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)

**РАЗРАБОТКА ПРИЛОЖЕНИЯ NOTEAPP**

Пояснительная записка к программе, разработанной по дисциплине «Новые технологии в программировании»

Студент гр. 588-2

\_\_\_\_\_\_\_\_\_\_\_\_ Робканов.К.Д

(подпись)

« »\_\_\_\_\_\_\_\_ 2021г.

Доцент каф. КСУП

\_\_\_\_\_\_\_\_\_\_\_\_ Горяинов А.Е.

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_2021г.

Томск 2021

**Содержание**

1 Назначение приложения…………………………………………………………..3

2 Группы пользователей и их функциональные возможности в приложении…..4

3 Стек технологий разработки……………………………………………………...5

4 Пользовательский интерфейс……………………………………………………..6

5 Диаграмма пакетов……………………………………………………………….10

6 Диаграмма классов……………………………………………………………….11

7 Описание тестирования приложения…………………………………………...12

8 Сборка установщика……………………………………………………………..15

9 Описание модели ветвления……………………………………………………..20

**1 Назначение приложения**

Пользовательское приложение NoteApp предназначено для ведения персональных записей и заметок. Приложение должно:

– Обеспечивать стабильную работу приложения при порядке 200 заметок.

– ОбеспечиватьВкатегоризацию заметок, навигацию по созданным заметкам.

– Предоставить инструменты для просмотра и редактирования заметок.

– Сохранять и восстанавливать заметки между сессиями приложения.

Выполнять промежуточные сохранения заметок на машине пользователя на случай аварийного завершения программы, отключения компьютера и так далее для защиты от потери данных.

**2 Группы пользователей и их функциональные возможности в приложении**

В приложении предусмотрена одна роль пользователя. Он имеет следующий набор функционала:

– Просмотр существующих заметок;

– Создание новой заметки, редактирование заметки и удаление заметки.

Категоризация и навигация по созданным заметкам по дате изменения.

**3 Стек технологий разработки**

Для реализации проекта был задан язык программирования С# на платформе .NET 4.7.2, набор библиотек Windows Forms для создания десктоп-приложений для операционной системы Windowsи среда разработки Visual Studio 2019. Системные требования обусловлены требованиями к платформе .NET.

Для поддержки сериализации и десериализации как механизма файлового сохранения данных в проекте разработки была использована библиотека Newtonsoft JSON.NET.

Для проведения юнит-тестирования проекта логики приложения в среде Visual Studio была использована библиотека NUnit c ее пакетами NUnit, так как она обеспечивает достаточную гибкость описания тестов, требуемую для данного проекта.

Для создания сценария сборки установочного пакета и компиляции установщика используется программа Inno Setup.

**4 Пользовательский интерфейс**

Вся работа с приложением осуществляется через два окна пользовательского интерфейса – основное окно с полным списком заметок и окно редактирования информации выбранной или новой заметки. Для дополнительной информации о программе и ее разработчике предусмотрено специальное окно, вызываемое из основного.

На рисунке 4.1 показано главное окно приложения (NoteApp). В этом окне пользователь в левой части окна видит список заметок, созданных им, в правой части он видит информацию о выбранной заметке.

