Омский Научный центр Сибирского отделения Российской академии наук

Региональная общественная организация «Омский совет ректоров»

Омское региональное отделение Всероссийской общественной организации

«Русское географическое общество»

Детская областная общественная организация

«Научное общество учащихся «Поиск»

Бюджетное образовательное учреждение дополнительного образования города Омска «Центр творчества «Созвездие»

52-я

Межрегиональная научно-практическая конференция

школьников и учащейся молодежи

Тема: «Daily.Maps, приложение-органайзер с возможностью работы с картами»

Учебно-исследовательская работа

Научное направление: Информатика и программирование

*Выполнил:*ученик 9 класса  
БОУ г. Омска «Лицей №64»  
Кайков Артем Кириллович  
  
*Научный руководитель:*педагог д.о.

БОУ ДО «Центр творчества «Созвездие»  
Морозов А. Д.

Омск – 2020

Содержание

[Введение 3](#_Toc35855964)

[Цель 4](#_Toc35855965)

[Задачи 4](#_Toc35855966)

[Анализ существующих решений 4](#_Toc35855967)

[Выбор технологий реализации 5](#_Toc35855968)

[Функциональные возможности приложения 5](#_Toc35855969)

[Хранение данных 7](#_Toc35855970)

[Создание алгоритма оптимизации времени 9](#_Toc35855971)

[Техническая часть 10](#_Toc35855972)

[Вывод 12](#_Toc35855973)

[Источники 13](#_Toc35855974)

# Введение

Сегодня темп жизни людей заметно ускорился. Ежедневно им приходится выполнять большое количество заданий, зачастую расположенных в разных частях города. В такой спешке очень легко забыть что-то важное или не уследить за выполнением задания. Конечно, можно использовать приложения-органайзеры, но если задания находятся в разных частях города, человек вынужден пользоваться несколькими приложениями одновременно. Также большинство органайзеров не учитывают затраты времени на транспорт и выполнение задания. А с добавлением каждого нового задания становится всё труднее не нарушить установленное расписание.

Поэтому я решил создать мобильное приложение, сочетающие функции органайзера и работу с картами. Разработка такого приложения позволит людям не только эффективнее планировать своё время, но и, например, сохранять свои часто используемые маршруты, делиться ими или подготовить маршрут для другого человека.

# Цель

Целью работы стала разработка приложения-органайзера «Daily.Maps» с возможностями построения маршрутов по карте и оптимизацией времени.

# Задачи

1. Анализ существующих решений
2. Выявление технологий, необходимых для реализации проекта
3. Реализация графического пользовательского интерфейса
4. Определение оптимального формата хранения данных
5. Создание алгоритма оптимизации пользовательского маршрута
6. Тестирование и устранение ошибок в работе приложения

# Анализ существующих решений

Разрабатываемое приложение не имеет прямых аналогов, но сочетает функции двух видов существующих сервисов или приложений: органайзеров и карт.

Поэтому первым этапом в реализации проекта был анализ уже существующих на рынке органайзеров с целью определения минимально необходимого набора функций органайзера. После изучения вариантов, предлагаемых площадкой Google Play Market, были выбраны пять самых популярных органайзеров: «Any.do», «Todoist», «TickTick», «Microsoft To Do», и «Wunderlist». Все они обладали несколькими базовыми функциями – добавление заданий, их редактирование, выполнение и удаление.

Некоторые из них также обладают и другими возможностями, но эти функции, как правило, платные. Например, в приложении «Any.do» заплатив за подписку, вы получаете доступ к тонкой настройке уведомлений: они могу приходить через определенный промежуток времени или когда вы приближаетесь к отмеченному на карте месту.

Помимо функций органайзера, Daily.Maps должно работать с картами: привязать задание к месту на карте, строить маршруты и отображать время их начала и выполнения, создавать шаблоны на основе выбранных заданий, которыми можно делиться с другими пользователями.

# Выбор технологий реализации

Целевой платформой разработки выступала ОС *Android* [1], и соответствующий ей язык программирования *Kotlin* [2].

