**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ   
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Руководитель проекта, старший преподаватель факультета компьютерных наук департамента «Программная инженерия»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М. Д. Шадрин  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г. | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2021 г. | |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. Инв. № |  | | Подп. и дата |  | | Инв. № подл. | **RU.17701729.06.05-01 ПЗ 01-1** | | **Клиентская часть мобильного приложения для планирования дедлайнов  Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.06.05-01 ПЗ 01-1-ЛУ**  **Исполнитель**  Студент группы БПИ 195  \_\_\_\_\_\_\_\_\_\_\_/Н. Д. Зубарева/  «\_\_\_»\_\_\_\_\_\_\_\_\_\_2021 г.    **Москва 2021** | | |  | |

**УТВЕРЖДЕНО**  
**RU.17701729.06.05-01 ПЗ 01-1-ЛУ**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. Инв. № |  | | Подп. и дата |  | | Инв. № подл. | **RU.17701729.06.05-01 ПЗ 01-1**  \_\_\_\_\_\_\_\_ | | **Клиентская часть мобильного приложения для планирования дедлайнов Пояснительная записка**  **RU.17701729.06.05-01 ПЗ 01-1**  **Листов 44**  Москва 2021  СОДЕРЖАНИЕ  [1. ВВЕДЕНИЕ 3](#_Toc72158486)  [1.1 Наименование программы 3](#_Toc72158487)  [1.2 Документы, на основании которых ведется разработка 3](#_Toc72158488)  [2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 4](#_Toc72158489)  [2.1 Назначение программы 4](#_Toc72158490)  [2.1.1 Функциональное назначение 4](#_Toc72158491)  [4.1.1 Эксплуатационное назначение 4](#_Toc72158492)  [5. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 5](#_Toc72158493)  [5.1 Постановка задачи на разработку программы 5](#_Toc72158494)  [5.2 Описание алгоритма и функционирования программы 5](#_Toc72158495)  [5.2.1 Описание алгоритма программы 5](#_Toc72158496)  [5.2.2 Обоснование выбора алгоритма решения задачи 8](#_Toc72158497)  [5.3 Описание и обоснование выбора метода организации входных и выходных данных 9](#_Toc72158498)  [5.3.1 Описание метода организации входных и выходных данных 9](#_Toc72158499)  [5.3.2 Обоснование выбора метода организации входных и выходных данных 9](#_Toc72158500)  [5.4 Описание и обоснование выбора организации хранения данных 9](#_Toc72158501)  [5.5 Описание и обоснование выбора состава технических и программных средств 9](#_Toc72158502)  [5.5.1 Требования к составу и параметрам технических средств 9](#_Toc72158503)  [5.5.2 Требования к информативной и программной совместимости 10](#_Toc72158504)  [6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 11](#_Toc72158505)  [6.1 Предполагаемая потребность 11](#_Toc72158506)  [6.2 Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 11](#_Toc72158507)  [6.2.1 Отечественные аналоги 11](#_Toc72158508)  [6.2.2 Зарубежные аналоги 11](#_Toc72158509)  [6.2.3 Преимущества данной программы 11](#_Toc72158510)  [СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 12](#_Toc72158511)  [ПРИЛОЖЕНИЕ 1 13](#_Toc72158512)  [ПРИЛОЖЕНИЕ 2 14](#_Toc72158513)  [ПРИЛОЖЕНИЕ 3 17](#_Toc72158514)  [ПРИЛОЖЕНИЕ 5 43](#_Toc72158515) |  |

1. **ВВЕДЕНИЕ**
   1. **Наименование программы**

**Наименование программы на русском языке:** «Клиент-серверное мобильное приложение для планирования дедлайнов»

**Наименование программы на английском языке:** «Client-Server Mobile Application For Deadline Planning»

**Условное обозначение:** планер

* 1. **Документы, на основании которых ведется разработка**

Программа выполняется в рамках темы курсового проекта в соответствии с учебным планом подготовки бакалавров по направлению 09.03.04 «Программная инженерия» Национального исследовательского университета «Высшая школа экономики», факультет компьютерных наук, департамент программной инженерии.

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 "Программная инженерия" и утвержденная академическим руководителем тема курсового проекта.

1. **НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ**
   1. **Назначение программы**
      1. Функциональное назначение

Приложение предназначено для организации планирования, а именно для составления, редактирования, отображения планов и ассоциированной с ними информации.

Функции, помогающие это реализовать:

1. создание, редактирование и удаление планов, подзадач, категорий,
2. отображение планов в списке с возможностью сортировки по срочности или важности
3. отображение планов на календаре
4. отображение планов в матрице Эйзенхауэра
   * 1. Эксплуатационное назначение

Приложение может быть использовано любым человеком, который хочет заниматься планированием своей деятельности с помощью смартфона на платформе Android. Наша программа поможет это сделать.

1. **ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ**
   1. **Постановка задачи на разработку программы**

При постановке задачи был проведен опрос потенциальных пользователей и на основании результатов были выделены основные функции, которые нужны в приложении такого рода. Таким образом, задача заключалась в написании программы, которая должна позволять пользователю:

1. Выполнять следующие действия с категориями дедлайнов:
   1. Создать категорию дедлайнов;
   2. Редактировать категорию дедлайнов;
   3. Удалить категорию дедлайнов;
   4. Показать все дедлайны категории;
2. Выполнять следующие действия с подзадачами:
   1. Создать новую подзадачу;
   2. Редактировать подзадачу;
   3. Удалить подзадачу;
3. Выполнять следующие действия с дедлайнами:
   1. Создать новый дедлайн;
   2. Редактировать дедлайн;
   3. Удалить дедлайн;
4. Отображать дедлайны пользователя в виде:
   1. списка;
   2. календаря на месяц;
5. Сортировать дедлайны
   1. по сроку выполнения;
   2. по важности;
6. Настраивать отображение выполненных дедлайнов:
   1. скрывать выполненные дедлайны;
   2. показывать выполненные дедлайны;
7. Визуализировать дедлайны через матрицу Эйзенхауэра
   1. **Описание алгоритма и функционирования программы**
      1. Описание алгоритма программы
         1. Описание общей схемы работы системы

Работа программы разделена на несколько логических блоков:

1. получение данных пользователя
2. создание планов и связанной с ними информации
3. изменение планов и связанной с ними информации
4. отображение планов и связанной с ними информации

Каждая часть работы происходит посредством интерфейса и взаимодействия пользователя с ним. В целом работа приложения связана с жизненным циклом планирования: план создается, редактируется, наблюдается, возможно, переносится или повторяется, отмечается как выполненный.

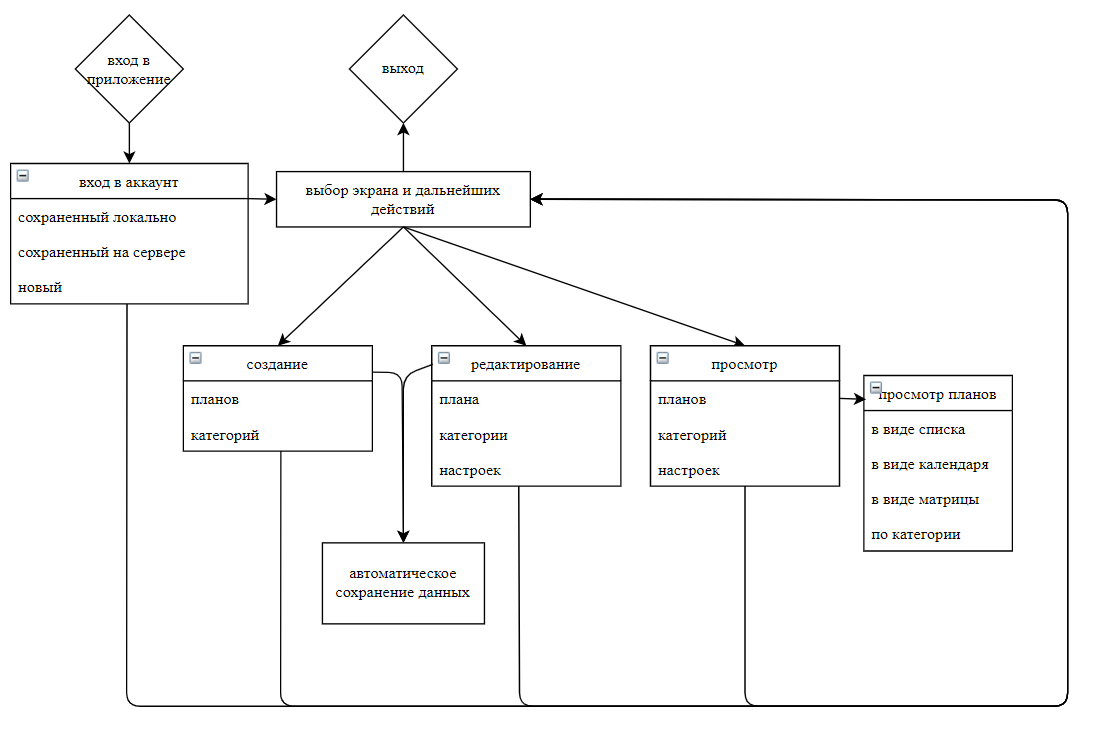


Рисунок 1 – Общая блок-схема программы.