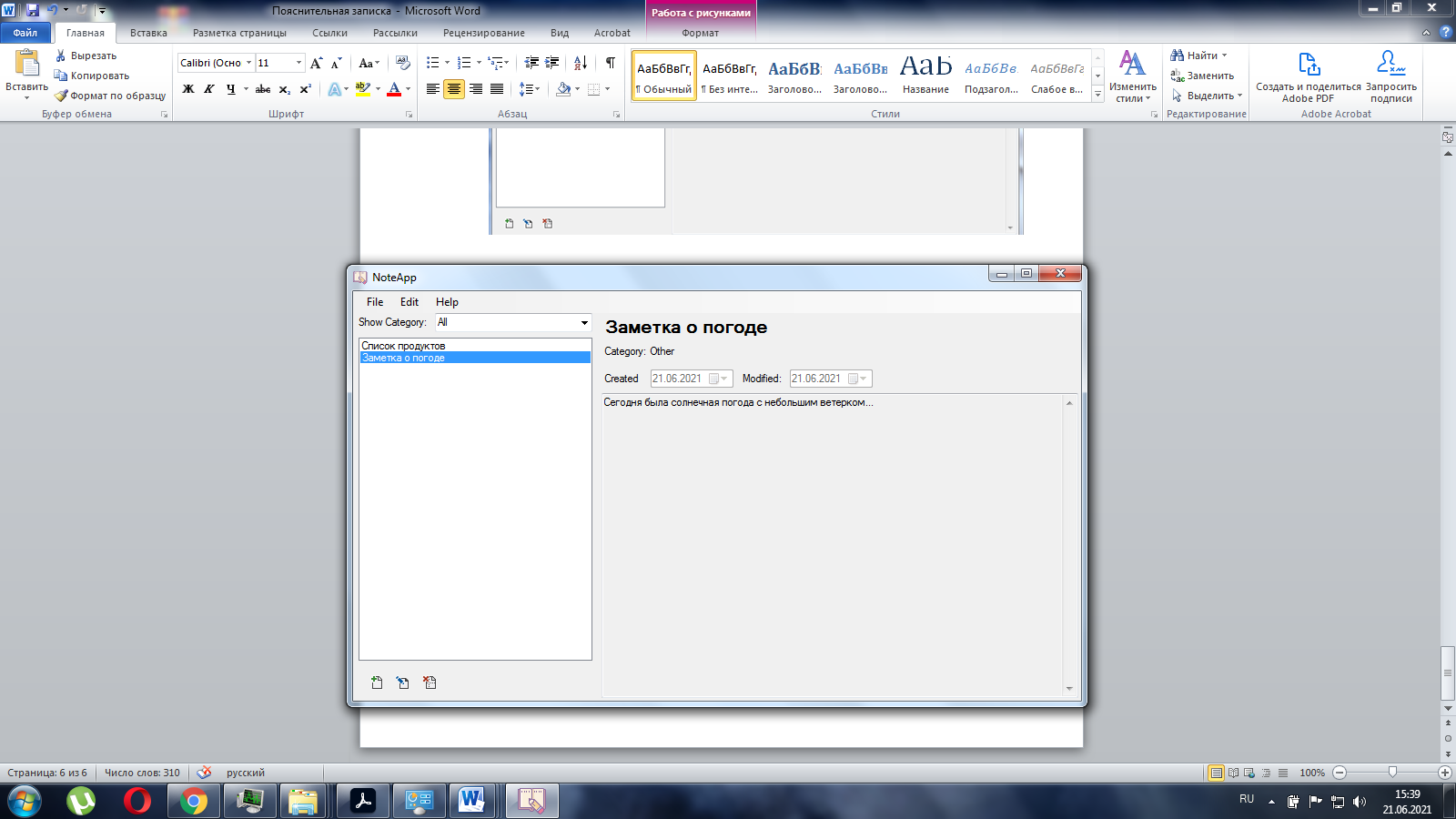


Рисунок 4.1 – Главное окно программы.

Для работы с заметками на форме есть элементы, вызывающие функции создания, редактирования, удаления записей и соответствующую форму. Они выполнены в виде кнопок с рисунками под списком заметок. Также все эти действия продублированы в верхнем меню главного окна во вкладке Edit.

При выборе заметки в списке, выбранная заметка отображается в правой панели. Главное окно не позволяет редактировать содержимое заметки – только просмотр.

На рисунке 4.2 показано окно создания и редактирования заметки (Add/Edit Note). В этом окне пользователь может создать или отредактировать уже существующую заметку.

