

Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Операционные системы и системное программирование
(ОСиСП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

ПРОГРАММНОЕ СРЕДСТВО

«Daily Planner»

БГУИР КП 1–40 01 01 003 ПЗ

Студент: гр. 951006 Белоусов А. В.

Руководитель: асс. Жиденко А. Л.

Минск 2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
1 АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПС	6
1.1 Примеры существующих аналогов	6
1.2 Формирование требований к проектируемому программному средству	11
2 РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА	12
3 ОБОСНОВАНИЕ ТЕХНИЧЕСКИХ ПРИЕМОВ ПРОГРАММИРОВАНИЯ	14
3.1 Обоснование выбора языка программирования	14
3.2 Обоснование выбора библиотек WinApi	14
4 ТЕСТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА.....	15
5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ	24
5.1 Начало работы	24
5.2 Добавление задачи	24
5.3 Изменение задачи.....	26
5.4 Удаление задач	27
5.5 Сохранение изменений после закрытия приложения	28
ЗАКЛЮЧЕНИЕ	30
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	31
ПРИЛОЖЕНИЕ А	32
A1 Текст программного модуля resource.h.....	32
A2 Пространство имён buttons.....	32
A3 Пространство имён util	34
A4 Пространство имён source	37
A5 Пространство имён mainwndproc	39
A6 Пространство имён editdlgproc	42
A7 Пространство имён todoview.....	45
A8 Класс PlannerRepository	47
A9 Класс PlannerEntry	49

ВВЕДЕНИЕ

Жизнь человека в современном мире полна информации. Зачастую эта информация является просто цифровым шумом. Мы теряемся в этом потоке, постоянно отвлекаемся, теряем концентрацию, и в итоге тратим время впустую. Часто кажется, что весь день вы работаете, а дела не закрываются. Задача планера – помочь вам отследить, куда уходит ваше время. Человеку для личного развития крайне важно отслеживать свою эффективность, что невозможно без контроля времени.

В потоке дел мы рискуем забыть что-то важное. Хорошо всегда иметь под рукой ежедневник, в который можно вносить «входящие» большие и маленькие дела. Потом они разделятся по важности и срочности, и вы избежите эффекта «горы дел».

Когда ты записываешь дела, они приобретают структуру и порядок. Когда в голове каша, обязательно что-то вылетит из головы. Если задачи вынести на бумагу, легче понять, какие из них требуют немедленного решения.

Если вы вводите привычку начинать день со списка задач, вы тренируете дисциплину и память. Написаны сотни книг о том, что вы вероятнее всего забудете что-то, если держать все в голове. Организованность – это один из самых важных критериев достижения успеха.

Четко сформулированные задачи, разнесенные во времени, уменьшают уровень стресса и тревожности. Доказано, что маленькие задачи («отправить мейл») и крупные («выбрать новый компьютер») влияют на подсознание одинаково. Даже самая маленькая невыполненная задача вносит раздражение и поднимает уровень стресса. Когда ты вносишь задачу в график, в голове ты четко понимаешь, когда ты вернешься к её выполнению. Это снижает уровень стресса от огромной лавины дел, на которые нужно найти время.

Записи в ежедневник помогают делить крупные задачи на составные части. Дробление помогает взглянуть на вопрос с разных сторон и уменьшить масштаб проблемы. Это способствует тому, чтобы, наконец, начать выполнение большого и сложного дела, к которому «не знаешь, как подступить».

Система ведения ежедневника подразумевает, что это игра «в долгую». Иметь планер – это значит взглянуть на свою жизнь в перспективе и понять, куда уходит время и силы, осознать свои ошибки, проанализировать свою деятельность и решить, как действовать с большей эффективностью для достижения своих целей.

1 АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПС

1.1 Примеры существующих аналогов

1.1.1 Microsoft To-Do

Организационные инструменты редко существуют в «вакууме». Если вы подключены к экосистеме Microsoft с помощью электронной почты Outlook и работы в Office, вам может быть интересно узнать, что у Microsoft есть собственное приложение для ведения дел.

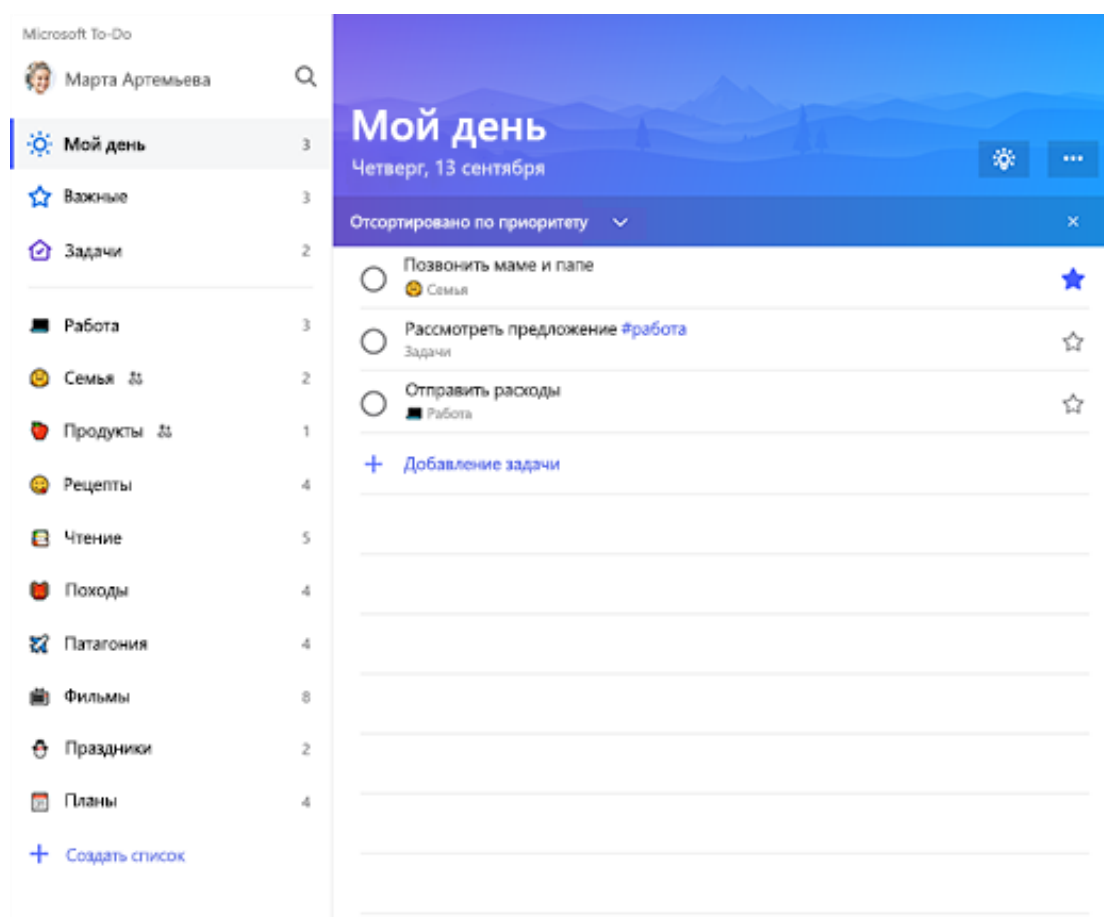


Рисунок 1.1 – Интерфейс приложения «Microsoft To-Do»

Microsoft To-Do, созданная командой Wunderlist после того, как Microsoft приобрела это приложение в 2015 году. Продукт от «мелкомягких» поразительно похож на Wunderlist — и это неплохо; постановка новых задач проста, и она предлагает почти то же самое с точки зрения инструментов и функций. Microsoft To-Do отличается тем, что делает акцент на My Day, а именно на идее, что вы начинаете каждый день с чистого листа, а в начале каждого дня уделяете время тому, чтобы записать, чего вы хотите достичь в этот день. Это философия жизни без суеты, и она направлена на то, чтобы

пользователи сосредоточились на том, что происходит здесь и сейчас. Это подходит не для всех, и если вы любите планировать заранее, то Microsoft To-Do это тоже позволяет. В него даже встроен интеллектуальный инструмент подсказок, который предложит вам задачи на основе вашей предыдущей записи.

Кажется, в будущем Microsoft To-Do будет синхронизироваться с Wunderlist, и вы сможете импортировать задачи Wunderlist, если вы будете переходить из этого приложения. В будущем планируется интеграция с другими службами Microsoft, что тоже очень удобно.

1.1.2 AnyDo

Any.do — отличное приложение с простым и понятным интерфейсом, которое обеспечивает быстрое и простое управление задачами и даже интегрируется с приложением iOS Reminders. Таким образом, вы можете сообщить Siri напоминание, и оно появится в Any.do. Однако синхронизация работает только в одном направлении: удаление задач из Any.do не приведет к его удалению из приложения напоминаний для iPhone, но, если вы используете Any.do в качестве основного приложения, это не будет проблемой. Есть также дополнительные удобные функции, такие как автоматическая сортировка списков продуктов и «планировка дня», чтобы помочь расставить приоритеты для задач.

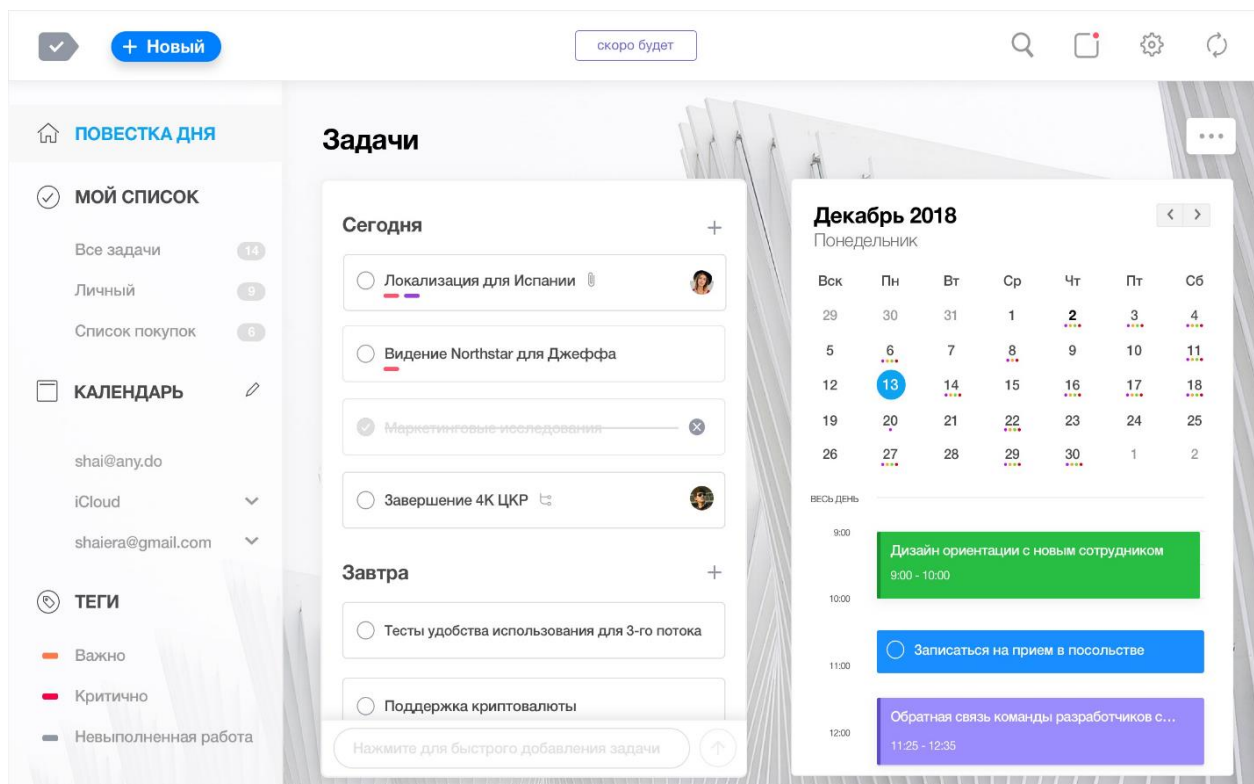


Рисунок 1.2 – Интерфейс приложения «AnyDo»

Синхронизация между ПК, планшетом и телефоном — еще одно приятное дополнение к поддержке iOS и Android. За переход на премиум-версию требуется ежемесячная плата в размере 3,50\$ для Android и 5\$ для iOS, но она расширяет возможности приложения благодаря неограниченным повторяющимся задачам, цветным меткам и флажкам, напоминаниям о местоположении и 100 ГБ для хранения файлов.