* + - 1. Описание получения данных пользователя

Получение данных происходит либо при использовании сервера и сохраненных там данных, либо из сохраненных на устройстве файлов SharedPreferences [15], подробную информацию о них можно найти в пункте 5.3.1 данного документа.

* + - 1. Описание создания и изменения планов и ассоциированной информации

Новый план пользователь может создать на экранах списка и календаря. При этом открывается экран редактирования плана, и пользователь задает информацию о плане, такую как: его название, тематическую категорию, к которой он относится, заметки, дату выполнения, крайний срок выполнения, важность, подзадачи, опции переноса и повторения. Далее при нажатии на кнопку сохранения информация плана обновляется и сохраняется.

* + - 1. Описание переноса плана

Опция переноса плана – одна из полезных функций при работе с задачами, которую можно реализовать разными способами. В случае этого приложения перенос означает то, что начало работы с планом (дата его выполнения) откладывается, то есть к ней прибавляется то число дней, которое указано в опции переноса (день, неделя, месяц или год). Это происходит, если план не выполнен, а дата его выполнения прошла. При этом крайний срок выполнения не переносится.

Это можно проиллюстрировать ситуацией, когда, например, крайний срок выполнения плана назначен на воскресенье текущей недели, а дата выполнения – на понедельник, у пользователя есть неделя на выполнение плана, но пользователь не приступает к работе до тех пор, пока не наступит суббота – в таком случае, если у плана настроен перенос на следующий день, дата его выполнения будет перемещаться по всем дням недели, пока, наконец, в субботу пользователь не выполнит план, и тогда его перенос прекратится.

В приложении попытка переноса всех планов осуществляется при входе в приложение до их отображения, таким образом информация о планах отображается актуально.

* + - 1. Описание повторения плана

Повторение плана может казаться похожим на перенос, но представляет собой другую, хотя и не менее полезную функцию. Оно нужно для повторяющихся событий, у которых каждый раз обновляется и дата выполнения, и крайний срок. Таким образом, если у плана настроено повторение, то, по выполнении этого плана, и к его дате выполнения, и к крайнему сроку будет прибавлено то число дней, через которое план должен быть повторен.

Это можно проиллюстрировать как еженедельное задание, которое открывается в понедельник и закрывается во вторник – как только пользователь выполняет текущий план, появляется такой же на следующей неделе.

* + - 1. Описание различных видов отображения планов

Для удобства взаимодействия с планами в приложении они отображаются в разных форматах, что помогает пользователю лучше понимать свое расписание в зависимости от интересующих характеристик.

* + - * 1. Описание отображения в списке и сортировок

Отображение на экране списка реализуется с двумя списками планов и подробной информацией о каждом плане. Два списка плана соотносятся с еще не выполненными и уже выполненными задачами. Это нужно для того, чтобы выполненные задачи не отвлекали пользователя от более актуальных. Деление по этим двум спискам, как и ожидается, происходит по значению статуса выполнения плана.

Для того, чтобы отображение в каждом из списков было более информативным, существует два способа сортировки планов в списке: по важности плана и по срочности плана.

В первом случае планы сортируются по убыванию целочисленного параметра важности, так что самые важные планы оказываются сверху, это помогает пользователю обозначить приоритетные задачи.

Во втором случае планы упорядочиваются по увеличению даты крайнего срока выполнения. Это выводит в верхнюю часть списка планы, у которых остается меньше всего времени для выполнения.

При равенстве сравниваемых параметров планы сортируются по времени создания.

* + - * 1. Описание отображения в календарном виде

Часто бывает удобно просматривать планы не все сразу, а на конкретный день, в зависимости от даты выполнения. Это реализуется с помощью экрана календаря, где, помимо списка планов есть окно календаря на месяц. Планы для списка фильтруются в соответствии с выбранной датой и отображаются, если дата их выполнения совпадает с выбранной. Также на этом экране при добавлении плана его дата автоматически выставляется выбранной в календаре.

* + - * 1. Описание отображения в виде матрицы Эйзенхауэра

Метод деления планов на категории в соответствии с сочетанием важности и срочности описан в книге «Семь навыков высокоэффективных людей» [1] и в целом является известным и распространенным способом выбора приоритетов и стратегий выполнения дел при планировании. В случае этого приложения планы делятся на четыре стандартные категории: срочные и важные, срочные и неважные, несрочные и важные, несрочные и неважные, причем все эти списки включают в себя только невыполненные планы, поскольку их не имеет смысла учитывать при таком разбиении.

Важность определяется в соответствии с численной оценкой важности плана – так как она лежит в пределах от 0 до 5, планы с важностью от 0 до 2 включительно считаются неважными, а планы с важностью от 3 до 5 включительно – важными.

Срочность определяется, как и на экране списка, в соответствии с крайним сроком выполнения. План считается срочным, если на его выполнение осталось не больше двух дней.

Стоит также отметить, что во избежание информационной перегрузки матрицы планы в ней отображаются «в облегченном виде», то есть их верстка включает в себя название, даты выполнения и крайнего срока и важность, все остальные параметры можно увидеть и отредактировать при нажатии, но в списках показана только наиболее важная в этом контексте информация.

* + - * 1. Описание отображения в виде планов в одной категории

В настройках категории есть возможность просмотреть все планы, относящиеся к выбранной категории. Это еще один способ отображения, он помогает оценить все планы в какой-либо тематической группе. Соответственно в такой список попадают планы, категория которых соответствует выбранной категории.

* + 1. Обоснование выбора алгоритма решения задачи

Одной из основных целей данного приложения было наглядное отображение информации, которое помогло бы пользователям планировать свою деятельность более эффективно. По этой причине были реализованы разные подходы, которые считаются полезными при целеполагании (многие из них описаны в [1] и [2]).

Так, при отображении, создании и редактировании плана важным является наличие подзадач, так как дробление плана на более простые и понятные шаги считается одним из важных условий создания выполнимого плана, как и временное ограничение, которое реализовано двумя датами – временем выполнения и крайним сроком, что создает продолжительный период и помогает с планированием. Также заметки о плане предоставляют возможность обозначать ресурсы для плана, подробности о задаче.

Выбор методов переноса и повторения плана обоснован персональным опытом разработчиков и более востребованными методами по результатам опроса потенциальных пользователей – нужными функциями считается возможность отложить выполнение плана и возможность повторять что-то в формате, близком к расписанию.

При выборе способов визуализации использовались наиболее распространенные методологии визуализации и структурирования планов, описываемые в [1] и [2], как раздельное отображение выполненных и текущих планов, концентрация на более срочных планах, более важных планах, сочетанию срочности и важности, датам выполнения как спискам на конкретный день, по тематической категории. В приложении совершена попытка сортировать и фильтровать планы по максимальному числу осмысленных признаков, чтобы у пользователя с почти любым предпочитаемым способом систематизации планов была возможность найти подходящее отображение планов при использовании приложения.

* 1. **Описание и обоснование выбора метода организации входных и выходных данных**
     1. Описание метода организации входных и выходных данных

Входные данные представляют собой JSON файл с сохраненными данными пользователя при наличии предыдущего сохранения, остальные же данные задаются через интерфейс приложения при работе с ним. Ввод посредством интерфейса осуществляется с помощью текстовых полей, выпадающих списков с вариантами, доступными для выбора, кнопок, полей выбора даты и времени, соответствующих параметрам и проверяющих корректность вводимых данных.

Выходные данные представляют собой отображение внесенной информации в интерфейсе, так как приложение предназначено для взаимодействия с данными, а не для экспортирования их куда-либо. Также происходит сохранение JSON файлов с данными пользователя.

