

Лабораторная работа № 10

«Разработка пользовательского интерфейса приложения, реализующего принципы инкапсуляции на платформе .NET Framework»

1 Цель работы

Целью лабораторной работы является изучение основ принципов инкапсуляции в приложениях на платформе .NET Framework.

2 Задание на лабораторную работу

Вариант 9.

1. Создать класс сотрудника вуза (Employee) и предусмотреть в нем соответствующие поля (ФИО, должность, факультет, пол, дата рождения и другие). Среди полей необходимо предусмотреть поля, необходимые для авторизации пользователя, которая будет осуществляться по одному из предложенных способов (пара логин/пароль, код авторизации, авторизация с помощью ключа).
2. Создать интерфейс формы для регистрации нового пользователя. При регистрации должны быть указаны поля авторизации. Поля, представляющие собой справочники (должность, факультет, пол), должны быть представлены на форме в виде выпадающих списков (ComboBox).
3. Создать интерфейс формы для авторизации существующего пользователя. Предусмотреть возможность авторизации различными способами, указанными в задании 1.
4. Добавить методы для авторизации пользователя различными способами. Объект user должен быть создан только при успешной авторизации пользователя. Данные пользователей проверяются по имеющимся файлам в папке employees. По результату авторизации должно выводиться соответствующее сообщение.

3 Ход выполнения

В первую очередь создаем интерактивное меню, с помощью пользовательского элемента управления «UserControl».

Для реализации всех заданий мы создаем несколько панелей, на которые заносим различные элементы и делаем их все типа false, и в случае нажатия различных кнопок, будут вызываться различные панели.

Для реализации первого, третьего и четвертого задания, мы просто создаем несколько кнопок (1 – логин/пароль, 2 – код доступа, 3 – ключ). Если пользователь выбирает первое, то ему необходимо ввести свои данные, которые были введены в 1 задании, если второе, тогда можно нажать на кнопку «Зайти на почту» и пользователя перекинет на страничку, где будет написан код доступа, при вводе которого далее выведется сообщение об успешной авторизации. Если же пользователь выберет третье, то нужно будет ввести код из файла (для этого нужно будет найти файл на компьютере и ввести его в приложение). В случае, если пользователь вводит что-то не так, высвечивается ошибка. Результат работы программы представлен на рисунках 1-7.

