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 CONTACTSAPP**

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

Студент гр. 588-2

\_\_\_\_\_\_\_\_\_\_\_\_/ Комков В.Д.

(подпись)

« 20 » сентября 2021г.

Руководитель старший научный сотрудник, доцент каф. КСУП

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

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_20\_\_г.

Томск 2021

**Оглавление**

1 Назначение приложения3

2 Группы пользователей и их функциональные возможности в приложении4

3 Стек технологий разработки5

4 Пользовательский интерфейс6

5 Диаграмма пакетов9

6 Диаграмма классов10

7 Описание тестирования приложения11

8 Сборка установщика12

9 Описание модели ветвления14

**1 Назначение приложения**

Пользовательское приложение ContactsApp предназначено для ведения персональных контактов и контактов. Приложение должно:

1) Обеспечивать стабильную работу приложения при порядке 200 контактов.

2) Обеспечивать поиск, навигацию по созданным контактам по фамилии и имени.

3) Предоставить инструменты для просмотра и редактирования контактов.

4) Сохранять и восстанавливать контакты между сессиями приложения.

5) Выполнять промежуточные сохранения контактов на машине пользователя на случай аварийного завершения программы, отключения компьютера и т.д. – 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:

1. Просмотр существующих контактов.
2. Создание нового контакта, редактирование контакта и удаление контакта.
3. Навигация по созданным контактам.

**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 2017. 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 контактов и окно редактирования информации выбранной или нового контакта. Для дополнительной информации о программе и ее разработчике предусмотрено специальное окно, вызываемое из основного.

При запуске приложения появляется основное окно (рисунок 4.1). Оно разделено на две части по вертикали: в левой части находится список всех контактов, которые представлены в виде их названий, а на правой панели отображается информация о выбранной контакте.

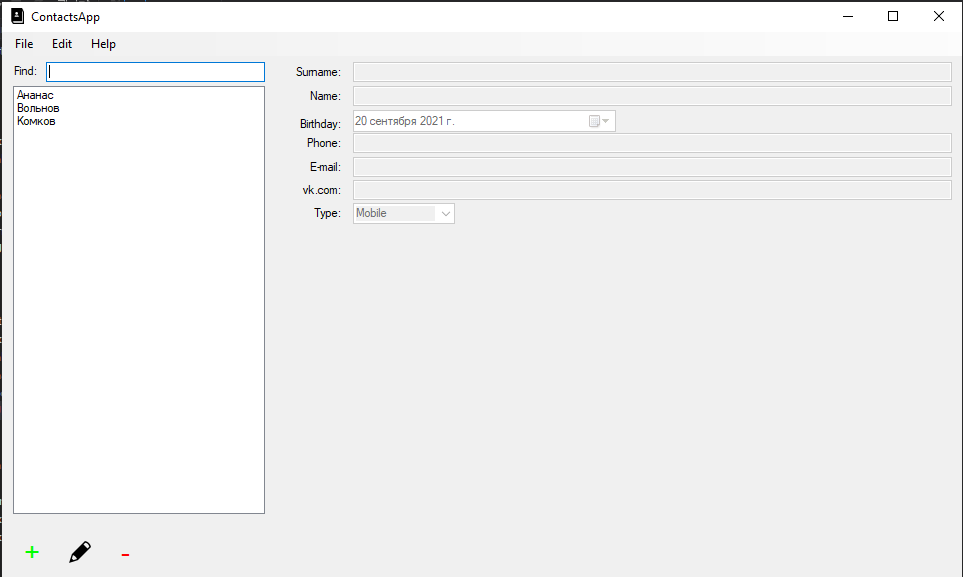


Рисунок 4.1 – Интерфейс главного окна программы

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

В случае добавления нового контакта или изменения старой вызывается окно редактирования контакта (рисунок 4.2). Если запись редактируется, то открывшееся окно содержит информацию этого контакта. При создании нового контакта открывается пустое окно, котором необходимо заполнить данными для нового контакта. Чтобы внесенные изменения были зафиксированы, необходимо выйти из формы по нажатию клавиши ОК, в противном случае все изменения будут отменены.

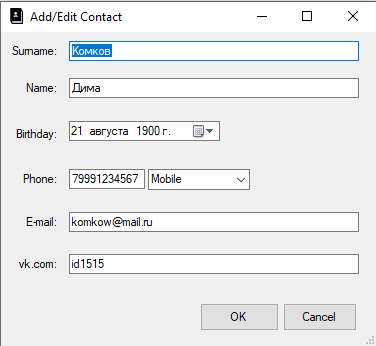


Рисунок 4.2 – Интерфейс окна для создания и редактирования контакта

При вызове окна изменения или добавления контакта происходит передача данных сначала из основной формы в вызываемую, а затем после подтверждения изменений осуществляется обратная передача данных и сохранение их в списке всех контактов. 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.

