

Описание конечного результата.

Описание задачи:

Необходимо написать вк бота, работающего от лица сообщества, который будет пересылать помеченные сообщения из одной беседы(source) в ряд других бесед (targets). Бот должен время от времени отправлять свой статус в личные сообщения выбранным людям, чтобы быть уверенными, что бот работает. Управлять ботом нужно с использованием команд.

Описание команд:

Формат команды	Где прописывать	Описание
«reg_target [число_идентификатор] [«title»]»	В target беседе	Беседа добавляется в список targets. Беседа ассоциируется с число_идентификатор. Число_идентификатор должно быть уникальным для каждой беседы. Если этот параметр не передан, то он генерируется программой. Title будет отображаться при пересылке сообщений из этой беседы(на будущее).
«reg_source [«title»]»	В source беседе	Беседа становится source. Title будет отображаться при пересылке сообщений из этой беседы.
«reg_checker»	В личке	Личка будет использована для отправки состояния бота.
«del_target»	В target беседе	Удаляет зарегистрированную беседу из списка targets.
«del_source»	В source беседе	Удаляет зарегистрированную беседу из списка source.
«del_checker»	В личке	Останавливает отправку статуса в этой личке.

Пометки сообщений:

- «_всем_ [«title»]» - сообщение содержащее такую строчку будет отправлено всем targets. Title будет помещен в тексте сообщения бота.
- «_всем_важно_ [«title»]» - сообщение содержащее такую строчку будет переслано всем targets с пометкой @all. Title будет помещен в тексте сообщения бота.

Детальные требования.

Описание работы программы:

При запуске бот считывает файл **config.json** и начинает слушание порта, указанного в файле. Если файл не удалось открыть или порт недоступен – Fatal Error. Бот находит режим работы в поле **«mode»**. Если поля нет, или оно не содержит **«work»** или **«config»** – Fatal Error. На порт 80 от вк будут приходить json файлы с ивентами. Подробнее в разделе приложение. Дальше в зависимости от режима.

Режим «config»:

В этом режиме происходит настройка бота с помощью всех команд. Пересылка сообщений не работает.

- **«reg_target** [число_идентификатор] [«title»]» - надо чтобы и число_идентификатор и id беседы были свободны в таблице(см. конфиг).
Если id беседы является source, то сообщение об ошибке в данную беседу.
Если id беседы используется с другим числом идентификатом, то сообщение об ошибке в данную беседу.
Если число_идентификатор уже занято, то сообщение об ошибке в данную беседу.
- **«reg_source** [«title»]» - надо чтобы source был не занят и id не лежал в targets.
Если source уже существует, то сообщение об ошибке в данную беседу.
Если id находится в targets, то сообщение об ошибке в данную беседу.
- **«reg_checker»** - отправляется боту в личные сообщения.
Если пользователь уже имеется, то сообщение об ошибке в личку.
- **«del_target»** - прописывается в target беседе.
Если беседа не является target, то сообщение об ошибке.
- **«del_target_by_num** число_идентификатор» - эксклюзивная команда в личку бота. Доступна только Godlike пользователям(см. конфиг).

Удаляет беседу связанную с число_идентификатор из targets.

Если беседа не является target, то сообщение об ошибке в личку.

Если число_идентификатор не удалось распознать в сообщении, то сообщение об ошибке в личку.

- **«del_source»** - прописывается в source беседе.
Если беседа не является source, то сообщение об ошибке
- **«del_checker»** - прописывается в личке с ботом.
Если айди не является checker, то сообщение об ошибке.
- **«print_map»** - эксклюзивная команда в личку бота. Доступна только Godlike пользователям(см. Конфиг). Отправляет таблицу зарегистрированных бесед в личку.
- **«set_title»** - на будущее.
- **«del_title»** - на будущее.

Режим «work»:

Обязательно наличие source беседы, а также хотя бы одной target беседы. Иначе Fatal error.

В этом режиме бот просто пересылает сообщения с тегами из source беседы в targets. Никакие команды не работают в этом режиме.

Приложение:

VK confirmation:

Для подключения Callback API сервер должен вернуть заданную строку.

Адрес:

Для получения уведомлений нужно подтвердить адрес сервера. На него будет отправлен **POST-запрос**, содержащий JSON:
{ "type": "confirmation", "group_id": 220604034 }

Строка, которую должен вернуть сервер: 3cac4c26

Подтвердить

Секретный ключ:

Сохранить

VK event json:

В таком формате vk будет присылать уведомления на сервер.

```
JSON
{
  "type": <тип события>,
  "object": <объект, инициировавший событие>,
  "group_id": <ID сообщества, в котором произошло событие>
}
```

Например:

```
JSON
{
  "type": "group_join",
  "object": {
    "user_id": 1,
    "join_type": "approved"
  },
  "group_id": 1
}
```

Нас интересует событие «message_new». «client_info» не содержит ничего важного для нашего бота, так что будет игнорироваться.