1.1.3 Google Tasks

Google Tasks такой же простой, как и список дел на бумаге. Это потрясающе минималистичное и хорошо разработанное приложение, которое делает именно то, что должно, и не более того. Вы можете создавать задачи, составлять их описание, а затем добавлять подзадачи. Они появляются в маркированном списке, и можно пометить каждую подзадачу завершенной, когда придет время. Каждая задача находится под списком, и количество списков, которые вы можете создать, не ограничено.

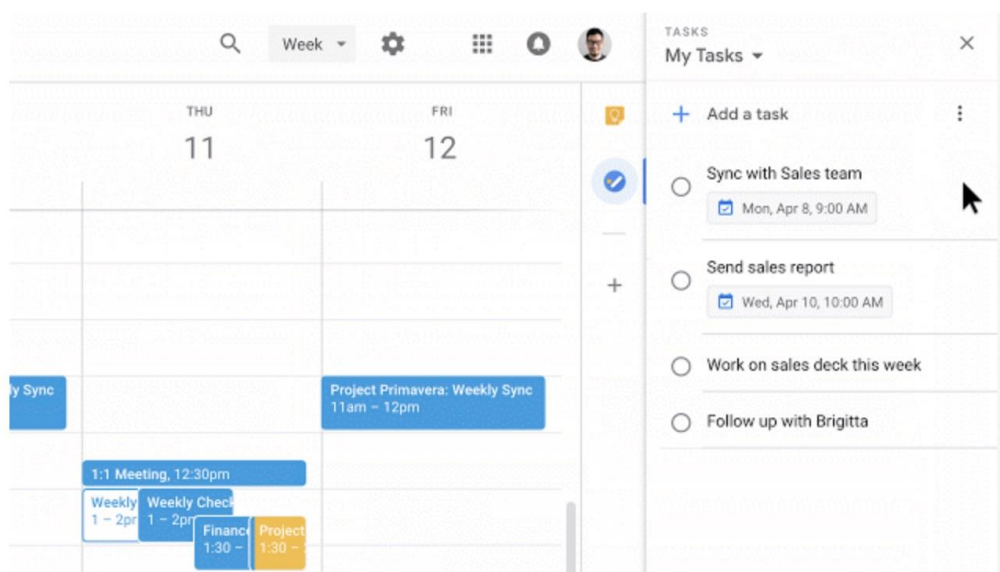


Рисунок 1.3 – Интерфейс приложения «Google Tasks»

Вы можете иметь список покупок, список дел и многое другое. В обмен на простоту Google Tasks теряем некоторые более глубокие теги и организационные функции, которые вы можете найти в других приложениях. Задачи Google доступны на iOS и Android. Если вы используете Gmail в Интернете, вы можете увидеть обзор своих задач в правом краю интерфейса, рядом с приложениями Календарь и Google Keep.

1.1.4 Todoist

Если вам нужно специальное приложение со списком дел, то к Todoist стоит присмотреться. Это одно из крупнейших приложений с огромным количеством пользователей и проверенной годами эффективностью. Вы можете зарегистрироваться с помощью своего профиля в Facebook или учетной записи Google, и начать работу так же просто, как ввести свою первую задачу и нажать «Отправить».

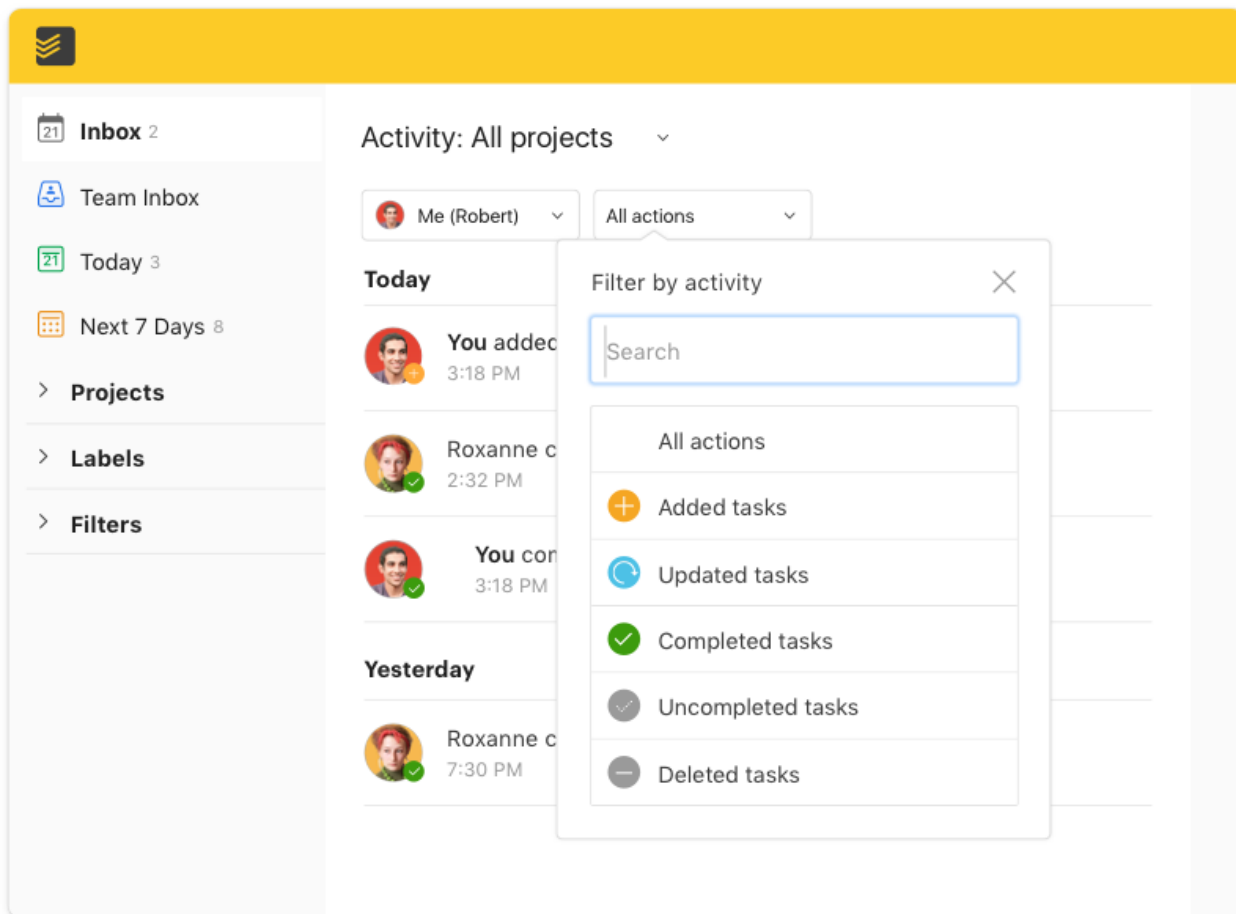


Рисунок 1.4 – Интерфейс приложения «Todoist»

Можно установить крайний срок выполнения, а также назначить приоритет задаче, или положить ее в группу схожих задач. Выполнение включает в себя установку галочки рядом с задачей, и есть определенное удовлетворение от этого действия и его анимации. Настройка напоминаний о задачах, дополнительные активные проекты, комментарии к задачам и автоматическое резервное копирование — все это дополнительные функции, а подписка на Todoist Premium будет приносить вам около 29\$ в год. Тем не менее, если вы опробовали его и считаете, что оно того стоит, то 29\$ — это разумно для годовой цены.

1.1.5 Evernote

Evernote — еще один список задач, который содержит множество других опций, и, вместе с тем, это приложение для заметок. Он поддерживает несколько способов делать заметки и напоминания, включая голосовые заметки, полные списки и фотографии. Тем не менее, Evernote делает еще один шаг вперед благодаря возможности добавлять видео и прикреплять документы Word или PDF-файлы к своим спискам. Как и в случае с Google Keep, каждый раз, когда вы добавляете что-то в Evernote, он будет синхронизироваться на всех ваших устройствах, но в отличие от Keep, есть опция автономного доступа к вашим файлам — но только если вы платите за версию Premium.

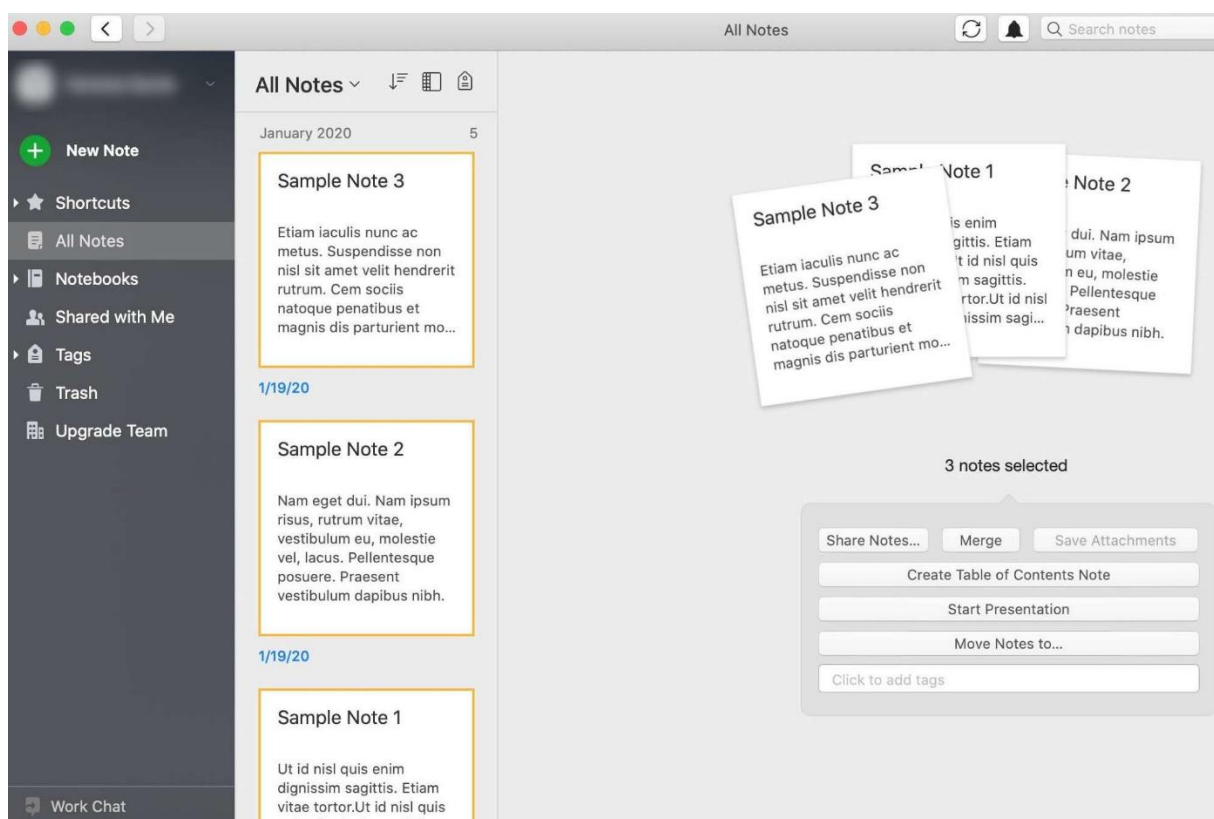


Рисунок 1.5 – Интерфейс приложения «Evernote»

Evernote бесплатен в версии Evernote Basic, и стоит 8\$ в месяц или 70\$ в год за Evernote Premium. Но у него есть целый ряд дополнительных опций, таких как автономный доступ, дополнительное хранилище и возможность добавить поддержку паролей для блокировки ваших ноутбуков. Тем не менее, только вы можете решить, стоит ли это ваших денег или нет, и мы всегда рекомендуем какое-то время использовать бесплатную версию, чтобы оценить, стоит ли вам улучшать ее.

1.2 Формирование требований к проектируемому программному средству

Целью данной курсовой работы является создание приложения-ежедневника для операционной системы Windows.