Также есть потребность хранить пользовательские данные, например, созданные задания и шаблоны. Для их хранения были рассмотрены несколько вариантов баз данных, в результате был выбран комплекс программных решений *Firebase* [3], в который входит документно-ориентированная база данных *Firebase Firestore*, она гарантирует доступ к данным в любое время, так как в случае отсутствия соединения, последние сохраненные данные выгружаются из кэша устройства.

Так как в приложение должны быть интегрированы карты, были использованы решения от Google [4] – *Maps SDK* и *Directions API*. В последующих версиях карты планируется реализовать при помощи решений *Mapbox* [5] по причине более гибкой настройки и меньшей цены использования.

Для исполнения графического интерфейса приложения была применена библиотека *Material Design* [6], которая поставляет самые актуальные и удобные в использовании виджеты.

# Функциональные возможности приложения

Перед тем, как приступить непосредственно к созданию backend-логики, необходимо определить, как пользователь будет взаимодействовать с приложением. И реализовать основные интерфейсы.

Так как сервис требует обязательную регистрацию, необходимо её упростить, поэтому была добавлена возможность авторизации через Google-аккаунт. После этого пользователь попадает на главный экран, на котором отображаются его задания (Рисунок 1). Он должен иметь возможность работать со своими задачами, поэтому у каждого из них есть кнопка, которая вызывает контекстное меню взаимодействия (Рисунок 2). В нем представлены основные необходимые функции.

Рис. 1 Главный экран со списком заданий

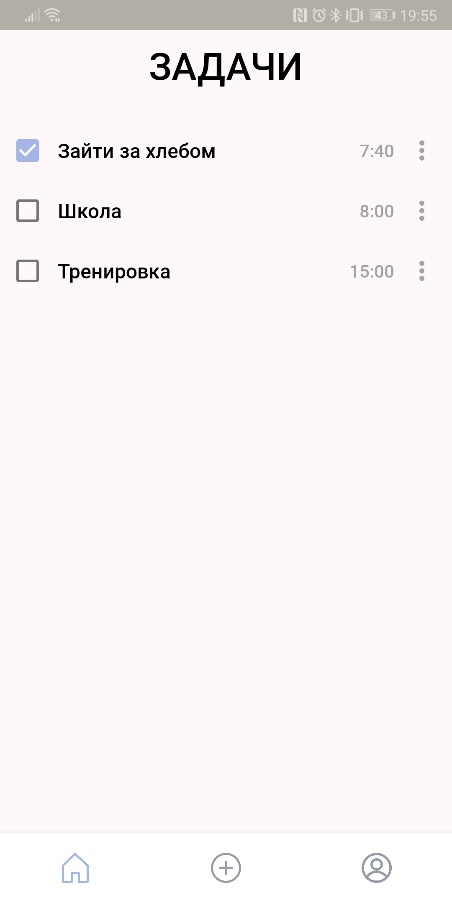
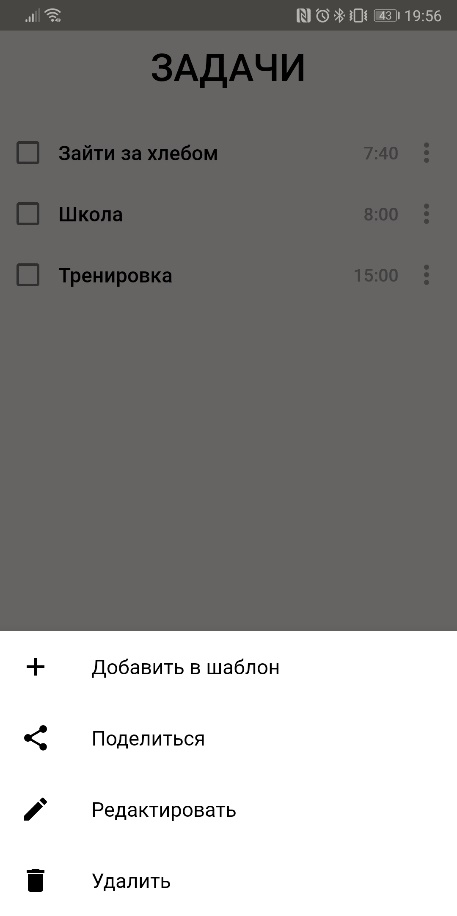