* + 1. Обоснование выбора метода организации входных и выходных данных

Описанный выше формат входных данных был выбран в соответствии с требованиями, предъявленными к входным данным в техническом задании, также формат JSON файлов читаем и понятен при использовании, разнообразно экспортируется в том числе при сохранении в базы данных. Ввод через интерфейс сделан таким образом, чтобы пользователю было наиболее привычно с ним взаимодействовать.

Выходные данные в части вывода в интерфейс организованы также с целью сделать отображение информации в приложении максимально информативным, при этом не перегружая интерфейс.

* 1. **Описание и обоснование выбора организации хранения данных**

Файлы хранятся в памяти, ассоциированной с приложением на устройстве, в виде JSON файлов в SharedPreferences [15]. Это удобный способ иметь к ним доступ только изнутри приложения, также он не требует много места, легок в работе. Так как данные пользователя по объему вполне подходят для хранения таким образом, этот способ более предпочтителен, чем локальные базы данных, как для временного кеширования, так и для постоянного локального хранения.

* 1. **Описание и обоснование выбора состава технических и программных средств**

Программные и технические средства выбраны на основании минимальных требований, необходимых для работы программы, предназначенной для смартфонов на платформе Android. Платформа Android выбрана в силу ее распространенности и удобства для разработки. В силу того, что приложение предусматривает локальное сохранение данных, необходимо достаточное количество памяти на устройстве.

* + 1. Требования к составу и параметрам технических средств

Для корректной работы программы необходимо выполнение следующих требований к техническим средствам:

1. Смартфон на базе платформы Android;
2. Экран с разрешением 720х1280, 1080х1920 или 2560×1440 пикселей;
3. Не менее 256 Мб оперативной памяти;
4. Не менее 64 Мб встроенной памяти или памяти на SD-карте.
   * 1. Требования к информативной и программной совместимости

Для корректной работы программы необходимо выполнение следующих требований к программным средствам:

1. Лицензионная операционная система Android 8.1 или выше;
2. Доступ к сети интернет для установки.
3. **ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ**
   1. **Предполагаемая потребность**

Программа ориентирована на эксплуатацию отдельными пользователями, имеющими мобильное устройство и желание заняться систематизацией своих планов.

* 1. **Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами**

При проведении анкетирования потенциальных пользователей были выделены следующие наиболее популярные аналоги нашего программного продукта:

* + 1. Отечественные аналоги

1. Мобильное приложение “Todobit” [18]
2. Мобильное приложение “HSE App” [11]
3. Набор скриптов на Си “дедлайнобот” (github разработчика [6])
   * 1. Зарубежные аналоги
4. Мобильное приложение “Google-календарь” [10]
5. Мобильное приложение “Notion” [13]
6. Мобильное приложение “TickTick” [17]
   * 1. Преимущества данной программы

В сравнении с аналогами приложение обладает следующими преимуществами:

1. минималистичный пользовательский интерфейс;
2. использование технологий визуализации планирования;
3. широкий набор функций;
4. бесплатность

**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Кови, Р.С., 1989. Семь навыков высокоэффективных людей. – М.: [Альпина Паблишер](https://www.ozon.ru/publisher/alpina-pablisher-859012/), 2019.
2. Цели по SMART: подробный обзор [Электронный ресурс] //URL: <http://powerbranding.ru/marketing-strategy/smart-celi/> (Дата обращения: 01.04.2021, режим доступа: свободный)
3. Activity [Электронный ресурс] //URL: https://developer.android.com/reference/kotlin/android/app/Activity (Дата обращения: 01.04.2021, режим доступа: свободный)
4. Android Bottom Navigation Example in Kotlin [Электронный ресурс] //URL: https://blog.mindorks.com/android-bottom-navigationview-in-kotlin (Дата обращения: 01.04.2021, режим доступа: свободный)
5. Babich, Nick. A Comprehensive Guide To Mobile App Design [Электронный ресурс] // URL: <https://www.smashingmagazine.com/2018/02/comprehensive-guide-to-mobile-app-design/> (Дата обращения: 01.04.2021, режим доступа: свободный)
6. Cortan122 [Электронный ресурс] //URL: <https://github.com/Cortan122> (Дата обращения: 01.04.2021, режим доступа: свободный)
7. DatePicker in Kotlin [Электронный ресурс] //URL: https://www.geeksforgeeks.org/datepicker-in-kotlin/ (Дата обращения: 01.04.2021, режим доступа: свободный)
8. Documentation for app developers [Электронный ресурс] // URL: <https://developer.android.com/docs?hl=ru/> (Дата обращения: 01.04.2021, режим доступа: свободный)
9. Fragments [Электронный ресурс] //URL: <https://developer.android.com/guide/fragments> (Дата обращения: 01.04.2021, режим доступа: свободный)
10. Google-календарь [Электронный ресурс] //URL: <https://play.google.com/store/apps/details?id=com.google.android.calendar> (Дата обращения: 01.04.2021, режим доступа: свободный)
11. HSE App [Электронный ресурс] //URL: <https://play.google.com/store/apps/details?id=com.chichkanov.hseapp> (Дата обращения: 01.04.2021, режим доступа: свободный)
12. LMS [Электронный ресурс] //URL: https://lms.hse.ru (Дата обращения: 01.04.2021, режим доступа: свободный)
13. Notion [Электронный ресурс] //URL: <https://apps.apple.com/ru/app/notion-notes-projects-docs/id1232780281> (Дата обращения: 01.04.2021, режим доступа: свободный)
14. RecyclerView [Электронный ресурс] //URL: https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView (Дата обращения: 01.04.2021, режим доступа: свободный)
15. Save key-value data [Электронный ресурс] // URL: https://developer.android.com/training/data-storage/shared-preferences (Дата обращения: 01.04.2021, режим доступа: свободный)
16. Spinners [Электронный ресурс] // URL: https://developer.android.com/guide/topics/ui/controls/spinner (Дата обращения: 01.04.2021, режим доступа: свободный)
17. TickTick [Электронный ресурс] //URL: <https://play.google.com/store/apps/details?id=com.ticktick.task> (Дата обращения: 01.04.2021, режим доступа: свободный)
18. Todobit [Электронный ресурс] //URL: <https://play.google.com/store/apps/details?id=org.todobit.android> (Дата обращения: 01.04.2021, режим доступа: свободный)
19. UML class diagrams [Электронный ресурс] //URL: <https://www.jetbrains.com/help/idea/class-diagram.html> (Дата обращения: 01.04.2021, режим доступа: свободный)

ПРИЛОЖЕНИЕ 1

**ТЕРМИНОЛОГИЯ**

Таблица 1 – Терминология

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| **Дедлайн, план** | Это некоторая задача, которую пользователь хочет выполнить, в приложении она описывается экземпляром класса Plan. |
| **Подзадача, подплан** | Это подзадача, этап выполнения какого-либо плана, в приложении описывается классом Subplan. |
| **Категория** | Это тематическая категория для задач, в приложении она описывается экземпляром класса Category. |
| **Матрица Эйзенхауэра** | Это стратегия систематизации задач (описана в [1]), разделяющая их на 4 категории в соответствии с их важностью и срочностью. В случае нашего приложения это матрица с двумя столбцами и двумя колонками, в которой в соответствии с разделением на категории записаны планы пользователя. |

ПРИЛОЖЕНИЕ 2

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ**

**Классы пакета-библиотеки possumLib приложения PossumPlanner**

*Таблица 2 – Классы пакета-библиотеки possumLib приложения PossumPlanner*

|  |  |
| --- | --- |
| **Класс** | **Назначение** |
| User | Класс для хранения информации о пользователе, списке его планов, категорий. |
| Plan | Класс для плана с его параметрами: статусом выполнения, категорией, подпланами, сроками выполнения, заметками и т.д. |
| Subplan | Класс для подзадачи, содержит ее название и статус выполнения. |
| Category | Класс для тематической категории плана, содержит название и цвет. |
| possibleRepetitions | Класс перечисления для возможных значений повторения плана. |
| possiblePutOffs | Класс перечисления для возможных значений переноса плана. |
| possibleReminders | Класс перечисления для возможных значений напоминаний о плане. |

**Классы пакета ui приложения PossumPlanner**

*Таблица 3 - Классы пакета ui приложения PossumPlanner*

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| MainActivity | Класс activity для экрана входа в приложение с помощью учетной записи или локально сохраненных данных. |
| BottomNavigationScreen | Класс activity для настройки нижней панели навигации по приложению для перемещения между фрагментами отображений в виде списка, календаря, матрицы Эйзенхауэра и экраном настроек. |