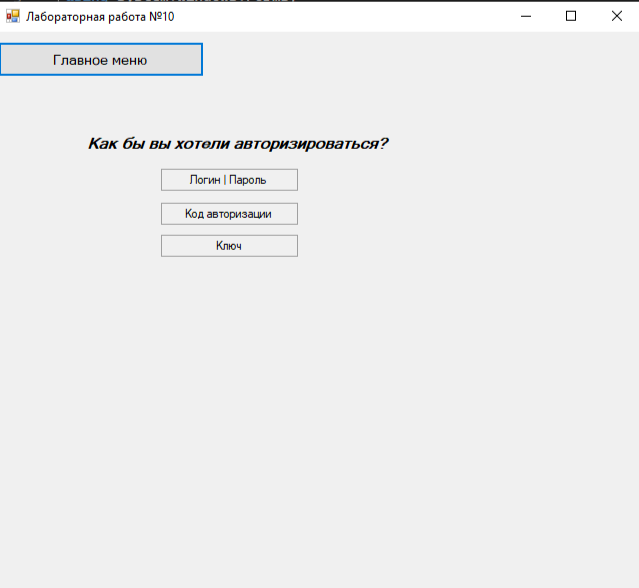


Рисунок 1 – Как пользователь может авторизоваться

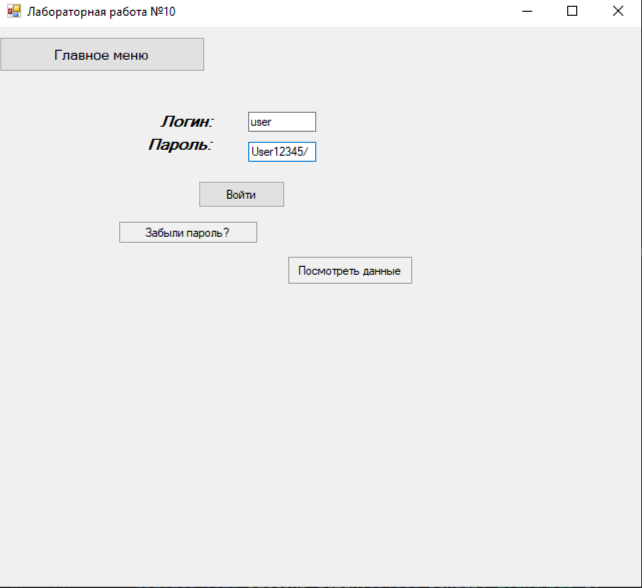


Рисунок 2 – Авторизация с помощью логина и пароля

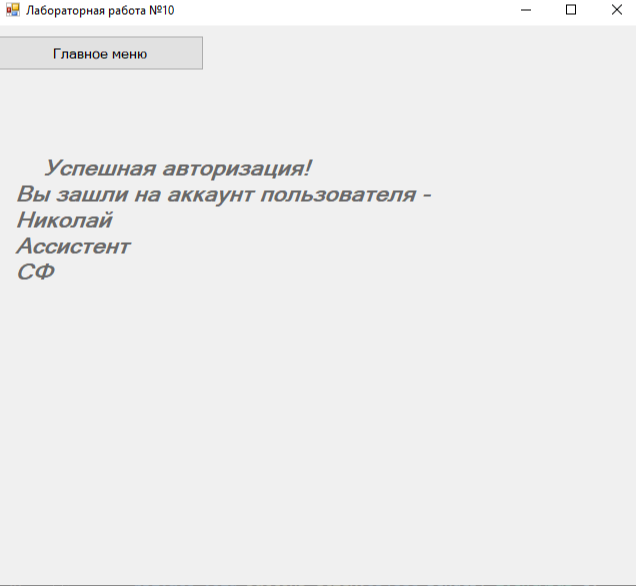


Рисунок 3 – Сообщение об успешной авторизации

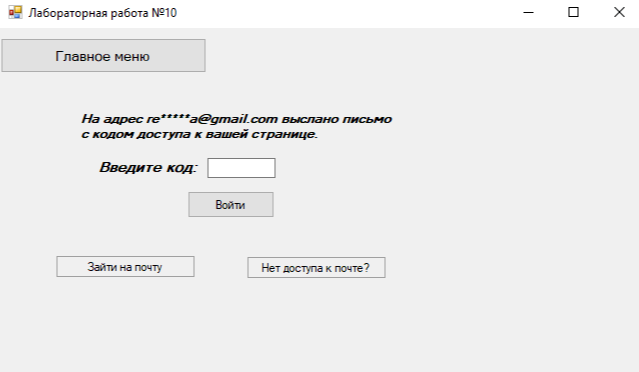


Рисунок 4 – Авторизация через код, высланный на почту

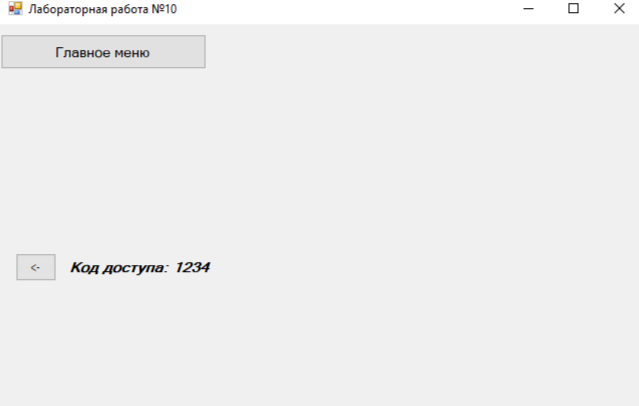


Рисунок 5 – Код на почте и ключ доступа

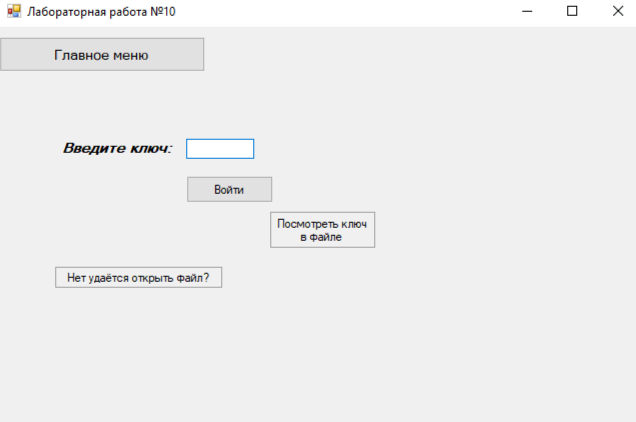


Рисунок 6 – Авторизация через ключ доступа

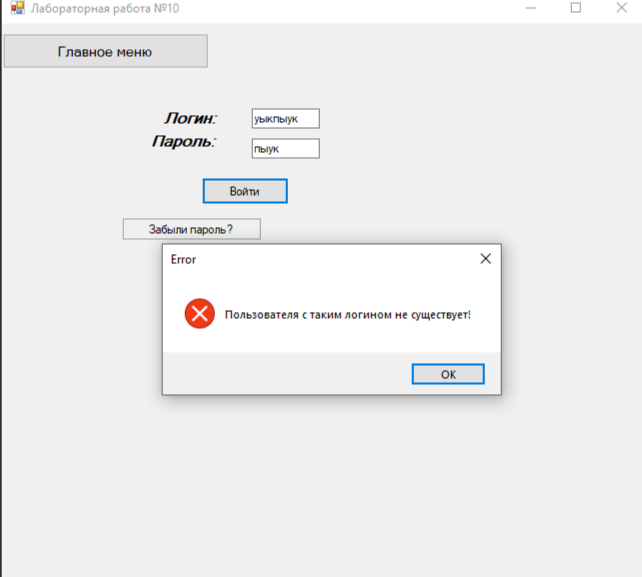


Рисунок 7 – Ошибка при вводе пароля и логина, которые не существуют

Для реализации второго задания мы добавляем на форму различные textbox и combobox. И когда пользователь вводит данные для своего аккаунта, они записываются в файл под названием employees. Также мы создаем условие, чтобы пользователь ввел пароль, который содержит заглавные, строчные буквы, цифры, специальные символы, и не содержит пробелов. В случае, если пользователь введет что-то не так, выскочит ошибка, а если все хорошо, то выведется сообщение об успешной регистрации. Результат работы программы по первому заданию представлен на рисунках 8-11.

