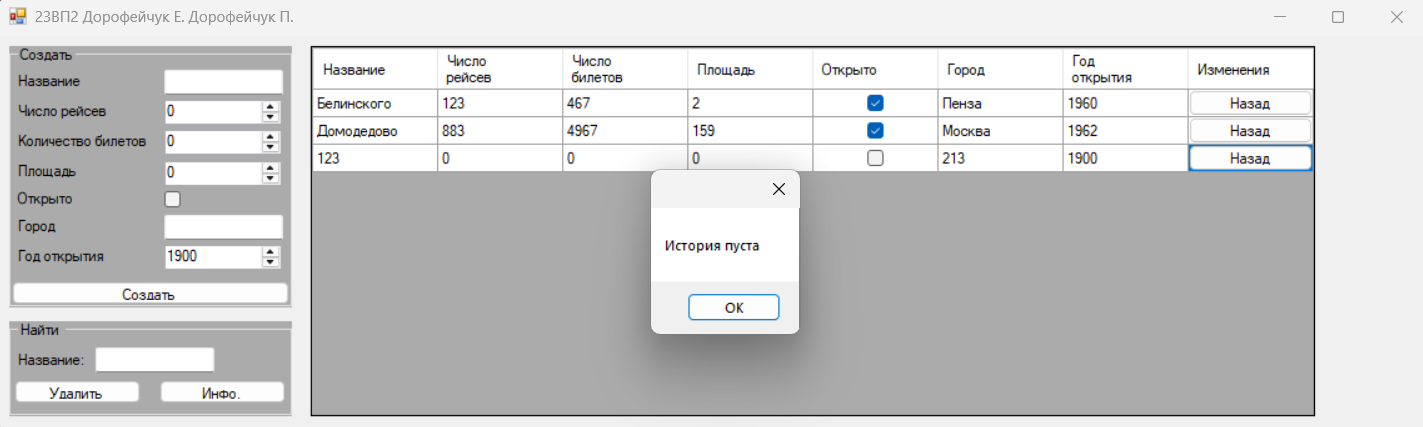


Рисунок 2 – Отмена изменений

Диаграмма вариантов использования

Диаграмма вариантов использования представлена на рисунке 3.

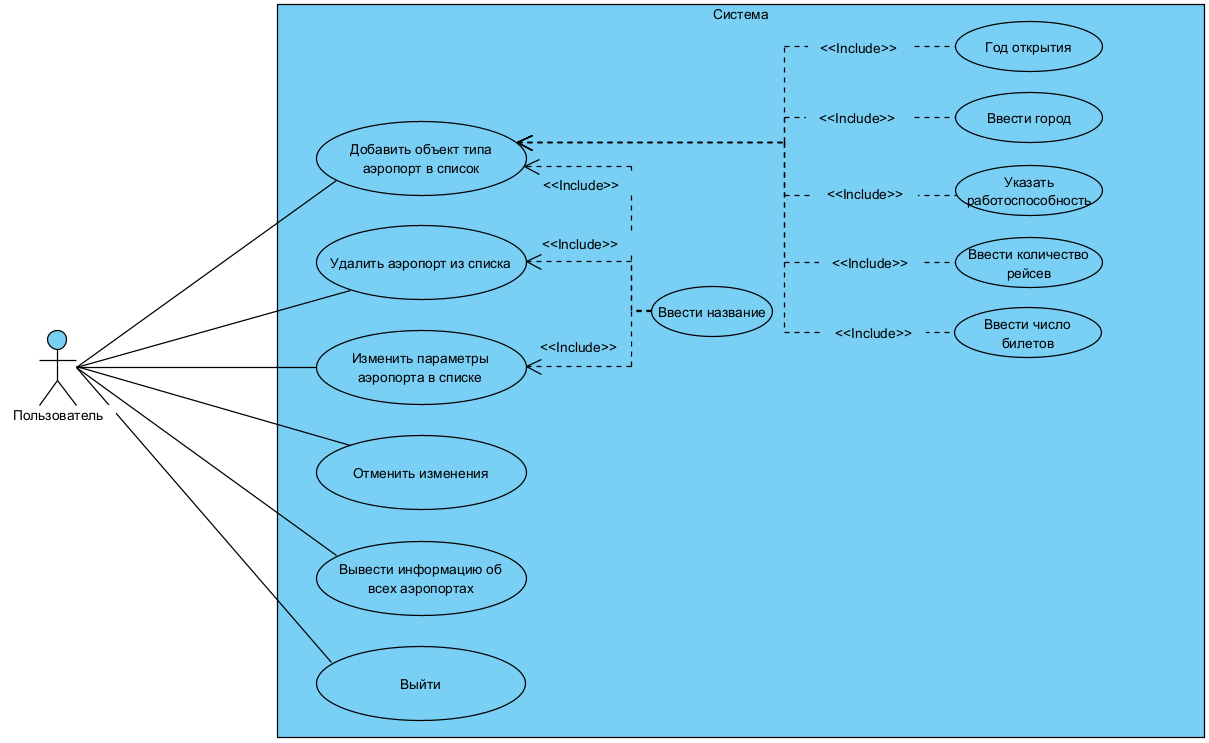


Рисунок 3 – Диаграмма вариантов использования

Вывод

В холе лабораторной работы мы смогли реализовать паттерн Хранитель, а также ознакомились с другими структурными паттернами, такими как Цепочка обязанностей, Итератор, Посредник, Наблюдатель, Состояние и т.д.

Мы создали класс Memento, который хранит в себе прошлые значения объекта типа Airport, а именно: Name (Название), City (Город) и YearOfConstruction (Год основания). Также был добавлен класс History, который является контейнерным классом содержащий в себе объекты типа Memento.

Таким образом мы научились работать с поражающими паттернами.