**Классы пакета ui.deadlineEdit приложения PossumPlanner**

*Таблица 4 - Классы пакета ui.deadlineEdit приложения PossumPlanner*

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| DeadlineEditActivity | Класс activity для экрана с подробным отображением плана для его редактирования и задания его параметров. |
| DeadlineEditAdapter | Класс адаптер для списка подпланов, находящегося на экране плана, настраивает взаимодействие с элементами списка. |
| DeadlineEditViewHolder | Класс для настройки вида в адаптере, задает дополнительные связи в отображении элемента списка. |

**Классы пакета ui.categoryEdit приложения PossumPlanner**

*Таблица 5 - Классы пакета ui.categoryEdit приложения PossumPlanner*

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| CategoryEditActivity | Класс activity для экрана с подробным отображением категории для ее редактирования и просмотра планов, относящихся к этой категории. |
| CategoryEditAdapter | Класс адаптер для списка планов, находящегося на экране категории, настраивает взаимодействие с элементами списка, позволяет открывать экран редактирования плана при нажатии на план. |
| CategoryEditViewHolder | Класс для настройки вида в адаптере, задает дополнительные связи в отображении элемента списка. |

**Классы пакета ui.list приложения PossumPlanner**

*Таблица 6 - Классы пакета ui.list приложения PossumPlanner*

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| ListFragment | Класс фрагмент для отображения планов в виде списка, показывает списки выполненных и невыполненных планов, позволяет сортировать их и добавлять новые планы. |
| NestedListAdapter | Класс адаптер для сворачиваемых списков – его элементами являются список выполненных дел с заголовком и список невыполненных дел с заголовком, настраивает логику взаимодействия с ними, позволяет сворачивать и разворачивать оба списка. |
| ViewHolder | Класс холдер для связи объекта сворачиваемого списка и лейаута элемента списка сворачиваемых списков. |
| ListForRV | Вспомогательный класс для элемента сворачиваемого списка, содержит название, характеристику свернутости и название способа сортировки. |
| DeadlinesListAdapter | Класс адаптер для списка планов, являющегося частью элемента сворачиваемого списка. Позволяет открывать экран редактирования плана при нажатии на него. |
| ListViewHolder | Класс холдер для связи объекта списка планов и лейаута лдя элемента списка планов, настраивает поля с информацией в соответствии с режимом отображения и информацией о плане. |

**Классы пакета ui.calendar приложения PossumPlanner**

*Таблица 7 - Классы пакета ui.calendar приложения PossumPlanner*

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| CalendarFragment | Класс фрагмент для отображения вида календаря, содержит окошко календаря на месяц, в котором можно перелистывать месяца, нажимать на конкретные даты, и список планов, соответствующих выбранной дате. |
| CalendarAdapter | Класс адаптер для списка планов, находящегося на экране календаря, настраивает взаимодействие с элементами списка, позволяет открывать экран редактирования плана при нажатии на план. |
| CalendarViewHolder | Класс холдер для связи отображаемого элемента – плана и полей лейаута, соответствующего элементу списка. |

**Классы пакета ui.graph приложения PossumPlanner**

*Таблица 8 - Классы пакета ui.graph приложения PossumPlanner*

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| GraphFragment | Класс фрагмент, содержащий четыре списка планов, разбитых по срочности и важности в матрицу Эйзенхауэра, планы отображаются в менее подробном виде и открывают экран редактирования при нажатии. |

**Классы пакета ui.settings приложения PossumPlanner**

*Таблица 9 - Классы пакета ui.settings приложения PossumPlanner*

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| SettingsFragment | Класс фрагмент для экрана настроек, позволяет переходить с него к настройкам различных функций. |
| SettingsAdapter | Класс адаптер для списка настраиваемых функций, находящегося на экране настроек, позволяет открывать экран редактирования того или иного блока при нажатии на надпись. |
| UserSettingsActivity | Экран для настройки параметров учетной записи пользователя. |

**Классы пакета ui.settings.categories приложения PossumPlanner**

*Таблица 10 - Классы пакета ui.settings.categories приложения PossumPlanner*

|  |  |
| --- | --- |
| **Название** | **Назначение** |
| SettingsCategoriesScreen | Класс для экрана настроек категорий, содержащий список категорий пользователя. |
| SettingsCategoriesAdapter | Класс адаптер для списка категорий пользователя, находящегося на экране настроек категорий, позволяет открывать экран редактирования категории при нажатии на ее название. |
| SettingsCategorуViewHolder | Класс холдер, нужен для связи объекта категории и отображаемого в списке элемента, соответствующего ее названию. |

ПРИЛОЖЕНИЕ 3

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ, ПОЛЕЙ И СВОЙСТВ**

**Описание членов класса User**

*Таблица 11 – Описание членов класса User*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public | void | String, String, ArrayList<Plan>, ArrayList<Category> | | Конструктор с параметрами для пользователя, задает значения полей. |
| toString | public | String | - | | Метод, формирующий на основании объекта строку с основной информацией о нем. |
| addCategory | public | void | Category | | Метод добавления категории, добавляет категорию к списку категорий, также добавляет название категории в список их названий. |
| deleteCategory | public | void | Category | | Метод удаления категории, удаляет категорию из списка категорий, также удаляет ее название из списка их названий. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| name | public | String | | Поле, свойство с автоматическими геттером и сеттером для имени пользователя, в сеттере проверяется то, что длина нового имени лежит в пределах от 1 до 9. Дефолтное значение «new user». | |
| token | public | String | | Поле, свойство с автоматическими геттером и сеттером для токена авторизации, в сеттере проверяется то, что длина нового токена лежит в пределах от 1 до 9. Дефолтное значение «token». | |
| plans | public | ArrayList<Plan> | | Поле, свойство с автоматическими геттером и сеттером для списка планов пользователя. Дефолтное значение - пустой ArrayList. | |
| categories | public | ArrayList<Category> | | Поле, свойство с автоматическими геттером и сеттером для списка категорий пользователя. В сеттере также настроено добавление названий всех новых категорий в список названий. Дефолтное значение - пустой ArrayList. | |
| categories\_names | public | ArrayList<String> | | Поле, свойство с автоматическими геттером и сеттером для списка названий категорий пользователя. Дефолтное значение - пустой ArrayList. | |

**Описание членов класса Plan**

*Таблица 12 – Описание членов класса Plan*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public |  | String, Boolean, Category, String, Int, LocalDateTime, LocalDateTime, ArrayList<Subplan>, possibleRepetitions, possiblePutOffs, possibleReminders, Double | | Конструктор с параметрами для плана, задает значения полей. |
| isUrgent | public | Boolean | - | | Метод, выдающий значение срочности или несрочности плана. Критерий срочности – то, что до крайнего срока выполнения остается не более двух дней с текущего момента времени. |
| toString | public | String | - | | Метод, формирующий на основании объекта строку с основной информацией о нем |
| tryRepeat | public | void | - | | Метод, пытающийся повторить план. Если у плана есть значение повторения, он повторяется в зависимости от того, какое стоит значение повторения. В соответствии с ним к дате выполнения и крайнему сроку выполнения прибавляется либо один день, либо одна неделя, либо один месяц, либо один год, и статус выполнения меняется на невыполненный. Этот метод вызывается при отметке плана как выполненного.. |
| tryPutOff | public | void | - | | Метод, пытающийся перенести план. Если план истек и у него есть значение переноса, он переносится в зависимости от того, какое стоит значение переноса. В соответствии с ним к дате выполнения прибавляется либо один день, либо одна неделя, либо один месяц, либо один год. |
| isOverdue | public | Boolean | - | | Метод, выдающий значение того, истек ли срок выполнения плана. Критерий истечения срока – план не отмечен как выполненный и текущий момент времени после даты выполнения плана. |
| setPutOff | public | void | String | | Метод, пытающийся задать значение переноса плана в соответствии с передаваемым строковым значением. Если значение не соответствует переносу на следующий день, неделю, месяц или год, задается значение отсутствия переноса. |
| setRepetition | public | void | String | | Метод, пытающийся задать значение повторения плана в соответствии с передаваемым строковым значением. Если значение не соответствует повторению каждый день, неделю, месяц или год, задается значение отсутствия повторения. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| title | public | String | | Поле, свойство с автоматическими геттером и сеттером для названия плана, в сеттере проверяется то, что длина нового названия лежит в пределах от 1 до 19. Дефолтное значение «новый план». | |
| isFinished | public | Boolean | | Поле, свойство с автоматическими геттером и сеттером для статуса завершенности плана. В сеттере проверяется, что значение меняется на противоположное. Дефолтное значение – false. | |
| category | public | Category | | Поле, свойство с автоматическими геттером и сеттером для тематической категории плана. Дефолтное значение – пустая категория. | |
| importance | public | Int | | Поле, свойство с автоматическими геттером и сеттером для целочисленной оценки важности плана. В сеттере есть проверка на то, что новое значение лежит в пределах от 0 до 5. Дефолтное значение – 1. | |
| repetition | public | possibleRepetitions | | Поле, свойство с автоматическими геттером и сеттером для повторения плана. Дефолтное значение – possibleRepetitions.NONE. | |
| subplans | public | ArrayList<Subplan> | | Поле, свойство с автоматическими геттером и сеттером для списка подпланов пользователя. Дефолтное значение - пустой ArrayList. | |
| putOff | public | possiblePutOffs | | Поле, свойство с автоматическими геттером и сеттером для переноса плана. Дефолтное значение – possiblePutOffs.NONE. | |
| reminder | public | possibleReminders | | Поле, свойство с автоматическими геттером и сеттером для напоминаний о плане. Дефолтное значение – possibleReminders.NONE. | |
| time | public | Double | | Поле, свойство с автоматическими геттером и сеттером для времени, нужного для выполнения плана. Дефолтное значение – 0.0. | |
| date | public | LocalDateTime | | Поле, свойство с автоматическими геттером и сеттером для даты выполнения плана. Дефолтное значение – LocalDateTime.now. | |
| deadline | public | LocalDateTime | | Поле, свойство с автоматическими геттером и сеттером для крайнего срока выполнения плана. Дефолтное значение – LocalDateTime.now. | |
| notes | public | String | | Поле, свойство с автоматическими геттером и сеттером для заметок о плане. Дефолтное значение – «». | |

