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)

ПРИЛОЖЕНИЕ CONTACTS APP

Отчет по дисциплине “Новые технологии в программировании”

Выполнил:

студент гр. 580-2

\_\_\_\_\_\_Сеченов В.В.

«4» июня 2022 г.

Проверил:

Доцент кафедры КСУП

\_\_\_\_\_\_\_\_\_\_\_\_Горяинов А.Е.

оценка

«\_\_\_» июня 2022 г.

Томск, 2022

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 4](#_Toc137267625)

[1 НАЗНАЧЕНИЕ ПРИЛОЖЕНИЯ 3](#_Toc137267626)

[2 ГРУППЫ ПОЛЬЗОВАТЕЛЕЙ И ИХ ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ В ПРИЛОЖЕНИИ 4](#_Toc137267627)

[3 СТЕК ТЕХНОЛОГИЙ РАЗРАБОТКИ. СИСТЕМНЫЕ ТРЕБОВАНИЯ 5](#_Toc137267628)

[4 ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС 6](#_Toc137267629)

[5 ДИАГРАММА ПАКЕТОВ ПРИЛОЖЕНИЯ 11](#_Toc137267630)

[6 ДИАГРАММА КЛАССОВ ПРИЛОЖЕНИЯ 12](#_Toc137267631)

[7 ОПИСАНИЕ ТЕСТИРОВАНИЯ ПРИЛОЖЕНИЯ 14](#_Toc137267632)

[8 ОПИСАНИЕ СБОРКИ УСТАНОВЩИКА 15](#_Toc137267633)

[9 ОПИСАНИЕ МОДЕЛИ ВЕТВЛЕНИЯ 16](#_Toc137267634)

# 1 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.

# 2 ГРУППЫ ПОЛЬЗОВАТЕЛЕЙ И ИХ ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ В ПРИЛОЖЕНИИ

В данном приложении существует только одна группа пользователей, которым доступен весь функционал приложения (создание, удаление и редактирование контактов).

# 3 СТЕК ТЕХНОЛОГИЙ РАЗРАБОТКИ. СИСТЕМНЫЕ ТРЕБОВАНИЯ

Для создания приложения использовался следующий стек технологий:

* C# является языком программирования для создания всего проекта;
* Windows Forms используется для создания пользовательского интерфейса приложения;
* Newtonsoft.Json используется для сериализации контактов в формат JSON и десериализации из этого формата для сохранения в файл и загрузки из файла;
* NUnit используется для автоматического юнит-тестирования классов.

# 4 ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС

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

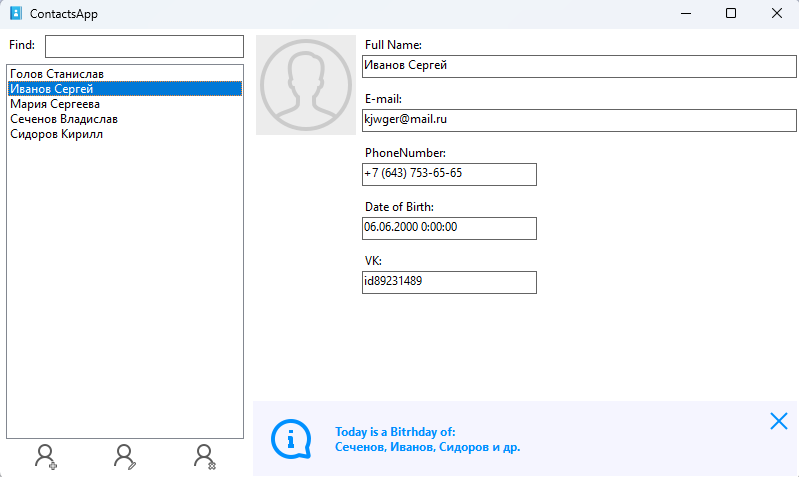


Рисунок 4.1 – Макет главного окна приложения ContactsApp

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. 4.2).

В левой панели главного окна над списком контактов расположено текстовое поле для поиска контактов по подстроке. При вводе пользователем подстроки, в списке контактов должны отобразиться только контакты, данные которых содержат введенную подстроку (т.е. проверяются не только полное имя, но и почта, номер телефона, дата рождения, аккаунт в VK). Если из текстового поля стереть подстроку, то в панели должны отобразиться все существующие контакты.

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 незаполнено (установлена лишь дата рождения по умолчанию). Для редактирования уже существующего контакта все поля должны быть предзаполнены данными текущего контакта.

