# ralph

*Выпуск 7.1.1*

dadyarri

мая 22, 2020

# Оглавление

**Содержание**



Ralph - бот ВКонтакте, упрощающий рутинные задачи старост студенческих групп.

# Глава 1

## Функционал

Бот управляется с помощью встроенной клавиатуры ВКонтакте.

Для ее вызова в первый раз необходимо отправить сообщение с текстом «Начать»

- Призыв **[Нужен доступ Администратора]**:
  - Общий призыв (отправляет в беседу упоминание всех студентов)
  - Призыв выбранных студентов
- Создание рассылки по Потокам **[Нужен доступ Администратора]**:
  - Общий канал
  - Тестовый канал
  - Расписание
  - Обновления
- Каждым потоком можно управлять через меню «Управление рассылками»
- Пользовательские настройки **[Нужен доступ Администратора]**:
  - Смена беседы
  - Выбор формата призыва (с именами или без)
- Получение расписания:
  - на сегодня
  - на завтра
  - на послезавтра
  - на любую дату в формате ДД-ММ-ГГГГ
- Веб-интерфейс:
  - Получение ссылок авторизации в веб-интерфейсе бота

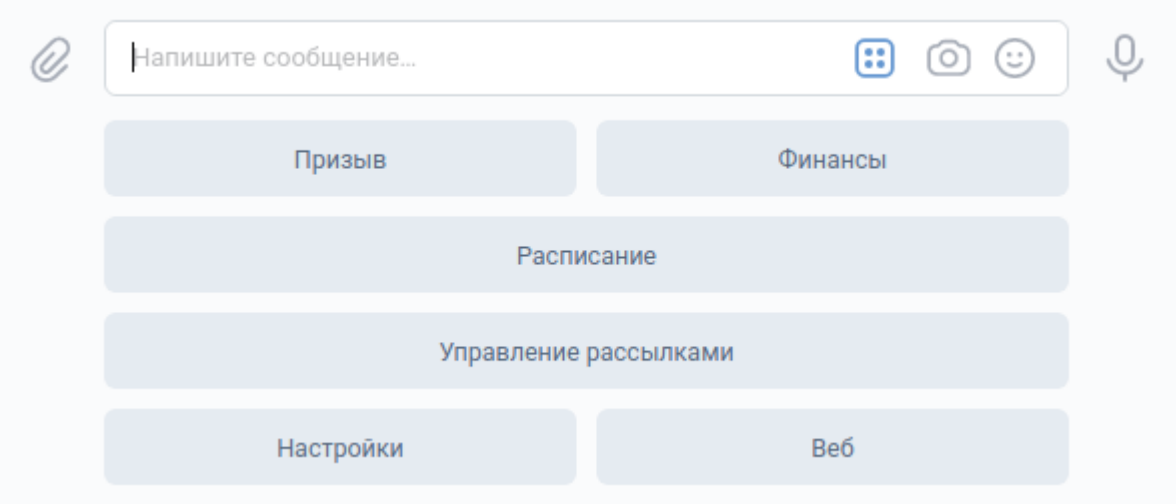
## 1.1 Руководство Администратора

**Содержание**

### 1.1.1 Начало работы

Для того, чтобы начать работу с ботом, нужно отправить боту, связанным с вашим учебным заведением (см. *Список подключенных учебных заведений*) сообщение с текстом «Начать» (то же действие выполняет «Старт»).

В ответ появляется клавиатура с главным меню. Она выглядит так:

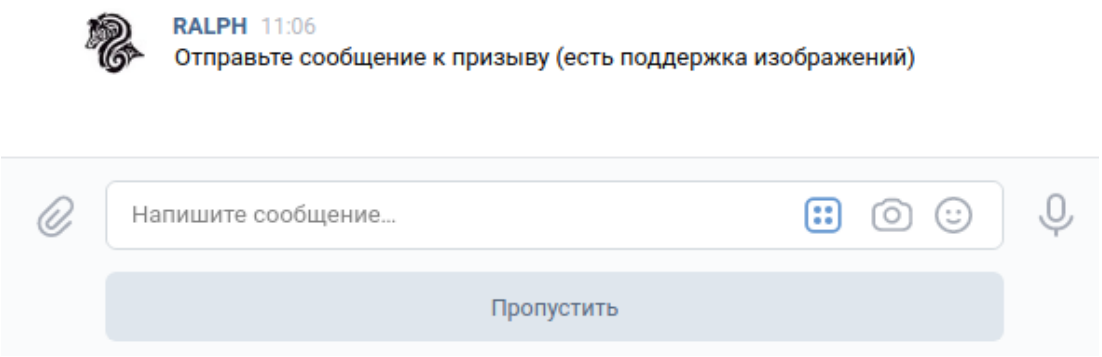



### 1.1.2 Описание функционала

#### 1.1.2.1 Призыв

*Позволяет отправить в беседу студентов упоминание (@id...) выбранных обучающихся или всех разом.*

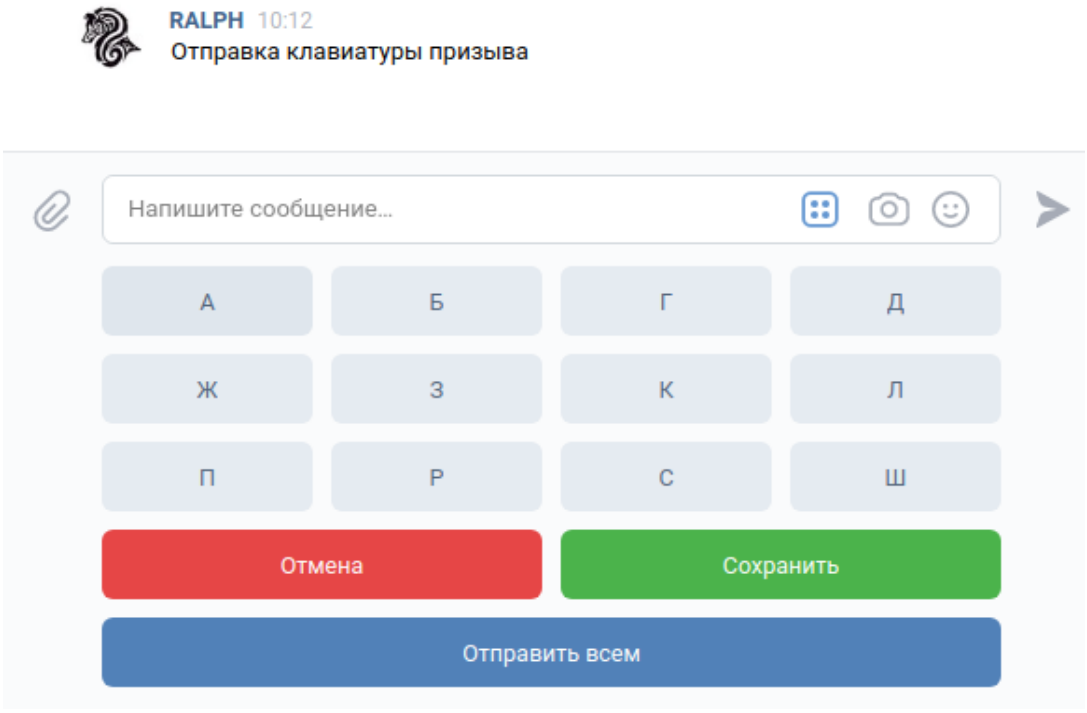
Чтобы отправить призыв, нужно выбрать в главном меню одноимённую кнопку и следовать дальнейшим указаниям:

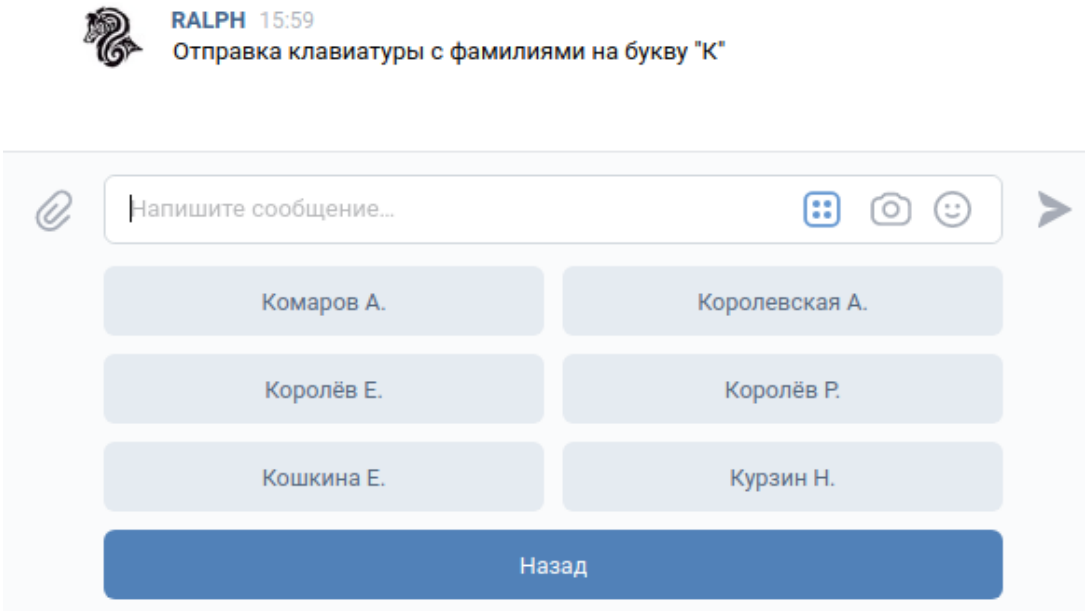



Сообщение к призыву можно не указывать.

Далее нужно выбрать получателей призыва (получат все участники беседы, но уведомление придёт только выбранным)

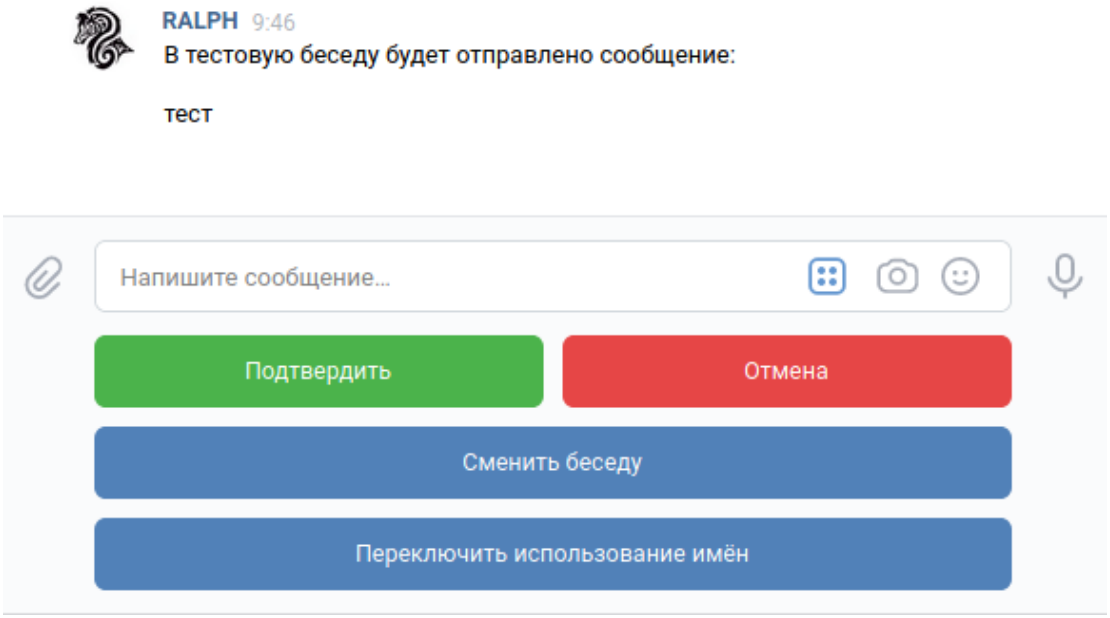
Каждая кнопка с буквой - это подменю, в котором собраны все фамилии, начинающиеся на эту букву. С помощью этих кнопок можно собрать список людей, которых нужно призвать.