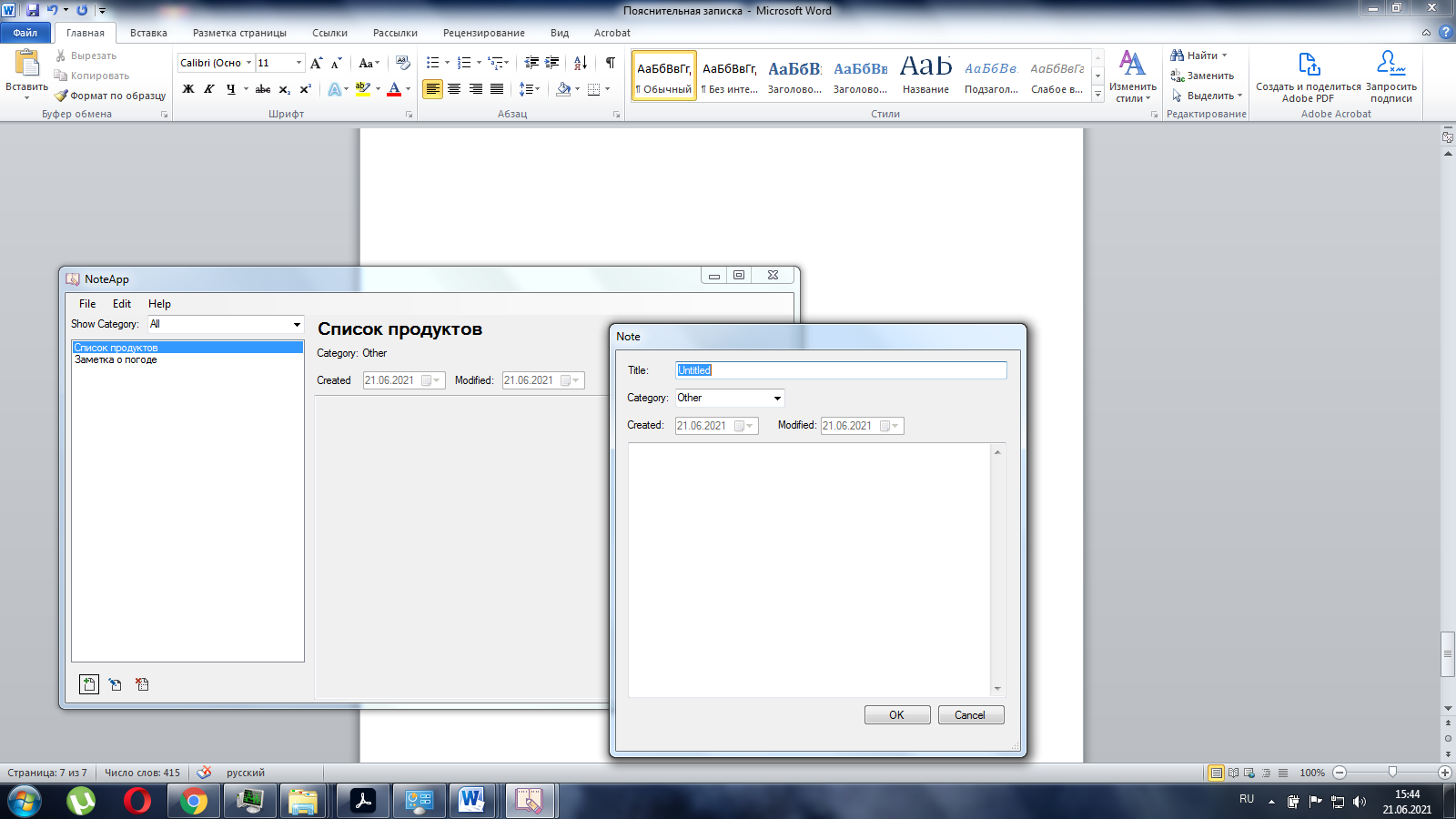


Рисунок 4.2 – Окно создания/редактирования заметки.

При нажатии на кнопку OK окно создания заметки закрывается, в список заметок главного окна добавляется новая заметка. При редактировании текущей заметки, нажатие на кнопку OK должно обновить название заметки в списке заметок, и обновить отображаемую заметку в правой панели приложения. При нажатии кнопки Cancel создание/редактирование заметки отменяется и исходная заметка остается без изменений.

В случае ввода пользователем некорректных данных (нарушение допустимой длины названия заметки), поле окрашивается в красный цвет (Рисунок 4.3).И заметка с некорректными данными не сможет добавится в список заметок.