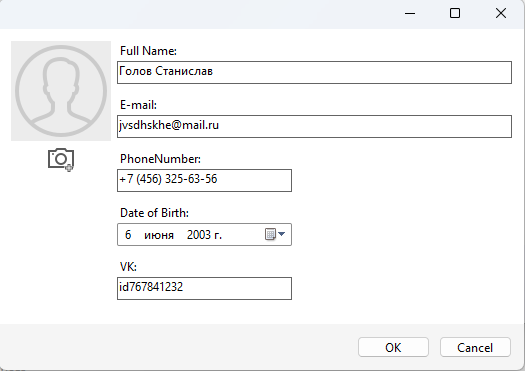


Рисунок 4.2 – Макет окна редактирования контакта в приложении ContactsApp

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.

По нажатию на F1 в главном окне открывается окно About о приложении: (см. рис. 4.3).

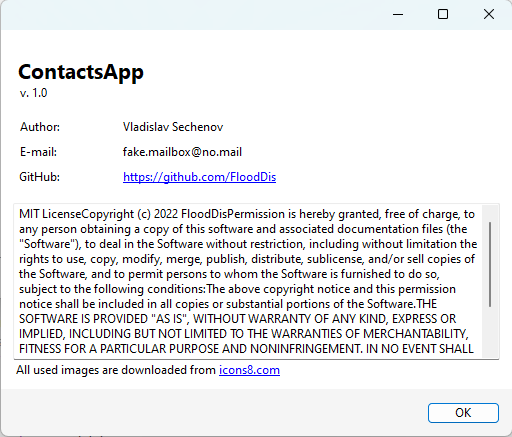


Рисунок 4.3 – Макет окна «About»

Окно «About» содержит название, номер версии приложения, имя автора, почту, рабочую ссылку на аккаунт пользователя на GitHub, текст лицензии приложения (по умолчанию MIT License), указание о правах на использованные изображения (обязательное условие бесплатного использования изображений). Окно должно закрываться по нажатию на кнопку OK или клавишу Esc.

Таким образом, в программе должно быть реализовано три окна:

- Главное окно;

- Окно создания/редактирования контакта;

- Окно «About».

Верстка главного окна должна быть адаптивной. Окно «About» и окно создания/редактирования данных контакта имеют 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: а) создания нового контакта; б) удаления контакта; в) закрытии приложения.

# 5 ДИАГРАММА ПАКЕТОВ ПРИЛОЖЕНИЯ

На рисунке 5.1 изображена диаграмма пакетов приложения.

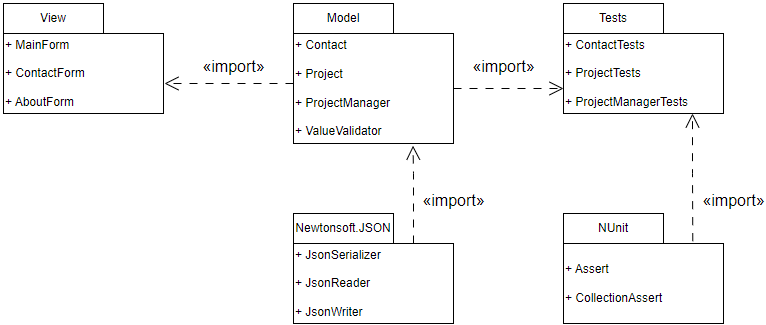


Рисунок 5.1 – Диаграмма пакетов приложения

# 6 ДИАГРАММА КЛАССОВ ПРИЛОЖЕНИЯ

На рисунке 6.1 изображена концептуальная диаграмма классов.

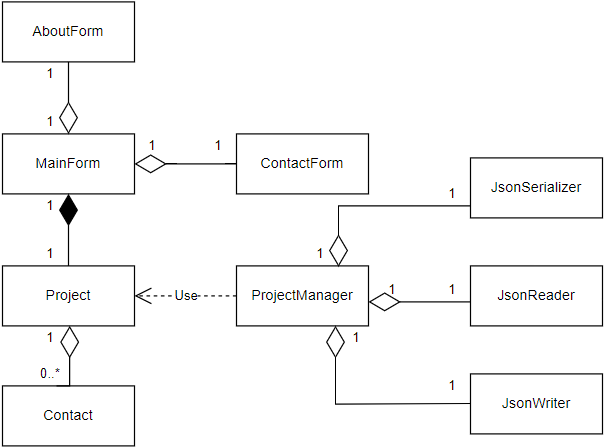


Рисунок 6.1 – Концептуальная диаграмма классов

На рисунке 6.2 изображена детальная диаграмма классов бизнес-логики.

