Министерство образования и науки 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)

ТЕХНОЛОГИИ РАЗРАБОТКИ ПО

Пояснительная записка к практике

по дисциплине “ГПО”



Обучающийся гр. 588-2

Е.В. Кабанова (подпись) (И.О. фамилия)

(дата)

|  |  |
| --- | --- |
|  | Руководитель практики от  профильной организации:  Доцент кафедры КСУП  (должность, ученая степень, звание)  \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ Калентьев А.А.  оценка (подпись) (И.О. Фамилия)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата) |

|  |  |
| --- | --- |
|  | Руководитель практики от  Университета:  Доцент кафедры КСУП  (должность, ученая степень, звание)  \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ Калентьев А.А.  оценка (подпись) (И.О. Фамилия)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата) |

Томск 2020**Реферат**

Отчет по практике 39 с., 44 рис., 0 табл., 6 источников.

Ключевые слова: РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, ДЕСКТОПНОЕ ПРИЛОЖЕНИЕ, CONTACTSAPP, GIT, БИЗНЕС-ЛОГИКА, C#, ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС, WINDOWS FORMS, ЮНИТ-ТЕСТИРОВАНИЕ.

**Оглавление**

[1 Введение 3](#_Toc55685671)

[2 Назначение приложения 4](#_Toc55685672)

[3 Стек технологий разработки. Системные требования 4](#_Toc55685673)

[4 Пользовательский интерфейс 4](#_Toc55685674)

[5 Диаграмма пакетов 8](#_Toc55685675)

[6 Диаграмма классов проекта бизнес-логики 8](#_Toc55685676)

[7 Описание тестирования приложения 9](#_Toc55685677)

[8 Описание сборки установщика 10](#_Toc55685678)

[9 Описание модели ветвления 13](#_Toc55685679)

[10 Заключение 15](#_Toc55685680)

[Список источников 16](#_Toc55685681)

# **1 Введение**

Цель практики – освоение процесса разработки программного обеспечения, посредствам учебно-методического материала «Теория разработки ПО». Результатом выполнения практики является desktop приложения выполненное согласно назначенному варианту задания. Разработка приложения ведется на языке C# с использованием набора компонентов Windows Forms. В ходе выполнения практики необходимо последовательно выполнить следующие этапы: установить все требуемые приложения, освоить систему версионного контроля Git c использованием сервиса GitHub, написать все необходимые классы логики, разработать пользовательский интерфейс приложения, организовать автоматическое тестирование логики приложения и сборку установщика, а также подготовить техническую документацию в виде пояснительной записки к проекту с UML-диаграммами, ретроспективы по процессу разработки, календарного плана и сметы на разработку аналогичного приложения.

Согласно назначенному варианту в результате выполнения данной работы должно быть разработано и протестировано пользовательское приложение ContactsApp, предназначенное для ведения и хранения контактов.

# **2 Назначение приложения**

Пользовательское приложение ContactsApp, предназначено для ведения и хранения 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.

Приложение-референс: контакты мобильного телефона

# **3 Стек технологий разработки. Системные требования**

Разработка приложения предполагается на языке C# с использованием набора компонентов Windows Forms. Несмотря на то, что использование Windows Forms в реальных проектах на 2020 год крайне мало, данный набор компонентов относительно прост в изучении и хорошо подходит для учебных задач.

Программы и сервисы, необходимые для разработки приложения:

1. Среда разработки: Microsoft Visual Studio 2018 Community
2. Вспомогательные плагины для среды разработки: JetBrains Resharper.
3. Система версионного контроля: git с использованием сервиса GitHub.com
4. Сборка установочных пакетов: InnoSetup
5. Microsoft Word

# **4 Пользовательский интерфейс**

После запуска приложения перед пользователем появляется главное окно (рис. 4.1). Двухколоночная верстка главного окна содержит список всех контактов в левой панели и отображает текущий выбранный контакт в правой панели. В списке контактов показаны фамилии контактов, в один момент времени может быть выбран только один контакт (далее – текущий контакт).