В результате сравнения аналогов программного обеспечения и анализа предметной области в данном курсовом проекте предполагается создание приложения, имеющего следующий функционал. Необходимо создать:

1. Механизм для добавления, изменения и удаления задач
2. Два вида задач: события (ивенты), имеющие продолжительность во времени и напоминания, имеющие только временную отметку
3. Возможность доступа к элементам управления несколькими способами, например, путём клика по кнопке, выбора пункта меню или нажатия комбинации клавиш на клавиатуре (горячие клавиши).
4. Механизм контроля за сохранением данных

2 РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА

При загрузке приложения происходит чтение файла данных и загрузка их во внутреннюю структуру представления данных, представляющую собой список из экземпляров класса задачи. Каждый такой экземпляр содержит название задачи, дату и время начала, дату и время окончания задачи, а также её тип: задачи вида напоминания не хранят данных в полях даты и времени окончания, а временная метка задачи хранится в полях даты и времени начала задачи [1].

При запуске приложения в окне отображается таблица с колонками, каждая из которых представляет собой поле класса задачи; элемент контроля «чек-бокс» напротив каждой задачи, характеризующий её состояние: в случае если элемент отмечен, задача считается выполненной; кнопки управления, такие как добавление задачи, изменение задачи, удаление задачи, удаление выполненных задач.

При нажатии на кнопку добавления задачи открывается модальное окно со следующими элементами управления: текстовое поле для ввода названия задачи с клавиатуры, группа радиокнопок «напоминание» и «событие» для выбора типа задачи, две группы элементов управления выбора даты и времени для выбора даты и времени начала и даты и времени окончания задачи. В случае если пользователь выбирает тип задачи напоминание, элементы выбора даты и времени окончания задачи блокируются, и разблокировываются в противном случае.

В элементе управления выбора даты есть возможность как ввести дату вручную с клавиатуры (для этого необходимо сделать клик в рабочей области элемента и начать ввод), так и выбрать дату из выпадающего элемента управления календарь, в котором можно удобно выбрать дату, например, в следующем году или через несколько месяцев.

В элементе управления выбора времени также есть возможность ввода с клавиатуры (для этого также необходимо кликнуть в рабочей области и осуществить ввод) и выбора необходимого времени с помощью стрелок в правой части элемента. Для этого необходимо кликнуть по паре цифр, характеризующих часы или минуты, а затем, кликая по стрелкам, изменить значение. За один клик можно изменить значение часов на ± 1 час, значение минут на ± 1 минуту.

В нижней части модального окна расположены кнопки подтверждения и отмены внесенных изменений. В случае клика по кнопке отмены, диалоговое окно закрывается, а все действия пользователя в диалоговом окне теряются. В

случае клика по кнопке подтверждения (ОК), модальное окно также закрывается, а все действия пользователя сохраняются и сразу же отображаются в таблице.

При нажатии на кнопку обновления задачи также открывается модальное диалоговое окно, в котором все значения полей, кнопок и элементов выбора даты и времени отображаются в соответствии с выбранной задачей. При необходимости пользователь может внести корректировки способами, описанными выше. Далее также следует либо подтвердить, либо отменить изменения.

При нажатии на кнопку удаления задачи необходимо предварительно выбрать задачу, подлежащую удалению, в таблице, путем одиночного клика по ней (фон задачи изменится на синий). После чего следует нажать на клавишу удаления, выбранная задача пропадет из списка. В случае, если задача не будет выбрана, после клика по кнопке удаления задачи на экране отобразится соответствующее сообщение.

При нажатии на кнопку удаления всех выполненных задач, из таблицы пропадут все задачи, отмеченные, как выполненные (элемент управления «чек-бокс» отмечен). В случае, если таковые задачи отсутствуют, ничего не произойдет.

После совершения таких действий, как добавление новой задачи, изменение существующей, удаление выделенной задачи, удаление выполненных задач либо изменение состояния задачи (перевод из невыполненного состояния в выполненное либо наоборот), система переводится в несохраненное состояние. Для перевода ее обратно в сохраненное состояние необходимо либо нажать комбинацию клавиш `ctrl + S`, либо в меню File выбрать пункт Save [2].

В случае, если пользователь переведет систему в несохраненное состояние и попытается закрыть приложение, система оповестит его о несохраненных изменениях и предложит сохранить внесенные изменения, не сохранять внесенные изменения, либо вернуться назад в приложение.

3 ОБОСНОВАНИЕ ТЕХНИЧЕСКИХ ПРИЕМОВ ПРОГРАММИРОВАНИЯ

3.1 Обоснование выбора языка программирования

В качестве языка программирования для реализации данного программного средства был выбран C++ — компилируемый статически типизированный язык программирования общего назначения. C++ позволяет создавать как простые приложения и утилиты, так и коммерческие системы, поддерживает множество парадигм программирования. Язык широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также развлекательных приложений (игр). [3]

3.2 Обоснование выбора библиотек WinApi

Windows API — общее наименование набора базовых функций интерфейсов программирования приложений операционных систем семейств Microsoft Windows корпорации «Майкрософт». Предоставляет прямой способ взаимодействия приложений пользователя с операционной системой Windows. Для создания программ, использующих Windows API, корпорация «Майкрософт» выпускает комплект разработчика программного обеспечения, который называется Platform SDK и содержит документацию, набор библиотек, утилит и других инструментальных средств для разработки [4].

Windows API спроектирован для использования в языке Си для написания прикладных программ, предназначенных для работы под управлением операционной системы MS Windows. Работа через Windows API — это наиболее близкий к операционной системе способ взаимодействия с ней из прикладных программ. Более низкий уровень доступа, необходимый только для драйверов устройств, в текущих версиях Windows предоставляется через Windows Driver Model [5].

Windows API представляет собой множество функций, структур данных и числовых констант, следующих соглашениям языка Си. В то же время конвенция вызова функций отличается от cdecl, принятой для языка C: Windows API использует stdcall (winapi). Все языки программирования, способные вызывать такие функции и оперировать такими типами данных в программах, исполняемых в среде Windows, могут пользоваться этим API. В частности, это языки C++, C#, Pascal, Visual Basic и многие другие.

4 ТЕСТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

Таблица 4.1 – Результаты ручного тестирования программы

№	Тест	Ожидаемый результат	Полученный результат
1	Наличие главного меню: 1. Запустить приложение 2. Дождаться окончания загрузки главного окна 3. Проверить элементы окна	1. Приложение запускается 2. На экране отображается главное окно приложения 3. Отображаются следующие элементы: – таблица задач – кнопка создания задачи – кнопка изменения задачи – кнопка удаления задачи – кнопка удаления выполненных задач	1. Приложение запускается 2. На экране отображается главное окно приложения 3. Отображаются следующие элементы: – таблица задач – кнопка создания задачи – кнопка изменения задачи – кнопка удаления задачи – кнопка удаления выполненных задач
2	Кликабельность кнопки создания задачи: 1. Кликнуть по кнопке создания задачи	1. Отображается модальное диалоговое окно со следующими элементами: – текстовое поле названия задачи – две радиокнопки типа задачи – два элемента	1. Отображается модальное диалоговое окно со следующими элементами: – текстовое поле названия задачи – две радиокнопки типа задачи – два элемента

Продолжение таблицы 4.1

№	Тест	Ожидаемый результат	Полученный результат
		<p>выбора даты начала и окончания задачи</p> <p>— два элемента выбора времени начала и окончания задачи</p> <p>— кнопки сохранения и отмены внесенных изменений</p>	<p>выбора даты начала и окончания задачи</p> <p>— два элемента выбора времени начала и окончания задачи</p> <p>— кнопки сохранения и отмены внесенных изменений</p>
3	<p>Кликабельность текстового поля:</p> <p>1. Кликнуть по текстовому полю</p> <p>2. Осуществить ввод</p>	<p>1. В области ввода текстового поля мигает курсор</p> <p>2. В области ввода текстового поля отображается вводимый текст</p>	<p>1. В области ввода текстового поля мигает курсор</p> <p>2. В области ввода текстового поля отображается вводимый текст</p>
4	<p>Кликабельность радиокнопок выбора типа задачи:</p> <p>1. Кликнуть по кнопке выбора типа задачи напоминание</p> <p>2. Кликнуть по кнопке выбора типа задачи событие</p>	<p>1. Элементы управления выбора даты и времени окончания задачи заблокировались</p> <p>2. Элементы управления выбора даты и времени окончания задачи разблокировались</p>	<p>1. Элементы управления выбора даты и времени окончания задачи заблокировались</p> <p>2. Элементы управления выбора даты и времени окончания задачи разблокировались</p>
5	<p>Возможность ввода в элемент управления выбора даты начала (окончания) задачи вручную:</p> <p>1. Кликнуть по элементу</p>	<p>1. Текущее значение элемента управления выделилось</p> <p>2. В рабочей области элемента управления</p>	<p>1. Текущее значение элемента управления выделилось</p> <p>2. В рабочей области элемента управления</p>

Продолжение таблицы 4.1

№	Тест	Ожидаемый результат	Полученный результат
	управления выбора даты начала (окончания) задачи 2. Осуществить ввод	отображается ввод	отображается ввод
6	Возможность ввода в элемент управления выбора даты начала (окончания) задачи при помощи элемента управления календарь: 1. Кликнуть по элементу управления выбора даты начала (окончания) задачи 2. Кликнуть по иконке выбора даты из календаря 3. Выбрать дату в календаре	1. Текущее значение элемента управления выделилось 2. На экране отобразился элемент управления календарь 3. Элемент управления календарь скрылся, в рабочей области элемента управления отображается выбранная дата	1. Текущее значение элемента управления выделилось 2. На экране отобразился элемент управления календарь 3. Элемент управления календарь скрылся, в рабочей области элемента управления отображается выбранная дата
7	Возможность ввода в элемент управления выбора времени начала (окончания) задачи вручную: 1. Кликнуть по элементу управления выбора времени начала (окончания) задачи 2. Осуществить ввод	1. Текущее значение элемента управления выделилось 2. В рабочей области элемента управления отображается ввод	1. Текущее значение элемента управления выделилось 2. В рабочей области элемента управления отображается ввод

Продолжение таблицы 4.1

№	Тест	Ожидаемый результат	Полученный результат
8	<p>Возможность ввода в элемент управления выбора времени начала (окончания) задачи при помощи элементов управления стрелка:</p> <ol style="list-style-type: none"> 1. Кликнуть по элементу управления выбора времени начала (окончания) задачи 2. Кликнуть по иконке стрелки + шаг (- шаг) 	<ol style="list-style-type: none"> 1. Текущее значение элемента управления выделилось 2. В рабочей области элемента управления отображается изменённое значение времени (+- шаг) 	<ol style="list-style-type: none"> 1. Текущее значение элемента управления выделилось 2. В рабочей области элемента управления отображается изменённое значение времени (+- шаг)
9	<p>Кликабельность кнопки отмены внесённых изменений:</p> <ol style="list-style-type: none"> 1. Кликнуть по кнопке отмены внесённых изменений 	<ol style="list-style-type: none"> 1. Модальное диалоговое окно скрывается, в таблице задач не происходит никаких изменений 	<ol style="list-style-type: none"> 1. Модальное диалоговое окно скрывается, в таблице задач не происходит никаких изменений
10	<p>Кликабельность кнопки сохранения внесённых изменений:</p> <ol style="list-style-type: none"> 1. Кликнуть по кнопке сохранения внесённых изменений 	<ol style="list-style-type: none"> 1. Модальное диалоговое окно скрывается, в таблице задач отображаются изменения, введённые в диалоговом окне 	<ol style="list-style-type: none"> 1. Модальное диалоговое окно скрывается, в таблице задач отображаются изменения, введённые в диалоговом окне
11	<p>Кликабельность кнопки изменения задачи после выбора задачи в таблице задач:</p> <ol style="list-style-type: none"> 1. Кликнуть по задаче в таблице задач 	<ol style="list-style-type: none"> 1. Выбранная задача выделилась 2. Отображается диалоговое окно с элементами управления, заполненными в соответствии с 	<ol style="list-style-type: none"> 1. Выбранная задача выделилась 2. Отображается диалоговое окно с элементами управления, заполненными в соответствии с