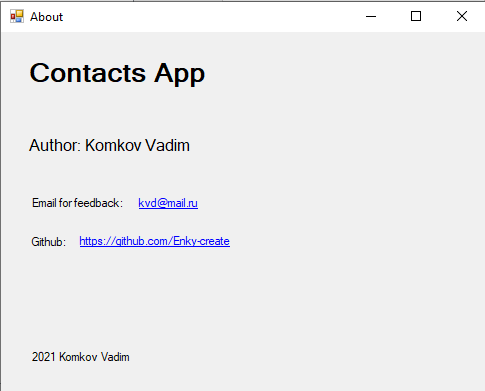


Рисунок 4.3 – Интерфейс окна дополнительной информации

Оставшийся пункт меню главного окна File реализует выход из программы.

**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.

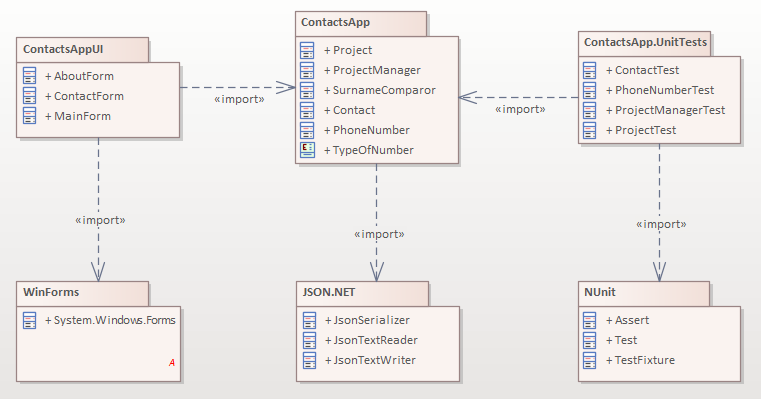


Рисунок 5.1 – Диаграмма пакетов приложения

**6 Диаграммы классов**

Проект логики приложения состоит из следующих 4 классов:

1. Contact – класс контакта;
2. TypeOfNumber – перечисление категорий контактов;
3. Project – класс списка контактов;
4. ProjectManager – класс для работы с файловым сохранением.

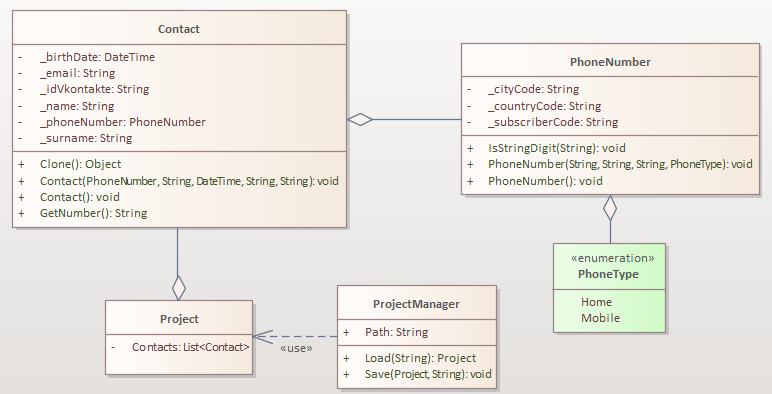
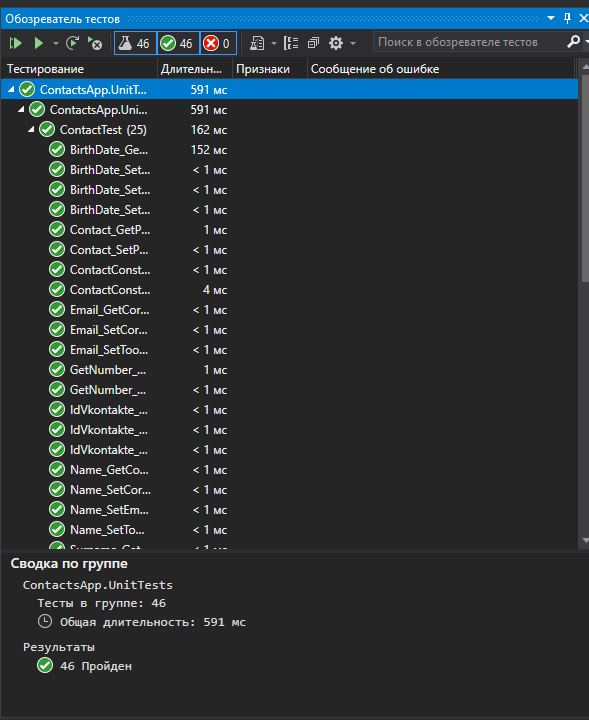
Диаграмма классов приложения представлена на рисунке 6.1.