RALPH 10:12
Отправка клавиатуры призыва
Напишите сообщение...
А
Б
Г
Д
Ж
З
К
Л
П
Р
С
Ш
Отмена
Сохранить
Отправить всем


RALPH 15:59
Отправка клавиатуры с фамилиями на букву "К"
Напишите сообщение...
Комаров А.
Королевская А.
Королёв Е.
Королёв Р.
Кошкина Е.
Курзин Н.
Назад

Когда список будет собран, нажмите на кнопку «Сохранить»

Вам будет показано сформированное сообщение с подтверждением отправки:




Нажатие «Подтвердить» отправляет сообщение в беседу; Нажатие «Отмена» возвращает в главное меню и очищает сохранённые данные; Нажатие «Сменить беседу» переключает активную беседу (тестовая/основная) и повторно спрашивает разрешения на отправку. Нажатие «Переключить использование имён» перегенерирует сообщение с изменёнными настройками использования имён

#### 1.1.2.2 Управление рассылками

В отличие от простых пользователей Администраторы способны также вручную отправлять рассылки любого типа.

При нажатии на кнопку «Управление рассылками» открывается подменю со списком доступных рассылок.

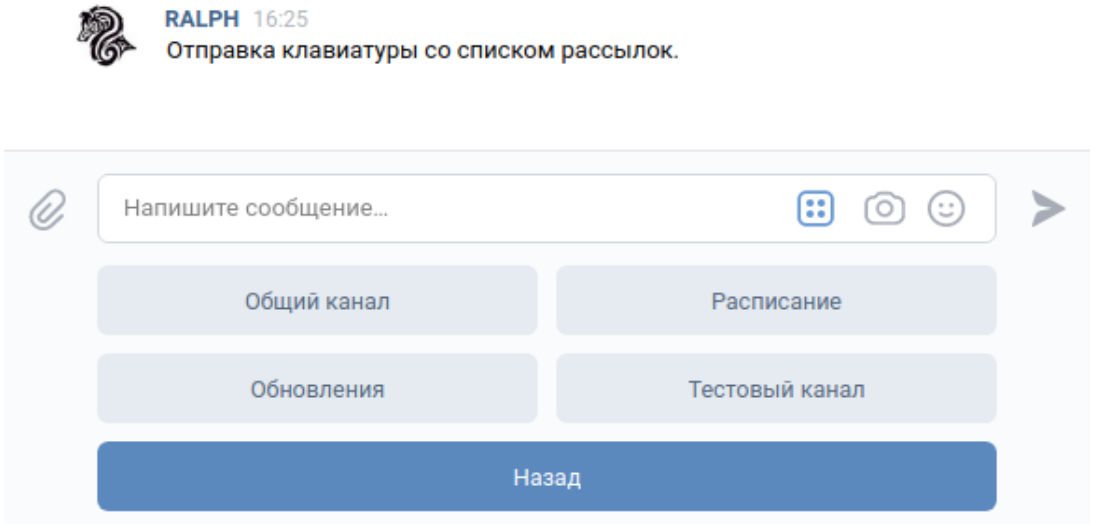
Каждая из этих рассылок это подменю, из которого доступно отправка рассылки или управление подписками:

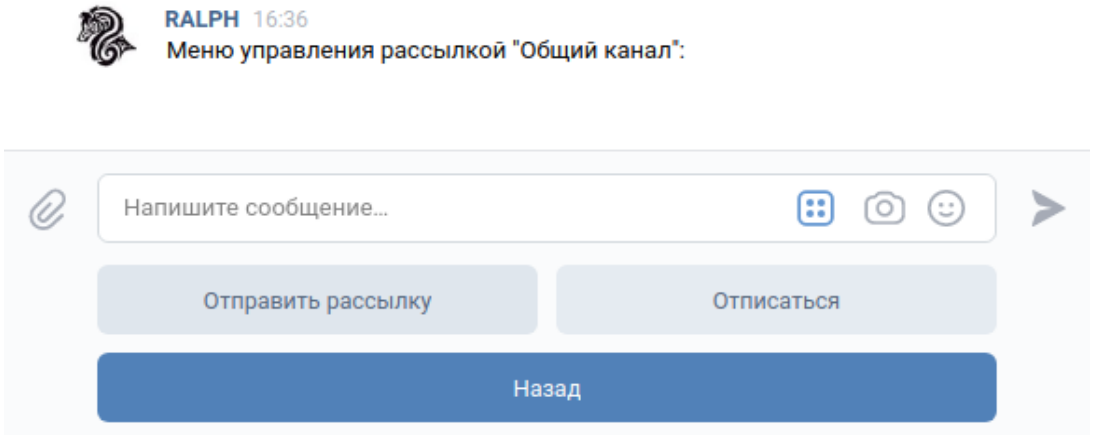
##### Отправка рассылок

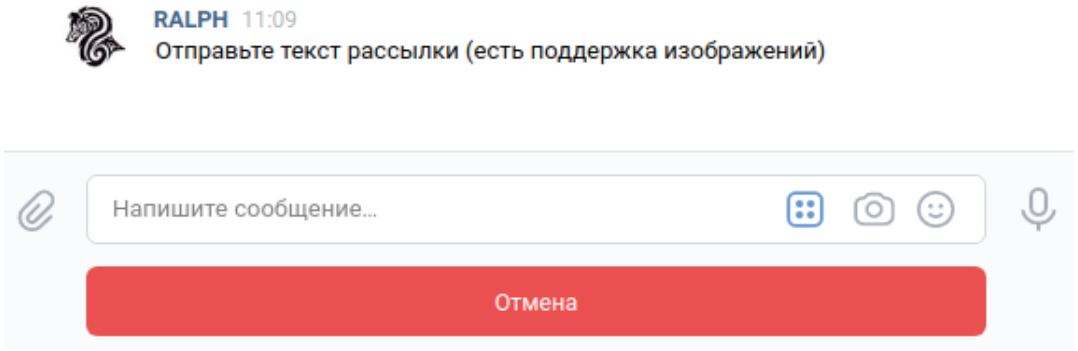
При нажатии этой кнопки вам будет предложено ввести текст рассылки:

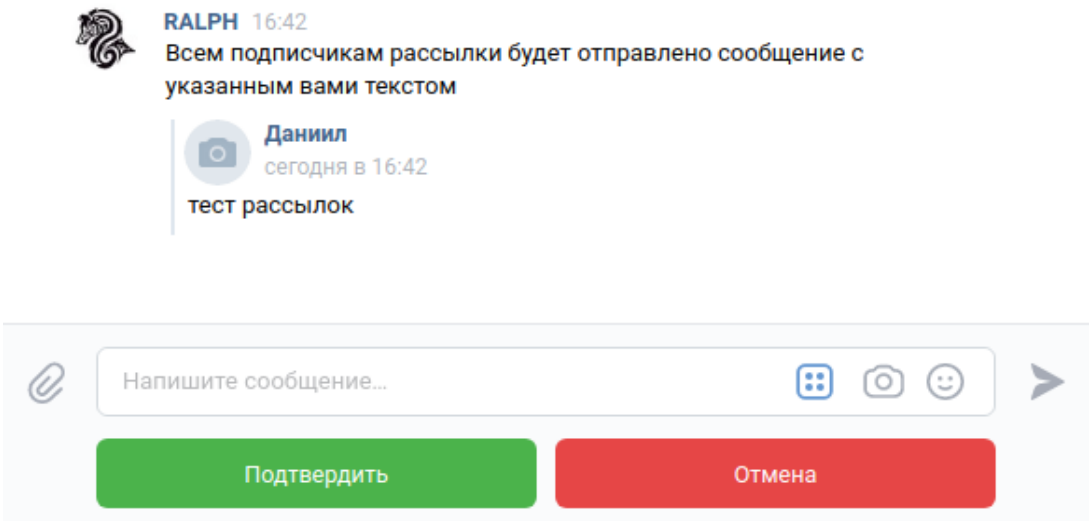
После сохранения текста вы сможете подтвердить отправку и сообщение автоматически улетит подписчикам выбранной вами рассылки:

Нажатие кнопки «Подтвердить» отправляет рассылку, кнопки «Отмена» удаляет сохраненные данные и возвращается на экран со списком клавиатур.


RALPH 16:25
Отправка клавиатуры со списком рассылок.
Напишите сообщение...
Общий канал
Расписание
Обновления
Тестовый канал
Назад


RALPH 16:36
Меню управления рассылкой "Общий канал":
Напишите сообщение...
Отправить рассылку
Отписаться
Назад


RALPH 11:09
Отправьте текст рассылки (есть поддержка изображений)
Напишите сообщение...
Отмена




#### Подписки

У каждого пользователя есть статус подписки на любую рассылку. По умолчанию подписки выглядят так:

- Общий канал: активно
- Расписание: активно
- Обновления: неактивно
- Тестовый канал: неактивно

Любой пользователь всегда может самостоятельно изменить свой список подписок.

### 1.1.2.3 Расписание

Нажатие кнопки «Расписание» открывает подменю с выбором даты для получения расписания.

Доступные варианты:

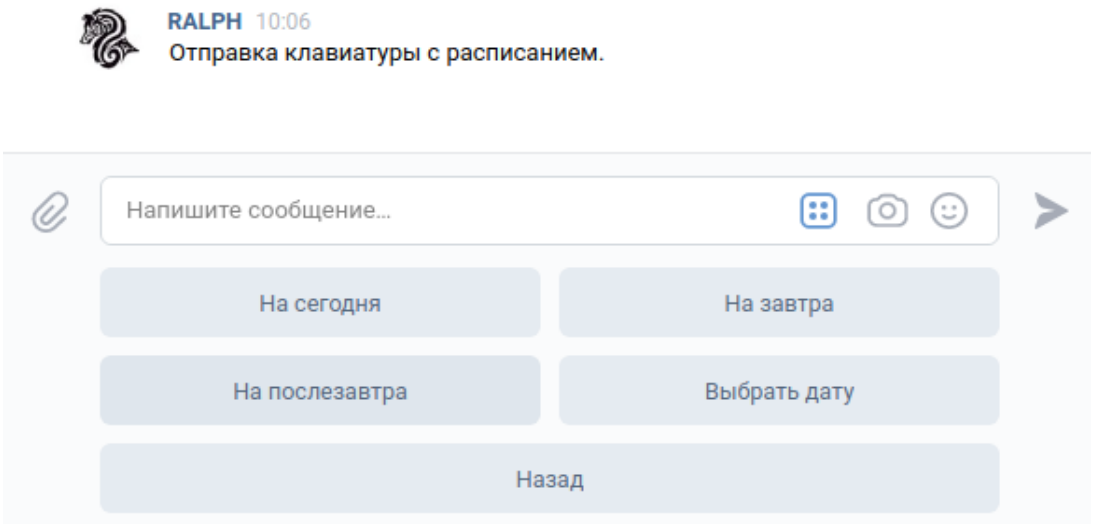
- на сегодня
- на завтра
- на послезавтра
- на любую дату (в формате ДД-ММ-ГГГГ)

При запросе расписания на любую дату бот спросит на какую дату нужно получить расписание. При этом встроена проверка на валидность

Пример **валидной даты**: 13-02-2020

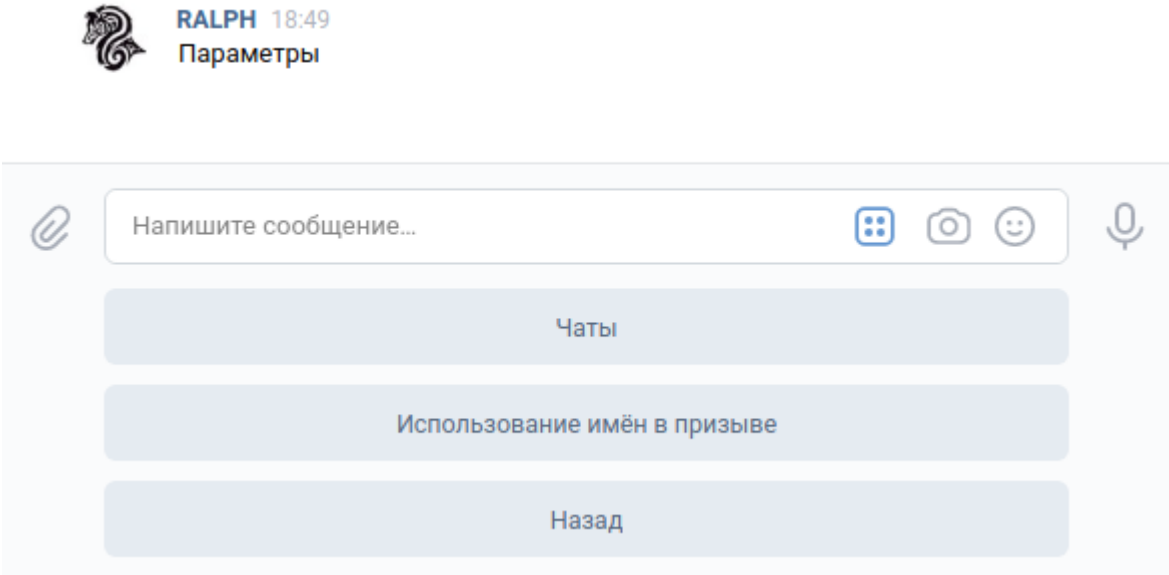
Примеры **невалидной даты**:

- 13.12.2020 (использованы точки вместо дефисов)
- 13/12/2020 (использованы слеши вместо дефисов)
- и т.д.