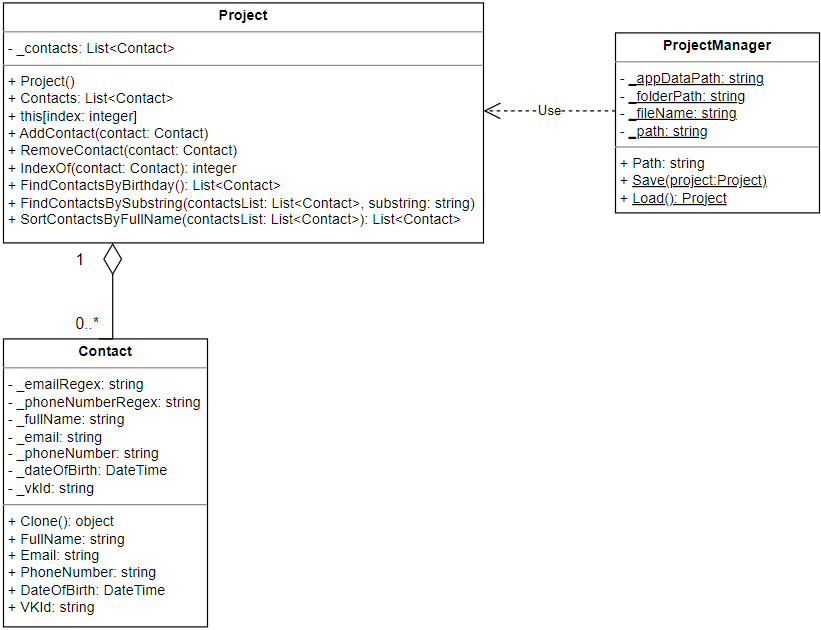


Рисунок 6.2 – Детальная диаграмма классов бизнес-логики

# 7 ОПИСАНИЕ ТЕСТИРОВАНИЯ ПРИЛОЖЕНИЯ

Основным видом тестирования приложения является юнит-тестирование. Юнит-тестированию подлежат класс Contact, хранящий информацию о контакте, и класс Project, хранящий в себе список контактов.

Тестированию подлежат все методы классов Contact и Project. Степень покрытия приложения юнит-тестами составляет 90%.

Также для тестирования работоспособности всего приложения проводилось приемочное тестирование. Для этого:

1. Создается несколько контактов в приложении;
2. Проверяется отображение данных контакта, выбранного из списка;
3. Созданные контакты редактируются;
4. Часть созданных контактов удаляются;
5. Все пункты, описанные выше, повторяются, но с поиском по подстроке;
6. Приложение закрывается и снова открывается для проверки сохранения и загрузки раннее созданных контактов.

# 8 ОПИСАНИЕ СБОРКИ УСТАНОВЩИКА

Для сборки установщика используется Inno Setup. Inno Setup – система создания инсталляторов для Windows-программ с открытым исходным кодом.

Сборка установщика осуществляется при запуске скрипта Inno Setup. Для успешной сборки установщика перед запуском скрипта необходимо запустить программу в конфигурации Release. Собранный установщик помещается в папку Output.

В процессе сборки установщика не используется обфускация кода, в приложение не внедряются функции защиты и лицензирования.

# 9 ОПИСАНИЕ МОДЕЛИ ВЕТВЛЕНИЯ

При создании приложения использовалась распределенная система управления версиями Git.

Основное отличие Git от любой другой системы контроля версий  — это подход к работе со своими данными. Концептуально, большинство других систем хранят информацию в виде списка изменений в файлах. Эти системы представляют хранимую информацию в виде набора файлов и изменений, сделанных в каждом файле, по времени (обычно это называют контролем версий, основанным на различиях).

Git не хранит и не обрабатывает данные таким способом. Вместо этого, подход Git к хранению данных больше похож на набор снимков миниатюрной файловой системы. Каждый раз, когда вы делаете коммит, то есть сохраняете состояние своего проекта в Git, система запоминает, как выглядит каждый файл в этот момент, и сохраняет ссылку на этот снимок. Для увеличения эффективности, если файлы не были изменены, Git не запоминает эти файлы вновь, а только создаёт ссылку на предыдущую версию идентичного файла, который уже сохранён. Git представляет свои данные как **поток снимков**.

Моделью ветвления была выбрана GitFlow. GitFlow – методология работы с Git, в ней определяется, какие виды веток необходимы проекту и как выполнять слияние между ними.

В процессе создания приложения из методологии GitFlow использовались только ветки develop и feature. Для каждой новой функциональности создавалась отдельная ветка с названием вида “features/<название функциональности>”.