**Описание членов класса possibleRepetitions**

*Таблица 13 – Описание членов класса possibleRepetitions*

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Назначение* |
| NONE | public | possibleRepetitions | Значение повторения, задающее отсутствие повторения плана. |
| DAILY | public | possibleRepetitions | Значение повторения, задающее повторение плана каждый день. |
| WEEKLY | public | possibleRepetitions | Значение повторения, задающее повторение каждую неделю. |
| MONTHLY | public | possibleRepetitions | Значение повторения, задающее повторение каждый месяц. |
| EACHYEAR | public | possibleRepetitions | Значение повторения, задающее повторение на следующий год. |

**Описание членов класса possiblePutOffs**

*Таблица 14 – Описание членов класса possiblePutOffs*

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Назначение* |
| NONE | public | possiblePutOffs | Значение переноса, задающее отсутствие переноса плана. |
| NEXTDAY | public | possiblePutOffs | Значение переноса, задающее перенос на следующий день. |
| NEXTWEEK | public | possiblePutOffs | Значение переноса, задающее перенос на следующую неделю. |
| NEXTMONTH | public | possiblePutOffs | Значение переноса, задающее перенос на следующий месяц. |
| NEXTYEAR | public | possiblePutOffs | Значение переноса, задающее перенос на следующий год. |

**Описание членов класса possibleReminders**

*Таблица 15 – Описание членов класса possibleReminders*

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Назначение* |
| NONE | public | possibleReminders | Значение напоминания, задающее отсутствие напоминаний о плане. |
| H3BEFORE | public | possibleReminders | Значение напоминания, задающее напоминания о плане за 3 часа до его крайнего срока. |
| H12BEFORE | public | possibleReminders | Значение напоминания, задающее напоминания о плане за 12 часов до его крайнего срока. |
| H24BEFORE | public | possibleReminders | Значение напоминания, задающее напоминания о плане за 24 часа до его крайнего срока. |
| H72BEFORE | public | possibleReminders | Значение переноса, задающее напоминания о плане за 72 часа до его крайнего срока. |

**Описание членов класса Subplan**

*Таблица 16 – Описание членов класса Subplan*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public | void | String, Boolean | | Конструктор с параметрами для подзадачи, задает значения полей. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| title | public | String | | Поле, свойство с автоматическими геттером и сеттером для названия подзадачи, в сеттере проверяется то, что длина нового имени лежит в пределах от 1 до 19. Дефолтное значение «новый подплан». | |
| isFinished | public | Boolean | | Поле, свойство с автоматическими геттером и сеттером для статуса завершенности подзадачи. В сеттере проверяется, что значение меняется на противоположное. Дефолтное значение – false. | |

**Описание членов класса Category**

*Таблица 17 – Описание членов класса Category*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public | void | String, Int | | Конструктор с параметрами для категории, задает значения полей. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| name | public | String | | Поле, свойство с автоматическими геттером и сеттером для названия категории, в сеттере проверяется то, что длина нового имени лежит в пределах от 1 до 19. Если задается строка «без категории», полю присваивается значение пустой строки. Дефолтное значение «новая категория». | |
| colour | public | Int | | Поле, свойство с автоматическими геттером и сеттером для цвета, ассоциированного с категорией, закодированного целым числом. Дефолтное значение – 0. | |

**Описание членов класса MainActivity**

*Таблица 18 – Описание членов класса MainActivity*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreate | public | void | Bundle? | | Оверрайд стандартного метода Activity, включает лейаут, настраивает видимость кнопок. |
| onStart | public | void | - | | Оверрайд стандартного метода Activity, вызывает тот же метод у базового класса. |
| LogIn | public | void | View | | Метод для регистрации нового пользователя, считывает логин и пароль из тектовых полей для ввода, обращается к менеджеру аутентификации за проверкой правильности, в случае верных данных настраивает новый аккаунт и обновляет интерфейс. |
| SignIn | public | void | View | | Метод для входа пользователя в существующий аккаунт, считывает логин и пароль из тектовых полей для ввода, обращается к менеджеру аутентификации за проверкой наличия такого аккаунта, в случае верных данных входит в аккаунт и обновляет интерфейс. |
| updateUI | private | void | FirebaseUser? | | Метод обновления интерфейса для пользователя из базы данных, выводится сообщение об авторизации и открывается экран навигации на основе информации пользователя, иначе выводятся сообщения о неправильных данных. |
| updateUI | private | void | String | | Метод для вывода получаемого сообщения в текстовое поле. |
| exampleUser | private | void | - | | Метод для создания нового пользователя с планами и категориями «например». |
| getSaved | public | void | View | | Метод для получения сохраненных локально данных пользователя – пытаемся получить SharedPreferences, если не удается, вызываем метод создания нового пользователя. |
| save | private | void | - | | Метод сохранения информации пользователя в SharedPreferences. |
| onPause | public | void | - | | Оверрайд стандартного метода Activity, при ее прекращении вызывается метод сохранения данных. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| login\_ | private | String | | Поле для строкового логина пользователя, дефолтное значение - "". | |
| password\_ | private | String | | Поле для строкового пароля пользователя, дефолтное значение - "". | |
| authManager | private | AuthManager | | Поле для менеджера аутентификации – объект для взаимодействия с сервером, дефолтное значение - AuthManager(). | |
| visitedSecond | private | Boolean | | Поле для значения того, заходит ли пользователь на этот экран впервые за сессию, либо нет. В случае повторного попадания на экран нет необходимости заново обращаться к серверу или локальному хранилищу за данными пользователя. Дефолтное значение – false. | |
| new\_user | private | User | | Поле для объекта пользователя, в которое при получении сохраненной информации будет записана она, либо, если ее нет, будет записан новый пользователь. Дефолтное значение - User(). | |

**Описание членов класса BottomNavigationScreen**

*Таблица 19 – Описание членов класса BottomNavigationScreen*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreate | public | void | Bundle? | | Оверрайд стандартного метода Activity, включает лейаут нижнего меню навигации, проверяет актуальность всех планов (проходит по планам и вызывает метод переноса), передает данные в дочерние фрагменты. |
| save | private | void | - | | Метод сохранения информации пользователя в SharedPreferences. |
| onPause | public | void | - | | Оверрайд стандартного метода Activity, при ее прекращении вызывается метод сохранения данных. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| passed\_user | public | User | | Поле для передаваемого из основной activity пользователя, статическое поле, чтобы объект был по ссылке и обновлялся при изменениях. Дефолтное значение - User(). | |