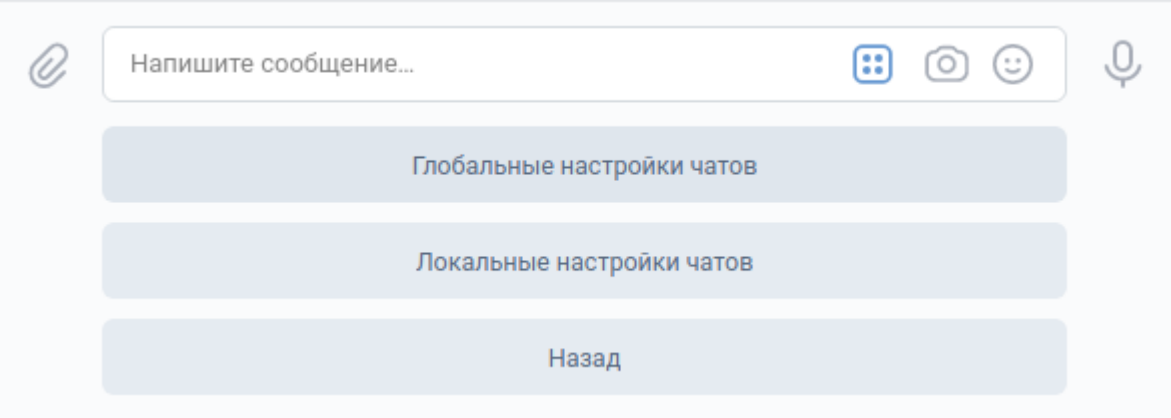

#### 1.1.2.4 Настройки

Это подменю с двумя опциями, которые можно изменить для себя (то есть глобальные параметры они не затрагивают)



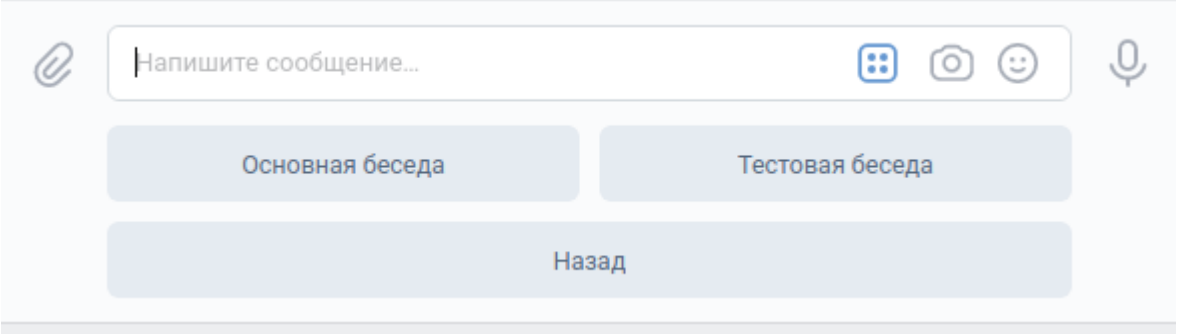
### Чаты

Это подменю в котором можно выбрать уровень настроек (глобальный - влияет на группу, локальный - на текущего администратора)




### Глобальные настройки

Перечисляет ассоциированные с группой пользователя чаты и позволяет их настроить




Когда в группе пользователя не настроены все чаты (тестовый/основной) и в кэше есть неразмеченные чаты, появляется кнопка «Зарегистрировать чат», которая позволяет присвоить чату тип (тестовый/основной)

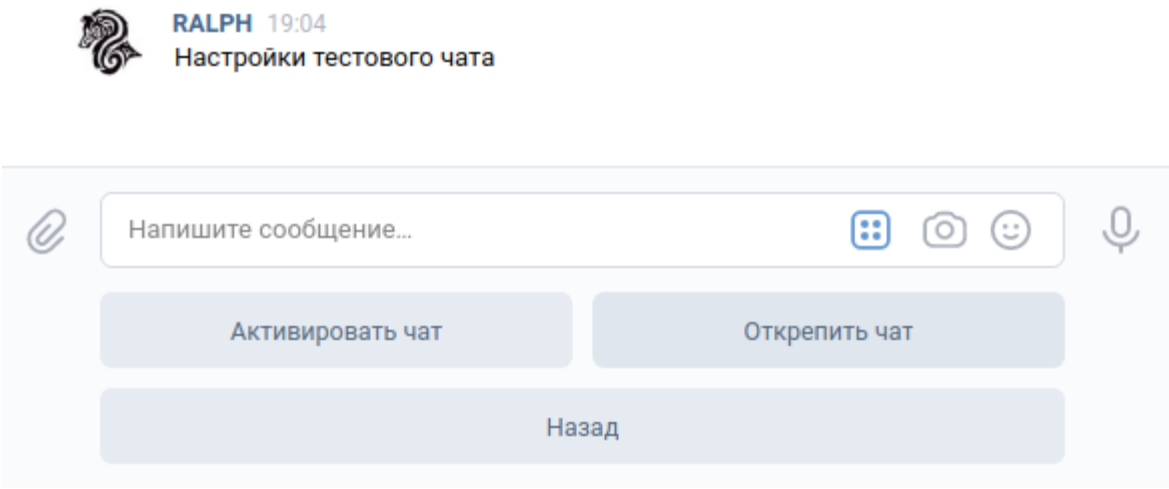
### Настройка чата

Каждый чат, прикрепленный к группе может быть выбран как активный (но только один одновременно)

В активный чат будет приходить автоматическая рассылка (с расписанием, например)

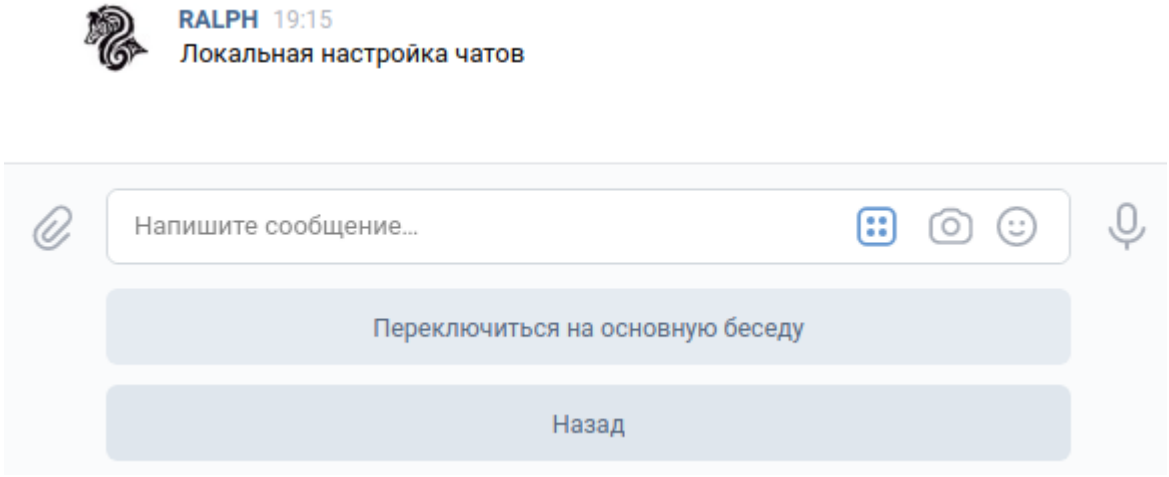
Когда вы активируете чат, другой автоматически становится неактивным

Открепить чат значит сообщить боту, что чат больше не будет использоваться по своему назначению. По факту это удаление. Разница лишь в том, что бот поместит этот чат в специальный кэш, куда попадают все неразмеченные чаты, в том числе, чаты, в которые бота только что пригласили и его можно будет назначить обратно.




### Локальные настройки

Позволяет выбрать в какой чат будут приходить Призывы от этого администратора. Не влияет на активность чата на уровне группы



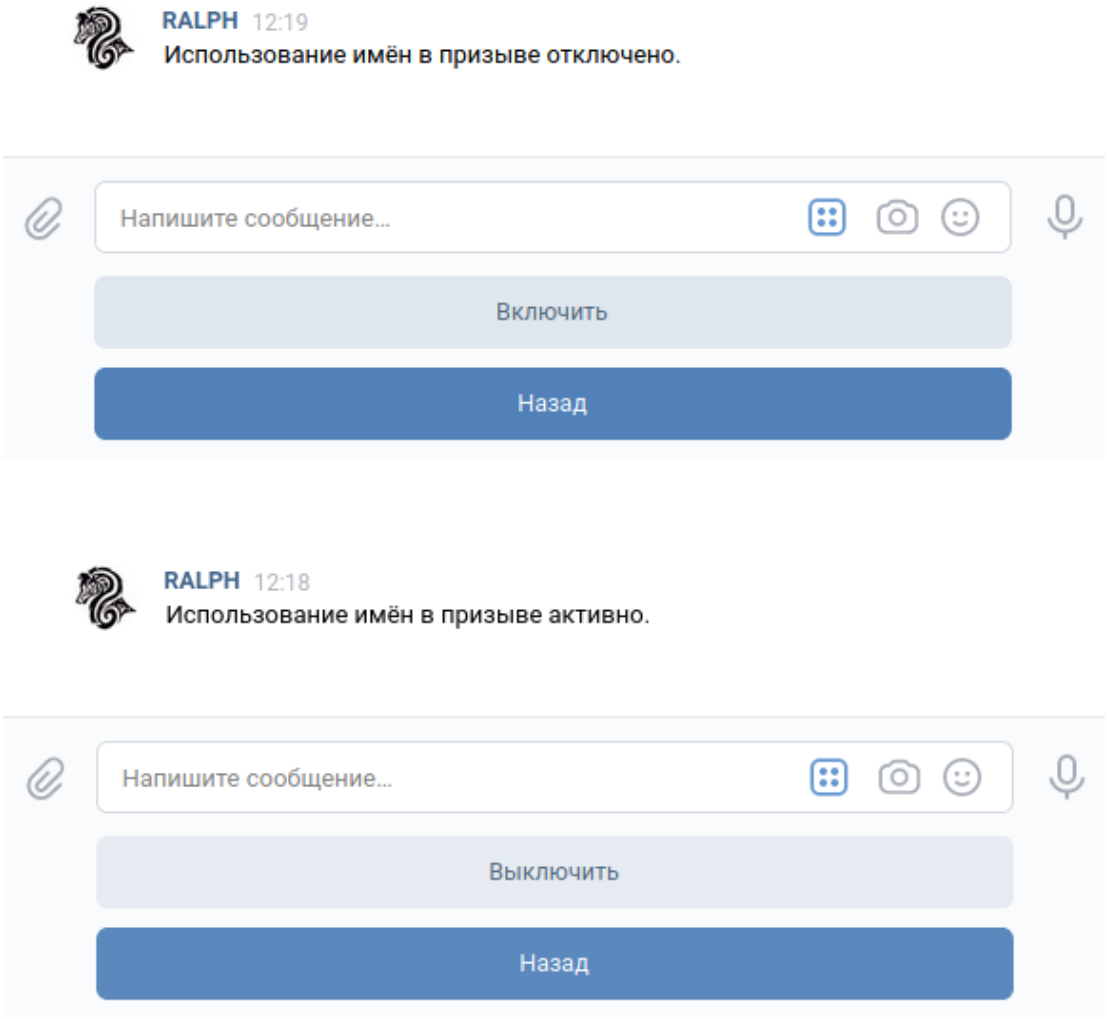

### Использование имён в призыве

При нажатии на кнопку «Использование имён в призыве» бот сообщит о текущем состоянии опции и предложит переключиться.