Продолжение таблицы 4.1

№	Тест	Ожидаемый результат	Полученный результат
	2. Кликнуть по кнопке изменения задачи	выбранной задачей	выбранной задачей
12	Кликабельность кнопки изменения задачи без выбора задачи в таблице задач: 1. Кликнуть по кнопке изменения задачи	1. Отображается модальное диалоговое окно с сообщением о необходимости выбора задачи до нажатия кнопки изменения задачи	1. Отображается модальное диалоговое окно с сообщением о необходимости выбора задачи до нажатия кнопки изменения задачи
13	Кликабельность кнопки удаления задачи после выбора задачи в таблице задач: 1. Кликнуть по задаче в таблице задач 2. Кликнуть по кнопке удаления задачи	1. Выбранная задача выделилась 2. Выбранная задача скрылась в таблице задач	1. Выбранная задача выделилась 2. Выбранная задача скрылась в таблице задач
14	Кликабельность кнопки удаления задачи без выбора задачи в таблице задач: 1. Кликнуть по кнопке удаления задачи	1. Отображается модальное диалоговое окно с сообщением о необходимости выбора задачи до нажатия кнопки удаления задачи	1. Отображается модальное диалоговое окно с сообщением о необходимости выбора задачи до нажатия кнопки удаления задачи
15	Кликабельность кнопки удаления выполненных задач при наличии выполненных задач в таблице задач:	1. Выполненные задачи скрылись в таблице задач	1. Выполненные задачи скрылись в таблице задач

Продолжение таблицы 4.1

№	Тест	Ожидаемый результат	Полученный результат
	1. Кликнуть по кнопке удаления выполненных задач		
16	Кликабельность кнопки удаления выполненных задач при отсутствии выполненных задач в таблице задач: 1. Кликнуть по кнопке удаления выполненных задач	1. В таблице задач не происходит никаких изменений	1. В таблице задач не происходит никаких изменений
17	Кликабельность неотмеченного элемента управления «чек-бокс»: 1. Кликнуть по неотмеченному элементу управления «чек-бокс»	1. Элемент управления перешёл в отмеченное состояние	1. Элемент управления перешёл в отмеченное состояние
18	Кликабельность отмеченного элемента управления «чек-бокс»: 2. Кликнуть по отмеченному элементу управления «чек-бокс»	2. Элемент управления перешёл в неотмеченное состояние	2. Элемент управления перешёл в неотмеченное состояние
19	Кликабельность кнопки закрытия приложения при отсутствии несохраненных изменений:	1. Приложение закрывается	1. Приложение закрывается

Продолжение таблицы 4.1

№	Тест	Ожидаемый результат	Полученный результат
	1. Кликнуть по кнопке закрытия приложения		
20	<p>Кликабельность кнопки закрытия приложения при наличии несохраненных изменений и выборе сохранения изменений:</p> <ol style="list-style-type: none"> 1. Кликнуть по кнопке закрытия приложения 2. Кликнуть по кнопке сохранения изменений 	<ol style="list-style-type: none"> 1. Отображается модальное диалоговое окно с предложением сохранения несохраненных изменений и тремя кнопками выбора: «да», «нет», «отмена» 2. Изменения сохраняются, приложение закрывается 	<ol style="list-style-type: none"> 1. Отображается модальное диалоговое окно с предложением сохранения несохраненных изменений и тремя кнопками выбора: «да», «нет», «отмена» 2. Изменения сохраняются, приложение закрывается
21	<p>Кликабельность кнопки закрытия приложения при наличии несохраненных изменений и выборе несохранения изменений:</p> <ol style="list-style-type: none"> 1. Кликнуть по кнопке закрытия приложения 2. Кликнуть по кнопке несохранения изменений 	<ol style="list-style-type: none"> 1. Отображается модальное диалоговое окно с предложением сохранения несохраненных изменений и тремя кнопками выбора: «да», «нет», «отмена» 2. Изменения не сохраняются, приложение закрывается 	<ol style="list-style-type: none"> 1. Отображается модальное диалоговое окно с предложением сохранения несохраненных изменений и тремя кнопками выбора: «да», «нет», «отмена» 2. Изменения не сохраняются, приложение закрывается
22	Кликабельность кнопки закрытия приложения при наличии	1. Отображается модальное диалоговое окно с предложением	1. Отображается модальное диалоговое окно с предложением

Продолжение таблицы 4.1

№	Тест	Ожидаемый результат	Полученный результат
	<p>несохраненных изменений и выборе отмены выхода:</p> <ol style="list-style-type: none"> 1. Кликнуть по кнопке закрытия приложения 2. Кликнуть по кнопке отмены выхода из приложения 	<p>сохранения несохраненных изменений и тремя кнопками выбора:</p> <p>«да», «нет», «отмена»</p> <ol style="list-style-type: none"> 2. Модальное диалоговое окно скрывается 	<p>сохранения несохраненных изменений и тремя кнопками выбора:</p> <p>«да», «нет», «отмена»</p> <ol style="list-style-type: none"> 2. Модальное диалоговое окно скрывается
23	<p>Работоспособность комбинации клавиш CTRL + N, эквивалентных нажатию кнопки создания задачи:</p> <ol style="list-style-type: none"> 1. Нажать на клавиатуре CTRL + N 	<ol style="list-style-type: none"> 1. Отображается модальное диалоговое окно со следующими элементами: <ul style="list-style-type: none"> – текстовое поле названия задачи – две радиокнопки типа задачи – два элемента выбора даты начала и окончания задачи – два элемента выбора времени начала и окончания задачи – кнопки сохранения и отмены 	<ol style="list-style-type: none"> 2. Отображается модальное диалоговое окно со следующими элементами: <ul style="list-style-type: none"> – текстовое поле названия задачи – две радиокнопки типа задачи – два элемента выбора даты начала и окончания задачи – два элемента выбора времени начала и окончания задачи – кнопки сохранения и отмены

Продолжение таблицы 4.1

№	Тест	Ожидаемый результат	Полученный результат
		внесённых изменений	внесённых изменений
24	Кликабельность пункта меню сохранения изменений: 1. Кликнуть пункт меню File> Save	1. Внесённые изменения сохранились	1. Внесённые изменения сохранились
25	Работоспособность комбинации клавиш CTRL + S, эквивалентных нажатию пункта меню сохранения изменений: 1. Нажать на клавиатуре CTRL + S	1. Внесённые изменения сохранились	1. Внесённые изменения сохранились

временная метка окончания должна быть строго больше временной метки начала задачи. В случае попытки сохранения неверно заполненной задачи отобразится сообщение с ошибкой (рис. 5.3). При нажатии на кнопку «Отмена» внесённые изменения не применяются, а форма редактирования задачи закрывается.

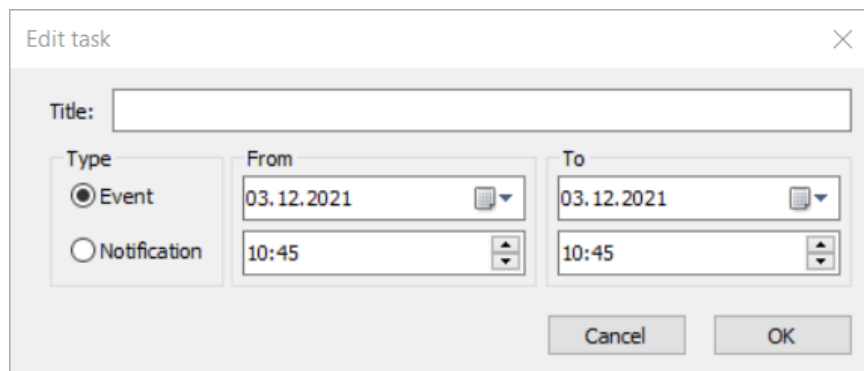


Рисунок 5.2 – Диалоговое окно редактирования задачи

При выборе типа задачи напоминание («Notification»), секция «To», содержащая поля для ввода даты и времени окончания блокируется (рис.5.4).

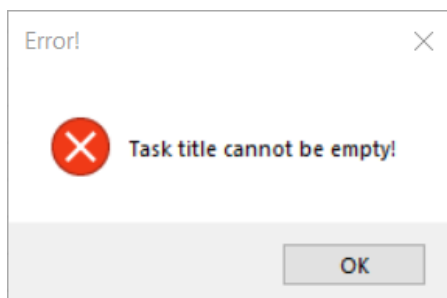


Рисунок 5.3 – Сообщение с причиной ошибки сохранения формы

Дату и время можно вводить как вручную, так и с помощью вспомогательных элементов: у элемента выбора даты справа отображается иконка календаря, по нажатию на которую открывается мини-календарь, с возможностью перехода по месяцам, годам, десятилетиям и столетиям (рис.5.5).

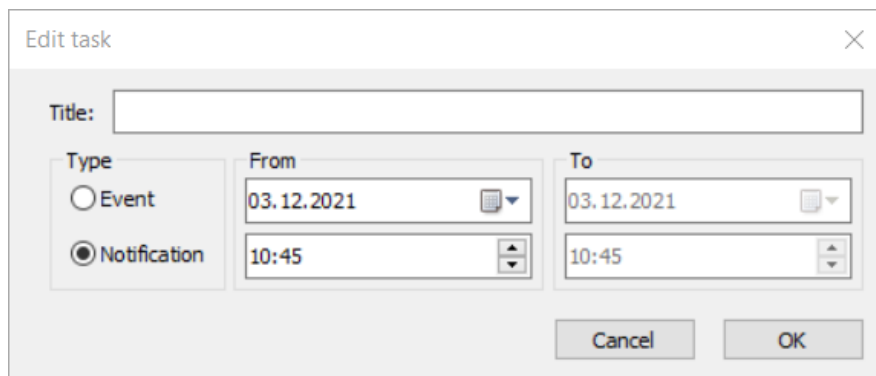


Рисунок 5.4 – Выбран тип задачи напоминание

У элемента выбора времени справа расположены две стрелки, с помощью которых можно увеличивать/уменьшать часы/минуты на 1 час/минуту.

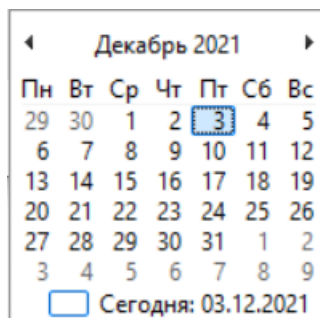


Рисунок 5.5 – Вспомогательный элемент выбора даты календарь

5.3 Изменение задачи

Для изменения задачи необходимо выбрать ее в таблице задач, а затем кликнуть по кнопке «Update task...». В случае, если задача не выбрана, отобразится соответствующее сообщение.

В открывшемся модальном диалоговом окне все поля формы заполнены в соответствии с выбранной задачей (рис. 5.6).

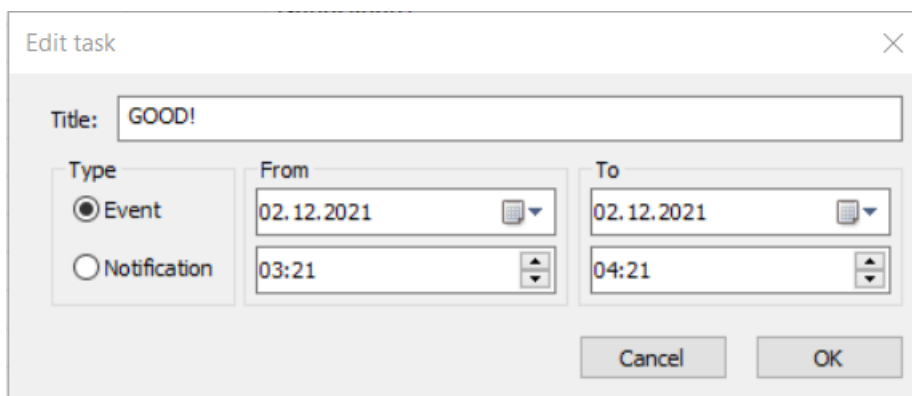


Рисунок 5.6 – Заполненное в соответствии с выбранной задачей диалоговое окно

Теперь можно изменить любые параметры задачи, правила те же, что и при создании: название не должно быть пустым, а временная метка начала должна быть строго меньше временной метки окончания задачи. При нажатии на кнопку «ОК» все внесенные изменения применятся и сразу же отобразятся в таблице задач; для отмены применения изменений следует нажать «Cancel».

5.4 Удаление задач

Для удаления одной задачи необходимо выбрать ее в таблице задач и нажать «Delete task». В случае, если задача не будет выбрана, отобразится соответствующее сообщение.

Выбранная задача пропадёт из списка задач (рис. 5.7, 5.8).