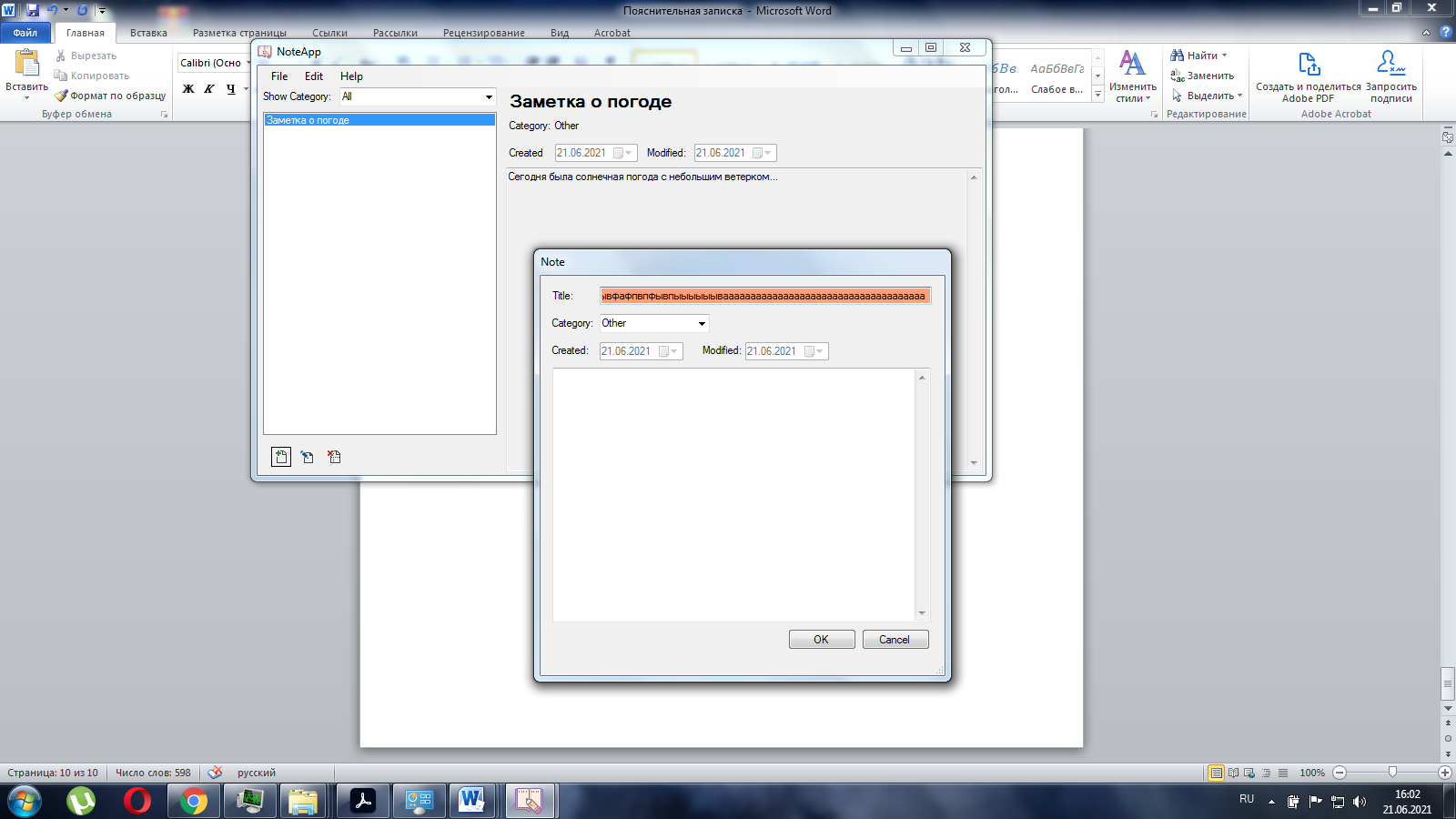


Рисунок 4.3 – Демонстрация превышения длины имени

При вызове окна изменения или добавления заметки происходит передача данных сначала из основной формы в вызываемую, а затем после подтверждения изменений осуществляется обратная передача данных и сохранение их в списке всех заметок. Удаление происходит с вызовом диалогового окна, на котором требуется подтвердить выбранное действие.

Из основного окна программы можно открыть еще одно окно (рисунок 4.3), содержащее информацию о приложении и его разработчике. Для этого необходимо в меню главного окна перейти в раздел About. В этой форме нет доступных для использования элементов.