Тип события:

- `message_new` — входящее сообщение.

Формат поля `object`:

- Для версий API 5.103 и выше: объект, содержащий следующие поля:
- `message` — [личное сообщение](#);
- `client_info` — [информация о функциях, доступных пользователю](#).

Личное сообщение содержит важные для нас поля

`int id` — уникальный идентификатор сообщения.

`int peer_id` — уникальный идентификатор назначения(кому отправили).

`int from_id` — уникальный идентификатор отправителя.

`string text` — текст сообщения.

Работа с VK API:

URL — <https://api.vk.com/method/<METHOD>?<PARAMS>>

METHOD может быть например status.get, message.send и прочие (см. <https://dev.vk.com/api/api-requests>).

PARAMS обязательно должен содержать минимум два параметра — «**access_token**» и «**v**».

«**v**» - указывает с какой версией API мы работаем. На данный момент актуальная «5.131».

«**access_token**» см далее.

Access_token:

Вот что написано на сайте вк. «**Для работы с методами API** вам необходимо передавать в запросах **специальный ключ доступа** — access_token. Он представляет собой строку из латинских букв и цифр и может соответствовать отдельному пользователю, сообществу или вашему приложению». **access_token** можно будет получить из настроек нашего сообщества(см. **конфиг**). Будет действовать до тех пор, пока не удалён вручную.

Коды ошибок VK:

При попытке выполнить запрос сайт может вернуть ошибку.

<https://dev.vk.com/reference/errors> — здесь можно посмотреть все коды. Рассмотрим основные из них.

- Код ошибки 5. Неверный токен. Может возникнуть при работе с API. Нужно попытаться снова через **1, 2, 5 секунд** — если не получилось то прервать выполнение операции. Такое бывает, но редко, что вк не узнает токены. Как правило быстро проходит (~5 минут).
- Код ошибки 15. Доступ запрещен. Нужно попытаться снова через **1, 2, 5 секунд** — если не получилось то прервать выполнение операции.
- Код ошибки 1 или 10. Неизвестная ошибка. Может возникнуть при работе с API. Нужно попытаться снова через **1, 2, 5 секунд** — если не получилось то прервать выполнение операции (одноразовая ошибка).
- Код ошибки 6. Слишком много запросов в секунду. Может возникнуть при работе с API. Нужно выдержать паузу в 2 секунды и повторить снова. И вообще таких случаев лучше не допускать. Будут тормозить программу. Использовать метод **messages.send peer_ids** чтобы избежать проблемы.
- При остальных кодах нужно попытаться снова через 1, 2, 5 секунд — если не получилось то прервать выполнение операции.

Логирование:

Первая часть:

Бот должен складывать в лог файл информацию для дебага о каждом этапе выполнения с помощью Boost.Log. Максимальный размер лог файла берётся из конфига.

Вторая часть:

Время от времени бот должен отправлять определенным людям(status checker-ам) сообщение, позволяющее понять, что бот живой. Также бот должен отправлять им сообщение, если в source беседе появилось любое новое сообщение. Это позволит понять, не потеряна ли связь с сервером уведомлений вконтакте.

Структура config.json:

- «**mode**» - «work» or «config». (**обязательное**)
- «**token**» - токен сообщества бота. (**обязательное**)
- «**secret_string**» - строка для подтверждения связи с вк. (**обязательное**)
- «**port**» - порт который будет слушать сервер. (**обязательное**)
- «**target_chats**» - с парами (идентификатор : {peer_id беседы, title беседы(**необязательное**)}) — map. (**обязательное для work**)
- «**source_chat**» - peer_id чата. (**обязательное для work**)
- «**status_checkers**» - peer_ids кому отправлять время от времени состояние бота. (**необязательное**)
- «**log_file_max_size**» - максимальный размер log файла, в Мб. (**необязательное**)
- «**godlike_ids**» - id пользователей с эксклюзивными правами. (**необязательное**)

Используемые библиотеки:

- "jsoncpp" чтобы работать с json.
- "Boost.Log" ну тут понятно да.
- "libhttpserver" простая и вроде как приятная библиотека для поднятия сервера.

- "cpr" для легкого использования http запросов.