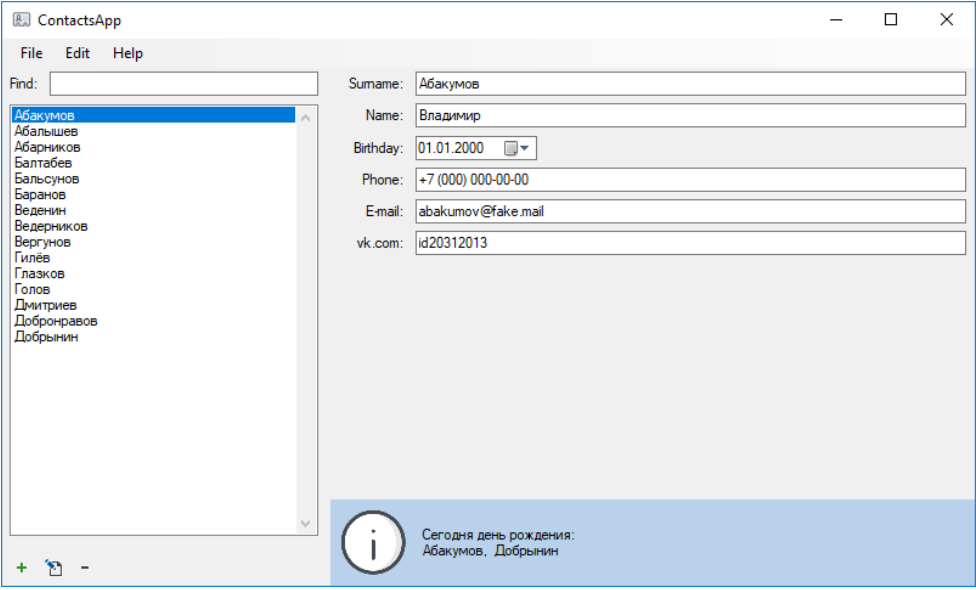


Рисунок 4.1 – Макет главного окна приложения ContactsApp

На панели со списком контактов внизу располагаются три кнопки в виде пиктограмм: Add Contact («Создать новый контакт»), Edit Contact («Редактировать текущий контакт»), Remove Contact («Удалить текущий контакт»).

При выборе контакта в списке, выбранный контакт отображается в правой панели. Главное окно не позволяет редактировать содержимое контакта – только просмотр. Также в правой панели под текущим контактом отображается информационная панель с сегодняшними именинниками (см. рис. 4.1).

При нажатии на кнопку Add Contact и Edit Contact появляется окно создания/редактирования контакта в диалоговом режиме (рис. 4.2). Для нового контакта окно изначально не заполнено (установлена лишь дата рождения по умолчанию). Для редактирования уже существующего контакта все поля должны быть предзаполнены данными текущего контакта.

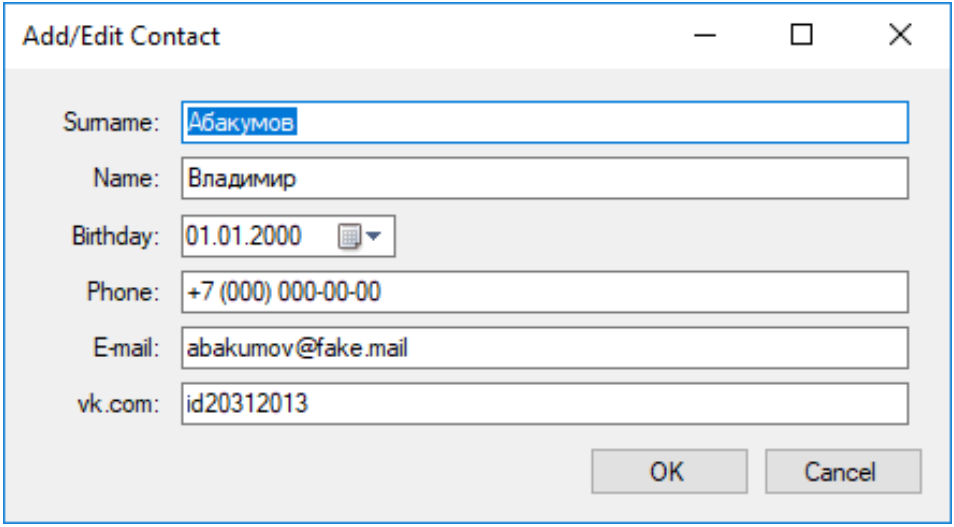


Рисунок 4.2 – Макет окна редактирования контакта в приложении ContactsApp

При нажатии на кнопку OK окно создания контакта закрывается, в список контактов главного окна добавляется новый контакт. При редактировании текущей контакта, нажатие на кнопку OK должно обновить фамилию контакта в списке контактов (если фамилия текущего контакта была изменена/исправлена), и обновить отображаемый контакт в правой панели приложения. При нажатии кнопки Cancel создание/редактирование контакта отменяется (новый контакт не добавляется, исходный контакт остается без изменений).