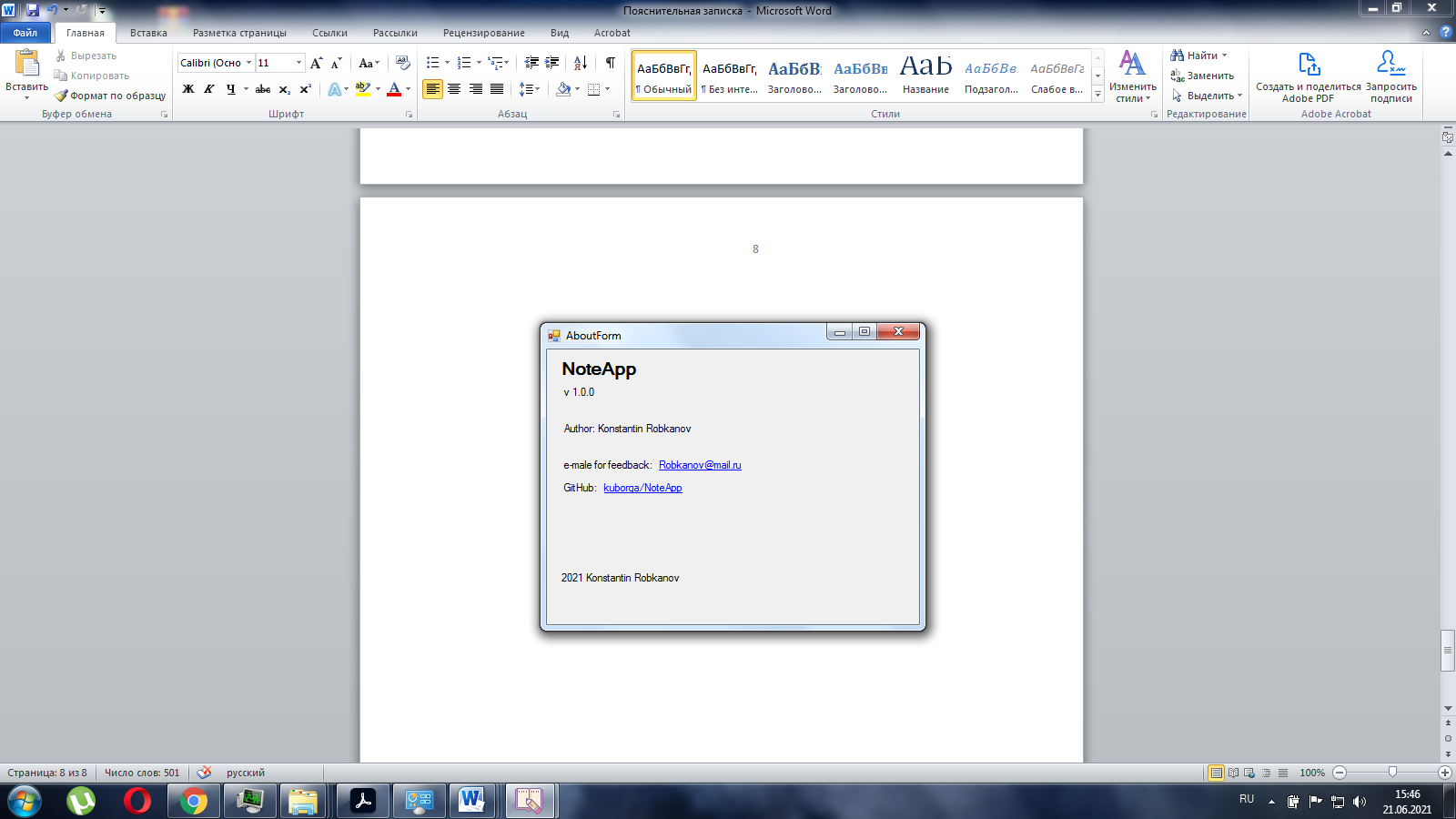


Рисунок 4.3 – Интерфейс окна дополнительной информации

Таким образом, в приложении реализовано три окна:

– главное окно;

– окно создания/редактирования заметки;

– окно «About».

Верстка главного окна и окна создания/редактирования заметки адаптивны. Окно «About» имеет фиксированный размер.

Загрузка заметок осуществляется при запуске программы до вывода главного окна пользователю, сохранение заметок в файл выполняется в случаях: а) создания новой заметки; б) удаления заметки; в) закрытия приложения.

Также в приложении запоминается текущая заметка. При закрытии приложения индекс текущей заметки сохраняется в файл, а при запуске – загружается из файла, что позволяет при повторном открытии приложения отобразить пользователю последнюю просмотренную им заметку.

**5 Диаграмма пакетов**

Диаграмма пакетов отображает архитектуру приложения, разделенную на отдельные пакеты – библиотеки. Внутри библиотек описываются доступные извне классы, между пакетами рисуются направленные линии, обозначающие связи между библиотеками. Диаграмма пакетов разработанного приложения представлена на рисунке 5.1.

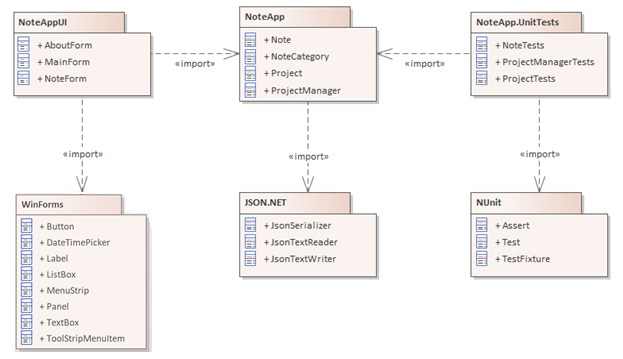


Рисунок 5.1 – Диаграмма пакетов приложения

**6 Диаграммы классов**

Диаграмма классов приложения представлена на рисунке 6.1.