**На что это влияет**

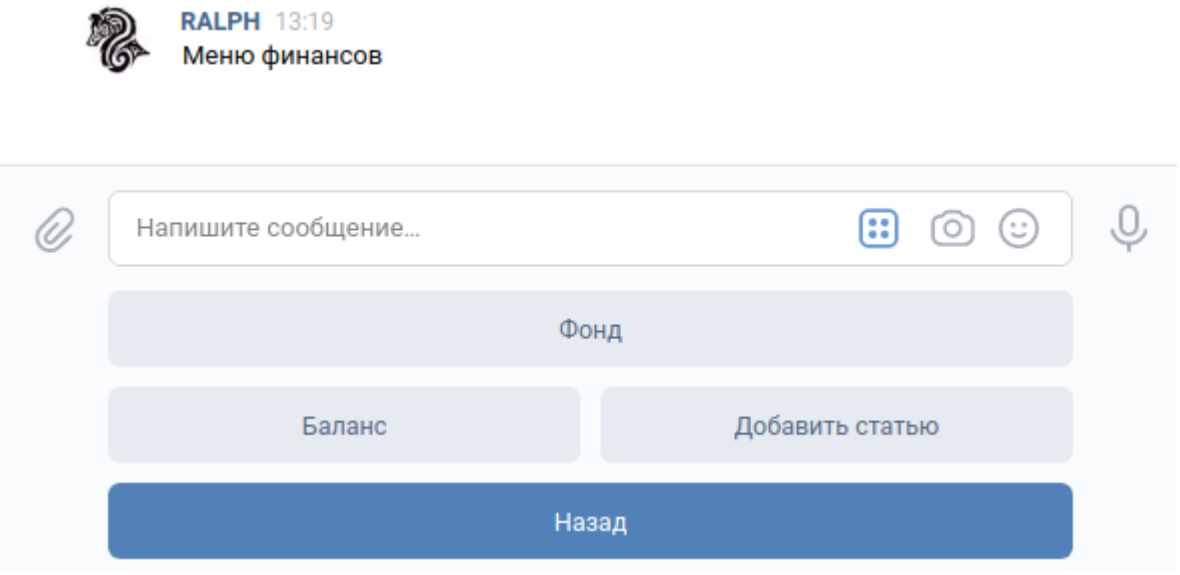
На формат сообщения призыва. Если опция выключена, призыв выглядит так:

Если включена, тогда так:


RALPH 12:19
Использование имён в призыве отключено.
Напишите сообщение...
Включить
Назад
RALPH 12:18
Использование имён в призыве активно.
Напишите сообщение...
Выключить
Назад
RALPH 12:59
В тестовую беседу будет отправлено сообщение:
!!!
вот так выглядит призыв, если опция использования имён неактивна
RALPH 13:02
В тестовую беседу будет отправлено сообщение:
Тимур, Серафим, Станислав
а так - если включена

#### 1.1.2.5 Финансы

Блок Финансов содержит набор функций, используемых для управления бюджетом группы. Её меню выглядит так:




**Важно:** Бот фактически не управляет бюджетом, а лишь вносит записи в базу данных. То есть положив деньги на карту, не стоит ожидать, что бот об этом узнает и внесет соответствующую запись, это нужно делать вручную через интерфейсы бота, о которых ниже.

##### Главное меню

##### Категории

Они появляются по мере создания новых категорий расходов. По нажатию на них открывается меню категории

### Баланс

С помощью этой кнопки можно получить состояние бюджета группы с учетом всех операций во всех категориях

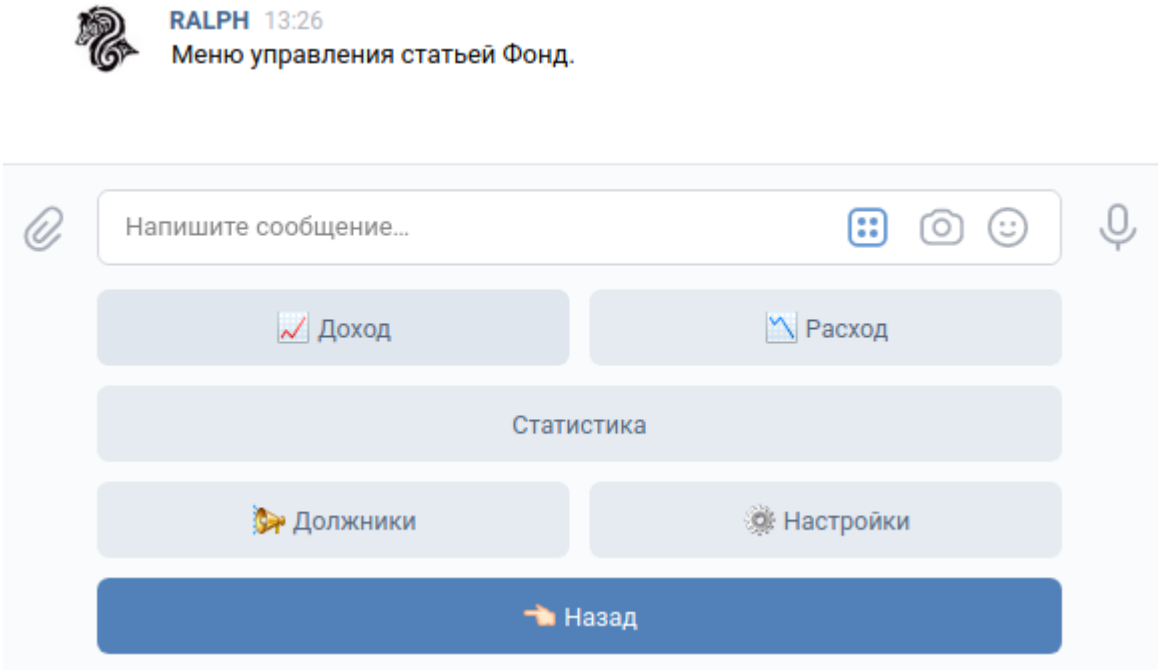
### Добавить статью

Здесь можно создать новую категорию расходов. Формат сообщения: <**имя**>,<**сумма сбора**>

Пробелы не важны. Главное - отделить сумму сбора от названия запятой. Иначе бот будет ругаться на неверный формат сообщения.

### Меню категории

Через меню категории расхода ведется все основное управление бюджетом.

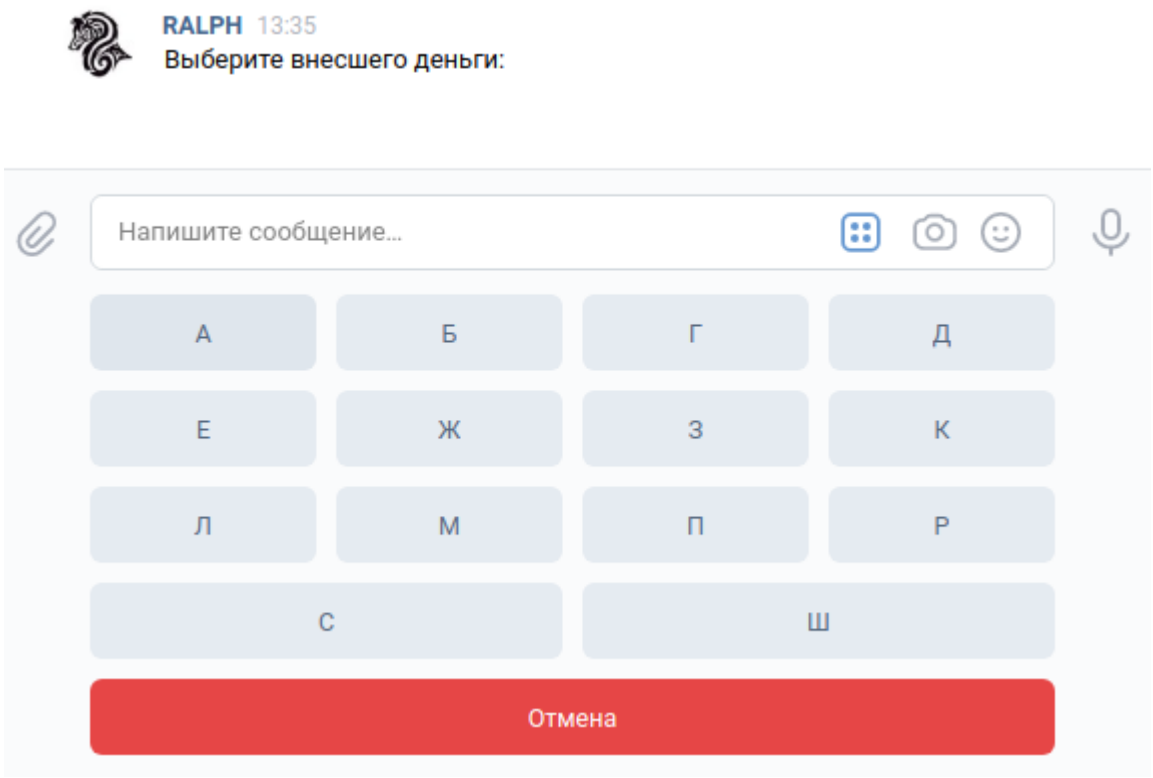



### Доход

Меню «Доход» создает запись о внесении средств на счет группы. Для этого нужно будет выбрать человека, внесшего деньги и сумму взноса. Для выбора человека предоставляется похожая менюшка, как и в меню «Призыв». Разница лишь в том, что после выбора человека подтверждать ничего не нужно, автоматически производится переход на следующий этап.

Далее нужно будет отправить сообщение с суммой взноса, которое должно содержать только число. В противном случае, бот откажется принимать значение и будет ожидать сообщения в нормальной форме.

После этого запись о взносе будет создана и пользователя перенесет в меню категории расхода




### Расход

Меню «Расход» создает запись о тратах в рамках категории. Нужно просто указать сумму расходов.

### Статистика

Меню «Статистика» вычисляет некоторые данные о сборе в рамках категории.