**Описание членов класса DeadlineEditActivity**

*Таблица 20– Описание членов класса DeadlineEditActivity*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreate | public | void | Bundle? | | Оверрайд стандартного метода Activity, включает лейаут редактирования плана, настраивает информацию всех полей в соответствии с информацией пользователя, задает методы нажатия на кнопки и поля, информацию в спиннерах для выпадающих списков. |
| showDateTimePicker | private | void | TextView | | Метод для настройки открытия диалогов выбора даты и времени для выбора даты и крайнего срока выполнения плана, открывают по очереди календарь и диалог выбора времени и задают выбранные значения в текст текстового поля из аргументов. |
| onItemSelected | public | void | AdapterView<\*>, View?, Int, Long | | Оверрайд стандартного метода AdapterView.OnItemSelectedListener, нужно для того, чтобы работал спиннер с выбором категории, при том, что вся логика вынесена из этих методов. |
| onNothingSelected | public | void | AdapterView<\*> | | Оверрайд стандартного метода AdapterView.OnItemSelectedListener, нужно для того, чтобы работал спиннер с выбором категории, при том, что вся логика вынесена из этих методов. |
| refresh | private | void | RecyclerView | | Метод для обновления вида списка подзадач, чтобы информация отображалась после изменений. |
| save | public | void | View | | Метод для сохранения плана в соответствии с изменениями. Задает значения полей изменяемого плана в соответствии с введенными в поля ввода данными, проводит проверки ввода, закрывает экран. |
| delete | public | void | View | | Метод для удаления плана, исключает редактируемый план из списка планов пользователя. |
| save | private | void | - | | Метод сохранения информации пользователя в SharedPreferences, нужен для обновления информации об измененном плане. |
| onPause | public | void | - | | Оверрайд стандартного метода Activity, при ее прекращении вызывается метод сохранения данных. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| deadlineToEdit | public | Plan | | Поле для передаваемого из основной activity редактируемого плана, статическое поле, чтобы объект был по ссылке и обновлялся при изменениях. Дефолтное значение - Plan(). | |
| userToEdit | public | User | | Поле для передаваемого из основной activity пользователя, статическое поле, чтобы объект был по ссылке и обновлялся при изменениях. Дефолтное значение - User(). | |
| adapter | private | DeadlineEditAdapter? | | Поле для адаптера списка подпланов дедлайна, дефолтное значение – null. | |

**Описание членов класса DeadlineEditAdapter**

*Таблица 21 – Описание членов класса DeadlineEditAdapter*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateViewHolder | public | void | ViewGroup, Int | | Оверрайд стандартного метода адаптера ресайкл вью, создающий вью на основе лейаута. |
| onBindViewHolder | public | void | DeadlineEditViewHolder, Int | | Оверрайд стандартного метода адаптера ресайкл вью, который связывает элементы отображения с элементами отображаемого списка подпланов. |
| getItemCount | public | Int | - | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий количество элементов в отображаемом списке. |
| getItem | private | Subplan | Int | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий подплан по индексу в массиве отображаемых подпланов. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| subplans | private | ArrayList<Subplan> | | Поле для отображаемого списка подпланов. | |

**Описание членов класса DeadlineEditViewHolder**

*Таблица 22 – Описание членов класса DeadlineEditViewHolder*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| bind | public | void | Sublan | | Оверрайд стандартного метода вью холдера адаптера ресайкл вью, который связывает информацию отображаемого объекта – подплана и поля лейаута для него. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| name | private | TextView | | Поле для названия подплана. | |
| check | private | CheckBox | | Поле для коробочки с галочкой выполнения, связывается со статусом выполнения подплана. | |
| subplan\_note | private | Subplan | | Поле для отображаемого подплана, дефолтное значение - Subplan(). | |

**Описание членов класса CategoryEditActivity**

*Таблица 23 – Описание членов класса CategoryEditActivity*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreate | public | void | Bundle? | | Оверрайд стандартного метода Activity, включает лейаут редактирования категории, настраивает информацию всех полей в соответствии с информацией пользователя, задает методы нажатия на кнопки и планы. |
| refresh | private | void | RecyclerView | | Метод для обновления вида списка планов, чтобы информация отображалась после изменений. |
| categorySave | public | void | View | | Метод для сохранения категории в соответствии с изменениями. Задает значения названия изменяемой категории в соответствии с введенным в поле ввода, для этого сначала удаляет текущую категорию (чтобы из названий удалилось ее название), затем добавляет новую, закрывает экран. |
| categoryDelete | public | void | View | | Метод для удаления категории, исключает редактируемую категорию из списка категорий пользователя с удалением ее названия, закрывает экран. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| categoryToEdit | public | Category | | Поле для передаваемой из основной activity редактируемой категории, статическое поле, чтобы объект был по ссылке и обновлялся при изменениях. Дефолтное значение - Category(). | |
| userWithCategoryToEdit | public | User | | Поле для передаваемого из основной activity пользователя, статическое поле, чтобы объект был по ссылке и обновлялся при изменениях. Дефолтное значение - User(). | |
| adapter | private | CategoryEditAdapter? | | Поле для адаптера списка планов в категории, дефолтное значение – null. | |

**Описание членов класса CategoryEditAdapter**

*Таблица 24 – Описание членов класса CategoryEditAdapter*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateViewHolder | public | void | ViewGroup, Int | | Оверрайд стандартного метода адаптера ресайкл вью, создающий вью на основе лейаута. |
| onBindViewHolder | public | void | CategoryEditViewHolder, Int | | Оверрайд стандартного метода адаптера ресайкл вью, который связывает элементы отображения с элементами отображаемого списка планов. |
| getItemCount | public | Int | - | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий количество элементов в отображаемом списке. |
| getItem | private | Plan | Int | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий план по индексу в массиве отображаемых планов. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| plans | private | ArrayList<Plan> | | Поле для отображаемого списка планов. | |

**Описание членов класса CategoryEditViewHolder**

*Таблица 25 – Описание членов класса DeadlineEditViewHolder*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public | void | - | | Метод настраивает реакцию на нажатие на элемент списка планов – открытие экрана редактирования выбранного плана. |
| bind | public | void | Plan | | Оверрайд стандартного метода вью холдера адаптера ресайкл вью, который связывает информацию отображаемого объекта –плана и поля лейаута для него. |
| openDeadlineScreenEdit | private | void | Plan | | Метод для открытия экрана редактирования выбранного плана. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| name | private | TextView | | Поле для текстового поля для названия плана. | |
| check | private | CheckBox | | Поле для галочки выполненности плана. | |
| plan | private | Plan | | Поле для текущего объекта плана, с которым связывается элемент отображения, дефолтное значение - Plan(). | |
| rating | private | RatingBar | | Поле для оценки важности плана. | |
| notes | private | TextView | | Поле для текста с заметками о плане. | |
| category | private | TextView | | Поле для текстового поля с названием категории плана. | |
| date | private | TextView | | Поле для текстового поля с датой выполнения плана. | |
| deadline | private | TextView | | Поле для текстового поля с крайним сроком выполнения плана. | |

**Описание членов класса ListFragment**

*Таблица 26 – Описание членов класса ListFragment*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateView | public | View? | LayoutInflater,  ViewGroup?, Bundle? | | Метод для настройки связывания объектов в списках через автоматический databinding, настройка спиннера для выбора способа сортировки планов. |
| openDeadlineScreen | private | void | Plan | | Метод для открытия экрана редактирования плана при создании нового плана. |
| onDestroy | public | void | - | | Метод для завершения работы фрагмента, освобождается переменная связи. |
| onItemSelected | public | void | AdapterView<\*>, View?, Int, Long | | Оверрайд стандартного метода AdapterView.OnItemSelectedListener, нужно для того, чтобы работал спиннер с выбором способа сортировки, при выборе какого-либо варианта вызывается обновление вида списков с выбранным режимом упорядочивания. |
| onNothingSelected | public | void | AdapterView<\*> | | Оверрайд стандартного метода AdapterView.OnItemSelectedListener, нужно для того, чтобы работал спиннер с выбором способа сортировки планов, при отсутствии выбранного элемента происходит обновление сортировки с дефолтным значением «по срочности». |
| refresh | private | void | RecyclerView | | Метод для обновления вида списка планов, чтобы информация отображалась после изменений. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| userList | public | User | | Статическое поле для передаваемого извне пользователя, нужно для обновления информации, дефолтное значение - User(). | |
| parentActivityList | public | BottomNavigationScreen | | Статическое поле для передаваемой снаружи родительской activity, нужно для обновления информации, дефолтное значение - BottomNavigationScreen(). | |
| \_binding | private | FragmentListBinding? | | Поле для переменной автоматической связи databinding между объектом списка и лейаутом, соответствующим ему, дефолтное значение – null. | |
| binding | private | FragmentListBinding | | Поле для переменной, инкапсулирующей \_binding и гарантирующей отсутствие null в переменной связи. | |
| listList | private | ArrayList<ListForRV> | | Поле для отображаемого списка заголовков списков планов, дефолтное значение - ArrayList<ListForRV>(). | |
| nestedListAdapter | private | NestedListAdapter | | Поле для адаптера сворачиваемых списков. | |