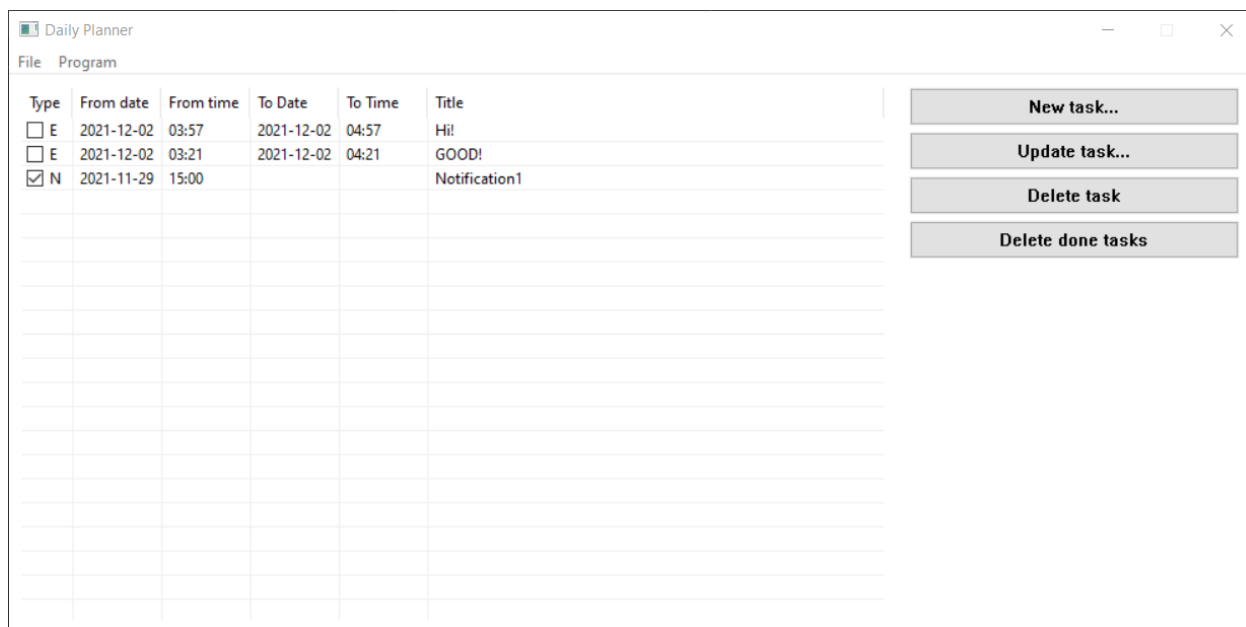


Рисунок 5.7 – Таблица задач до удаления первой задачи

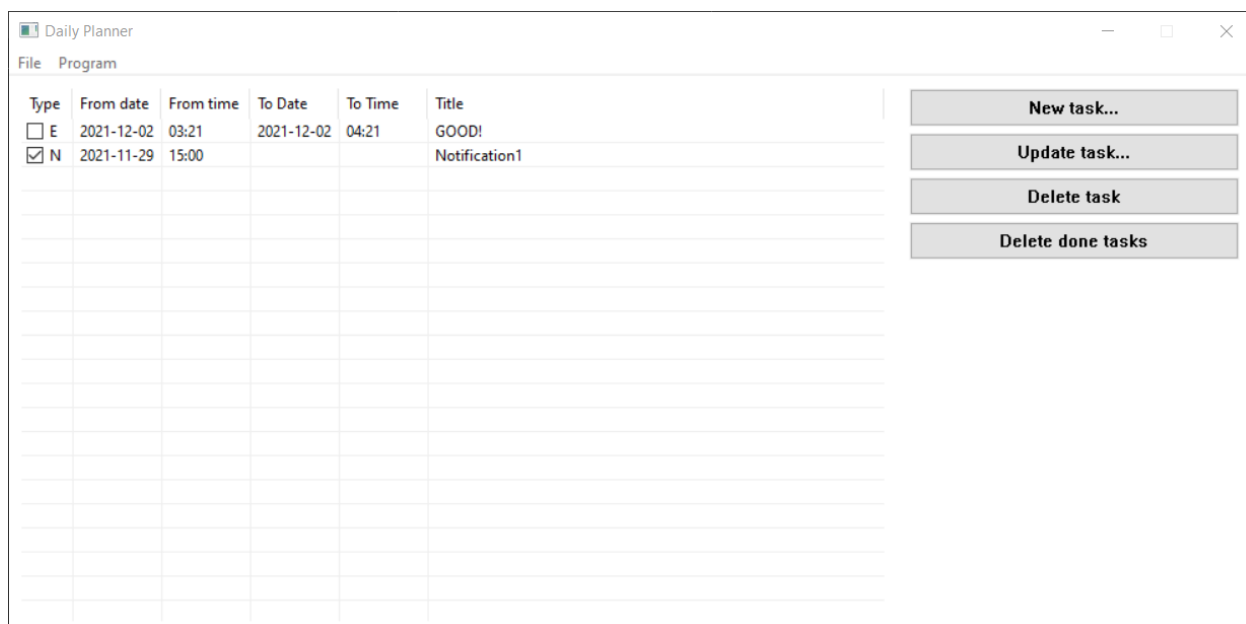


Рисунок 5.8 – Таблица задач после удаления первой задачи

Для удобного удаления всех выполненных задач необходимо кликнуть по кнопке «Delete done tasks». В этом случае, если в таблице есть выполненные

задачи, они будут удалены и перестанут отображаться в таблице, в противном случае ничего не изменится.

5.5 Сохранение изменений после закрытия приложения

В приложении есть возможность сохранять изменения после его закрытия, чтобы можно было продолжить работу через какое-то время. Для того, чтобы сохранить изменения следует кликнуть пункт меню File> Save, либо нажать клавиши CTRL + S, в этом случае все изменения сохранятся файл и можно будет безопасно закрыть приложение (рис. 5.9).

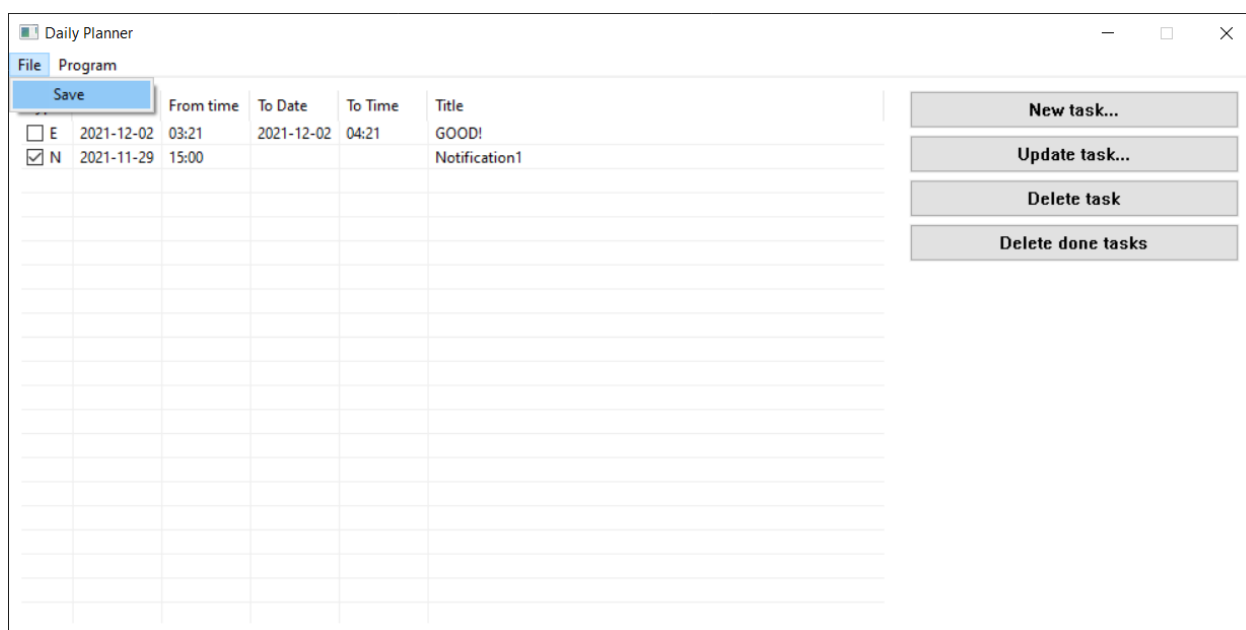


Рисунок 5.9 – Пункт меню File> Save

Приложение переходит в несохраненное состояние в случае, если изменился статус какой-либо из задач (выполнена/не выполнена), была добавлена новая задача, изменилась существующая либо были удалены какие-либо задачи.

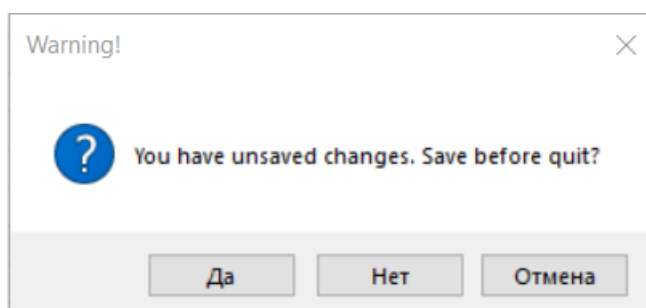


Рисунок 5.10 – Сообщение, отображаемое в случае попытки закрытия приложения без сохранения изменений

В случае если пользователь переведет приложение в несохраненное состояние, не сохранит изменения в файл и попытается закрыть приложение, отобразится соответствующее сообщение с несколькими действиями (рис. 5.10).

В случае клика по кнопке «Да» все внесенные изменения сохраняются, а приложение закроется; если пользователь кликнет по кнопке «Нет», изменения не сохраняются, приложение также закроется. По нажатию на кнопку «Отмена» закроется модальное окно с сообщением, позволяя продолжить работу с приложением.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы было разработано программное средство «Daily Planner», представляющее собой электронный ежедневник, который можно использовать для фиксации и контроля своих задач, планирования времени. Приложение имеет простой, «user-friendly» интерфейс, в качестве цветовой палитры и дизайна были выбраны стандартные для операционной системы Windows 10 решения.

В процессе проектирования были изучены необходимые для разработки части библиотеки WinApi, в частности работа с элементом управления таблица «List View», элементами управления кнопка «Button» и выбора даты и времени «Date Time Picker», а также изучен механизм работы WinApi-приложений, цикл обработки сообщений.

Также были исследованы существующие аналоги, определены цели и задачи проектирования. На этапе разработки были определены все возможные способы взаимодействия с приложением, алгоритмы работы в каждом из них. В руководстве пользователя доступным языком описаны правила и способы работы с приложением. Программное средство было протестировано и отлажено.

В дальнейшем возможны изменения, расширяющие функционал программного средства.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] Центр разработки для Windows [Электронный ресурс] Режим доступа: <https://docs.microsoft.com/ru-ru/windows/win32/api/>
- [2] Литвиненко Н. А. Технология программирования на C++ - БХВ-Петербург, 2010
- [3] Рихтер Дж. Windows для профессионалов: создание эффективных Win32-приложений с учетом специфики 64-разрядной версии Windows: Пер. с англ. — 4-е изд. — СПб.: Питер; М.: Издательско-торговый дом "Русская редакция", 2003.
- [4] Шупак Ю. А. Win32 API. Разработка приложений для Windows. — СПб.: Питер, 2008
- [5] Верма Р. Д. Справочник по функциям Win32 API. — 2-е изд., перераб. и доп. — М.: Горячая линия — Телеком, 2005.