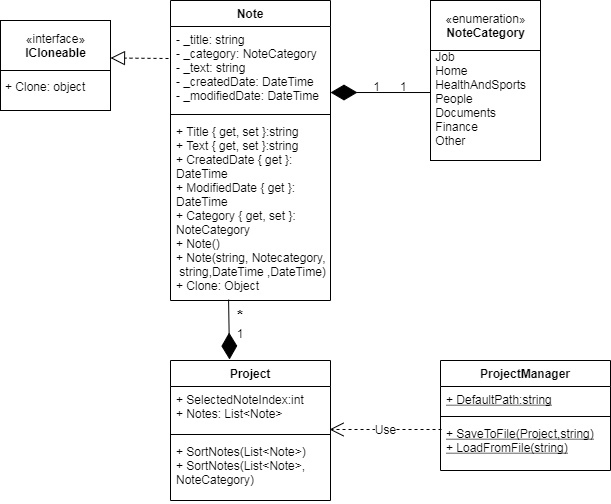


Рисунок 6.1 – Диаграмма классов бизнес-логики

**7 Описание тестирования приложения**

В данном проекте основным типом тестирования было юнит-тестирование. Это тестирование минимальных модулей архитектуры, максимально изолированных друг от друга. Так как минимальными модулями архитектуры, как правило, являются классы, в разработанном приложении тестировался проект бизнес-логики.

Полный список юнит-тестов представлен на рисунке 7.1.

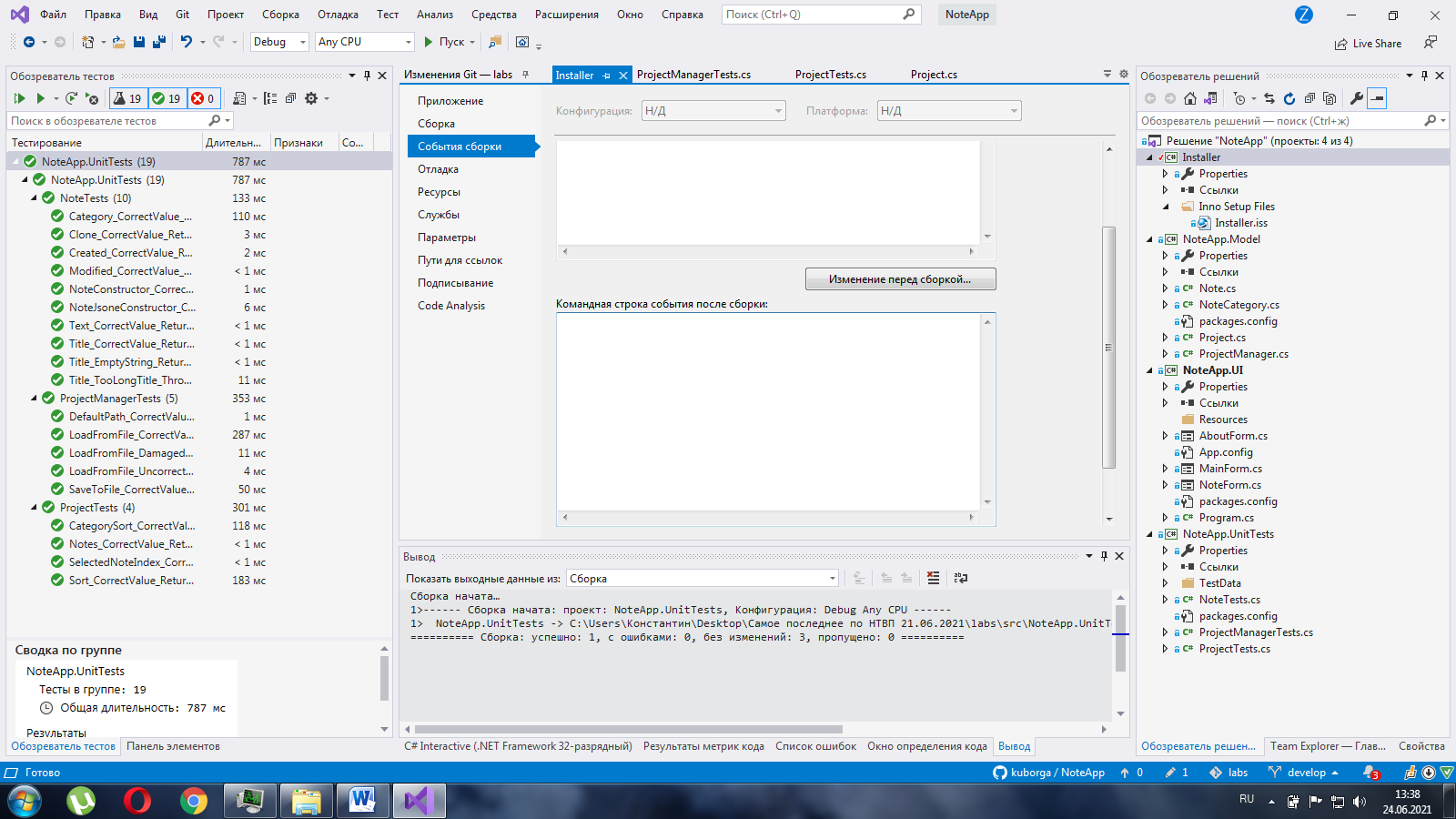


Рисунок 7.1 – Полный список юнит-тестов

Для приёмочного тестирования проводилась следующая последовательность действий:

1.Установка приложения на компьютер с помощью собранного установочного пакета.