**Описание членов класса NestedListAdapter**

*Таблица 27 – Описание членов класса NestedListAdapter*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateViewHolder | public | void | ViewGroup, Int | | Оверрайд стандартного метода адаптера ресайкл вью, создающий вью на основе лейаута. |
| onBindViewHolder | public | void | ViewHolder, Int | | Оверрайд стандартного метода адаптера ресайкл вью, который связывает элементы отображения с элементами отображаемого списка сворачиваемых списков планов, задаются два элемента – выполненные и невыполненные планы, задается сортировка в зависимости от выбранной. |
| getItemCount | public | Int | - | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий количество элементов в отображаемом списке сворачиваемых списков. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| listList | private | ArrayList<ListForRV> | | Поле для отображаемого списка заголовков списков планов, дефолтное значение - ArrayList<ListForRV>(). | |
| thisAdapter | private | NestedListAdapter? | | Поле адаптера списка сворачиваемых списков. | |
| adapter | private | DeadlinesListAdapter? | | Поле адаптера списка планов внутри списка сворачиваемых списков планов. | |

**Описание членов класса ListForRV**

*Таблица 28 – Описание членов класса ListForRV*

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Назначение* |
| name | public | String | Поле для названия заголовка сворачиваемого списка, дефолтное значение - “”, при использовании будет либо «завершенные планы», либо «незавершенные планы». |
| expand | public | Boolean | Поле для значения того, свернут ли список планов под этим заголовком, дефолтное значение – false. |
| sort | public | String | Поле для обозначения правила, в соответствии с которым сортируются планы в списке с этим заголовком, дефолтное значение - “по срочности”. |

**Описание членов класса DeadlinesListAdapter**

*Таблица 29 – Описание членов класса DeadlinesListAdapter*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateViewHolder | public | void | ViewGroup, Int | | Оверрайд стандартного метода адаптера ресайкл вью, создающий вью на основе лейаута, здесь в зависимости от переменной режима настраивается либо полноценное отображение планов, либо облегченное – для матрицы Эйзенхауэра. |
| onBindViewHolder | public | void | ListViewHolder, Int | | Оверрайд стандартного метода адаптера ресайкл вью, который связывает элементы отображения с отображаемыми планами. |
| getItemCount | public | Int | - | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий количество элементов в отображаемом списке планов. |
| getItem | private | Plan | Int | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий план по индексу в массиве отображаемых планов. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| deadlines | private | ArrayList<Plan> | | Поле для отображаемого списка планов. | |
| mode | private | String | | Поле для режима отображения – в случае значения «light» планы отображаются в облегченном, менее подробном виде, это нужно при использовании этого класса как адаптера на экране графики для матрицы Эйзенхауэра. | |

**Описание членов класса ListViewHolder**

*Таблица 25 – Описание членов класса ListViewHolder*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public | void | - | | Метод настраивает реакцию на нажатие на элемент списка планов – открытие экрана редактирования выбранного плана. |
| bind | public | void | Plan | | Оверрайд стандартного метода вью холдера адаптера ресайкл вью, который связывает информацию отображаемого объекта –плана и поля лейаута для него, в случае облегченного отображения плана некоторые поля не отображаются. |
| openDeadlineScreenEdit | private | void | Plan | | Метод для открытия экрана редактирования выбранного плана. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| *mode* | *private* | *String* | | Поле для режима отображения – в случае значения «light» планы отображаются в облегченном, менее подробном виде, это нужно при использовании этого класса как адаптера на экране графики для матрицы Эйзенхауэра. | |
| plan | private | Plan | | Поле для текущего объекта плана, с которым связывается элемент отображения, дефолтное значение - Plan(). | |

**Описание членов класса CalendarFragment**

*Таблица 30 – Описание членов класса CalendarFragment*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateView | public | View? | LayoutInflater,  ViewGroup?, Bundle? | | Метод для настройки вида в соответствии с лейаутом. Настраивается вид календаря, его реакции на взаимодействия с ним, настраивается добавление плана на текущую выбранную дату. |
| openDeadlineScreen | private | void | Plan | | Метод для открытия экрана редактирования плана при создании нового плана. |
| refresh | private | void | RecyclerView | | Метод для обновления вида списка планов, чтобы информация отображалась после изменений, также тут происходит фильтрация планов в соответствии с совпадением даты выполнения плана и выбранной даты в календаре, чтобы в календарном виде отображались планы на выбранную дату. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| userCalendar | public | User | | Статическое поле для передаваемого извне пользователя, нужно для обновления информации, дефолтное значение - User(). | |
| parentActivityCalendar | public | BottomNavigationScreen | | Статическое поле для передаваемой снаружи родительской activity, нужно для обновления информации, дефолтное значение - BottomNavigationScreen(). | |
| adapter | private | CalendarAdapter? | | Поле адаптера списка планов на экране календаря. | |

**Описание членов класса CalendarAdapter**

*Таблица 31 – Описание членов класса CalendarAdapter*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateViewHolder | public | void | ViewGroup, Int | | Оверрайд стандартного метода адаптера ресайкл вью, создающий вью на основе лейаута. |
| onBindViewHolder | public | void | CalendarViewHolder, Int | | Оверрайд стандартного метода адаптера ресайкл вью, который связывает элементы отображения с элементами отображаемого списка планов. |
| getItemCount | public | Int | - | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий количество элементов в отображаемом списке. |
| getItem | private | Plan | Int | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий план по индексу в массиве отображаемых планов. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| deadlines | private | ArrayList<Plan> | | Поле для отображаемого списка планов. | |

**Описание членов класса CalendarViewHolder**

*Таблица 25 – Описание членов класса CalendarViewHolder*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public | void | - | | Метод настраивает реакцию на нажатие на элемент списка планов – открытие экрана редактирования выбранного плана. |
| bind | public | void | Plan | | Оверрайд стандартного метода вью холдера адаптера ресайкл вью, который связывает информацию отображаемого объекта –плана и поля лейаута для него. |
| openDeadlineScreenEdit | private | void | Plan | | Метод для открытия экрана редактирования выбранного плана. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| name | private | TextView | | Поле для текстового поля для названия плана. | |
| check | private | CheckBox | | Поле для галочки выполненности плана. | |
| plan | private | Plan | | Поле для текущего объекта плана, с которым связывается элемент отображения, дефолтное значение - Plan(). | |
| rating | private | RatingBar | | Поле для оценки важности плана. | |
| notes | private | TextView | | Поле для текста с заметками о плане. | |
| category | private | TextView | | Поле для текстового поля с названием категории плана. | |
| date | private | TextView | | Поле для текстового поля с датой выполнения плана. | |
| deadline | private | TextView | | Поле для текстового поля с крайним сроком выполнения плана. | |

**Описание членов класса GraphFragment**

*Таблица 32 – Описание членов класса GraphFragment*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateView | public | View? | LayoutInflater,  ViewGroup?, Bundle? | | Метод для настройки вида в соответствии с лейаутом. Настраивается вид матрицы Эйзенхауэра, для этого создаются четыре списка и четыре адаптера соответственно, и планы пользователя классифицируются на эти четыре категории. Также настраивается кнопка с информацией про концепцию такой классификации. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| userGraph | public | User | | Статическое поле для передаваемого извне пользователя, нужно для обновления информации, дефолтное значение - User(). | |
| parentActivityGraph | public | BottomNavigationScreen | | Статическое поле для передаваемой снаружи родительской activity, нужно для обновления информации, дефолтное значение - BottomNavigationScreen(). | |
| adapterUI | private | DeadlinesListAdapter? | | Поле для адаптера списка планов, относящихся к категории срочных и важных, дефолтное значение - null. | |
| adapterUNI | private | DeadlinesListAdapter? | | Поле для адаптера списка планов, относящихся к категории срочных и неважных, дефолтное значение - null. | |
| adapterNUI | private | DeadlinesListAdapter? | | Поле для адаптера списка планов, относящихся к категории несрочных и важных, дефолтное значение - null. | |
| adapterNUNI | private | DeadlinesListAdapter? | | Поле для адаптера списка планов, относящихся к категории несрочных и неважных, дефолтное значение - null. | |