Рис. 2 Контекстное меню взаимодействия с заданием

При выборе пункта «Добавить в шаблон» на экране появится новый диалог, предлагающий пользователю выбрать один из существующих шаблонов или создать новый (Рисунок 3). При выборе пункта «Поделиться» появится меню выбора социальной сети, через которую будет отправлена ссылка на задание. При открытии этой ссылки, если у получателя уже установлено приложение, ему отобразится вся информация о задании и будет предложено добавить дубликат этого задания к себе. Помимо этого, воспользовавшись нижней навигационной панелью можно открыть диалог добавления нового задания (Рисунок 4).

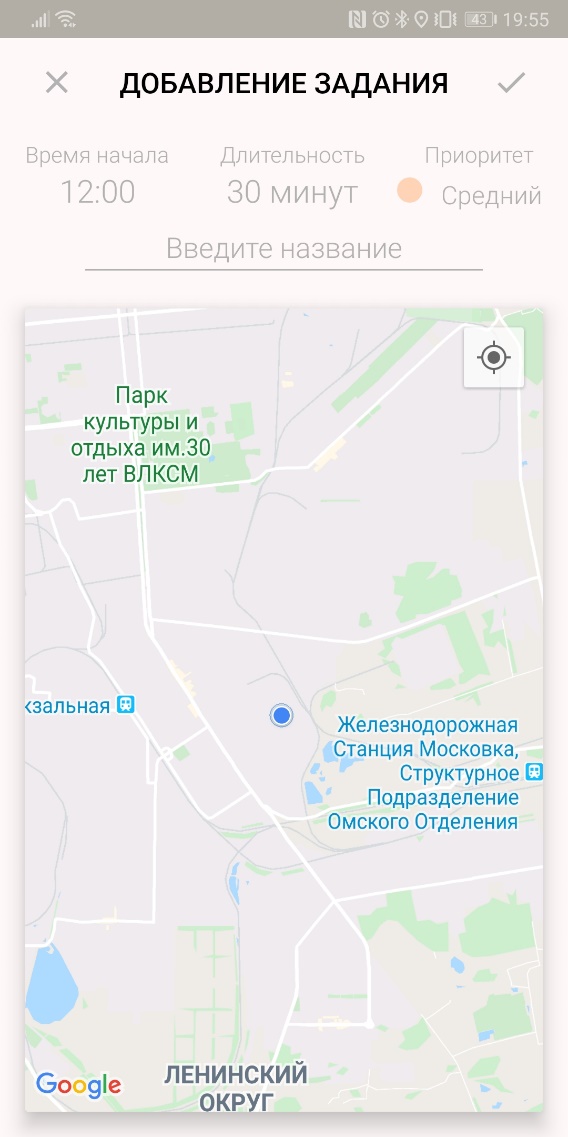
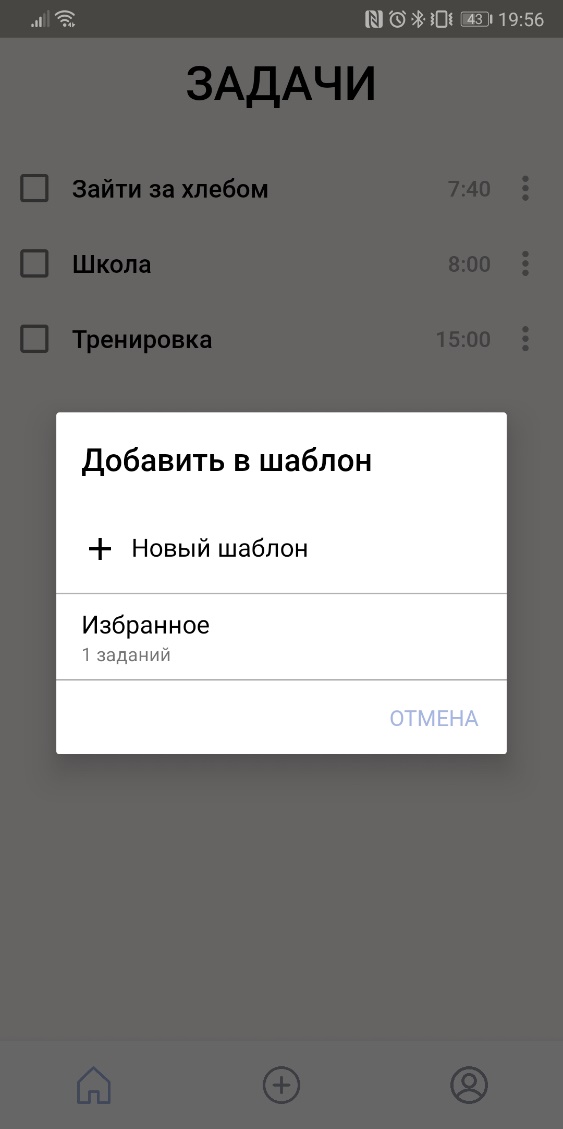