2. Запуск приложения. Окно программы должно быть пустым.

3. Создать три контакта в приложении с разными фамилиями.

4. Переключиться между контактами, убедиться, что смена текущего контакта в правой панели происходит корректно.

5. Ввести в поиск подстроку для поиска контактов - в списке контактов должны остаться только контакты, содержащие подстроку.

6. Введите в поиск подстроку, которой нет в фамилиях и именах контактов - список контактов должен быть пустым. Удалить подстроку поиска - список контактов должен восстановиться.

7. Выбрать любой контакт и нажать кнопку редактирования. Должно открыться окно редактирования контакта.

8. Ввести фамилию более 50 символов. Убедиться, что элемент управления с фамилией указывает на некорректное значение.

9. Ввести фамилию менее 50 символов. Убедиться, что элемент управления с фамилией становиться корректным.

10. Показать, что защита от некорректных значений также работает и для других полей.

11. Поменять фамилию контакта, отличную от исходной. Поменять номер телефона. Нажать «OK». Отредактированный контакт должен переместиться в списке контактов согласно алфавиту, отображаемые данные текущего контакта в правой панели также должны измениться.

12. Выбрать любой контакт и нажать кнопку редактирования. Должно открыться окно редактирования. Изменить фамилию контакта, номер телефона и e-mail. Нажать «Cancel». Исходный контакт должен остаться без изменений.

13. Удалить третий контакт.

14. Закрыть приложение. Должно произойти сохранение контактов в целевой файл.

15. Запустить приложение. В программе должны восстановиться контакты, созданные в предыдущую сессию.

16. Исследовательское тестирование программы руководителем.

**8 Сборка установщика**

Сборка проекта осуществляется в автоматическом режиме. Для сборки установочного пакета приложения используется программное обеспечение Inno Setup. С его помощью компилируется сценарий сборки (рисунок 8.1), создающий установочный пакет.

В сценарии сборки указывается различная информация о приложении, а также стандартный путь установки, название установочного пакета и дополнительные особенности установщика. Необходимо указать файлы, требуемые для работы приложения, такие как \*.exe и \*.dll. Остальные файлы (\*.pdb, \*.config, \*.manifest, \*.xml и другие возможные файлы) для работы приложения не нужны и исключаются из установочного пакета

Код сценария для установщика указан ниже.

; Script generated by the Inno Setup Script Wizard.

; SEE THE DOCUMENTATION FOR DETAILS ON CREATING INNO SETUP SCRIPT FILES!

#define MyAppName = "NoteApp"

#define MyAppVersion "1.0.0"

#define MyAppPublisher "Robkanov Konstantin"

#define MyAppURL "https://github.com/kuborga/NoteApp.git"

#define MyAppExeName "NoteAppUI.exe"

#define UninstallName "unins000.exe"

#define StartMenuFolderName "NoteApp"

#define AppIconName "iconMainForm.ico"

[Setup]

; NOTE: The value of AppId uniquely identifies this application. Do not use the same AppId value in installers for other applications.

; (To generate a new GUID, click Tools | Generate GUID inside the IDE.)

Это id приложения, которое однозначно идентифицирует приложение и записывается в реестре операционной системы.

AppId={{D337EE83-933F-477A-AC4D-F644D6866F87}

AppName = {#MyAppName}

AppVersion = {#MyAppVersion}

;AppVerName = {#MyAppName}{#MyAppVersion}

AppPublisher = {#MyAppPublisher}

AppPublisherURL = {#MyAppURL}

AppSupportURL = {#MyAppURL}

AppUpdatesURL = {#MyAppURL}

Папка установки приложения по умолчанию.

DefaultDirName = {autopf}\{#MyAppName}

DefaultGroupName = {#MyAppName}

ChangesAssociations = yes

DisableProgramGroupPage = yes

OutputDir = Installers

Название установочного файла,и дата создания установщика.