ПРИЛОЖЕНИЕ А

Исходный код программы

А1 Текст программного модуля resource.h

```
//{{NO_DEPENDENCIES}}
// Microsoft Visual C++ generated include file.
// Used by Resource.rc
//
#define IDD_EDIT_TASK 5
#define IDR_MAIN_MENU 101
#define IDS_LVC_FIRST 102
#define IDS_LVC_SECOND 103
#define IDS_LVC_THIRD 104
#define IDS_LVC_FOURTH 105
#define IDS_LVC_FIFTH 106
#define IDS_LVC_SIXTH 107
#define IDC_EDIT_TITLE 1001
#define IDC_T_EVENT 1002
#define IDC_T_NOTIFICATION 1003
#define IDC_DATEFROM 1005
#define IDC_TIMEFROM 1007
#define IDC_TIMETO 1008
#define IDC_DATETO 1010
#define ID_FILE 40001
#define IDM_FILE_SAVE 40002
#define ID_PROGRAM 40004
#define IDM_PROGRAM_ABOUT 40005

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE 105
#define _APS_NEXT_COMMAND_VALUE 40007
#define _APS_NEXT_CONTROL_VALUE 1008
#define _APS_NEXT_SYMED_VALUE 101
#endif
#endif
```

А2 Пространство имён buttons

А2.1 Текст программного модуля buttons.h

```
#pragma once

#include <windows.h>

#define IDB_NEW_TASK 3
#define IDB_UPD_TASK 4
#define IDB_DEL_TASK 5
#define IDB_DEL_DONE 6
#define IDB_GET_TODAY 7
```



```

extern HINSTANCE hInst;

HWND CreateNewTaskButton(HWND hWndParent);

HWND CreateUpdateTaskButton(HWND hWndParent);

HWND CreateDeleteTaskButton(HWND hWndParent);

HWND CreateDeleteDoneTasksButton(HWND hWndParent);

HWND CreateGetTodayTasksButton(HWND hWndParent);

```

A2.2 Текст программного модуля buttons.cpp

```

#include "buttons.h"

HWND CreateNewTaskButton(HWND hWndParent) {
    return CreateWindow(
        L"BUTTON",
        L"New task...",
        WS_TABSTOP | WS_VISIBLE | WS_CHILD,
        710, 10, 260, 30,
        hWndParent,
        (HMENU)IDB_NEW_TASK,
        hInst,
        NULL
    );
}

HWND CreateUpdateTaskButton(HWND hWndParent) {
    return CreateWindow(
        L"BUTTON",
        L"Update task...",
        WS_TABSTOP | WS_VISIBLE | WS_CHILD,
        710, 45, 260, 30,
        hWndParent,
        (HMENU)IDB_UPD_TASK,
        hInst,
        NULL
    );
}

HWND CreateDeleteTaskButton(HWND hWndParent) {
    return CreateWindow(
        L"BUTTON",
        L"Delete task",
        WS_TABSTOP | WS_VISIBLE | WS_CHILD,
        710, 80, 260, 30,
        hWndParent,
        (HMENU)IDB_DEL_TASK,
        hInst,
        NULL
    );
}

HWND CreateDeleteDoneTasksButton(HWND hWndParent) {
    return CreateWindow(
        L"BUTTON",
        L"Delete done tasks",
        WS_TABSTOP | WS_VISIBLE | WS_CHILD,
        710, 115, 260, 30,
        hWndParent,

```

```

        (HMENU)IDB_DEL_DONE,
        hInst,
        NULL
    );
}

HWND CreateGetTodayTasksButton(HWND hWndParent) {
    return CreateWindow(
        L"BUTTON",
        L"Get today tasks...",
        WS_TABSTOP | WS_VISIBLE | WS_CHILD,
        710, 150, 260, 30,
        hWndParent,
        (HMENU)IDB_GET_TODAY,
        hInst,
        NULL
    );
}

```

A3 Пространство имён util

A3.1 Текст программного модуля util.h

```

#pragma once

#include <windows.h>
#include <string>
#include <vector>
#include <sstream>
#include <iomanip>

#define TIME_DELIM L':'
#define DATE_DELIM L'-'

using namespace std;

LPWSTR s2ws(const string& s);

vector<string> split(string src, CHAR delimiter);
vector<wstring> wsplit(wstring src, WCHAR delimiter);

LPSYSTEMTIME timestampToSysTime(LPWSTR date, LPWSTR time);

void SysTimeToDate(LPSYSTEMTIME lpst, LPWSTR date);
void SysTimeToTime(LPSYSTEMTIME lpst, LPWSTR time);

LPWSTR getSaveFilePath(HWND hWnd, LPWSTR defaultName);

LPWSTR GetCurrentDate();

```

A3.2 Текст программного модуля util.cpp

```

#include "util.h"

```

```

LPWSTR s2ws(const string& s) {
    int sLength = (int)s.length() + 1;
    int bufLength = MultiByteToWideChar(CP_ACP, 0, s.c_str(), sLength, 0, 0);
    LPWSTR buf = new WCHAR[bufLength];
    MultiByteToWideChar(CP_ACP, 0, s.c_str(), sLength, buf, bufLength);
    return buf;
}

vector<string> split(string src, CHAR delimiter) {
    stringstream stream(src);
    string segment;
    vector<string> seglist;

    while (getline(stream, segment, delimiter)) {
        seglist.push_back(segment);
    }

    return seglist;
}

vector<wstring> wsplrit(wstring src, WCHAR delimiter) {
    wstringstream stream(src);
    wstring segment;
    vector<wstring> seglist;

    while (getline(stream, segment, delimiter)) {
        seglist.push_back(segment);
    }

    return seglist;
}

LPSYSTEMTIME timestampToSysTime(LPWSTR date, LPWSTR time) {
    vector<wstring> dateComponents = wsplrit(date, DATE_DELIM);
    vector<wstring> timeComponents = wsplrit(time, TIME_DELIM);
    LPSYSTEMTIME st = new SYSTEMTIME();
    st->wYear = stoi(dateComponents.at(0));
    st->wMonth = stoi(dateComponents.at(1));
    st->wDay = stoi(dateComponents.at(2));
    st->wHour = stoi(timeComponents.at(0));
    st->wMinute = stoi(timeComponents.at(1));
    return st;
}

void SysTimeToDate(LPSYSTEMTIME lpst, LPWSTR date) {
    WCHAR year[5] = { 0 }, month[3] = { 0 }, day[3] = { 0 };
    _itow_s(lpst->wYear, year, 5, 10);
    _itow_s(lpst->wMonth, month, 3, 10);
    _itow_s(lpst->wDay, day, 3, 10);

    if (lpst->wMonth < 10) {
        month[1] = month[0];
        month[0] = '0';
        month[2] = '\\0';
    }

    if (lpst->wDay < 10) {
        day[1] = day[0];
        day[0] = '0';
        day[2] = '\\0';
    }

    swprintf_s(date, 11, L"%s-%s-%s", year, month, day);
}

```

```

void SysTimeToTime(LPSYSTEMTIME lpst, LPWSTR time) {
    WCHAR hour[3] = { 0 }, minute[3] = { 0 };
    _itow_s(lpst->wHour, hour, 3, 10);
    _itow_s(lpst->wMinute, minute, 3, 10);

    if (lpst->wHour < 10) {
        hour[1] = hour[0];
        hour[0] = '0';
        hour[2] = '\\0';
    }

    if (lpst->wMinute < 10) {
        minute[1] = minute[0];
        minute[0] = '0';
        minute[2] = '\\0';
    }

    swprintf_s(time, 6, L"%s:%s", hour, minute);
}

LPWSTR getSaveFilePath(HWND hWnd, LPWSTR defaultName) {
    /*OPENFILENAME ofn;
    WCHAR szPath[100];
    ZeroMemory(&ofn, sizeof(OPENFILENAME));
    ofn.lStructSize = sizeof(OPENFILENAME);
    ofn.hwndOwner = hWnd;
    ofn.lpstrFile = szPath;
    ofn.lpstrFile[0] = '\\0';
    ofn.nMaxFile = 100;
    ofn.lpstrFilter = L"Text files\\0*.txt\\0";
    ofn.nFilterIndex = 1;

    BOOL fOk = GetSaveFileName(&ofn);
    if (fOk) {
        return szPath;
    }
    return nullptr;*/
    WCHAR szPath[MAX_PATH] = {};

    OPENFILENAME ofn = { sizeof(ofn) };
    ZeroMemory(&ofn, sizeof(ofn));
    ofn.hwndOwner = hWnd;
    ofn.lpstrFilter = L"Text Files\\0*.txt\\0";
    ofn.lpstrFile = szPath;
    ofn.nMaxFile = ARRAYSIZE(szPath);

    BOOL fOk = GetSaveFileName(&ofn);
    if (fOk)
    {
        // Open the file that was selected in
        // the Open File dialog
        return szPath;
    }
    return nullptr;
}

LPWSTR GetCurrentDate() {
    auto t = std::time(nullptr);
    auto tm = *std::localtime(&t);

    std::ostringstream oss;
    oss << std::put_time(&tm, "%Y-%m-%d");
    return s2ws(oss.str());
}

```

А4 Пространство имён source

А4.1 Текст программного модуля source.h

```
#pragma once
#pragma comment(lib, "comctl32.lib")

#include <windows.h>
#include <commctrl.h>

#include "resource.h"

#include "mainwndproc.h"
#include "editdlgproc.h"
#include "PlannerRepository.h"

HINSTANCE hInst;

HWND hWndParent;
HWND hLVTasks;

PlannerRepository repo;
int chosenTaskIndex = -1;

BOOL changed = FALSE;

LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);
```

А4.2 Текст программного модуля source.cpp

```
#include "source.h"

int WINAPI wWinMain(_In_ HINSTANCE hInstance, _In_opt_ HINSTANCE hPrevInstance,
    _In_ LPWSTR lpCmdLine, _In_ int nCmdShow) {

    hInst = hInstance;

    const wchar_t CLASS_NAME[] = L"daily-planner-main";

    WNDCLASSEX wcex = {};
    wcex.cbSize = sizeof(WNDCLASSEX);
    wcex.style = 0;
    wcex.lpfnWndProc = (WNDPROC)WindowProc;
    wcex.cbClsExtra = 0;
    wcex.cbWndExtra = 0;
    wcex.hInstance = hInstance;
    wcex.hIcon = 0;
    wcex.hCursor = LoadCursor(NULL, IDC_ARROW);
    wcex.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1);
    wcex.lpszMenuName = MAKEINTRESOURCE(IDR_MAIN_MENU);
    wcex.lpszClassName = CLASS_NAME;
    wcex.hIconSm = 0;

    RegisterClassEx(&wcex);

    INITCOMMONCONTROLSEX icex;
    icex.dwICC = ICC_LISTVIEW_CLASSES | ICC_DATE_CLASSES;
    InitCommonControlsEx(&icex);
```

```

hWndParent = CreateWindowEx(
    0,
    CLASS_NAME,
    L"Daily Planner",
    WS_OVERLAPPED | WS_CAPTION | WS_SYSMENU | WS_MINIMIZEBOX,
    CW_USEDEFAULT, 0,
    1000, 500,
    NULL,
    NULL,
    hInstance,
    NULL
);

if (!hWndParent) {
    return -1;
}

RegisterHotKey(hWndParent, HK_CTRL_N, MOD_CONTROL | MOD_NOREPEAT, 0x4E);
RegisterHotKey(hWndParent, HK_CTRL_S, MOD_CONTROL | MOD_NOREPEAT, 0x53);

ShowWindow(hWndParent, nCmdShow);

MSG msg = {};
while (GetMessage(&msg, NULL, 0, 0)) {
    TranslateMessage(&msg);
    DispatchMessage(&msg);
}

return 0;
}

LRESULT CALLBACK WindowProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    switch (uMsg) {
        case WM_CREATE:
            _WM_CREATE_MAIN(hWnd, uMsg, wParam, lParam);
            return 0;
        case WM_COMMAND:
            _WM_COMMAND_MAIN(hWnd, uMsg, wParam, lParam);
            return 0;
        case WM_HOTKEY:
            _WM_HOTKEY_MAIN(hWnd, uMsg, wParam, lParam);
            return 0;
        case WM_NOTIFY:
            _WM_NOTIFY_MAIN(hWnd, uMsg, wParam, lParam);
            return 0;
        case WM_CLOSE:
            _WM_CLOSE_MAIN(hWnd, uMsg, wParam, lParam);
            return 0;
        case WM_DESTROY:
            PostQuitMessage(0);
            return 0;
    }
    return DefWindowProc(hWnd, uMsg, wParam, lParam);
}

INT_PTR CALLBACK EditDlgProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    switch (uMsg) {
        case WM_INITDIALOG:
            _WM_INITDIALOG_DLG(hWnd, uMsg, wParam, lParam);
            return TRUE;
        case WM_COMMAND:
            _WM_COMMAND_DLG(hWnd, uMsg, wParam, lParam);
            return TRUE;
    }

```

```

    }
    return FALSE;
}

```

A5 Пространство имён mainwndproc

A5.1 Текст программного модуля mainwndproc.h

```

#pragma once

#include <windows.h>
#include <commctrl.h>

#include "resource.h"

#include "buttons.h"
#include "todoview.h"
#include "PlannerRepository.h"

#define HK_CTRL_N 1
#define HK_CTRL_S 2

extern HINSTANCE hInst;

extern HWND hLVTasks;

extern PlannerRepository repo;
extern int chosenTaskIndex;

extern BOOL changed;

INT_PTR CALLBACK EditDlgProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

void _WM_CREATE_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

void _WM_COMMAND_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

void _WM_HOTKEY_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

void _WM_NOTIFY_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

void _WM_CLOSE_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

void _file_save();

void _program_about(HWND hWnd);

void _new_task(HWND hWnd);

void _upd_task(HWND hWnd);

void _del_task(HWND hWnd);

void _del_done();

void _get_today(HWND hWnd);

BOOL isUnsavedChecks();