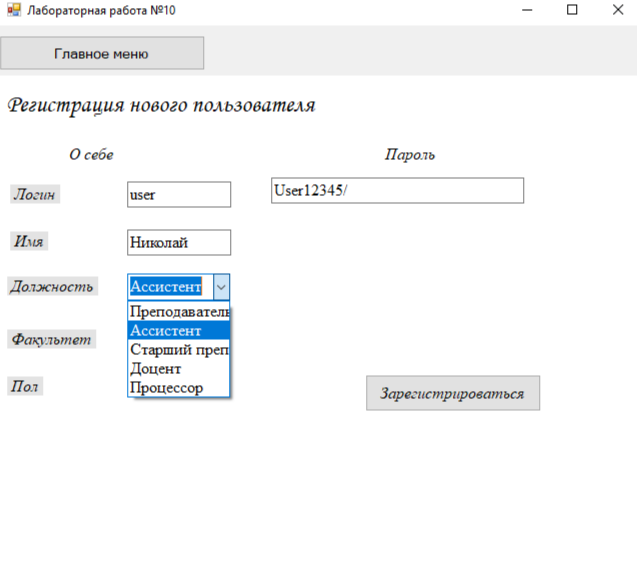


Рисунок 8 – Пример того, что написано в combobox

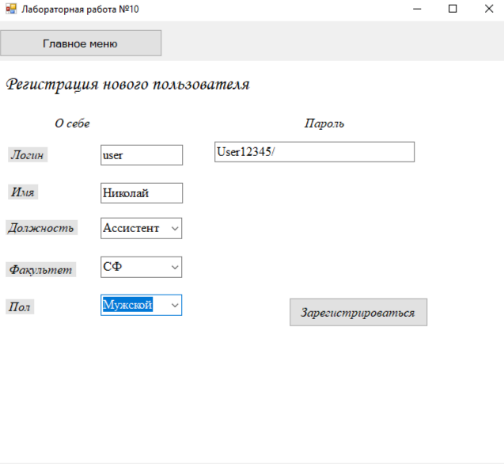


Рисунок 9 – Регистрация пользователя «Николая»

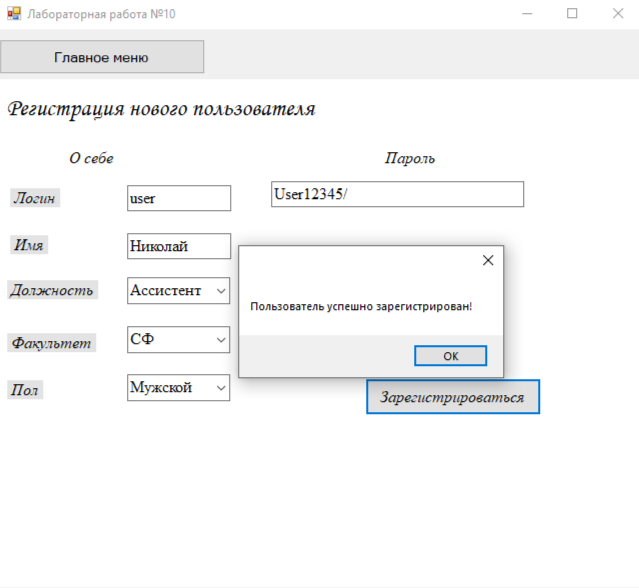


Рисунок 10 – Сообщение об успешной авторизации

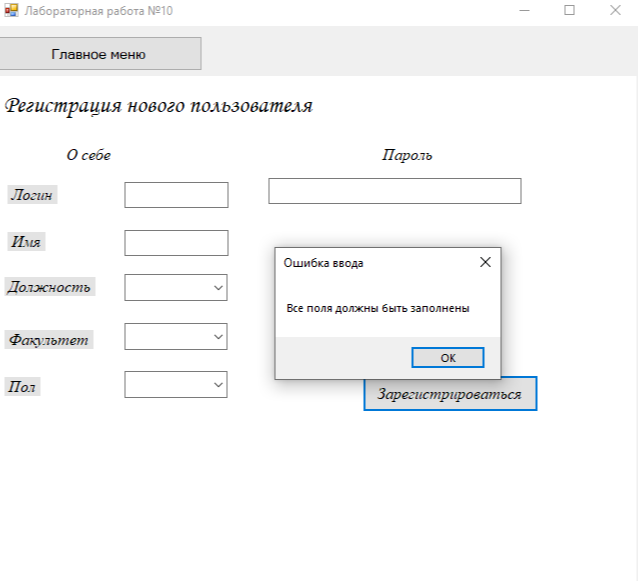


Рисунок 11 – Сообщение при неправильном вводе

Вывод: в результате выполнения лабораторной работы мы изучили основы принципов инкапсуляции на платформе .NET Framework.

Приложение А

Листинг программного кода языка C#

using System;

using System.IO;

using System.Windows.Forms;

namespace LW\_10

{

public partial class Ex\_1\_3 : UserControl

{

public Ex\_1\_3()

{

InitializeComponent();

}

Employee e = new Employee(" Иванов", " Иван", " Иванович - ", " Доцент ", " ФИТКБ ВГТУ ", "abcd", "1234", 1234, 1234);

private void button1\_Click(object sender, EventArgs e)

{

panel1.Visible = !panel1.Visible;

panel2.Visible = !panel2.Visible;

}

private void button2\_Click(object sender, EventArgs e)

{

panel1.Visible = !panel1.Visible;

panel3.Visible = !panel3.Visible;

}

private void button3\_Click(object sender, EventArgs e)

{

panel1.Visible = !panel1.Visible;

panel8.Visible = !panel8.Visible;

}

private void button4\_Click(object sender, EventArgs e)

{

panel2.Visible = !panel2.Visible;

panel5.Visible = !panel5.Visible;

}

private void button6\_Click(object sender, EventArgs e)

{

panel3.Visible = !panel3.Visible;

panel4.Visible = !panel4.Visible;

}

private void button10\_Click(object sender, EventArgs e)

{

panel3.Visible = !panel3.Visible;

panel4.Visible = !panel4.Visible;

}

private void button9\_Click(object sender, EventArgs e)

{

panel3.Visible = !panel3.Visible;

panel10.Visible = !panel10.Visible;

}

private void button16\_Click(object sender, EventArgs e)

{

panel8.Visible = !panel8.Visible;

panel9.Visible = !panel9.Visible;

}

private void button20\_Click(object sender, EventArgs e)

{

panel9.Visible = !panel9.Visible;

panel2.Visible = !panel2.Visible;

}

private void button19\_Click(object sender, EventArgs e)

{

panel9.Visible = !panel9.Visible;

panel3.Visible = !panel3.Visible;

}

private void button12\_Click(object sender, EventArgs e)

{

panel3.Visible = !panel3.Visible;

panel5.Visible = !panel5.Visible;

}

private void button8\_Click(object sender, EventArgs e)

{

panel5.Visible = !panel5.Visible;

panel8.Visible = !panel8.Visible;

}

private void button21\_Click(object sender, EventArgs e)

{

panel2.Visible = !panel2.Visible;

panel10.Visible = !panel10.Visible;

}

private void button5\_Click(object sender, EventArgs e)

{

StreamReader reader;

try

{

reader = new StreamReader(Directory.GetCurrentDirectory() + $"\\employees\\{Login\_input.Text}.txt");

string pass = reader.ReadLine().Remove(0, 8);

if (string.Compare(pass, password.Text) != 0)

throw new Exception("Неправильный пароль");

}

catch (FileNotFoundException)

{

MessageBox.Show("Пользователя с таким логином не существует!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

panel2.Visible = !panel2.Visible;

label14.Visible = !label14.Visible;

string username = reader.ReadLine().Remove(0, 18);

string position = reader.ReadLine().Remove(0, 11);

string faculty = reader.ReadLine().Remove(0, 11);

reader.Close();

label14.Text = ($" Успешная авторизация!\nВы зашли на аккаунт пользователя - \n{username}\n{position}\n{faculty}");

}

private void button23\_Click(object sender, EventArgs e)

{

panel2.Visible = !panel2.Visible;

panel11.Visible = !panel11.Visible;

}

private void button24\_Click(object sender, EventArgs e)

{

panel2.Visible = !panel2.Visible;

panel11.Visible = !panel11.Visible;

}

private void button7\_Click(object sender, EventArgs e)

{

try

{

this.e.Code = Convert.ToInt32(textBox1.Text);

if (this.e.Code != 1234)

{

MessageBox.Show("Вы ввели неправильный код!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

panel3.Visible = !panel3.Visible;

label14.Visible = !label14.Visible;

label14.Text = (" Успешная авторизация!\nВы зашли на аккаунт пользователя - \n" + this.e.Surname + this.e.Name + this.e.Middlename + " \n" + this.e.Post + this.e.Faculty);

}

}

catch

{

MessageBox.Show("Вы не ввели ничего, введите код авторизации!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void button18\_Click(object sender, EventArgs e)

{

try

{

this.e.Key = Convert.ToInt32(textBox2.Text);

if (this.e.Key != 1234)

{

MessageBox.Show("Вы ввели неправильный ключ!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

panel8.Visible = !panel8.Visible;

label14.Visible = !label14.Visible;

label14.Text = (" Успешная авторизация!\nВы зашли на аккаунт пользователя - \n" + this.e.Surname + this.e.Name + this.e.Middlename + " \n" + this.e.Post + this.e.Faculty);

}

}

catch

{

MessageBox.Show("Вы не ввели ничего, введите ключ!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void button17\_Click(object sender, EventArgs e)

{

if (openFileDialog1.ShowDialog() == DialogResult.Cancel)

return;

string filename = openFileDialog1.FileName;

string fileText = File.ReadAllText(filename);

richTextBox1.Text = fileText;

richTextBox1.Visible = !richTextBox1.Visible;

}

private void button22\_Click(object sender, EventArgs e)

{

panel8.Visible = !panel8.Visible;

panel10.Visible = !panel10.Visible;

}

}

}

Приложение B

Листинг программного кода языка C#

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace LW\_10

{

public partial class Ex\_2 : UserControl

{

public Ex\_2()

{

InitializeComponent();

}

private void registerButton\_Click\_1(object sender, EventArgs e)

{

string login = textBox3.Text;

string username = textBox2.Text;

string password = textBox1.Text;

string position = positionComboBox.SelectedItem?.ToString();

string faculty = facultyComboBox.SelectedItem?.ToString();

string gender = genderComboBox.SelectedItem?.ToString();

try

{

ValidateInput(username, password, position, faculty, gender);

// Создание нового файла с данными пользователя в папке employees

string folderPath = "employees";

if (!Directory.Exists(folderPath))

{

Directory.CreateDirectory(folderPath);

}

string userFilePath = Path.Combine(folderPath, $"{login}.txt");

using (StreamWriter writer = File.CreateText(userFilePath))

{

writer.WriteLine($"Пароль: {password}");

writer.WriteLine($"Имя пользователя: {username}");

writer.WriteLine($"Должность: {position}");

writer.WriteLine($"Факультет: {faculty}");

writer.WriteLine($"Пол: {gender}");

// Другие данные пользователя могут быть добавлены сюда

}

MessageBox.Show("Пользователь успешно зарегистрирован!");

}

catch (ArgumentException ex)

{

MessageBox.Show(ex.Message, "Ошибка ввода");

}

}

private void ValidateInput(string username, string password, string position, string faculty, string gender)

{

// Проверка наличия данных в полях

if (string.IsNullOrWhiteSpace(username) || string.IsNullOrWhiteSpace(password) ||

string.IsNullOrWhiteSpace(position) || string.IsNullOrWhiteSpace(faculty) ||

string.IsNullOrWhiteSpace(gender))

{

throw new ArgumentException("Все поля должны быть заполнены");

}

// Проверка пароля

if (!IsStrongPassword(password))

{

throw new ArgumentException("Пароль не соответствует требованиям (минимум 8 символов, цифры, заглавные и строчные буквы, спец. символы)");

}

}

private bool IsStrongPassword(string password)

{

// Проверка пароля на условия (минимум 8 символов, цифры, заглавные и строчные буквы, спец. символы)

var regex = new Regex(@"^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[^\da-zA-Z]).{8,}$");

return regex.IsMatch(password) && !password.Contains(" ");

}

}

}