Рис. 3 Диалоговое окно выбора шаблона

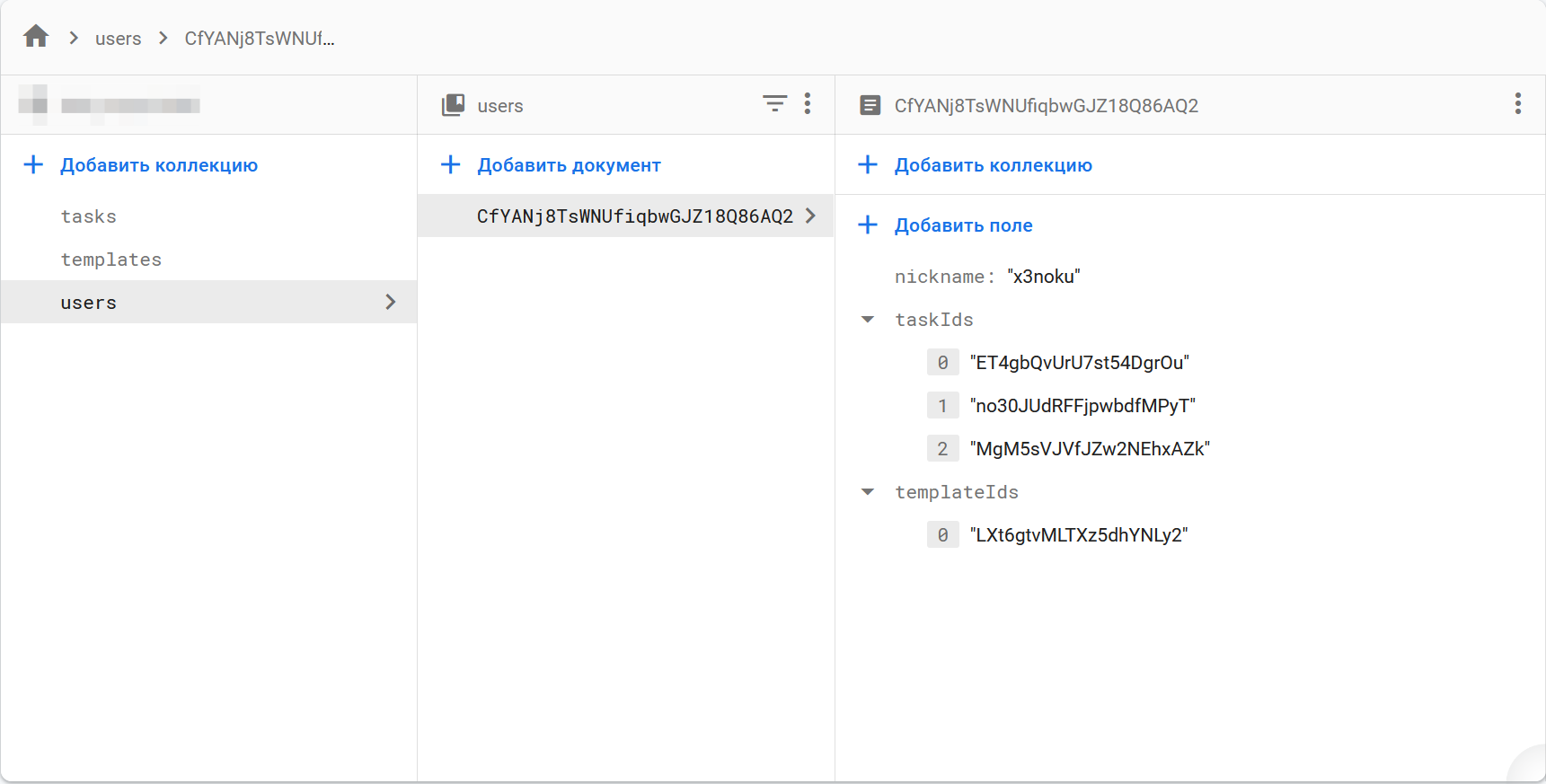
Рис. 4 Диалог добавления нового задания

В разделе профиля, куда можно также попасть через нижнюю навигационную панель, находится информация о вашем аккаунте, и список шаблонов, с которыми также можно взаимодействовать. После открытия одного из шаблонов, в случае наличия в нём заданий, отмеченных на карте, отображается кнопка «Составить маршрут», нажав на которую пользователь будет отправлен на страницу с картой и маршрутом, построенным по шаблону и оптимизированным алгоритмами приложения.

# Хранение данных

Очень важным шагом в проектировании является выявление оптимального формата хранения данных, ведь если он будет выбран неправильно, это сильно затруднит процесс разработки и поддержания проекта, а также может стать причиной дополнительных расходов за счёт увеличения количества запросов к серверу.

Для начала необходимо определить какую информацию мы будем хранить. Так как в приложении есть три основных логических объекта (пользователь, шаблон и задание), то и хранить придется только документы трёх типов, поэтому создадим в базе данных *Firestore* (Рисунок 5) три коллекции – *Users*, *Templates* и *Tasks*.



Сервис *Firebase Auth* обеспечивает создание записи пользователя в базе при его регистрации. Но, помимо его логина и зашифрованного пароля, также нужно хранить дополнительную информацию, поэтому рационально будет создать уникальный документ с информацией о данном пользователе в *Firestore*. Такой документ содержит в себе псевдоним и добавленные задания и шаблоны владельца аккаунта. При этом для упрощения дальнейшей работы и уменьшения веса документа, он будет хранить не сами задания и шаблоны, а только ссылки на их документы.