В случае ввода пользователем некорректных данных (нарушение допустимой длины фамилии, имени, указание невозможной даты рождения или неправильного номера телефона), данная ситуация должна быть обработана соответствующим образом.

При нажатии на кнопку Remove Contact главного окна текущий контакт удаляется. Перед удалением должно появиться окно с запросом на разрешение записи: «Do you really want to remove this contact: <Фамилия текущего контакта>». При нажатии на кнопку OK происходит удаление, при нажатии на кнопку Cancel удаление отменяется.

Меню главного окна содержит следующие пункты:

* File:
  + Exit (Выйти из приложения – Alt+F4)
* Edit:
  + Add Contact (Создать новый контакт)
  + Edit Contact (Редактировать текущий контакт)
  + Remove Contact (Удалить текущий контакт) 169
* Help
  + About (Вызвать окно «О программе» - F1) (см. рис. 4.3)

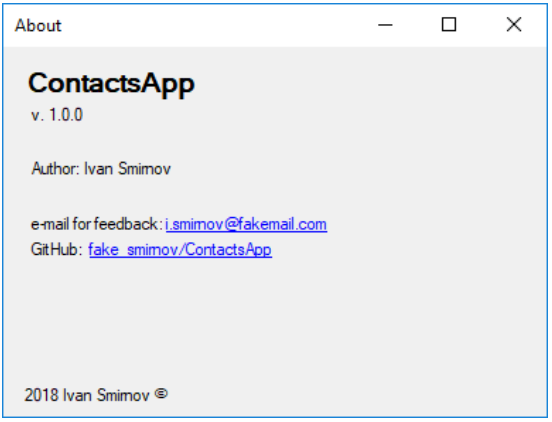


Рисунок 4.3 – Макет окна «About» приложения ContactsApp

Таким образом, в программе должно быть реализовано три окна:

* Главное окно.
* Окно создания/редактирования контакта.
* Окно «About».

Верстка главного окна и окна создания/редактирования контакта должна быть адаптивной. Окно «About» имеет фиксированный размер. Создание дополнительных элементов управления для уменьшения дублирования кода остается на усмотрение разработчика или руководителя. Загрузка контактов осуществляется при запуске программы до вывода главного окна пользователю, сохранение контактов в файл должно выполняться в случаях:

1. создания нового контакта;
2. удаления контакта;
3. закрытии приложения.

# **5 Диаграмма пакетов**

Помимо разрабатываемых проектов бизнес-логики, пользовательского интерфейса и юнит-тестов при создании приложения использовались сторонняя библиотека WinForms, JSON и NUnit.

Диаграмма пакетов приложения ContactsApp с указанием использованных классов представлена на рисунке 5.1:

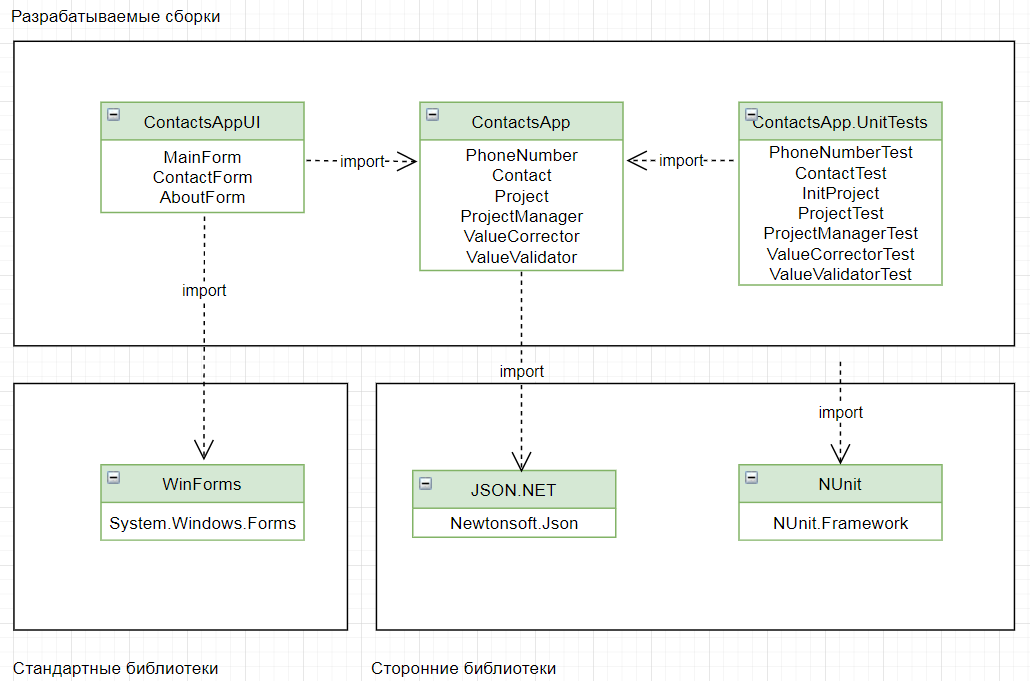


Рисунок 5.1 – Диаграмма пакетов приложения ContactsApp

# **6 Диаграмма классов проекта бизнес-логики**

Проект бизнес-логики приложения ContactsApp включает в себя 6 разработанных классов: PhoneNumber, Contact, Project, ProjectManager, ValueCorrector, ValueValidator и класс Json применяемый для сериализации объектов в классе ProjectManager.

Диаграмма классов проекта бизнес-логики с указанием сторонних библиотек представлена на рисунке 6.1:

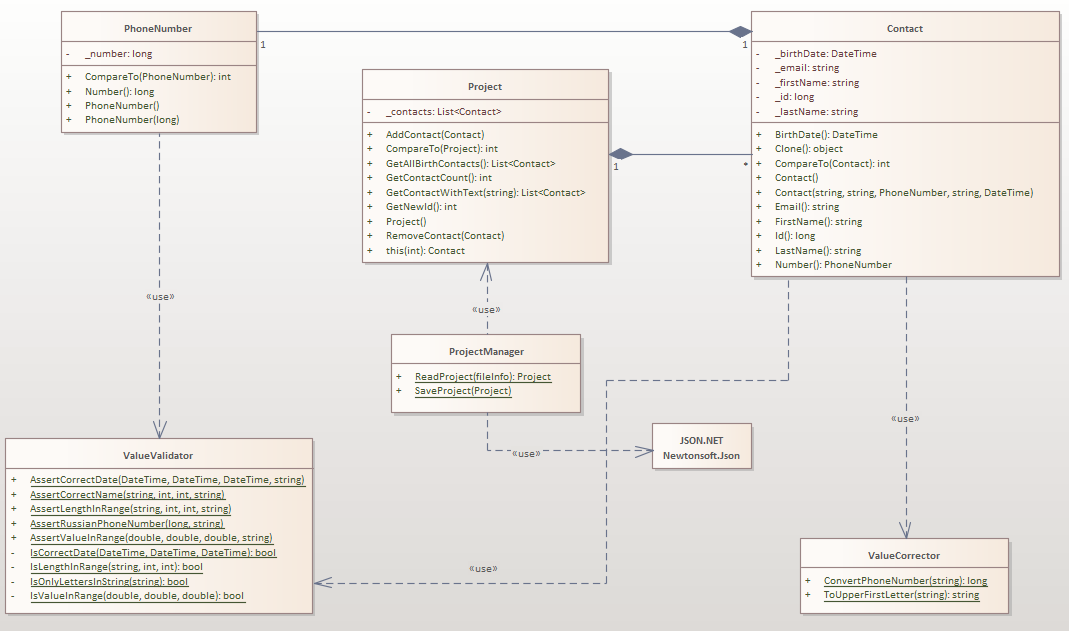


Рисунок 6.1 – Диаграмма классов проекта бизнес-логики

# **7 Описание тестирования приложения**

Разработка приложения ContactsApp подразумевает тестирование только классов проекта бизнес-логики. Для этого был создан соответствующие проект ContactsApp.UnitTests.