```

A5.2 Текст программного модуля mainwndproc.cpp

```
#include "mainwndproc.h"

void _WM_CREATE_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    hLVTasks = CreateView(hWnd);
    InitView(hLVTasks);

    CreateNewTaskButton(hWnd);
    CreateUpdateTaskButton(hWnd);
    CreateDeleteTaskButton(hWnd);
    CreateDeleteDoneTasksButton(hWnd);
    //CreateGetTodayTasksButton(hWnd);
}

void _WM_COMMAND_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    UpdateCheckStates(hLVTasks);
    switch (LOWORD(wParam)) {
        case IDM_FILE_SAVE:
            _file_save();
            break;
        case IDM_PROGRAM_ABOUT:
            _program_about(hWnd);
            break;
        case IDB_NEW_TASK:
            _new_task(hWnd);
            break;
        case IDB_UPD_TASK:
            _upd_task(hWnd);
            break;
        case IDB_DEL_TASK:
            _del_task(hWnd);
            break;
        case IDB_DEL_DONE:
            _del_done();
            break;
        case IDB_GET_TODAY:
            _get_today(hWnd);
            break;
    }
}

void _WM_HOTKEY_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    UpdateCheckStates(hLVTasks);
    switch (wParam) {
        case HK_CTRL_N:
            _new_task(hWnd);
            break;
        case HK_CTRL_S:
            _file_save();
            break;
    }
}

void _WM_NOTIFY_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    HandleWM_NOTIFY(hLVTasks, lParam);
}

void _WM_CLOSE_MAIN(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    if (changed == TRUE || isUnsavedChecks() == TRUE) {
        switch (MessageBox(hWnd, L"You have unsaved changes. Save before
quit?",
```



```

        L"Warning!", MB_YESNOCANCEL | MB_ICONQUESTION)) {
    case IDYES:
        UpdateCheckStates(hLVTasks);
        _file_save();
    case IDNO:
        DestroyWindow(hWnd);
        break;
    }
}
else {
    DestroyWindow(hWnd);
}
}

void _file_save() {
    repo.savePlanner();
    changed = FALSE;
}

void _program_about(HWND hWnd) {
    MessageBox(hWnd, L"Anton Belousov (c)2021\nDaily Planner v0.0.1", L"About",
    MB_OK | MB_ICONINFORMATION);
}

void _new_task(HWND hWnd) {
    DialogBox(hInst, MAKEINTRESOURCE(IDD_EDIT_TASK), hWnd, EditDlgProc);
}

void _upd_task(HWND hWnd) {
    int i = ListView_GetSelectionMark(hLVTasks);
    if (i == -1) {
        MessageBox(hWnd, L"You haven't selected any task!", L"Error!", MB_OK |
    MB_ICONERROR);
    }
    else {
        chosenTaskIndex = i;
        DialogBox(hInst, MAKEINTRESOURCE(IDD_EDIT_TASK), hWnd, EditDlgProc);
    }
}

void _del_task(HWND hWnd) {
    int i = ListView_GetSelectionMark(hLVTasks);
    if (i == -1) {
        MessageBox(hWnd, L"You haven't selected any task!", L"Error!", MB_OK |
    MB_ICONERROR);
    }
    else {
        repo.remove(i);
        UpdateView(hLVTasks);
        changed = TRUE;
    }
}

void _del_done() {
    if (repo.removeDone() == TRUE) {
        UpdateView(hLVTasks);
        changed = TRUE;
    }
}

void _get_today(HWND hWnd) {
    LPWSTR today = GetCurrentDate();
    wcscat_s(today, 100, L".txt");
    LPWSTR filePath = getSaveFilePath(hWnd, today);
    if (filePath != nullptr) {

```

```

        if (repo.saveToday(filePath, today) != TRUE) {
            MessageBox(hWnd, L"You don't have any tasks for today", L"Info",
MB_OK | MB_ICONASTERISK);
        }
    }
}

BOOL isUnsavedChecks() {
    for (int i = 0; i < repo.size(); i++) {
        PlannerEntry* pe = repo.get(i);
        BOOL currentCheckState = ListView_GetCheckState(hLVTasks, i);
        if (pe->isDone() != currentCheckState) {
            return TRUE;
        }
    }
    return FALSE;
}

```

A6 Пространство имён editdlgproc

A6.1 Текст программного модуля editdlgproc.h

```

#pragma once

#include <windows.h>
#include <windowsx.h>
#include <commctrl.h>

#include "resource.h"

#include "todoview.h"
#include "PlannerRepository.h"
#include "PlannerEntry.h"

extern HWND hLVTasks;

extern PlannerRepository repo;
extern int chosenTaskIndex;

extern BOOL changed;

void _WM_INITDIALOG_DLG(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

void _WM_COMMAND_DLG(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

void fillEditDlg(HWND hWnd);

```

A6.2 Текст программного модуля editdlgproc.cpp

```

#include "editdlgproc.h"

void _WM_INITDIALOG_DLG(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    HWND hTimeFrom = GetDlgItem(hWnd, IDC_TIMEFROM);
    HWND hTimeTo = GetDlgItem(hWnd, IDC_TIMETO);

    DateTime_SetFormat(hTimeFrom, L"HH:mm");
}

```

```

        DateTime_SetFormat(hTimeTo, L"HH:mm");

        if (chosenTaskIndex != -1) {
            fillEditDlg(hWnd);
        }
        else {
            CheckRadioButton(hWnd, IDC_T_EVENT, IDC_T_NOTIFICATION, IDC_T_EVENT);
        }
    }

void fillEditDlg(HWND hWnd) {
    HWND hDateFrom = GetDlgItem(hWnd, IDC_DATEFROM);
    HWND hTimeFrom = GetDlgItem(hWnd, IDC_TIMEFROM);
    HWND hDateTo = GetDlgItem(hWnd, IDC_DATETO);
    HWND hTimeTo = GetDlgItem(hWnd, IDC_TIMETO);

    PlannerEntry* chosenEntry = repo.get(chosenTaskIndex);

    SetDlgItemText(hWnd, IDC_EDIT_TITLE, chosenEntry->getTitle());
    LPSYSTEMTIME from = timestampToSysTime(chosenEntry->getDateFrom(),
chosenEntry->getTimeFrom());
    DateTime_SetSystemtime(hDateFrom, GDT_VALID, from);
    DateTime_SetSystemtime(hTimeFrom, GDT_VALID, from);
    if (!wcscmp(chosenEntry->getType(), PE_EVENT_T)) {
        CheckRadioButton(hWnd, IDC_T_EVENT, IDC_T_NOTIFICATION, IDC_T_EVENT);
        LPSYSTEMTIME to = timestampToSysTime(chosenEntry->getDateTo(),
chosenEntry->getTimeTo());
        DateTime_SetSystemtime(hDateTo, GDT_VALID, to);
        DateTime_SetSystemtime(hTimeTo, GDT_VALID, to);
    }
    else {
        CheckRadioButton(hWnd, IDC_T_EVENT, IDC_T_NOTIFICATION,
IDC_T_NOTIFICATION);
        EnableWindow(hDateTo, FALSE);
        EnableWindow(hTimeTo, FALSE);
    }
}

void _WM_COMMAND_DLG(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    HWND hDateFrom = GetDlgItem(hWnd, IDC_DATEFROM);
    HWND hTimeFrom = GetDlgItem(hWnd, IDC_TIMEFROM);
    HWND hDateTo = GetDlgItem(hWnd, IDC_DATETO);
    HWND hTimeTo = GetDlgItem(hWnd, IDC_TIMETO);
    HWND hTypeEvent = GetDlgItem(hWnd, IDC_T_EVENT);

    LPWSTR title;
    LPWSTR dateFrom;
    LPWSTR timeFrom;
    LPWSTR dateTo;
    LPWSTR timeTo;

    PlannerEntry* chosenEntry;

    SYSTEMTIME lpst = SYSTEMTIME{ 0 };

    switch (LOWORD(wParam)) {
        case IDC_T_EVENT:
            EnableWindow(hDateTo, TRUE);
            EnableWindow(hTimeTo, TRUE);
            break;
        case IDC_T_NOTIFICATION:
            EnableWindow(hDateTo, FALSE);
            EnableWindow(hTimeTo, FALSE);
            break;
        case IDOK:

```

```

        chosenEntry = chosenTaskIndex == -1
            ? new PlannerEntry()
            : repo.get(chosenTaskIndex);

        title = new WCHAR[100];
        GetDlgItemText(hWnd, IDC_EDIT_TITLE, title, 100);
        if (wcscmp(title, L"") == 0) {
            MessageBox(hWnd, L"Task title cannot be empty!", L"Error!",
MB_OK | MB_ICONERROR);
            break;
        }

        dateFrom = new WCHAR[11]{ 0 };
        dateFrom[10] = '\\0';
        DateTime_GetSystemtime(hDateFrom, &lpst);
        SysTimeToDate(&lpst, dateFrom);

        timeFrom = new WCHAR[6]{ 0 };
        timeFrom[5] = '\\0';
        DateTime_GetSystemtime(hTimeFrom, &lpst);
        SysTimeToTime(&lpst, timeFrom);

        if (Button_GetCheck(hTypeEvent) == BST_CHECKED) {
            dateTo = new WCHAR[11]{ 0 };
            dateTo[10] = '\\0';
            DateTime_GetSystemtime(hDateTo, &lpst);
            SysTimeToDate(&lpst, dateTo);

            timeTo = new WCHAR[6];
            timeTo[5] = '\\0';
            DateTime_GetSystemtime(hTimeTo, &lpst);
            SysTimeToTime(&lpst, timeTo);

            if (wcscmp(dateFrom, dateTo) >= 0 && wcscmp(timeFrom, timeTo) >=
0) {
                MessageBox(hWnd, L"Incorrect time interval!", L"Error!",
MB_OK | MB_ICONERROR);
                break;
            }

            delete chosenEntry->getDateTo();
            delete chosenEntry->getTimeTo();

            chosenEntry->setDateTo(dateTo);
            chosenEntry->setTimeTo(timeTo);
            chosenEntry->setType(const_cast<LPWSTR>(PE_EVENT_T));
        }
        else {
            delete chosenEntry->getDateTo();
            delete chosenEntry->getTimeTo();

            chosenEntry->setDateTo(NULL);
            chosenEntry->setTimeTo(NULL);
            chosenEntry->setType(const_cast<LPWSTR>(PE_NOTIFICATION_T));
        }

        delete chosenEntry->getTitle();
        delete chosenEntry->getDateFrom();
        delete chosenEntry->getTimeFrom();

        chosenEntry->setTitle(title);
        chosenEntry->setDateFrom(dateFrom);
        chosenEntry->setTimeFrom(timeFrom);

        if (chosenTaskIndex == -1) {

```

```

        repo.push_front(chosenEntry);
        UpdateView(hLVTasks);
    }

    InvalidateRect(hLVTasks, 0, TRUE);
    changed = TRUE;
case IDCANCEL:
    chosenTaskIndex = -1;
    EndDialog(hWnd, TRUE);
}
}

```

A7 Пространство имён todoview

A7.1 Текст программного модуля todoview.h

```

#pragma once

#include <windows.h>
#include <commctrl.h>

#include "resource.h"

#include "PlannerRepository.h"

#define LV_COLUMN_C 6

extern BOOL changed;

HWND CreateView(HWND hWndParent);

BOOL InitView(HWND hLV);

BOOL InitViewColumns(HWND hLV);

BOOL InitViewItems(HWND hLV, int cItems);

void HandleWM_NOTIFY(HWND hLV, LPARAM lParam);

void UpdateView(HWND hLV);

void UpdateCheckStates(HWND hLV);

```

A7.2 Текст программного модуля todoview.cpp

```

#include "todoview.h"

extern HINSTANCE hInst;
extern PlannerRepository repo;

HWND CreateView(HWND hWndParent) {
    HWND hLV = CreateWindow(
        WC_LISTVIEW,
        L"",
        WS_CHILD | WS_VISIBLE | LVS_REPORT | LVS_EDITLABELS | LVS_NOSORTHEADER,
        10, 10,