Документ шаблона должен также быть максимально легковесным, чтобы не тратить пользовательские ресурсы на загрузку данных, которые в отдельных случаях могут быть бесполезными. Поэтому его структура достаточно проста: название шаблона, уникальный номер владельца и список заданий, состоящий из ссылок на документы заданий. В дальнейшем планируется классифицировать шаблоны в зависимости от их содержимого и добавить соответствующее поле.

Структура файла задания наиболее простая, так как в них не содержится ссылок на дополнительные документы. Он содержит только: текст задания, его приоритет, время начала и длительность выполнения, координаты точки, к которой оно привязано и выполнено ли оно.

Такая структура базы данных позволяет всегда получать полный набор необходимых данных за максимально краткий промежуток времени.

# Создание алгоритма оптимизации времени

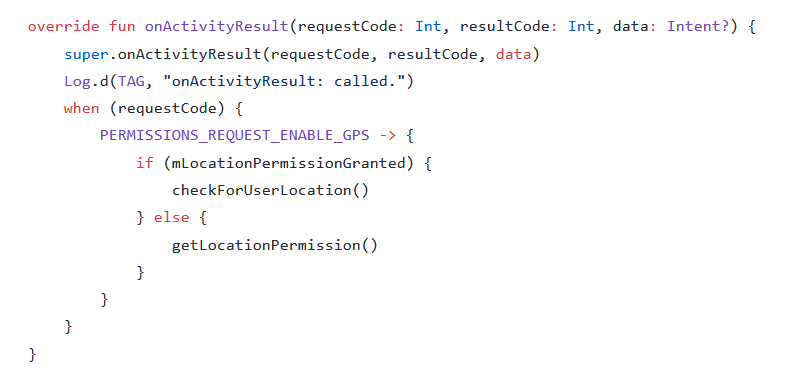
В основе алгоритма лежит концепция главенства более старшего приоритета над младшим. Приложение пожертвует младшими приоритетами в угоду старшему. Логической единицей, с которой работает приложение в данной версии, стал фрагмент маршрута. Происходит обработка списка заданий и деление его на фрагменты, где границами выступают задания с максимальными приоритетами. Первым делом мы проходим по списку заданий каждого из фрагментов и ищем конфликты. Если такие найдены, то фрагмент делится на правую и левую части, и они сравниваются по максимальному приоритету и их количеству. Если какая-либо из частей содержит в себе задание с большим приоритетом, то противоположная часть сдвигается на значение конфликта, а если обе части содержат задания с равным максимальным приоритетом, то части сдвигаются обратно пропорционально количеству таких заданий, в сумме по абсолютному значению сдвига получая значение конфликта.