OutputBaseFilename = NoteAppSetup {#SetupSetting("MyAppVersion") + GetDateTimeString('dd-mm-yyyy hh-nn-ss', '-', ':')}

SetupIconFile = "..\NoteAppUI\Resources\{#AppIconName}"

Compression = lzma

SolidCompression = yes

WizardStyle = modern

На каком языке будет производиться установка приложения,из выбранных ниже .

[Languages]

Name: "english"; MessagesFile: "compiler:Default.isl"

Name: "russian"; MessagesFile: "compiler:Languages\Russian.isl"

[Dirs]

Name: "{commonstartmenu}\{#StartMenuFolderName}"

Указываем, какие задачи будут выполнены вовремя установки.Например,нужен ли ярлык на рабочем столе?

[Tasks]

Name: "desktopicon"; Description: "{cm:CreateDesktopIcon}"; GroupDescription: "{cm:AdditionalIcons}"; Flags: unchecked

Указываем какие файлы используются установщиком для сборки.

[Files]

Source: "Release\\*.exe"; DestDir: "{app}"; Flags: ignoreversion

Source: "Release\\*.dll"; DestDir: "{app}"; Flags: ignoreversion

Source: "..\NoteAppUI\Resources\{#AppIconName}"; DestDir: "{app}"; Flags: ignoreversion

Указываем какие иконки у ярлыка на рабочем столе и в меню пуск.

[Icons]

Name: "{commonstartmenu}\{#StartMenuFolderName}\{#MyAppName}"; Filename: "{app}\{#MyAppExeName}"; IconFilename:"{app}\{#AppIconName}"

Name: "{commonstartmenu}\{#StartMenuFolderName}\{cm:UninstallProgram,{#MyAppName}}"; Filename: "{app}\{#UninstallName}"

Name: "{commondesktop}\{#MyAppName}"; Filename: "{app}\{#MyAppExeName}"; IconFileName: "{app}\{#AppIconName}"; Tasks:desktopicon

[Run]

Filename: "{app}\{#MyAppExeName}"; Description: "{cm:LaunchProgram,{#StringChange(MyAppName, '&', '&&')}}"; Flags: nowait postinstall skipifsilent

Для работы приложения необходимы только исполняемые файлы и библиотеки, соответственно, в установочный пакет помещаются только файлы с расширением \*.exe и \*.dll. Их можно указать с помощью масок имен файлов в секции [File] установочного сценария:

Source: "..\Release\\*.exe"; DestDir: "{app}"; Flags: ignoreversion

Source: "..\Release\\*.dll"; DestDir: "{app}"; Flags: ignoreversion

Таким образом, для разрабатываемого приложения в установочный пакет будут помещены следующие файлы:

1. NewtonsoftJson.dll – библиотека, необходимая для сериализации;
2. NoteApp.dll - скомпилированная библиотека проекта логики;
3. NoteAppUI.exe - скомпилированный исполняемый файл.

Так же необходимо добавить в свойстве проекта NoteAppUI сценарий, который будет исполнен после сборки проекта.

Данная строка создает папку InstallScripts в директории, где располагается файл NoteApp.sln.

md "$(SolutionDir)InstallScripts"

Данная строка создает папку Release в папке InstallScripts. Сюда будут помещены файлы с расширением exe и dll.

md "$(SolutionDir)InstallScripts\Release"

Данная команда создает папку Installers в папке InstallScripts. Сюда в дальнейшем помещен файл установщик.

md "$(SolutionDir)InstallScripts\Installers"

Данные команды копируют все файлы с расширением exe и dll из папки bin, куда собирается решение.

xcopy"$(ProjectDir)$(OutDir)\*.dll" "$(SolutionDir)InstallScripts\Release"

xcopy"$(ProjectDir)$(OutDir)\*.exe" "$(SolutionDir)InstallScripts\Release"

Ниже приведены команды,которые выполняются после сборки проекта Installer.

Данная команда запускает файл установщика.

"$(SolutionDir)packages\Tools.InnoSetup.6.1.2\tools\ISCC.exe" "$(SolutionDir)InstallScripts\installer.iss"

Данная команда удаляет папку Realese со всеми файлами внутри.

rd /s/q "$(SolutionDir)InstallScripts\Release"

**9 Описание модели ветвления**

При разработке приложения использовалась система версионного контроля Git в локальной файловой системе проекта. Фиксации производились из среды разработки Visual Studio. Ссылка на репозиторий: https://github.com/kuborga/NoteApp.git

Работа над проектом велась в двух ветках репозитория:

1. Main – ветка, содержащая проверенную, протестированную, и готовую к включению в сборщик версию проекта
2. Develop – основная ветка разработки.