Эта информация выводится в диалог:

- Всего сдали (количество человек)
- Всего не сдали (количество человек)
- Всего собрано (сумма)

### Должники

Меню «Должники» формирует в беседу призыв людей, не сдавших деньги на определенную статью. Процесс полностью автоматизирован, от пользователя требуется только подтвердить отправку и изменить параметры, если это необходимо. Параметры те же, что и у Призыва:

**См.также:**

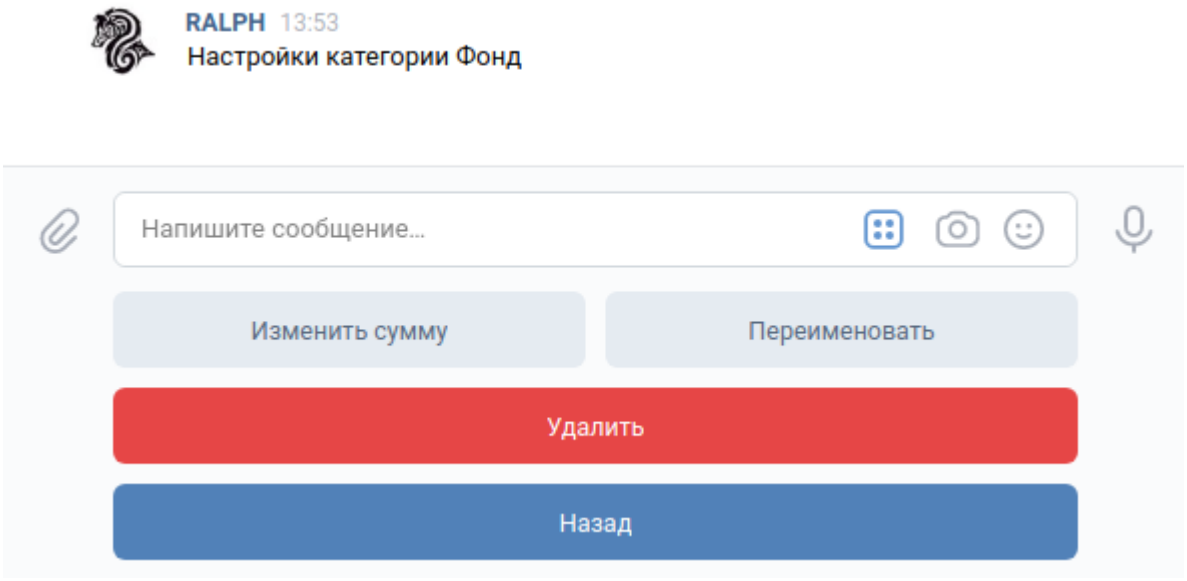
*Настройки*

#### Настройки категории

Меню «Настройки» позволяет сконфигурировать уже существующую категорию расходов

Доступны следующие параметры:

- Переименование
- Изменение суммы сбора
- Удаление




**Предупреждение:** Удаление категории ведет к удалению всех связанных записей (как расходов, так и доходов)

#### 1.1.2.6 Веб

Меню Веб показывает список групп, доступных пользователю для администрирования.

При переходе на одну из таких кнопок генерируется ссылка авторизации для входа под аккаунтом выбранной группы в веб-интерфейс.

Ссылка одноразовая и работает в течение 5 минут.

**Важно:** Поскольку ВК не дает пользователям выбора, на какие сайты переходить можно, а на какие нет, то без средств обхода vk.com/away.php попасть в веб-интерфейсы невозможно

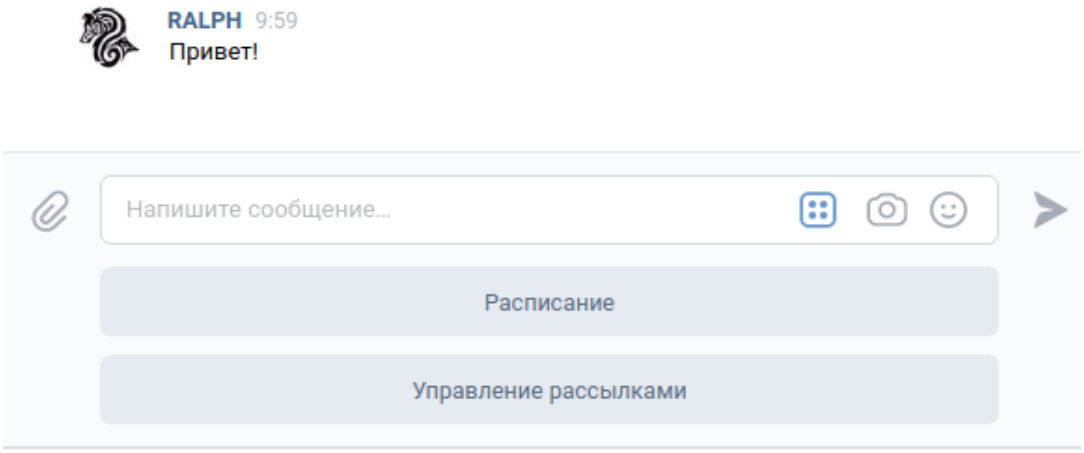
## 1.2 Руководство пользователя

**Содержание**



### 1.2.1 Начало работы

Для того, чтобы начать работу с ботом, нужно отправить ему сообщение с текстом «Начать» (то же действие выполняет «Старт»). В ответ появляется клавиатура с главным меню. Она выглядит так:




### 1.2.2 Описание функционала

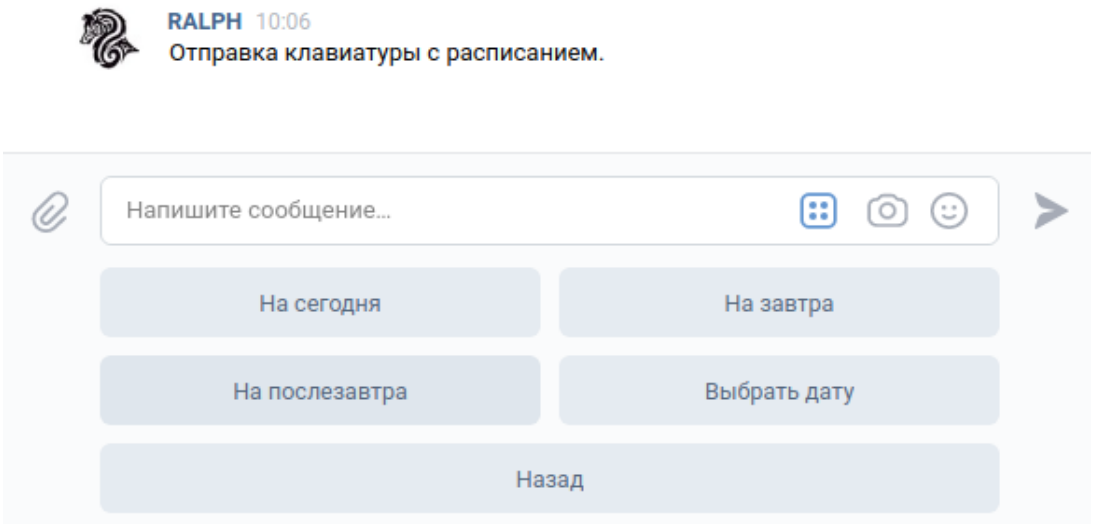
#### 1.2.2.1 Расписание

Нажатие кнопки «Расписание» открывает подменю с выбором даты для получения расписания.

Доступные варианты:

- на сегодня
- на завтра
- на послезавтра
- на любую дату (в формате ДД-ММ-ГГГГ)

При запросе расписания на любую дату бот спросит на какую дату нужно получить расписание. При этом встроена проверка на валидность




Пример **валидной даты**: 13-02-2020

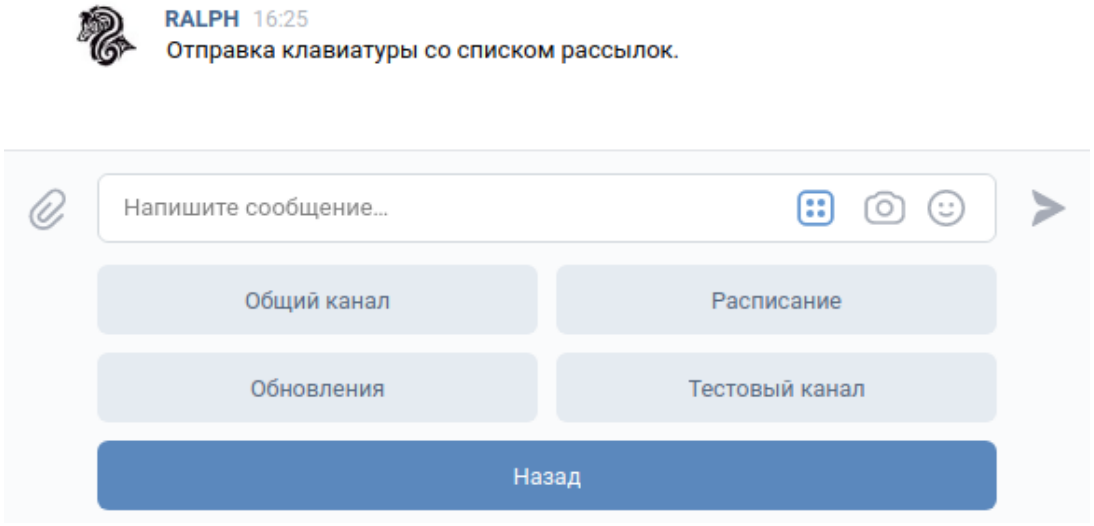
Примеры **невалидной даты**:

- 13.12.2020 (использованы точки вместо дефисов)
- 13/12/2020 (использованы слеши вместо дефисов)
- и т.д.

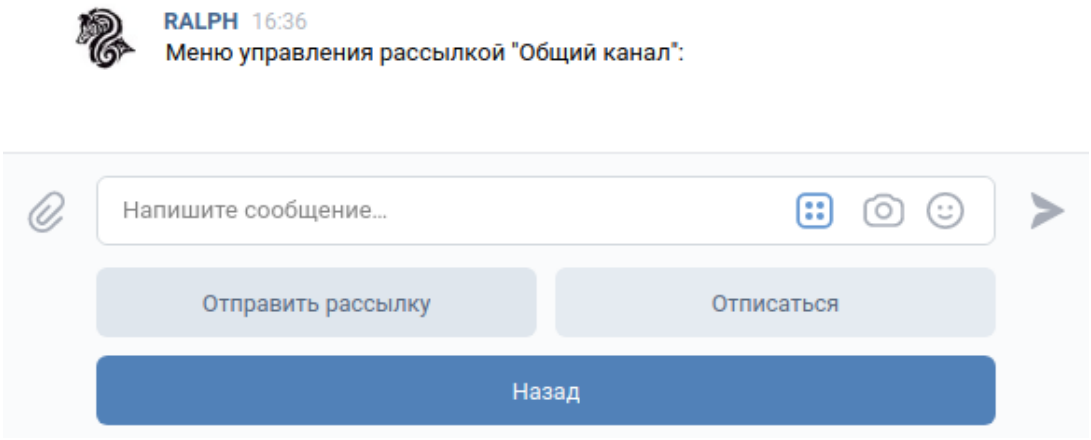
#### 1.2.2.2 Управление рассылками

В отличие от простых пользователей Администраторы способны также вручную отправлять рассылки любого типа.

При нажатии на кнопку «Управление рассылками» открывается подменю со списком доступных рассылок.



Каждая из этих рассылок это подменю, из которого доступно отправка рассылки или управление подписками:




#### Подписки

У каждого пользователя есть статус подписки на любую рассылку. По умолчанию подписки выглядят так:

- Общий канал: активно
- Расписание: активно
- Обновления: неактивно
- Тестовый канал: неактивно

Любой пользователь всегда может самостоятельно изменить свой список подписок.

## 1.3 Руководство по внесению вклада в разработку

**Содержание**

### 1.3.1 Создание ишью

Если вы нашли баг, косяк, недоработку, или хотели бы предложить новую функцию, вам нужно открыть ишью. (Вам понадобится аккаунт на Github, если его еще нет, зарегистрируйтесь здесь). Для создания ишью зайдите на специальную страницу в репозитории, ознакомьтесь с существующими ишью, и если похожей на вашу проблему, еще не описано нажмите на кнопку **New**.

Выберите подходящий шаблон.

- «Отчёт о баге», если нашли проблему
- «Запрос функции», если хотите предложить новую функцию или улучшение существующей функции

Укажите название проблемы/новой функции (её краткое описание в поле *Title*), и максимально подробное описание в поле *Leave a comment*

Следуйте приведённому шаблону, это поможет быстрее решить проблему.

Вы можете оставить фотографии, скриншоты (для багов это особенно важно), поле комментария поддерживает разметку Markdown.`Шпаргалка по Markdown <https://github.com/sandino/Markdown-Cheatsheet>`_

Описание бага должно содержать шаги для его повторения, чтобы разработчик мог убедиться в наличии бага и провести его поиск в коде.

В ближайшее время разработчик изучит вашу проблему и займется её решением.

### 1.3.2 Локальная разработка

Чтобы получить свежую копию репозитория вам нужно форкнуть репозиторий в свой аккаунт на GitHub (используя кнопку Fork), затем уже свою копию репозитория склонировать на рабочую машину

```
git clone https://github.com/<ваш_ник>/ralph
```

**Предупреждение:** Делая свои изменения вы должны прислушиваться к соглашениям, перечисленным в разделе *Соглашения*, в противном случае, ваш пул реквест скорее всего, будет отклонён.

Завершив работу над вашими улучшениями нужно выгрузить репозиторий на удаленный сервер

```
git push origin master
```

Затем перейдя в основной репозиторий нужно открыть пулреквест (запрос на слияние)

Если вы недавно выгрузили свои изменения на Github вы должны увидеть вверху страницы с репозиторием предложение пулреквеста

Нажмите на кнопку *Open pull request*, укажите название и описание пулреквеста и подтвердите создание запроса.

### 1.3.3 Соглашения

#### 1.3.3.1 Работа с ветками

Изменяя репозиторий, вы не должны работать с веткой *master*. Вам нужно создать свою ветку с именем, которое будет зависеть от того, что предстоит сделать:

- Фикс ишью: *<ваш_ник>/<номер_версии_в_которую_войдет_фикс>/fix-#<номер_ишью>*.
- Новая фишка: *<ваш_ник>/<номер_версии_в_которую_войдет_фикс>/<краткое_описание_фишки>*

Номер разрабатываемой версии можно спросить у мэйнтэйнера репозитория в Telegram (самый надёжный способ). Туда же можно задавать любые вопросы по коду, тестированию и пр.

Таких веток может быть несколько. Каждую из них нужно создавать из мастера.

Затем, когда работа с веткой будет завершена, нужно из мастера создать новую ветку с именем *wip/<номер_версии>*, в неё слить все ваши ветки, выгрузить её на GitHub, и создать запрос на слияние

#### 1.3.3.2 Наименование коммитов

1. По возможности коммит должен содержать изменения только одного файла (если это не рефактор, например).
2. Имя коммита должно быть на английском, начинаться с заглавной буквы, заканчиваться точкой и содержать краткое описание изменений, сделанных в этом коммите
3. Имя коммита не должно содержать глаголов в страдательном залоге (вместо *Added* нужно использовать *Add*, например).
4. Если коммит решает проблему, указанную в [issue](https://github.com/dadyarri/ralph/issues), нужно упомянуть в имени коммита номер этого ишью через знак решетки (например: *Fix issue #100 (<краткое описание ишью>).*)
5. В каждом коммите должно быть одно изменение, которое можно трактовать коротко и однозначно. Не допускаются коммиты, изменяющие 20 файлов с именем *«A lot of fixes.»*

## 1.4 Руководство по адаптированию

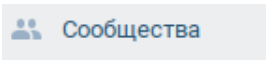
### 1.4.1 Подключение нового учебного заведения

**Содержание**

- *Настройка виртуального окружения*
- *Настройка базы данных*
- *Модификация кода*
- *Выгрузка на хостинг*
- *Загрузка исходного кода веб-интерфейса*
- *Выгрузка веб-интерфейса на сервер*

#### 1.4.1.1 Создание сообщества

Ботов ВКонтакте можно привязывать только к группам. Для её создания перейдите в раздел сообществ из бокового меню:




Там нажмите на кнопку «Создать сообщество»


В открывшемся меню выберите «Группа по интересам»

На ваше усмотрение заполните данные группы

Перейдите в раздел «Управление» нового сообщества

Затем в подраздел «Работа с API»

Здесь нужно создать токен сообщества со следующими правами:

*управление сообществом, сообщения сообщества, фотографии, документы, стена*

Скопируйте себе этот токен, в дальнейшем он нам понадобится.

**Предупреждение:** Никому не передавайте этот токен! Он позволяет управлять вашим сообществом

Перейдите во вкладку Long Poll API и включите его. Выберите версию API 5.103

Затем во вкладке типы событий установите все галочки

Далее, в разделе «Сообщения» нужно включить сообщения сообщества.

По желанию можно настроить приветствие. Это сообщение, которое получит пользователь, впервые открывший диалог с ботом.

Нажмите «Сохранить».

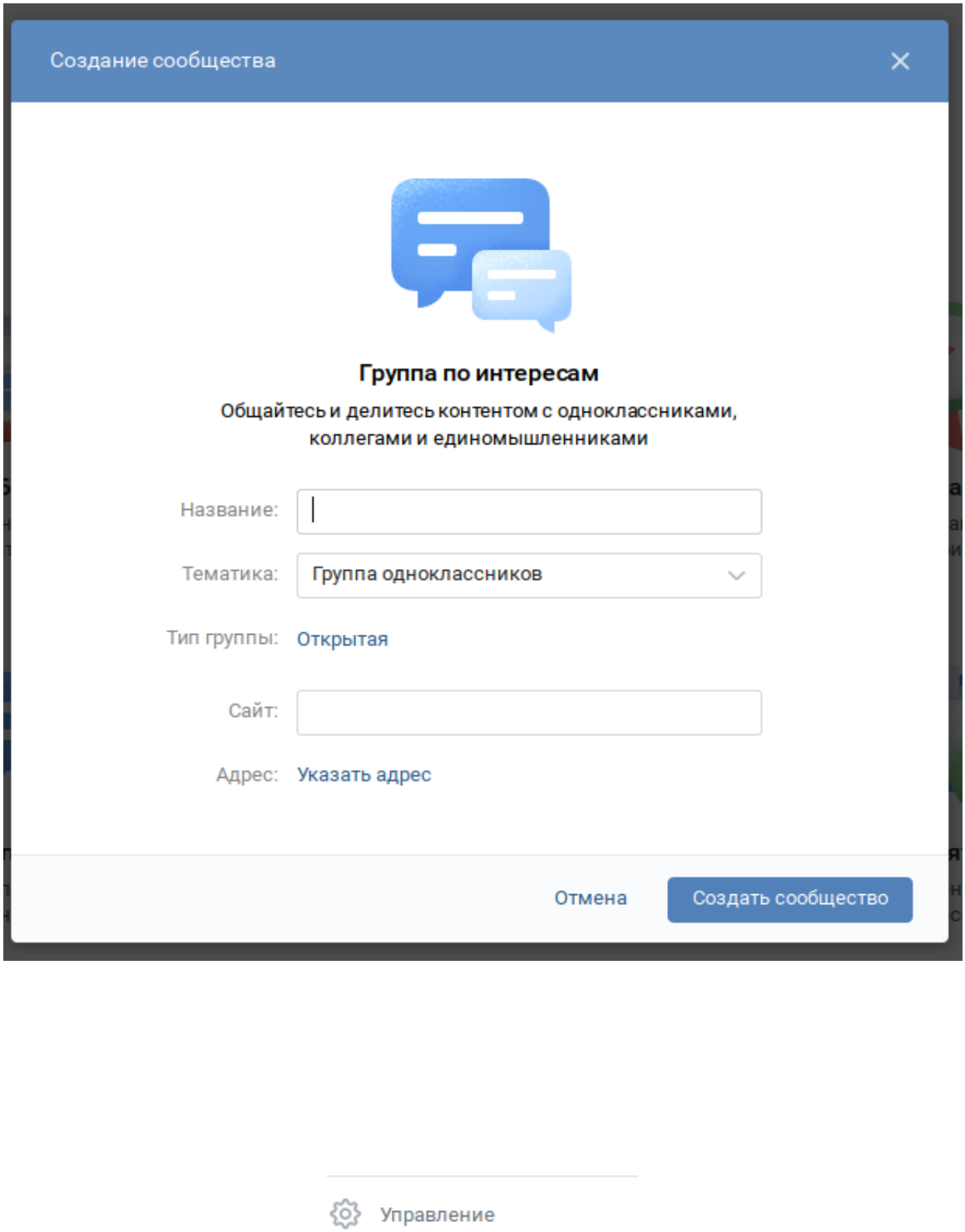
Затем, в подразделе «Настройки для бота» включите «Возможности ботов» и выберите обе галочки.

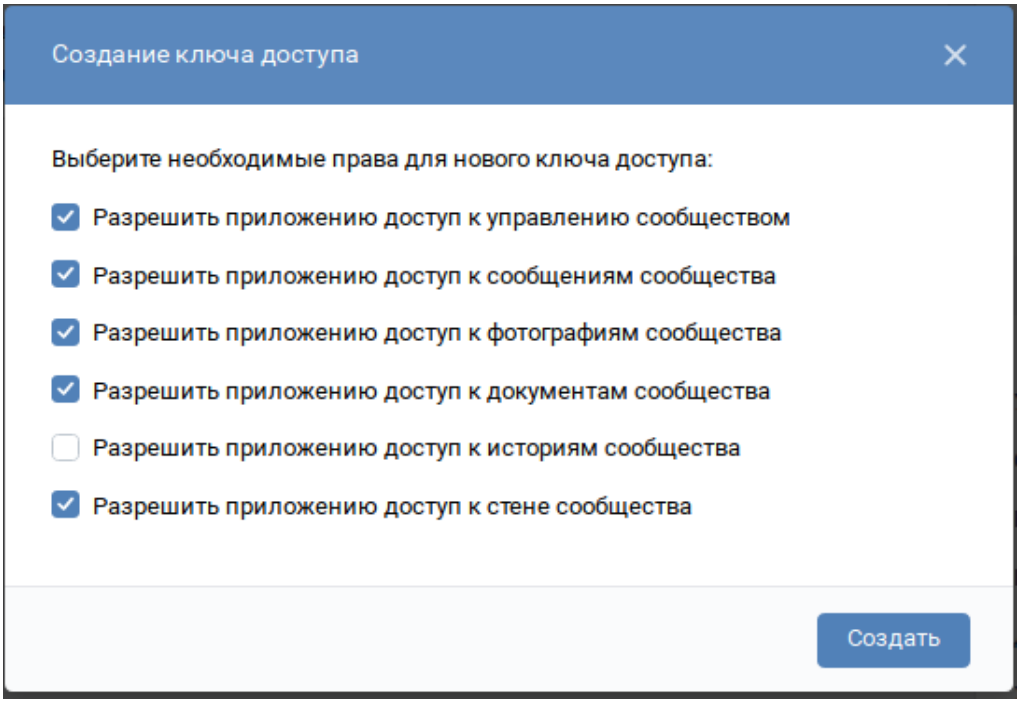
Снова нажмите «Сохранить»

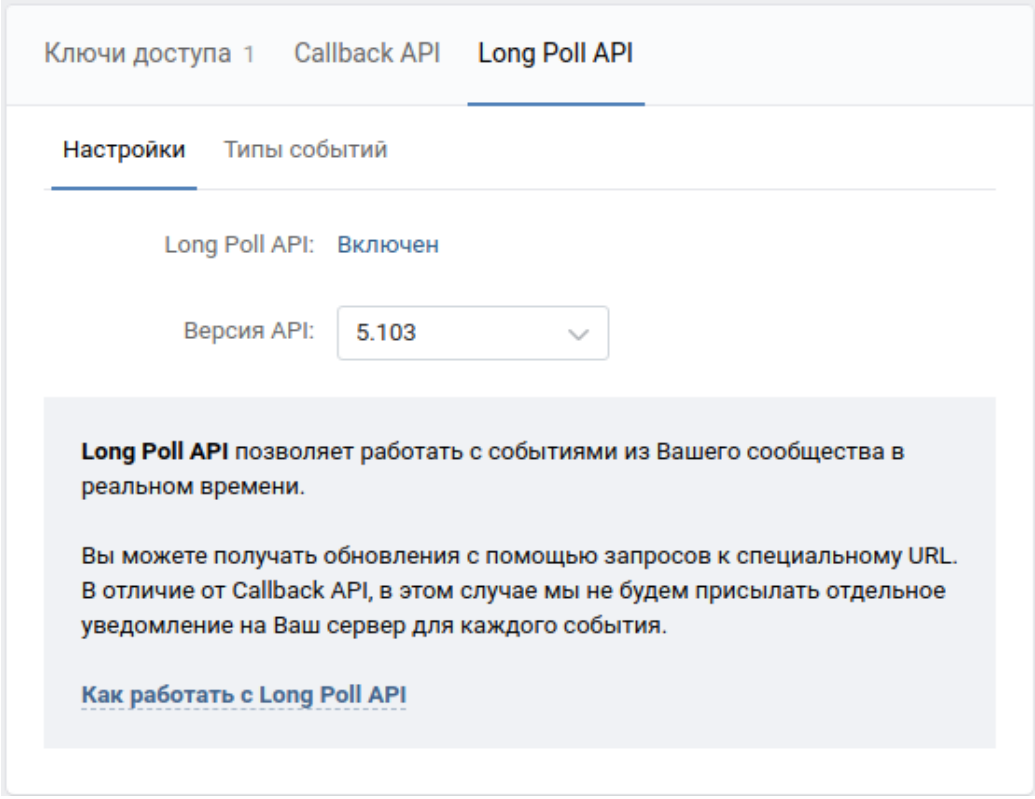
Создайте отдельную беседу для тестирования и обучения администраторов.

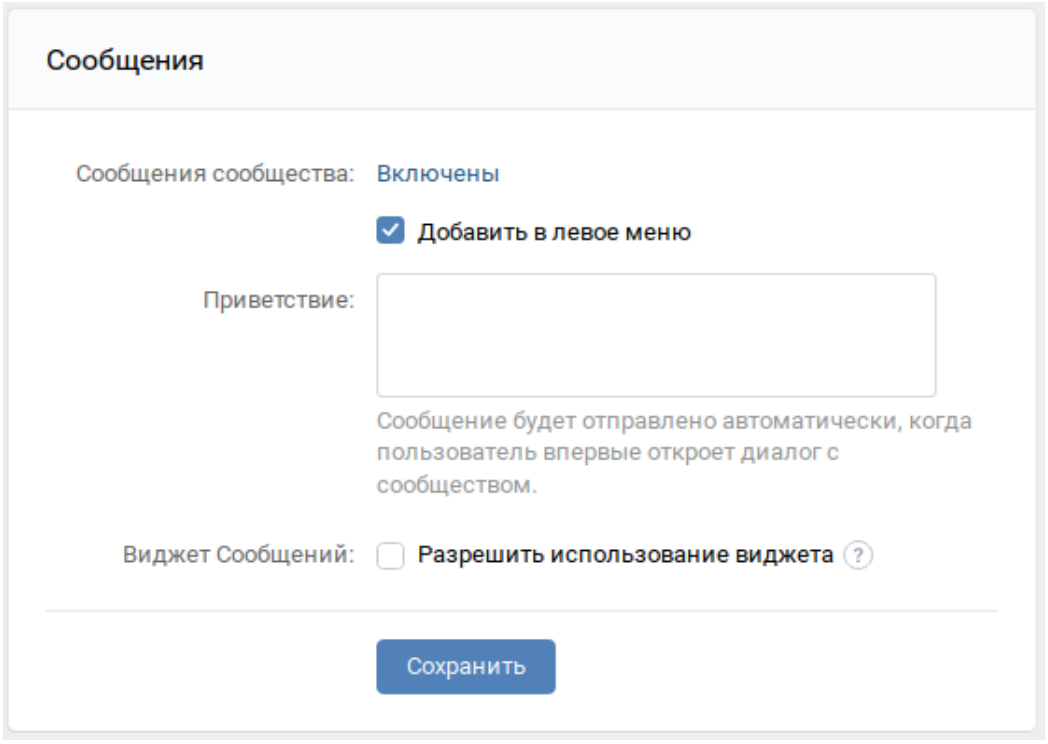
Вернитесь к сообществу. У вас появится меню с одной кнопкой - «Добавить в беседу».

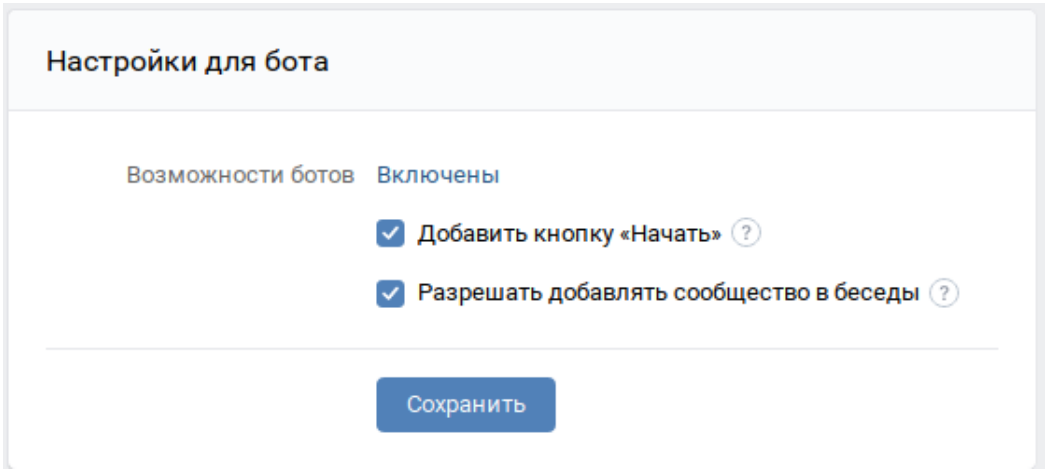

Моя страница
Мои сообщества
Сообщество ВКонтакте
Публикуйте материалы разных форматов, общайтесь с читателями, изучайте статистику и подключайте монетизацию. Для начала выберите тип сообщества.
Бизнес
Кафе, магазин, фитнес-клуб, банк, кинотеатр, мастерская
Тематическое сообщество
Новости и афиши, развлечения, тематические блоги и СМИ
Бренд или организация
Товар, фильм, компания, учебное заведение, благотворительный фонд
Группа по интересам
Учебная группа, тайное общество, объединение по интересам
Публичная страница
Музыкальный коллектив, общественное движение, блогер, спортивная команда
Мероприятие
Концерт, день рождения, выставка, вечеринка, мастер-класс, конференция

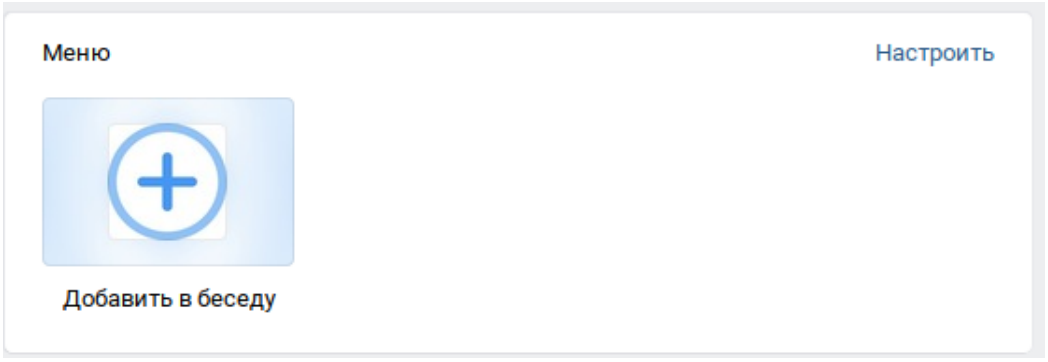

Создание сообщества
Группа по интересам
Общайтесь и делитесь контентом с одноклассниками, коллегами и единомышленниками
Название:
Тематика:
Группа одноклассников
Тип группы:
Открытая
Сайт:
Адрес:
Указать адрес
Отмена
Создать сообщество
Управление


Создание ключа доступа
Выберите необходимые права для нового ключа доступа:
Разрешить приложению доступ к управлению сообществом
Разрешить приложению доступ к сообщениям сообщества
Разрешить приложению доступ к фотографиям сообщества
Разрешить приложению доступ к документам сообщества
Разрешить приложению доступ к историям сообщества
Разрешить приложению доступ к стене сообщества
Создать


Ключи доступа 1
Callback API
Long Poll API
Настройки
Типы событий
Long Poll API: Включен
Версия API: 5.103
Long Poll API позволяет работать с событиями из Вашего сообщества в реальном времени.
Вы можете получать обновления с помощью запросов к специальному URL. В отличие от Callback API, в этом случае мы не будем присылать отдельное уведомление на Ваш сервер для каждого события.
Как работать с Long Poll API


Сообщения
Сообщения сообщества:
Включены
Добавить в левое меню
Приветствие:
Сообщение будет отправлено автоматически, когда пользователь впервые откроет диалог с сообществом.
Виджет Сообщений:
Разрешить использование виджета
Сохранить


Настройки для бота
Возможности ботов
Включены
Добавить кнопку «Начать»
Разрешать добавлять сообщество в беседы
Сохранить


Меню
Настроить
Добавить в беседу

Нажмите на неё и выберите обе беседы, в которых должен работать бот.

Вернитесь к беседам. Откройте список участников и найдите среди них своего бота. Сделайте его администратором беседы, чтобы дать ему доступ на написание сообщений в этой беседе. (это может сделать только действующий администратор или создатель)

Это действие нужно повторить с обоими беседами.

#### 1.4.1.2 Получение токена пользователя

Токен пользователя необходми для автоматической смены статуса группы с номером версии.

**См.также:**

Документация ВКонтакте о получении токена пользователя https://vk.com/dev/implicit_flow_user

#### 1.4.1.3 Настройка компьютера для работы

1. Зарегистрируйтесь на Github, если у вас еще нет там аккаунта (он понадобится для получения уведомлений о выходе новых версий)
2. Перейдите на страницу с репозиторием нажмите кнопку Watch, чтобы получать уведомления об обновлениях на электропочту. Выберите вариант Releases only, если не собираетесь участвовать в развитии основной ветки проекта
3. Затем нажмите кнопку Fork. Репозиторий скопируется в ваш аккаунт. Далее вы будете работать из своей копии.
4. Установите Git на свой компьютер.

   Для Windows используйте Git For Windows

   В Linux используйте стандартный менеджер пакетов

   ```
   sudo pacman -S git # В Arch
   ```

   или

   ```
   sudo apt install git # В Ubuntu
   ```

5. Настройте Git

   ```
   git config --global user.name ваше_имя

   git config --global user.email ваша_электропочта_с_github
   ```

6. Установите Python, если он еще не установлен

   (В установщике для Windows важно указать галочку Add Python to PATH)

7. Установите PostgreSQL

   Для Windows

   Для Linux:

   ```
   sudo apt install postgresql postgresql-contrib # Ubuntu
   ```

   ```
   sudo pacman -S postgresql # Arch
   ```

#### 1.4.1.4 Получение исходного кода бота

1. Создайте папку в которой будет хранится исходный код бота. Перейдите в нее из командной строки.
2. Загрузите последнюю доступную версию бота:

```
git clone https://github.com/ваш_ник/ralph

cd ralph
```

#### 1.4.1.5 Настройка переменных окружения

Переменные окружения - это особый файл с секретными данными, которые нельзя нигде публиковать.

В случае с Ральфом там хранится токен сообщества, токен администратора, URL доступа к базе данных, и для удобства - настройки модуля логгирования, идентификатор сообщества с ботом.

Создайте в папке с исходным кодом файл **.env** (Именно начинающийся с точки. В Windows могут возникнуть проблемы с этим, тогда создайте файл из IDE)

Вот готовый шаблон файла с переменными окружения:

```
DATABASE_URL="<ссылка доступа к базе данных. Ниже будет сказано, как её получить>"
GID_ID="<идентификатор группы с ботом>"
LOG_FMT="%(levelname)s: %(message)s" # Формат логов
LOG_LEVEL="20" # Уровень логгирования. Указывает на то, логи какого уровня печатать. Подробнее
↪здесь: `<https://docs.python.org/3/library/logging.html#levels>`_
VK_TOKEN="<токен сообщества>"
VK_USER_TOKEN="<токен администратора. Нужен для автоматического изменения номера версии в статусе
↪группы>"
```

В случае, если вы работаете в PyCharm для локального запуска бота вы можете использовать расширение EnvFile (Без него вы будете получать ошибку KeyError, говорящую о том, что не была найдена переменная окружения)

#### 1.4.1.6 Настройка хостинга

Я предлагаю использовать в качестве хостинга Heroku. У них есть бесплатный тариф, но с ограничением по трафику в 550 часов / месяц. По моему опыту этого достаточно. Для работы с Heroku нужно:

- зарегистрировать аккаунт
- установить CLI

Все действия из списка ниже нужно выполнять в папке Ralph со всеми исходниками бота

1. Авторизуемся в CLI:

```
heroku auth
```

2. Создаём приложение:

```
heroku create
```

Имя приложения будет сгенерировано автоматически.

3. Подключаем к приложению базу данных PostgreSQL:

```
heroku addons:create heroku-postgresql:hobby-dev
```

4. Получим ссылку для доступа к базе данных

```
heroku config:get DATABASE_URL
```

5. Скопируем всю ссылку и вставим в файл **.env**

6. Настроим переменные окружения на сервере:

```
heroku config:set GID_ID=идентификатор вашей группы
heroku config:set LOG_FMT=%(levelname)s: %(message)s
heroku config:set LOG_LEVEL=20
heroku config:set VK_TOKEN=токен сообщества
heroku config:set VK_USER_TOKEN=токен администратора
```

#### 1.4.1.7 Настройка виртуального окружения

1. Создадим виртуальное окружение в папке с ботом

```
python -m venv venv
```

2. Активируем его

   На Windows:

```
venv/Scripts/activate
```

   На Linux:

```
source venv/bin/activate
```

#### 1.4.1.8 Настройка базы данных

Для того чтобы скопировать нужную для работы бота структуру БД в вашу копию бота:

1. Откройте файл **db.pgsql**

2. Замените все вхождения «user» на имя вашего пользователя

---

**Примечание:** Структура ссылки, получаемой от Heroku: postgres://**username**:password@localhost/db_name

Вам нужно скопировать выделенную часть из вашей ссылки в меню Замена в редакторе

---

3. Примените дамп из файла **db.pgsql**

В Linux:

```
psql $DATABASE_URL < db.pgsql
```

В Windows:

```
psql %DATABASE_URL% < db.psql
```

После выполнения этой команды вы получите чистую БД с необходимой структурой

#### 1.4.1.9 Модификация кода

Код написан без жёсткой привязки к конкретному учереждению. Его можно легко адаптировать под любые нужды.

Модуль получения расписания хранится в файле scheduler.py.

Документация по этому модулю находится в отдельном файле.

Так же, для локального тестирования нужно создать файл **.env**, где будут записаны переменные окружения.

Чтобы сохранить в репозитории сделанные изменения, нужно сделать коммит:

Сохраним изменения:

```
git add .
```

И создадим коммит:

```
git commit -m "<краткое описание сделанных вами изменений>"
```

Отправим изменения на сервер

```
git push origin master
```

#### 1.4.1.10 Выгрузка на хостинг

1. Редактированный код можно выгрузить на сервер:

```
git push heroku master
```

2. И запустить:

```
heroku ps:scale bot=1 sch=1
```

#### 1.4.1.11 Загрузка исходного кода веб-интерфейса

1. Создайте папку в которой будет хранится исходный код веб-интерфейса. (Она не должна находиться в папке с ботом) Перейдите в нее из командной строки.
2. Загрузите последнюю доступную версию веб-интерфейса:

```
git clone https://github.com/dadyarri/ralph_cms

cd ralph_cms
```

3. Скопируйте файл **.env** из папки с ботом в папку с веб-интерфейсом

#### 1.4.1.12 Выгрузка веб-интерфейса на сервер

1. Создаём приложение:

```
heroku create
```

Имя приложения будет сгенерировано автоматически.

2. Настроим переменные окружения на сервере:

```
heroku config:set GID_ID=идентификатор вашей группы
heroku config:set LOG_FMT=%(levelname)s: %(message)s
heroku config:set LOG_LEVEL=20
heroku config:set VK_TOKEN=токен сообщества
heroku config:set VK_USER_TOKEN=токен администратора
heroku config:set DATABASE_URL=ссылка из файла .env
```

8. Теперь код можно выгрузить на сервер. Запуск произойдет автоматически:

```
git push heroku master
```

Бот готов к работе. Теперь осталось создать группу со студентами и настроить её.

**См.также:**

Создание группы

### 1.4.2 Создание группы в существующем учебном заведении

#### 1.4.2.1 Работа с веб-интерфейсом

Если вы попали на эту страницу после создания нового учебного заведения:

1. Перейдите по адресу, который вам был дан при создании приложения веб-интерфейса

Если вы добавляете новую группу к уже подключенному заведению:

1. Перейдите по адресу, который указан в колонке веб-интерфейс таблицы на этой странице

Вы увидите перечень созданных групп для этого учебного заведения

2. Нажмите на кнопку «Добавить группу»
3. Заполните предложенную форму

   Поле «Дескриптор расписания» - это уникальная строка, которая отличает группу от других на вашем сайте с расписанием

4. Сохраните изменения

   Вы вернетесь на страницу вашей группы

5. Теперь вам нужно добавить администратора для группы. Для этого нажмите на ссылку в зеленом уведомлении
6. Укажите желаемое имя пользователя, пароль и отправьте форму.

   Вы попадете на главную страницу приложения уже под своим аккаунтом

7. Откройте страницу вашей группы, перейдите в раздел список студентов и зарегистрируте в базе данных всех студентов вашей группы

8. Затем зайдите на страницы тех, кто должен иметь возможность управлять ботом и нажмите на кнопку «Назначить администратором»

#### 1.4.2.2 Работа с ботом

Если вы подключаете новое учебное заведение, откройте группу с ботом, которую вы создали

Если вы создаете новую группу в существующем заведении, откройте группу с ботом с этой страницы

1. Напишите боту «Начать». Если вы назначили свой профиль администратором (п. 8 предыдущего раздела), вы получите клавиатуру администратора. Значит мы можем продолжить настройку.
2. Откройте раздел «Настройки» -> «Чаты» -> Глобальные настройки чатов
3. Нажмите кнопку «Зарегистрировать чат». (Она появится, только если вы уже пригласили бота в беседы, с которыми он должен работать).
4. Выберите чат и определите его тип (основной/тестовый)
5. Повторите то же самое с другим чатом
6. Выберите один из чатов и Активируйте его. (Автоматическая рассылка с расписанием приходит в активный чат)

**Настройка завершена!**

### 1.4.3 Список подключенных учебных заведений

#### 1.4.3.1 Колледжи

| Название | Бот | Веб-интерфейс |
|---|---|---|
| ИвПЭК (г. Иваново) | https://vk.com/ralphb | https://ralph-cms.herokuapp.com |

## 1.5 Веб-интерфейс

С помощью веб-интерфейса можно:

1. Управлять любыми данными студентов
    1. Имя
    2. Фамилия
    3. Группа
    4. Подгруппа
    5. Статус
    6. Ссылка на страницу ВК
2. Создавать и удалять новые группы
3. Назначать и разжаловать администраторов бота

## 1.6 Документация модулей

### 1.6.1 Bot

Основной класс бота.

class bot.Bot
Класс, описывающий объект бота, включая авторизацию в API, и все методы бота.

token
Токен доступа к сообществу ВКонтакте

**Type** str

user_token
Токен доступа пользователя-администратора ВКонтакте (используется для обновления статуса сообщества)

**Type** str

gid
Идентификатор сообщества ВКонтакте

**Type** str

cid
Идентификатор активной беседы (используется в рассылке расписания)

**Type** str

kbs
Объект класса Keyboards, содержащий генераторы клавиатур

**Type** Keyboards

db
Объект класса Database, инициирующий подключение к базе данных

**Type** Database

admins
Список идентификаторов ВКонтакте пользователей, имеющих доступ администратора бота

**Type** List[str]

#### Пример

```
from bot import Bot
bot = Bot()
bot.auth()
```

auth()
Авторизация ВКонтакте, подключение к API

generate_mentions(*ids: str*, *names: bool*) → str
Генерирует строку с упоминаниями из списка идентификаторов

**Параметры**

- ids – Перечень идентификаторов пользователей
- names – Флаг, указывающий на необходимость использования имён

**Результат** Сообщение, упоминающее выбранных пользователей

**Тип результата** str

`is_admin(_id: int)` → bool
Проверяет, является ли пользователь администратором бота

**Параметры** `_id` – Идентификатор пользователя для проверки привелегий

**Результат** Флаг, указывающий на принадлежность текущего пользователя к касте Администраторов

**Тип результата** bool

`send_gui(pid: int, text: str = 'Привет!')` → NoReturn
Отправляет клавиатуру главного меню

**Параметры**

- `pid` – Получатель клавиатуры
- `text` – Сообщение, вместе с которым будет отправлена клавиатура

`send_mailing(m_id: int, text: str, group: int, attach: str = '')`
Генерирует строку с упоминаниями из списка идентификаторов

**Параметры**

- `group` – Номер группы для поиска подписчиков
- `m_id` – Идентификатор рассылки
- `text` – Сообщение рассылки
- `attach` – Список вложений, прикрепляемых к рассылке

`send_message(msg: str = '', pid: int = None, keyboard: str = '', attachment: str = None, user_ids: str = None, forward: str = '')` → NoReturn
Обёртка над API ВКонтакте, отправляющая сообщения

**Параметры**

- `msg` – Текст отправляемого сообщения
- `pid` – Идентификатор пользователя/беседы/сообщества получателя сообщения (*не нужен, если указан user_ids*)
- `keyboard` – JSON-подобная строка со встроенной клавиатурой
- `attachment` – Вложения к сообщению (**не работает**)
- `user_ids` – Перечень адресатов для отправки одного сообщения (*не нужен, если указан pid*)
- `forward` – Перечень идентификаторов сообщений для пересылки

`update_version()`
Обновляет версию в статусе группы с ботом

### 1.6.2 Schedule

class scheduler.Date
Вспомогательный класс, содержащий строковые представления дат

property day_after_tomorrow
Возвращает послезавтрашнюю дату в формате ГГГГ-ММ-ДД

property today
Возвращает сегодняшнюю дату в формате ГГГГ-ММ-ДД

property tomorrow
Возвращает послезавтрашнюю дату в формате ГГГГ-ММ-ДД, если сегодня суббота, завтрашнюю в любой другой день

class scheduler.Schedule(*date: str*, *gid: str = '324'*)
Класс, переводящий расписание из сырой веб-страницы в читаемую строку

date
Дата в формате ГГГГ-ММ-ДД, используемая для получения расписания

**Type** str

gid
Идентификатор группы, для которой нужно получать расписание

**Type** str

#### Пример

```
d = Date() # Новый объект конструктора дат
sch = Schedule(d.today) # Новый объект конструктора расписания
sch.get_raw() # Соединение с сервером, получение свежего
# расписания на сегодня
if sch.is_exist():  # Проверка наличия расписания
    text = sch.generate() # Генерация расписания в читаемом виде
```

clean() → List[list]
Чистит объект вебскрапера от мусорных данных и оставляет только то, что относится к расписанию

**Результат** Список со списками. Каждый вложенный список описывает пару ( номер, название предмета, преподавателя, кабинет)

**Тип результата** List[list]

generate() → str
Собирает расписание в читаемую строку

get_raw()
Подключается к серверу и получает расписание как объект вебскрапера

is_exist() → bool
Проверяет наличие расписания, основываясь на присутствии плашек «Расписание отсутствует» и «Расписание составлено, но не опубликовано» и содержимом таблицы с расписанием

**Результат** Флаг, указывающий на существование расписания

**Тип результата** bool

`scheduler.listen()`
Слушает сервер на предмет наличия расписания. Если находит - отправляет, иначе ждет 15 минут

`scheduler.send()`
Отправляет расписание в активную беседу и в ЛС подписчикам рассылки «Расписание»

# Содержание модулей Python

## b



## s

# Алфавитный указатель