**Описание членов класса SettingsFragment**

*Таблица 1– Описание членов класса SettingsFragment*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateView | public | View? | LayoutInflater,  ViewGroup?, Bundle? | | Метод для настройки вида в соответствии с лейаутом. Настраивается список тех, вещей, которые можно настраивать, прикрепляется адаптер. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| userSettings | public | User | | Статическое поле для передаваемого извне пользователя, нужно для обновления информации, дефолтное значение - User(). | |
| parentActivitySettings | public | BottomNavigationScreen | | Статическое поле для передаваемой снаружи родительской activity, нужно для обновления информации, дефолтное значение - BottomNavigationScreen(). | |
| adapter | private | SettingsAdapter? | | Поле адаптера списка настраиваемых блоков на экране настроек. | |

**Описание членов класса SettingsAdapter**

*Таблица 1– Описание членов класса SettingsAdapter*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| onCreateViewHolder | public | void | ViewGroup, Int | | Оверрайд стандартного метода адаптера ресайкл вью, создающий вью на основе лейаута. |
| onBindViewHolder | public | void | SettingsViewHolder, Int | | Оверрайд стандартного метода адаптера ресайкл вью, который связывает элементы отображения с элементами отображаемого списка категорий настроек. |
| getItemCount | public | Int | - | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий количество элементов в отображаемом списке настраиваемых функций. |
| getItem | private | Plan | Int | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий строку с названием настраиваемой функции по индексу в массиве отображаемых планов. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| settings | private | ArrayList<String> | | Поле для отображаемого списка настраиваемых функций. | |

**Описание членов класса SettingsViewHolder**

*Таблица 25 – Описание членов класса SettingsViewHolder*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public | void | - | | Метод настраивает реакцию на нажатие на элемент списка настраиваемых функций – открытие экрана настройки того, что пользователь выбрал по названию. |
| bind | public | void | Plan | | Оверрайд стандартного метода вью холдера адаптера ресайкл вью, который связывает название отображаемой опции для настройки с названиями возможных настраиваемых функций. |
| openSettingsScreenEdit | private | void | Plan | | Метод для открытия экрана настройки выбранной по названию функции, в зависимости от названия открываются разные экраны. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| name | private | TextView | | Поле для текстового поля для названия настраиваемой функции. | |

**Описание членов класса UserSettingsActivity**

*Таблица 1– Описание членов класса UserSettingsActivity*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | *Назначение* |
| onCreate | public | void | Bundle? | Оверрайд стандартного метода Activity, включает лейаут настроек учетной записи пользователя настраивает кнопки для всех функций. |
| deleteLocal | public | void | View | Метод для удаления локально сохраненных данных пользователя, очищает SharedPreferences. |
| logInWithNewAccount | public | void | View | Метод для входа в приложение с другой учетной записью с удалением текущей информации. |
| bindNewEmail | public | void | View | Метод для привязки нового почтового адреса к существующей учетной записи с сохранением информации. |

**Описание членов класса SettingsCategoriesScreen**

*Таблица 1– Описание членов класса SettingsCategoriesScreen*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreate | public | void | Bundle? | | Оверрайд стандартного метода Activity, включает лейаут редактирования категории, настраивает адаптер к списку категорий, задает методы нажатия на кнопки и названия категорий. |
| save | private | void | - | | Метод сохранения информации пользователя в SharedPreferences. |
| onPause | public | void | - | | Оверрайд стандартного метода Activity, при ее прекращении вызывается метод сохранения данных. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| userWithCategories | public | User | | Статическое поле для передаваемого извне пользователя, нужно для обновления информации, дефолтное значение - User(). | |
| parentSettingsCategories | public | BottomNavigationScreen | | Статическое поле для передаваемой снаружи родительской activity, нужно для обновления информации, дефолтное значение - BottomNavigationScreen(). | |
| adapter | private | SettingsCategoryAdapter? | | Поле адаптера списка названий категорий пользователя. | |

**Описание членов класса SettingsCategoriesAdapter**

*Таблица 1– Описание членов класса SettingsCategoriesAdapter*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| onCreateViewHolder | public | void | ViewGroup, Int | | Оверрайд стандартного метода адаптера ресайкл вью, создающий вью на основе лейаута. |
| onBindViewHolder | public | void | SettingsCategoryViewHolder, Int | | Оверрайд стандартного метода адаптера ресайкл вью, который связывает элементы списка категорий пользователя с элементами отображаемого списка названий категорий. |
| getItemCount | public | Int | - | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий количество элементов в отображаемом списке названий категорий. |
| getItem | private | Category | Int | | Оверрайд стандартного метода адаптера ресайкл вью, возвращающий категорию по индексу в массиве отображаемых категорий. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| categories | private | ArrayList<Category> | | Поле для отображаемого списка категорий. | |

**Описание членов класса SettingsCategoryViewHolder**

*Таблица 1– Описание членов класса SettingsCategoryViewHolder*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Методы** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | *Аргументы* | | *Назначение* |
| init | public | void | - | | Метод настраивает реакцию на нажатие на элемент списка названий категорий – открытие экрана редактирования категории, название которой было выбрано. |
| bind | public | void | Category | | Оверрайд стандартного метода вью холдера адаптера ресайкл вью, который связывает название отображаемой категории с полем названия и приравнивает поле категории к передаваемой отображаемой категории. |
| openCategoryEditScreen | private | void | Category | | Метод для открытия экрана редактирования выбранной по названию категории. |
| **Поля** | | | | | |
| *Имя* | *Мод. Доступа* | *Тип* | | *Назначение* | |
| name | private | TextView | | Поле для текстового поля для названия отображаемой категории. | |
| category | private | Category | | Поле для сохранения отображаемой категории. | |

ПРИЛОЖЕНИЕ 5

**ДИАГРАММА КЛАССОВ**

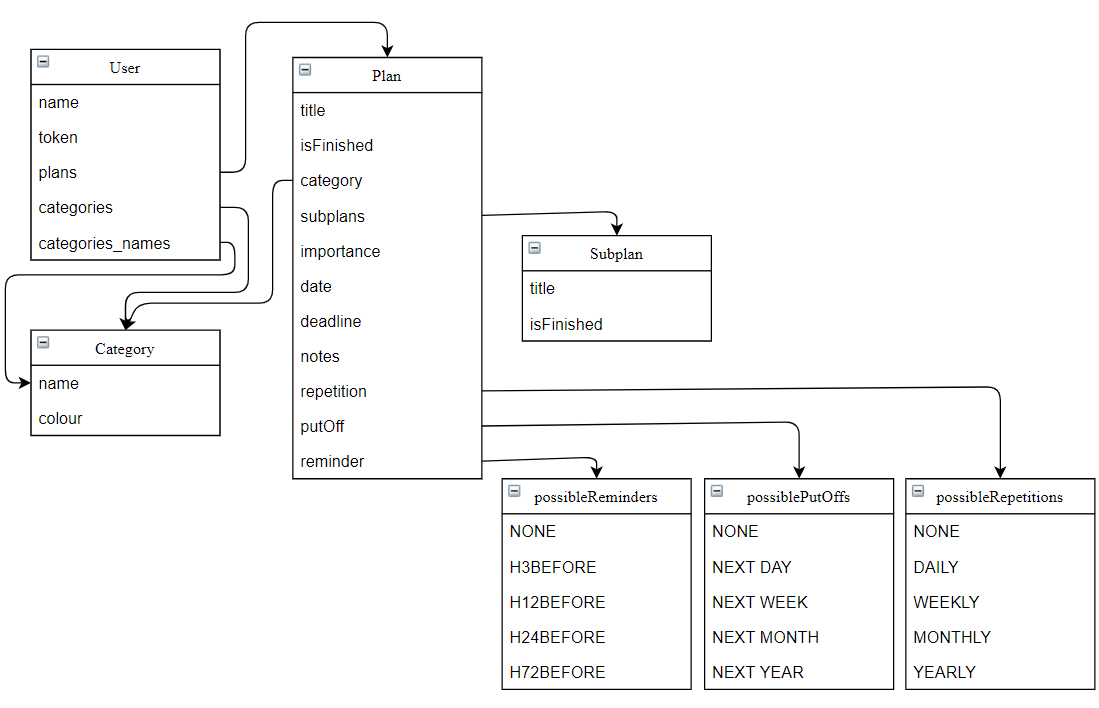


Рисунок 2 – Диаграмма классов библиотеки PossumLib.

**ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| измененных | замененных | Новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |