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:

\_\_\_\_\_\_\_\_ Черников А.И.

Проверил

доцент кафедры КСУП:

\_\_\_\_\_\_\_\_ Коцубинский В.П.

Дата: «\_\_\_»\_\_\_\_\_\_\_\_2020 г.

Томск 2020

Оглавление

[**1 Цель и задачи лабораторной работы** 3](#_Toc56003791)

[**2 Ход лабораторной работы** 4](#_Toc56003792)

[**Главное окно интерфейса MainForm** 4](#_Toc56003793)

[**Окно добавления или редактирования заметок NoteForm** 10](#_Toc56003794)

[**Окно About** 14](#_Toc56003795)

[**История коммитов** 15](#_Toc56003796)

[**3 Заключение** 16](#_Toc56003797)

**1 Цель и задачи лабораторной работы**

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

Задачи:

1. Изучить использование фреймворка пользовательского интерфейса Windows Forms и его компоненты.

2. Освоить компоненты и свойства для создания адаптивного дизайна.

3. Освоить обработку событий компонентов Windows Forms.

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 Windows Forms:

**Главное окно интерфейса MainForm**

Главное окно интерфейса MainForm, представлен на рисунке 2.1:

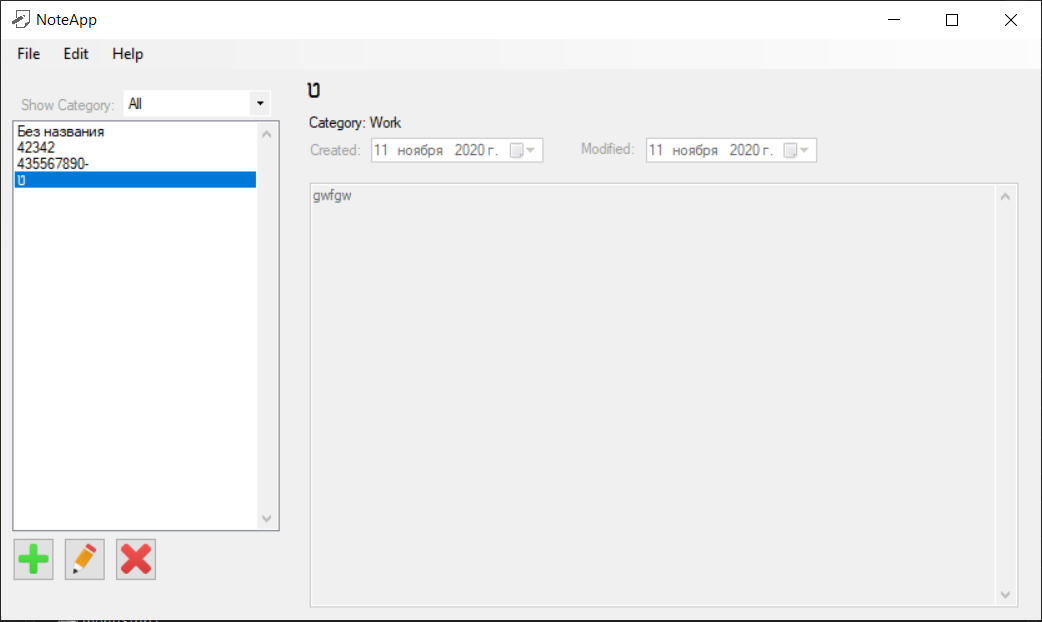


Рисунок 2.1 - Главное окно - MainForm

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

Первым делам заполним категориями CategorysComboBox:

CategorysComboBox.Items.Add("All");

foreach (Category element in Enum.GetValues(typeof(Category)))

{

CategorysComboBox.Items.Add(element);

}

Дальше создаем сущности allNotes и sortNotes. allNotes – это сущность отвечает за все заметки, а sortNotes – это та же allNotes, но отсортированнная по категориям. Код создания представлен ниже:

Project allNotes = new Project();

List<Note> sortNotes = new List<Note>();

Потом создаем функции CreateNote(), ChangeNote() и DeleteNote():

private void CreateNote()

{

//получаем выбранную заметку

Note newNote = new Note(string.Empty, Category.Work, string.Empty, DateTime.Now); //сама заметка

newNote.TimeLastChange = DateTime.Now;

var inner = new NoteForm(); //создаем форму

inner.Note = newNote; //передаем форме данные

//если было нажато Cancel завершаем выполнение обработчика

if (inner.ShowDialog() == DialogResult.OK)

{

var updatedNote = inner.Note; //забираем измененные данные

//добавляем новую заметку в список

allNotes.Glossary.Add(updatedNote);

var changeTitle = updatedNote.Name;

FillListbox();

TitlesListBox.SelectedItem = changeTitle;

ProjectManager.SaveToFile(allNotes);

}

}

private void ChangeNote()

{

//если заметка не выбрана завершаем выполнение обработчика(ничего не происходит при нажатии на "Изменить")

if (TitlesListBox.SelectedIndex == -1)

{

return;

}

//получаем выбранную заметку

var selectedIndex = TitlesListBox.SelectedIndex; //индекс нашей заметки в списке всех заметок allNotes

var selectedNote = sortNotes[selectedIndex]; //сама заметка

var inner = new NoteForm(); //создаем форму

inner.Note = selectedNote; //передаем форме данные

//если было нажато Cancel завершаем выполнение обработчика

if (inner.ShowDialog() == DialogResult.OK)

{

var updatedNote = inner.Note; //забираем измененные данные

//удалить и заменить старые данные

allNotes.Glossary.RemoveAt(allNotes.RealIndexes[selectedIndex]);

allNotes.Glossary.Add(updatedNote);

FillListbox();

var changeTitle = updatedNote.Name;

TitlesListBox.SelectedItem = changeTitle;

ProjectManager.SaveToFile(allNotes);

}

}

private void DeleteNote()

{

//если заметка не выбрана завершаем выполнение обработчика(ничего не происходит при нажатии на "Удалить")

if (TitlesListBox.SelectedIndex == -1)

{

return;

}

if (MessageBox.Show("Вы уверены что хотите удалить заметку?", "Удаление", MessageBoxButtons.OKCancel, MessageBoxIcon.Question) == DialogResult.OK)

{

//получаем выбранную заметку

var selectedIndex = TitlesListBox.SelectedIndex; //индекс нашей заметки в списке всех заметок allNotes

TitlesListBox.Items.RemoveAt(selectedIndex);

allNotes.Glossary.RemoveAt(allNotes.RealIndexes[selectedIndex]);

allNotes.\_currentNote = -1;

FillListbox();

ClearRigthPanel();

splitContainerNote.Panel2.Visible = false;

ProjectManager.SaveToFile(allNotes);

}

}

Функция FillListbox(), упомянутая в CreateNote(), ChangeNote() и DeleteNote(), работает с TitleListBox (заголовками заметок):

private void FillListbox()

{

//проверка на null(если заметок еще нет)

if (allNotes != null)

{

TitlesListBox.Items.Clear();

sortNotes = allNotes.SortWithSelectionCategory(CategorysComboBox.SelectedIndex);

{

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

{

TitlesListBox.Items.Add(sortNotes[i].Name);

}

}

}

}

Функция ClearRigthPanel():

private void ClearRigthPanel()

{

CategoryTextBox.Clear();

TitleTextBox.Clear();

NoteTextBox.Clear();

}

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

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

{

ChangeNote();

}

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

{

CreateNote();

}

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

{

DeleteNote();

}

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

{

Close();

}

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

{

CreateNote();

}

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

{

ChangeNote();

}

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

{

DeleteNote();

}

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

{

var about = new About(); //создаем форму

about.ShowDialog();

}

И код для работы TitlesListBox:

private void TitlesListBox\_SelectedIndexChanged\_1(object sender, EventArgs e)

{

//завершение метода если элемент не выбран

if (TitlesListBox.SelectedIndex == -1)

{

return;

}

allNotes.\_currentNote = allNotes.RealIndexes[TitlesListBox.SelectedIndex];

ProjectManager.SaveToFile(allNotes);

ClearRigthPanel();

splitContainerNote.Panel2.Visible = true;

sortNotes = allNotes.SortWithSelectionCategory(CategorysComboBox.SelectedIndex);

//заполняем данными правую часть окна

TitleTextBox.Text = sortNotes[TitlesListBox.SelectedIndex].Name;

CategoryTextBox.Text = "Category: " + sortNotes[TitlesListBox.SelectedIndex].Category;

CreateDateTimePicker.Value = sortNotes[TitlesListBox.SelectedIndex].TimeCreation;

ChangeDateTimePicker.Value = sortNotes[TitlesListBox.SelectedIndex].TimeCreation;

NoteTextBox.Text = sortNotes[TitlesListBox.SelectedIndex].Text;

}

**Окно добавления или редактирования заметок NoteForm**

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

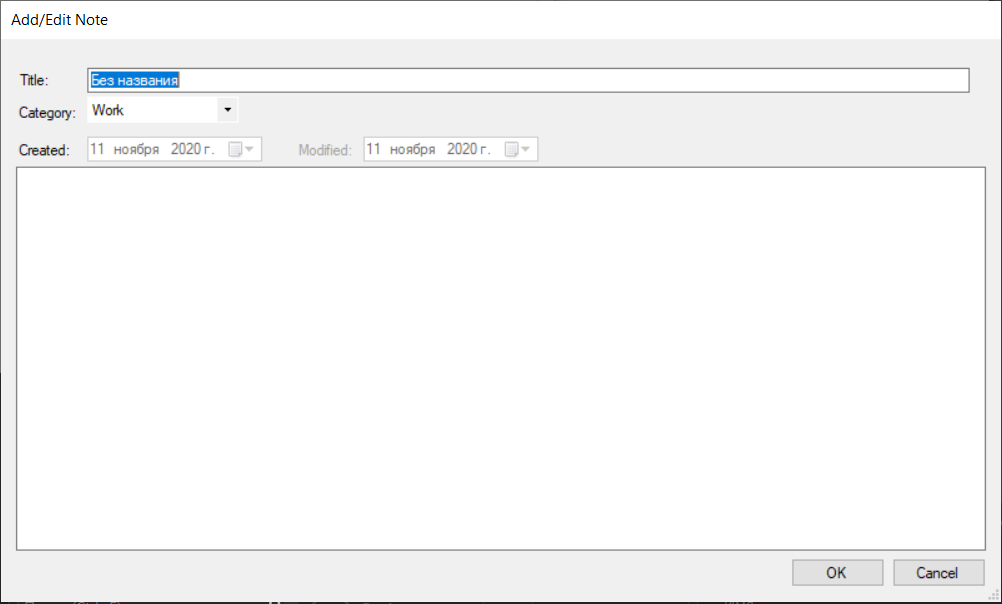


Рисунок 2.2 - NoteForm.

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

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

private Note \_note;

public Note Note

{

get { return \_note; }

set

{

\_note = value;

if (\_note != null)

{

TitleTextBox.Text = \_note.Name; //при отправлении заметки в это окно, автоматом заполнятся компоненты формы

CategorysComboBox.SelectedItem = \_note.Category;

CreateDateTimePicker.Value = \_note.TimeCreation;

ChangeDateTimePicker.Value = \_note.TimeLastChange;

NoteTextBox.Text = \_note.Text;

}

}

}

Дальше передаем CategorysComboBox наши возможные категории:

foreach (Category element in Enum.GetValues(typeof(Category)))

{

CategorysComboBox.Items.Add(element);

}

Потом создаем функции для проверки того, что в поля введены имя, текст и категория заметки:

private bool CheckTitle

{

get

{

if (TitleTextBox.Text == string.Empty)

{

return true;

}

else

{

return false;

}

}

}

private bool CheckText

{

get

{

if (NoteTextBox.Text == string.Empty)

{

return true;

}

else

{

return false;

}

}

}

private bool CheckCategory

{

get

{

if (CategorysComboBox.SelectedIndex == -1)

{

return true;

}

else

{

return false;

}

}

}

Определение кнопок OkButton и CancelButton:

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

{

if (CheckTitle || CheckText || CheckCategory)

{

MessageBox.Show("Введите название,текст и категорию");

}

else

{

\_note.TimeLastChange = DateTime.Now; // Изменение даты редактирования

\_note.Name = TitleTextBox.Text;

\_note.Text = NoteTextBox.Text;

var selectedIndex = CategorysComboBox.SelectedIndex;

\_note.Category = (Category)CategorysComboBox.Items[selectedIndex];

DialogResult = DialogResult.OK;

this.Close();

}

}

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

{

DialogResult = DialogResult.Cancel;

this.Close();

}

## **Окно About**

Окно «About» представлена на рисунке 2.3:

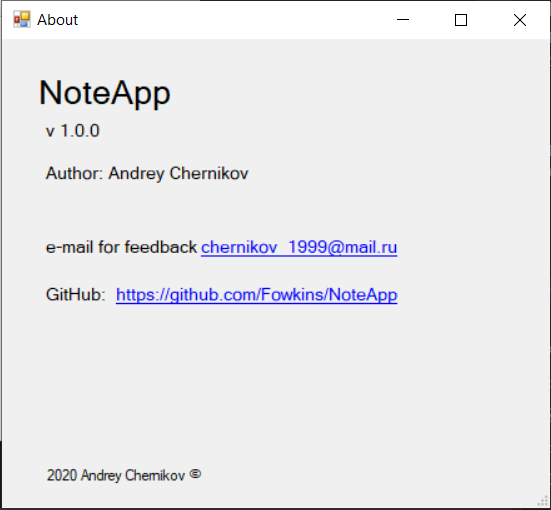


Рисунок 2.3 – Окно «About».

## **История коммитов**

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

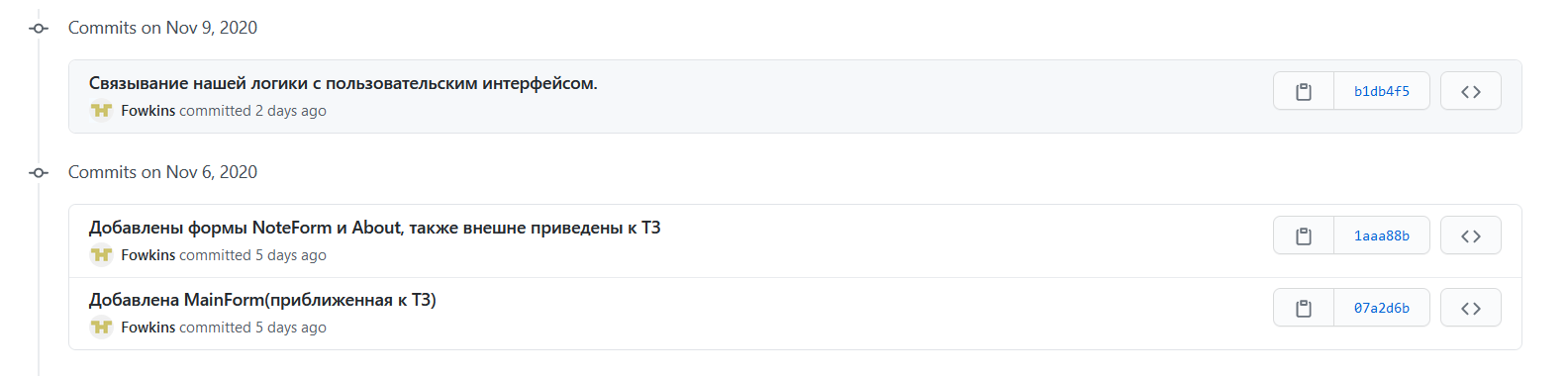


Рисунок 2.4 - История коммитов ветки develop.

# **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.