Рисунок 6.1 – Диаграмма проекта логики приложения

**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.

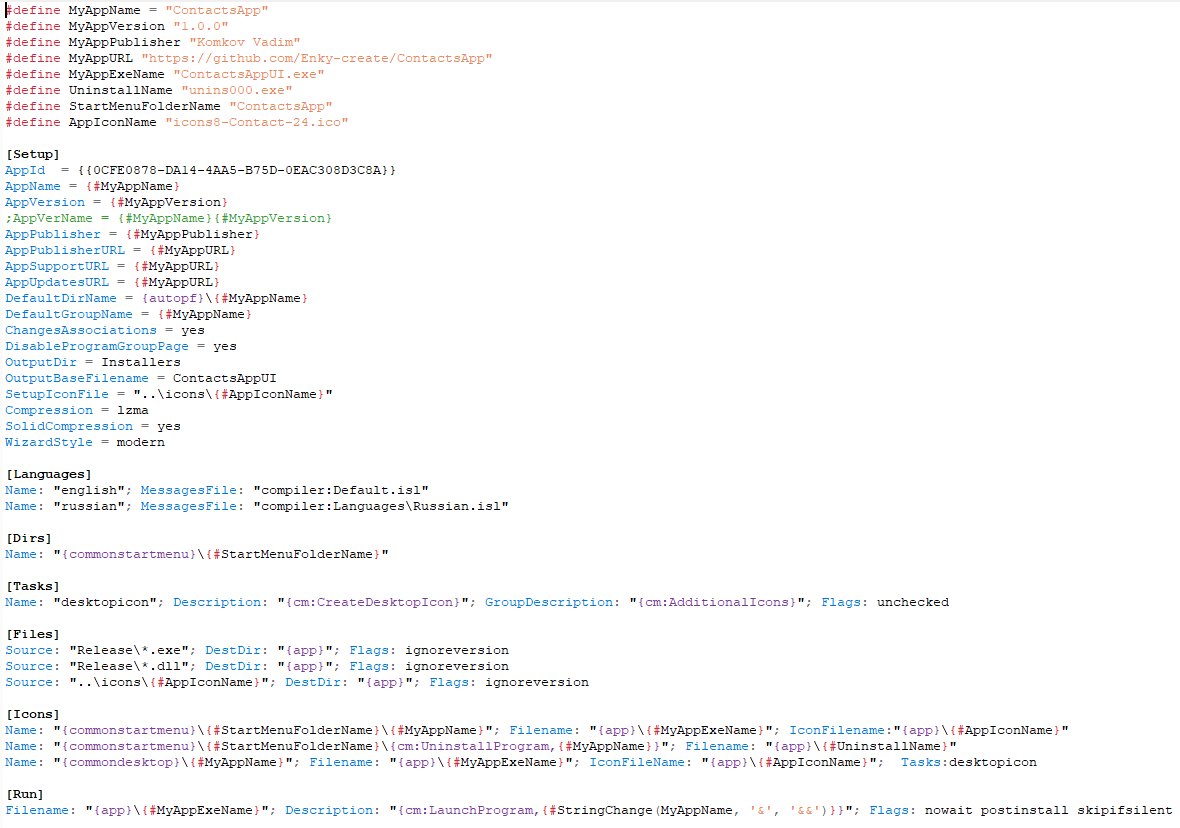
Были разработаны тесты, полностью покрывающие исходный код. На рисунке 7.1 представлена информация о проведении тестов.

 Рисунок 7.1 – Результаты тестирования

**8 Сборка установщика**

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.

 Рисунок 8.1 – Сценарий сборки установочного пакета

Сборка установочного файла осуществляется автоматически во время сборки программы. Установщик располагается в директории с исходным проектом. Сценарий событий после сборки проекта ContactsAppUI:

md "$(SolutionDir)InstallScripts"

md "$(SolutionDir)InstallScripts\Release"

md "$(SolutionDir)InstallScripts\Installers"

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

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

Для проекта Installer сценарий события после сборки имеет вид:

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

rd "$(SolutionDir)InstallScripts\Release"

Где:

«md» - создание директории;

«rd /S /Q» - удаление директории;

«xcopy» - копирование из одной директории в другую.

**9 Описание модели ветвления**

При разработке приложения использовалась система версионного контроля Git в локальной файловой системе проекта. Фиксации производились из среды разработки Visual Studio.

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.