После того, как мы полностью прошли по фрагменту и устранили все конфликты, проверяем не возник ли новый конфликт между крайним элементом и ограничителем, и если да, то начинаем последовательно сдвигать задания на значения конфликта от ограничителя, пока полностью не устраним все возможные конфликты. Однако такой способ применим только для ограниченных с одной стороны или вовсе неограниченных фрагментов. С ограниченными с двух сторон фрагментами после того, как приняты все возможные меры, проверяется присутствуют ли конфликты. В случае положительного ответа, пользователю будет предложено избавиться от задания с наименьшим приоритетом или изменить свое расписание.

# Техническая часть

Операционная система Android накладывает на приложения некоторые ограничения по использованию аппаратных ресурсов устройства. Например, перед тем как воспользоваться системой GPS нужно убедиться, что пользователь дал на это разрешение и включил геолокацию. Так как на странице с маршрутом знание местоположения устройства является необходимым, приходится сделать все возможное, чтобы получить доступ к нему и для этого нужно по максимуму упростить процесс одобрения для пользователя. В приведенном ниже листинге вы можете наблюдать три функции: *isServicesOK*, *isMapsEnabled*, *buildAlertMessageNoGps*.

Первая проверяет, что системные сервисы Google в норме, и в случае неполадок пытается предложить пользователю вариант решения. Вторая проверяет, включены ли на устройстве сервисы геолокации, если нет – вызывается функция *buildAlertMessageNoGps*. Та, в свою очередь, создает новое диалоговое окно, уведомляющее владельца устройства о необходимости включения геолокации. Оно сопровождается кнопкой, по нажатию на которую будет открыто системное окно настроек в необходимой вкладке. После выхода из этого окна будет вызван системный метод *onActivityResult*, который в случае наличия разрешения на использование системного сервиса начнёт отслеживать пользовательское местоположение, иначе вызовет функцию, открывающую соответствующее диалоговое окно.



# Вывод

В результате проделанной работы при помощи среды разработки Android Studio, языка программирования Kotlin было спроектировано и реализовано приложение-органайзер «Daily.Maps» с возможностями построения маршрутов по заданиям и оптимизацией времени.

В будущем проект будет совершенствоваться и портироваться на другие платформы.

Просмотреть исходный код проекта можно на сервисе GitHub– <https://github.com/x3noku/daily-maps-android/>. Скачать приложение можно по ссылке – <https://yadi.sk/d/e5Iqji1pXImAWw>

# Источники

1. Справка разработчика по *Android* // Developer Android URL: <https://developer.android.com/>
2. Справочник по языку *Kotlin* // kotlinlang.org URL: <https://kotlinlang.org/>
3. *Документация Firebase* // firebase.google.com URL: <https://firebase.google.com/docs/android/setup/>
4. Облачная платформа *Google* карт// cloud.google.com URL: <https://cloud.google.com/maps-platform/>
5. Документация *Mapbox* // docs.mapbox.com URL: <https://docs.mapbox.com/>
6. Сайт *Material Design* // material.io URL: <https://material.io/>
7. Главные принципы материального дизайна // Habr URL: <https://m.habr.com/ru/company/redmadrobot/blog/252773/>
8. Тайм-менеджмент и история органайзеров // Habr URL: <https://habr.com/ru/post/180277/>