Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ОТЧЕТ

К лабораторной работе №3 (Разработка пользовательского интерфейса)

по дисциплине «Системное программное обеспечение»

Выполнил студент

группы 538:

\_\_\_\_\_\_\_\_ Асмоловский А.В.

Проверил

доцент кафедры КСУП:

\_\_\_\_\_\_\_\_ Коцубинский В.П.

Дата: «\_\_\_»\_\_\_\_\_\_\_\_2020 г.

Томск 2020

Оглавление

[**1 Цель и задачи лабораторной работы** 3](#_Toc57375268)

[**2 Ход лабораторной работы** 4](#_Toc57375269)

[**Главное окно интерфейса MainForm** 4](#_Toc57375270)

[**Окно добавления или редактирования заметок EditNote** 8](#_Toc57375271)

[**Окно About** 10](#_Toc57375272)

[**История коммитов** 11](#_Toc57375273)

[**3 Заключение** 12](#_Toc57375274)

**1 Цель и задачи лабораторной работы**

Цель работы:изучить разработку адаптивного пользовательского интерфейса десктоп-приложения.

Задачи:

1.Изучить использование фреймворка пользовательского интерфейса WindowsForms и его компоненты.

2.Освоить компоненты и свойства для создания адаптивного дизайна.

3.Освоить обработку событий компонентов WindowsForms.

4.Реализовать передачу данных между разрабатываемыми формами и пользовательскими элементами управления.

5. Реализовать защиту от некорректного ввода.

**2 Ход лабораторной работы**

В ходе лабораторной работы мы реализовали следующие типы WindowsForms:

**Главное окно интерфейса MainForm**

Главное окно интерфейса MainForm, представлен на рисунке 2.1:

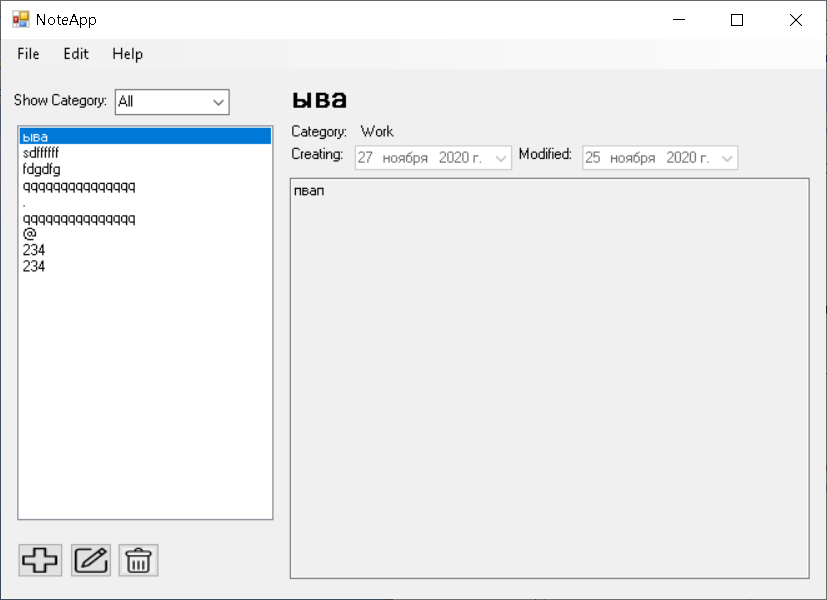


Рисунок 2.1 — Главное окно - MainForm

Также код, который осуществляет взаимодействие между представленным интерфейсом и разработанной раннее нами бизнес-логики:

Первым делом заполним категориями CategoryComboBox:

CategoryComboBox.Items.Add("All");

CategoryComboBox.Items.Add(NoteCategory.Work);

CategoryComboBox.Items.Add(NoteCategory.Home);

CategoryComboBox.Items.Add(NoteCategory.HealthAndSport);

CategoryComboBox.Items.Add(NoteCategory.Peoples);

CategoryComboBox.Items.Add(NoteCategory.Documents);

CategoryComboBox.Items.Add(NoteCategory.Finance);

CategoryComboBox.Items.Add(NoteCategory.Other);

Дальше создаем сущность AllNotes. AllNotes – это сущность отвечающая за все заметки

Код создания представлен ниже

Project AllNotes = newProject();

Код работы кнопок NewButton, ChangeButton, DeleteButton и верхнего элементов ToolStripMenu (exit, addNote, editNote, removeNote, about):

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

{

Note newNote = new Note()

{

LastChangeTime=DateTime.Now};

EditNote noteForm = new EditNote(newNote);

var result = noteForm.ShowDialog();

var realIndex = NotesList.SelectedIndex;

if (result == DialogResult.OK)

{

AllNotes.NoteList.Add(newNote);

ProjectManager.SaveToFile(AllNotes);

if (AllNotes.NoteList.Count != 0)//если есть заметка, то выводим ее

{

LastNote();

}

}

}

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

{

if(NotesList.SelectedIndex==-1)

{ return; }

var note = AllNotes.NoteList[NotesList.SelectedIndex];

var realIndex = NotesList.SelectedIndex;

EditNote noteForm = new EditNote(note);

var result =noteForm.ShowDialog();

if (result == DialogResult.OK)

{

note = noteForm.\_note;

AllNotes.NoteList.RemoveAt(realIndex);

AllNotes.NoteList.Add(note);

ProjectManager.SaveToFile(AllNotes);

NotesList.SelectedIndex = realIndex;

if (AllNotes.NoteList.Count != 0)//если есть заметка, то выводим ее

{

LastNote();

}

}

}

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

{

if (NotesList.SelectedIndex == -1)

{ return; }

DialogResult result = MessageBox.Show("Do you realy want to delete "+AllNotes.NoteList[NotesList.SelectedIndex].Name+" ?", "Удаление заметки",MessageBoxButtons.OKCancel);

if (result == DialogResult.OK)

{

AllNotes.NoteList.RemoveAt(NotesList.SelectedIndex);

ProjectManager.SaveToFile(AllNotes);

if (AllNotes.NoteList.Count == 0)

EmptyNote();

if (AllNotes.NoteList.Count != 0)

LastNote();

}

}

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

{

Application.Exit();

}

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

{

CreateButton\_Click(sender,e);

}

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

{

ChangeButton\_Click(sender, e);

}

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

{

DeleteButton\_Click(sender, e);

}

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

{

AboutForm form = new AboutForm();

form.ShowDialog();

}

И код для работы NoteList:

private void NotesList\_SelectedIndexChanged(object sender, EventArgs e)

{

if (NotesList.SelectedIndex == -1)

{

return;

}

HeadingLabel.Text = AllNotes.NoteList[NotesList.SelectedIndex].Name;// вывод всех данных на форму

SelectedCategoryLabel.Text = Convert.ToString((NoteCategory)AllNotes.NoteList[NotesList.SelectedIndex].Category);

NoteTextBox.Text = AllNotes.NoteList[NotesList.SelectedIndex].Text;

ModifiedDateTimePicker.Value = AllNotes.NoteList[NotesList.SelectedIndex].LastChangeTime;

CreationDateTimePicker.Value = AllNotes.NoteList[NotesList.SelectedIndex].CreatingTime;

CorrectNameLenght();

}

Метод CorrectNameLenght используемый в NoteList, который осуществляет проверку вводимых значений в поле Name:

private void CorrectNameLenght()//2 уровень проверки правильности ввода

{

for(int i =0;i<AllNotes.NoteList.Count;i++)

{

if(AllNotes.NoteList[i].Name.Length>50)

{

HeadingLabel.ForeColor = Color.Red;

}

else

HeadingLabel.ForeColor = Color.Black;

}

}

**Окно добавления или редактирования заметок EditNote**

EditNote – это окно для добавления или редактирования заметок, представлено на рисунке 2.2:

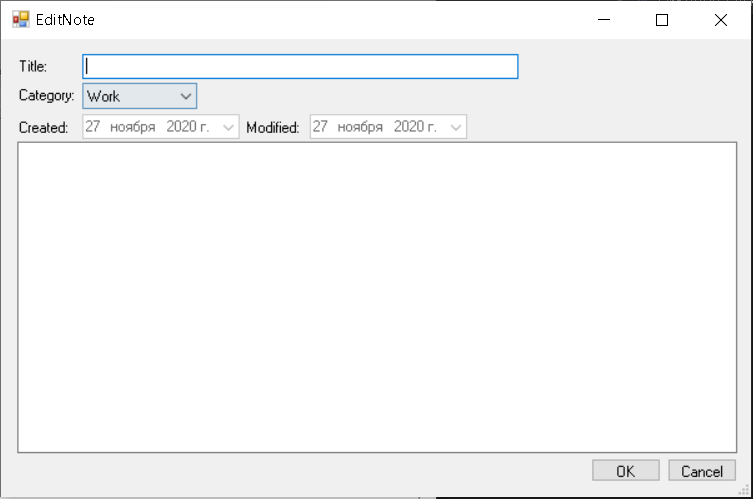


Рисунок 2.2 — EditNote.

Код, который осуществляет взаимодействие между представленным интерфейсом и разработанной раннее нами бизнес-логики:

Сначала создаем переменную типа Note и открытую функцию для передачи значения, созданной нами переменной в форму:

public Note \_note;

При нажатии кнопки OK на форме, выполняется следующий код:

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

{

\_note.Name = HeadingTextBox.Text;

if(Regex.IsMatch(\_note.Name, @"[^\w\.@-]"))//проверка спец символы

{

HeadingTextBox.ForeColor = Color.Red;

return;

}

else

HeadingTextBox.ForeColor = Color.Black;

\_note.Text = NoteText.Text;

\_note.Category = (NoteCategory)CategoryComboBox.SelectedItem;

\_note.LastChangeTime = ModifyingDateTimePicker2.Value;

this.DialogResult = DialogResult.OK;

this.Close();

}

При нажатии кнопки Cancel на форме, выполняется следующий код:

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

{

this.DialogResult = DialogResult.Cancel;

this.Close();

}

## **Окно About**

Окно «About» представлена на рисунке 2.3:

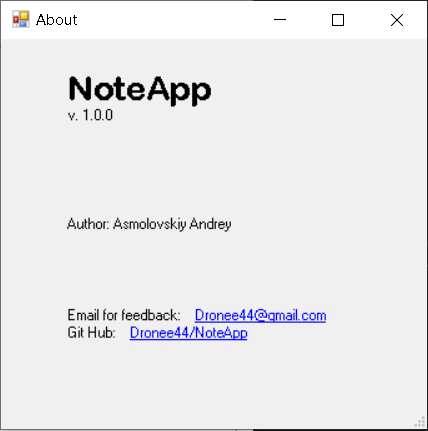


Рисунок 2.3 – Окно «About».

## **История коммитов**

Просмотрим историю коммитов ветки develop, чтобы убедиться, что мы все сделали верно:

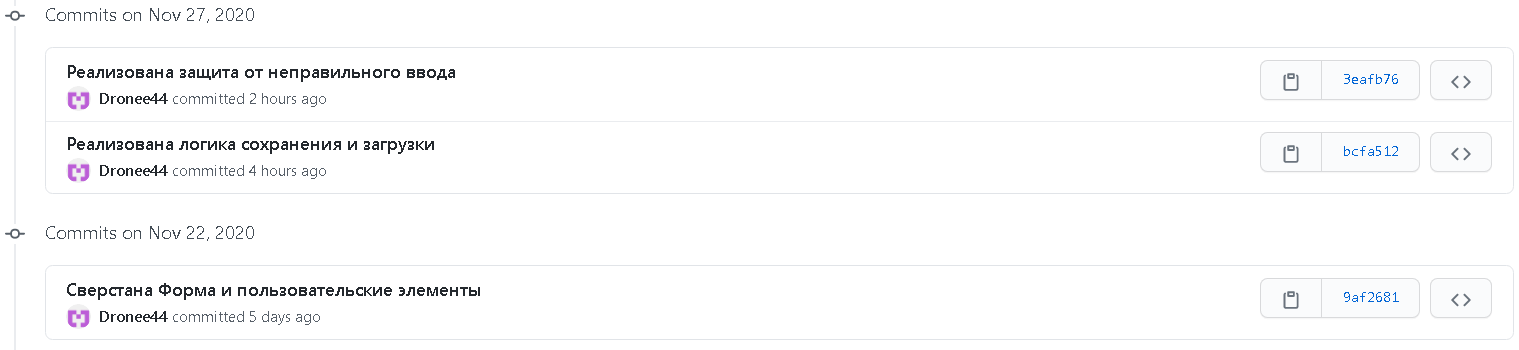


Рисунок 2.4 - История коммитов ветки develop.

# **3 Заключение**

В ходе данной лабораторной работы мы разработали пользовательский интерфейс нашей программы и связали его с нашей бизнес-логикой, разработанной в лабораторной работе №2. Тем самым мы получили первую рабочею версию программы NoteApp.

Задачи на лабораторную работы выполнены.