Тестирование проводилось автоматически с использованием библиотеки NUnit. Всего в результате разработки проекта юнит-тестов было разработано и проведено 52 теста.

Общая циклометрическая сложность и время выполнения тестов классов бизнес-логики и проекта в целом представлено на рисунке 7.1:

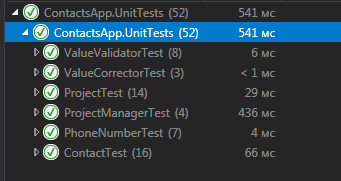


Рисунок 7.1 – Общая циклометрическая сложность и время выполнения тестов классов бизнес-логики и проекта в целом

Для проверки достаточности написанных тестов использовалось заранее установленное расширение JetBrains.

Анализ покрытия проекта тестами, выполненный с использованием Resharper представлен на рисунке 7.2:

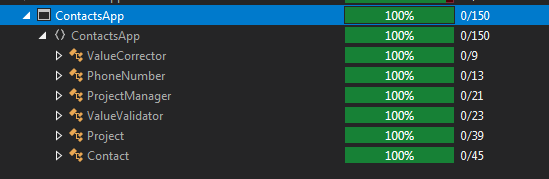


Рисунок 7.2 – Анализ покрытия проекта тестами, выполненный с использованием Resharper

# **8 Описание сборки установщика**

Прежде всего перед сборкой необходимо выполнить компиляцию решения в конфигурации Release. Таким образом получаем только необходимые для сборки файлы без дополнительного кода для отладки, который будет замедлять работу приложения.

Создание сценария для сборки установочного пакета и компиляция установщика производились автоматически в среде Visual Studio с использованием программы InnoSetup. Для этого в качестве событий после успешной сборки проекта ContactsAppUI были прописаны следующие команды (рисунок 8.1):

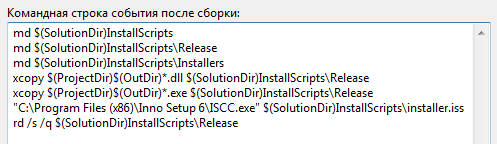


Рисунок 8.1 – События после успешной сборки проекта ContactsAppUI

В первую очередь после успешной сборки проекта в папке решения автоматически создается папка InstallScripts, в которой создаются подпапки Release и Installer. В папку Release помещаются файлы необходимые для сборки установочного пакета. Такие файлы имеют расширение \*.exe и \*.dll и представлены на рисунке 8.2:

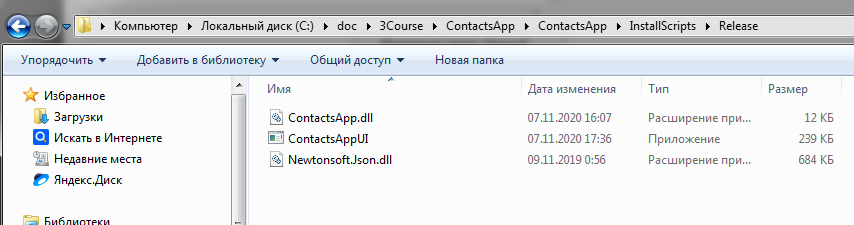


Рисунок 8.2 – Содержимое временной папки Release

Данная папка требуется только в процессе сборки, поэтому после создания установщика автоматически удаляется.

Далее происходит создание сценария установочного пакета при помощи программы InnoSetup. Сценарий сохраняется в формате installer.iss и помещается в папку InstallScripts.

Сценарий для сборки установочного пакета приложения ContactsApp представлен на рисунке 8.3:

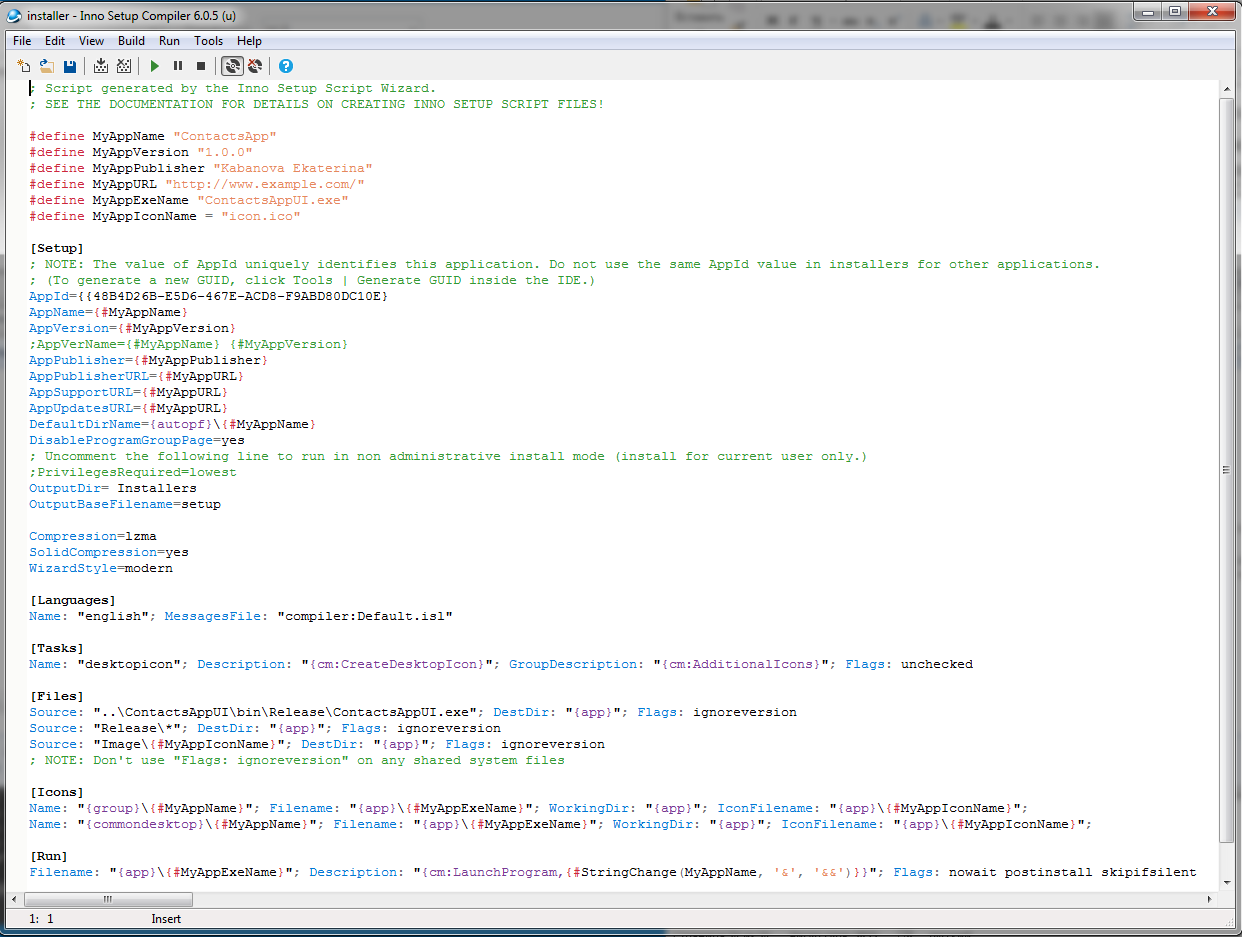


Рисунок 8.3 – Сценарий для сборки установочного пакета приложения ContactsApp

Сценарий разделен на область определения переменных, секции с инструкциями, а также может содержать код на языке Delphi для создания дополнительных, нестандартных окон в установщике.

В начале сценария можно объявить различные константы с помощью директивы #define. После директивы #define сначала указывается имя константы и через пробел её значение. В сценарии, представленном на рисунке 2.30, объявлены 6 констант: название приложения, версия выпуска, инициалы разработчика, адрес электронной почты, исполняемый файл проекта ContactsAppUI и имя файла с иконкой для приложения. При дальнейшей поддержке программы, в случае изменения, например, адреса почты достаточно будет исправить только значение переменной вместо исправления значений во всём сценарии.