```

```

        680, 420,
        hWndParent,
        NULL,
        hInst,
        NULL
    );
    ListView_SetExtendedListViewStyle(hLV, LVS_EX_FULLROWSELECT
        | LVS_EX_HEADERDRAGDROP
        | LVS_EX_LABELTIP
        | LVS_EX_CHECKBOXES
        | LVS_EX_GRIDLINES);
    return hLV;
}

BOOL InitView(HWND hLV) {
    InitViewColumns(hLV);
    InitViewItems(hLV, repo.size());
    return TRUE;
}

BOOL InitViewColumns(HWND hLV) {
    WCHAR szText[256];
    LVCOLUMN lvc;

    lvc.mask = LVCF_FMT | LVCF_MINWIDTH | LVCF_WIDTH | LVCF_TEXT | LVCF_SUBITEM;
    lvc.fmt = LVCFMT_LEFT | LVCFMT_FIXED_WIDTH;

    int* cxs = new int[6]{ 40, 70, 70, 70, 70, 360 };
    int* cxsmín = new int[6]{ 40, 70, 70, 70, 70, 70 };
    for (int iCol = 0; iCol < LV_COLUMN_C; iCol++) {
        lvc.pszText = szText;
        lvc.iSubItem = iCol;
        lvc.cx = cxs[iCol];
        lvc.cxMin = cxsmín[iCol];

        LoadString(hInst,
            IDS_LVC_FIRST + iCol,
            szText,
            sizeof(szText) / sizeof(szText[0]));

        ListView_InsertColumn(hLV, iCol, &lvc);
    }

    delete[] cxs;
    delete[] cxsmín;
    return TRUE;
}

BOOL InitViewItems(HWND hLV, int cItems) {
    LVITEM lvi;

    lvi.mask = LVIF_TEXT | LVIF_STATE;
    lvi.iSubItem = 0;
    lvi.state = 0;
    lvi.stateMask = 0;
    lvi.pszText = LPSTR_TEXTCALLBACK;

    for (int index = 0; index < cItems; index++) {
        lvi.iItem = index;
        ListView_InsertItem(hLV, &lvi);
        ListView_SetCheckState(hLV, index, repo.get(index)->isDone());
    }

    return TRUE;
}

```

```

void HandleWM_NOTIFY(HWND hLV, LPARAM lParam) {
    NMLVDISPINFO* plvdi;
    PlannerEntry* dispEntry;

    switch (((LPNMHDR)lParam)->code) {
    case LVN_GETDISPINFO:
        plvdi = (NMLVDISPINFO*)lParam;
        dispEntry = repo.get(plvdi->item.iItem);
        switch (plvdi->item.iSubItem) {
        case 0:
            plvdi->item.pszText = dispEntry->getType();
            break;
        case 1:
            plvdi->item.pszText = dispEntry->getDateFrom();
            break;
        case 2:
            plvdi->item.pszText = dispEntry->getTimeFrom();
            break;
        case 3:
            plvdi->item.pszText = dispEntry->getDateTo();
            break;
        case 4:
            plvdi->item.pszText = dispEntry->getTimeTo();
            break;
        case 5:
            plvdi->item.pszText = dispEntry->getTitle();
            break;
        }
        break;
    }
}

void UpdateView(HWND hLV) {
    ListView_DeleteAllItems(hLV);
    InitViewItems(hLV, repo.size());
}

void UpdateCheckStates(HWND hLV) {
    for (int i = 0; i < repo.size(); i++) {
        PlannerEntry* pe = repo.get(i);
        BOOL currentCheckState = ListView_GetCheckState(hLV, i);
        if (pe->isDone() != currentCheckState) {
            changed = TRUE;
            pe->setDone(currentCheckState);
        }
    }
}

```

A8 Класс PlannerRepository

A8.1 Текст программного модуля PlannerRepository.h

```

#pragma once

#include <iostream>
#include <fstream>
#include <string>
#include <vector>

```

```

#include "PlannerEntry.h"

#define DATA_FILE_PATH "D:\\OSaSP labs\\daily-planner\\x64\\Debug\\data.txt"

using namespace std;

class PlannerRepository {
    vector<PlannerEntry*> entries;
public:
    PlannerRepository();

    int size();
    void push_front(PlannerEntry* entry);
    PlannerEntry* get(int index);
    void remove(int i);
    BOOL removeDone();

    void savePlanner();
    BOOL saveToday(LPWSTR filePath, LPWSTR date);
};

```

A8.2 Текст программного модуля PlannerRepository.cpp

```

#include "PlannerRepository.h"

PlannerRepository::PlannerRepository() {
    string fileLine;
    ifstream in(DATA_FILE_PATH);
    if (in.is_open()) {
        while (getline(in, fileLine)) {
            entries.push_back(new PlannerEntry(fileLine));
        }
        in.close();
    }
}

int PlannerRepository::size() {
    return entries.size();
}

void PlannerRepository::push_front(PlannerEntry* entry) {
    entries.insert(entries.begin(), entry);
}

PlannerEntry* PlannerRepository::get(int index) {
    if (index < 0 || index >= entries.size()) {
        return nullptr;
    }
    return entries.at(index);
}

void PlannerRepository::remove(int i) {
    PlannerEntry* pe = entries.at(i);
    entries.erase(entries.begin() + i);
    delete pe;
}

BOOL PlannerRepository::removeDone() {
    PlannerEntry* pe;
    BOOL anyDone = FALSE;
    for (int i = 0; i < entries.size(); i++) {
        pe = entries.at(i);
    }
}

```



```

        if (pe->isDone()) {
            anyDone = TRUE;
            remove(i);
        }
    }
    return anyDone;
}

void PlannerRepository::savePlanner() {
    wofstream out;
    out.open(DATA_FILE_PATH);
    if (out.is_open()) {
        for (PlannerEntry* pe : entries) {
            out << pe->toString() << endl;
        }
        out.close();
    }
}

BOOL PlannerRepository::saveToday(LPWSTR filePath, LPWSTR date) {
    wofstream out;
    BOOL isAnyTasks = FALSE;
    for (PlannerEntry* pe : entries) {
        if ((wcscmp(pe->getType(), PE_EVENT_T) == 0
            && wcscmp(pe->getDateFrom(), date) <= 0
            && wcscmp(pe->getDateTo(), date) >= 0)
            || (wcscmp(pe->getType(), PE_NOTIFICATION_T) == 0
            && wcscmp(pe->getDateFrom(), date) == 0)) {
            if (!out.is_open()) {
                out.open(filePath);
            }
            isAnyTasks = TRUE;
            out << pe->toString() << endl;
        }
    }
    if (out.is_open()) {
        out.close();
    }

    return isAnyTasks;
}

```

A9 Класс PlannerEntry

A9.1 Текст программного модуля PlannerEntry.h

```

#pragma once

#include <windows.h>
#include <string>
#include <vector>

#include "util.h"

#define PE_NOTIFICATION_S 5
#define PE_EVENT_S 7

#define PE_NOTIFICATION_T L"N"
#define PE_EVENT_T L"E"

```

```

#define PE_UNDEFINED_T L"U"

#define PE_DELIMITER '|'

using namespace std;

class PlannerEntry {
    BOOL done;
    LPWSTR type;
    LPWSTR dateFrom;
    LPWSTR timeFrom;
    LPWSTR dateTo;
    LPWSTR timeTo;
    LPWSTR title;
public:
    PlannerEntry();
    PlannerEntry(BOOL done, LPCWSTR type, LPWSTR dateFrom, LPWSTR timeFrom,
LPWSTR dateTo, LPWSTR timeTo, LPWSTR title);
    PlannerEntry(string dataEntryString);
    ~PlannerEntry();

    wstring toString();

    BOOL isDone();
    LPWSTR getType();
    LPWSTR getDateFrom();
    LPWSTR getTimeFrom();
    LPWSTR getDateTo();
    LPWSTR getTimeTo();
    LPWSTR getTitle();

    void setDone(BOOL done);
    void setType(LPWSTR type);
    void setDateFrom(LPWSTR dateFrom);
    void setTimeFrom(LPWSTR timeFrom);
    void setDateTo(LPWSTR dateTo);
    void setTimeTo(LPWSTR timeTo);
    void setTitle(LPWSTR title);
};

```

A9.2 Текст программного модуля PlannerEntry.cpp

```

#include "PlannerEntry.h"

PlannerEntry::PlannerEntry() {
    this->done = FALSE;
    this->type = nullptr;
    this->dateFrom = nullptr;
    this->timeFrom = nullptr;
    this->dateTo = nullptr;
    this->timeTo = nullptr;
    this->title = nullptr;
}

PlannerEntry::PlannerEntry(BOOL done, LPCWSTR type, LPWSTR dateFrom, LPWSTR
timeFrom, LPWSTR dateTo, LPWSTR timeTo, LPWSTR title) {
    this->done = done;
    this->type = const_cast<LPWSTR>(type);
    this->dateFrom = dateFrom;
    this->timeFrom = timeFrom;
    this->dateTo = dateTo;
}

```

```

        this->timeTo = timeTo;
        this->title = title;
    }

PlannerEntry::PlannerEntry(string dataEntryString) {
    vector<string> seglist = split(dataEntryString, PE_DELIMITER);

    if (seglist.size() != PE_EVENT_S && seglist.size() != PE_NOTIFICATION_S) {
        done = FALSE;
        type = const_cast<LPWSTR>(PE_UNDEFINED_T);
        dateFrom = NULL;
        timeFrom = NULL;
        dateTo = NULL;
        timeTo = NULL;
        title = NULL;
    }
    else {
        done = stoi(seglist.at(0));
        type = s2ws(seglist.at(1));
        dateFrom = s2ws(seglist.at(2));
        timeFrom = s2ws(seglist.at(3));
        BOOL isEvent = !wcscmp(type, PE_EVENT_T);
        dateTo = isEvent ? s2ws(seglist.at(4)) : NULL;
        timeTo = isEvent ? s2ws(seglist.at(5)) : NULL;
        title = isEvent ? s2ws(seglist.at(6)) : s2ws(seglist.at(4));
    }
}

PlannerEntry::~PlannerEntry() {
    delete type;
    delete dateFrom;
    delete timeFrom;
    delete dateTo;
    delete timeTo;
    delete title;
}

wstring PlannerEntry::toString() {
    LPWSTR doneBuf = new WCHAR[2];
    _itow_s(done, doneBuf, 2, 10);

    LPWSTR result = new WCHAR[200];
    if (wcscmp(this->type, PE_EVENT_T) == 0) {
        swprintf_s(result, 200, L"%s|%s|%s|%s|%s|%s", doneBuf, type,
dateFrom, timeFrom, dateTo, timeTo, title);
    }
    else {
        swprintf_s(result, 200, L"%s|%s|%s|%s", doneBuf, type, dateFrom,
timeFrom, title);
    }

    delete[] doneBuf;
    return wstring(result);
}

BOOL PlannerEntry::isDone() {
    return done;
}

LPWSTR PlannerEntry::getType() {
    return type;
}

LPWSTR PlannerEntry::getDateFrom() {
    return dateFrom;
}

```

```

}

LPWSTR PlannerEntry::getTimeFrom() {
    return timeFrom;
}

LPWSTR PlannerEntry::getDateTo() {
    return dateTo;
}

LPWSTR PlannerEntry::getTimeTo() {
    return timeTo;
}

LPWSTR PlannerEntry::getTitle() {
    return title;
}

void PlannerEntry::setDone(BOOL done) {
    this->done = done;
}

void PlannerEntry::setType(LPWSTR type) {
    this->type = type;
}

void PlannerEntry::setDateFrom(LPWSTR dateFrom) {
    this->dateFrom = dateFrom;
}

void PlannerEntry::setTimeFrom(LPWSTR timeFrom) {
    this->timeFrom = timeFrom;
}

void PlannerEntry::setDateTo(LPWSTR dateTo) {
    this->dateTo = dateTo;
}

void PlannerEntry::setTimeTo(LPWSTR timeTo) {
    this->timeTo = timeTo;
}

void PlannerEntry::setTitle(LPWSTR title) {
    this->title = title;
}

```

Обозначение				Наименование		Дополнительные сведения			
				<u>Текстовые документы</u>					
БГУИР КП 1-40 01 01 003 ПЗ				Пояснительная записка		53 с.			
				<u>Графические документы</u>					
ГУИР 951006 003 ПД				Схема программы		Формат А1			