Секция [Setup] задаёт основные настройки установщика:

* AppId – уникальный идентификационный номер приложения для регистрации в реестре Windows, сгенерированный автоматически,
* AppName – название приложения, в том числе и для регистрации в реестре,
* AppVersion – версия приложения,
* AppPublisher – инициалы разработчика,
* DefaultDirName – путь для создания папки приложения на пользовательском устройстве,
* OutputDir – папка, для сохранения установщика,
* OutputBaseFilename – имя установочного файла,
* Compression – алгоритм сжатия (архивирования) файлов, помещаемых в установщик. OutputBaseFilename – имя выходного файла – файла установщика и др.

Секция [Tasks] позволяет создавать дополнительные окна в установщике с опциями, которые сможет установить пользователь. В сценарии, представленном на рисунке 2.30, такой опцией является установка ярлыка приложения на рабочем столе устройства.

Секция [Files] описывает файлы, которые необходимо поместить в установщик. В данном случае это исполнительный файл проекта ContactsAppUI, файлы временной папки Release и изображение, используемое в качестве иконки приложения.

Далее описывается установка иконки приложения.

Секция [Run] является необязательной. В этой секции можно указать любое количество программ для выполнения после того, как программа была успешно установлена.

# **9 Описание модели ветвления**

Разработка приложения ContactsApp велась с использованием системы контроля версий Git. Сразу после создания проект был размещен на удаленном репозитории сервиса GitHub с указанием лицензии MIT License. Для сокрытия конфигурационных файлов из системы контроля версий Git был задействован файл .gitignore.

Так как данная работа предназначена для индивидуального выполнения, при разработке приложения использовалась упрощенная модель ветвления GitFlow. Основная работа над проектом происходила в ветке разработки – develop. При корректной работе новой функциональности, изменения вносились в главную ветку репозитория – master.

Ссылка на удаленный репозиторий с приложением ContactsApp:

<https://github.com/KeV-project/ContactsApp>

# **10 Заключение**

В ходе выполнения данной работы были изучены основные принципы разработки программного обеспечения. Все полученные знания были закреплены на практике в процессе разработки десктоп-приложения для ведения и обработки контактов ContactsApp.

В процессе подготовки к созданию приложения были изучены основные принципы разработки на языке C# и набор компонентов Windows Forms.

Согласно техническому заданию были разработаны проекты бизнес-логики и пользовательского интерфейса. При помощи библиотеки NUnit были разработаны и проведены тесты для всех классов бизнес-логики.

После завершения разработки всех частей приложения при помощи программы InnoSetup был создан сценарий установочного пакета и скомпилирован установщик приложения.

На протяжении всего процесса разработка результаты публиковались в удаленном репозитории на сайте GitHub.

# **Список источников**

1. Технологии разработки программного обеспечения: методические указания к самостоятельной и лабораторным работам / А. Е. Горяинов, А.А. Калентьев, 2014. – 176 с.
2. Полное руководство по языку программирования С# 9.0 и платформе .NET 5 [Электронный ресурс] // <https://metanit.com>: Метанит – сайт о программировании. 2020. URL: <https://metanit.com/sharp/tutorial/> (дата обращения: 20.10.2020).
3. Руководство по программированию в Windows Forms [Электронный ресурс] // <https://metanit.com>: Метанит – сайт о программировании. 2020. URL: <https://metanit.com/sharp/windowsforms/> (дата обращения: 28.10.2020).
4. Интерактивный тур, для знакомства с основами Git [Электронный ресурс] // <https://githowto.com/ru>: GitHowTo – курс обучения Git на русском. 2020. URL: <https://githowto.com/ru/setup> (дата обращения: 17.09.2020).
5. Соглашения по оформлению кода команды RSDN [Электронный ресурс] // [https://rsdn.org](https://rsdn.org/): RSDN – Cообщество [русскоговорящих](https://ru.wikipedia.org/wiki/%D0%A0%D1%83%D1%81%D1%81%D0%BA%D0%BE%D1%8F%D0%B7%D1%8B%D1%87%D0%BD%D1%8B%D0%B5) разработчиков программного обеспечения. 2003. URL: <https://rsdn.org/article/mag/200401/codestyle.XML> (дата обращения: 29.10.20).
6. Cайт вопросов и ответов для программистов Stack Overflow [Электронный ресурс] // <https://ru.stackoverflow.com>: Stack Overflow - Cайт вопросов и ответов для программистов. 2020. URL: <https://ru.stackoverflow.com/> (дата обращения: 14.10.20).