Материалы к занятию

Ботов можно конечно же писать без использования каких либо сторонних библиотек, но существует огромное количество сторонних модулей, которые облегчают и ускоряют разработку бота.

Боты — специальные аккаунты в Telegram, созданные для того, чтобы автоматически обрабатывать и отправлять сообщения. Пользователи могут взаимодействовать с ботами при помощи сообщений, отправляемых через обычные или групповые чаты. Логика бота контролируется при помощи HTTPS запросов к API для ботов.

Что могут делать боты? Вот несколько примеров использования ботов:

- Интеграция с другими сервисами. Например, бот может отправлять комментарии или управлять «умным домом». Или, например, отправлять вам уведомления при совершении каком-то действия или события.
- Утилиты и инструменты. Бот может отображать погоду, переводить тексты или предупреждать о предстоящих событиях по вашему запросу.
- Одно- и многопользовательские игры. Бот может поиграть с вами в шашки или шахматы, проводить викторины и так далее.
- Социальные сервисы. Бот может находить вам собеседника, основываясь на ваших общих интересах и увлечениях.
- Все, что вам захочется. Бота можно запрограммировать для чего угодно. Разве что посуду они помыть не смогут.

Как уже было сказано ранее, роботы — особые аккаунты, которые не требуют номера телефона при создании. По сути, эти аккаунты играют роль интерфейса к вашему сервису, который работает на удалённом сервере.

Самое интересное в роботах это то, что для их создания вам не нужно изучать низкоуровневые методы работы с MTProto и шифрованием — общение с роботом организовано при помощи обычного HTTPS интерфейса с упрощёнными методами Telegram API. Он называется Bot API.

Для того, чтобы создать бота есть специальный Бот. Просто напишите пользователю @BotFather и следуйте его инструкциям. Для взаимодействия с ботом нам потребуется TOKEN для нашего бота. Токен - некий уникальный ключ, который позволяет телеграм понимать с каким ботом происходит взаимодействие.

Чем бот отличается от обычного аккаунта?

- У роботов нет статусов «онлайн» и «был в сети», вместо этого отображается надпись «бот».
- Для ботов выделено ограниченное место на серверах — все сообщения будут удалены по прошествии определённого срока после обработки.
- Боты не могут сами начать общение с пользователем. Пользователь должен либо добавить робота в группу, либо первым начать с ним диалог. Для этого можно использовать ссылки вида t.me/<bot_username> или поиск по имени пользователя.
- Имя пользователя у робота должно заканчиваться на «bot» (например, controllerbot).
- При добавлении в конференцию, по умолчанию робот не получает всех сообщений.
- Роботы никогда не спят, не едят и не жалуются (если только вы не запрограммируете их на обратное).

У ботов Telegram есть много уникальных возможностей — например, кастомизированные клавиатуры, дополнительные интерфейсы для команд по умолчанию, внешнее связывание и специальные режимы приватности для групп.

Пользователи могут общаться с вашим ботом напрямую из поля ввода сообщения — из любого чата или группы. Для этого нужно всего лишь набрать имя пользователя вашего бота и запрос для поиска. Получив запрос, бот может возвратить какие-либо результаты. Как только пользователь нажмёт на один из них, он сразу же отправится в чат. Таким образом можно запрашивать контент от бота в чатах, группах или каналах.

С помощью ботов пользователи могут играть в HTML5-игры в группах или приватных чатах. Игровая платформа Telegram поможет составить таблицу рекордов и оповещать пользователей об изменении рейтинга.

Одна из самых необычных возможностей Bot API — кастомизированные клавиатуры. При передаче сервером ответа есть возможность передать команду на отображение

специальной клавиатуры с предустановленными вариантами ответа. Клиент Telegram, получив сообщение, отобразит пользователю вашу клавиатуру. Нажатие на клавишу сразу же отправит на сервер соответствующую команду. Таким образом можно значительно упростить взаимодействие робота с пользователем. На данный момент для отображения на клавише могут использоваться эмодзи и текст.

Команды представляют собой более гибкий способ общения с ботом. Рекомендуется следующий синтаксис:
/команда [необязательный] [аргумент]

Команда должна начинаться с символа косой черты «/» и не может быть длиннее 32 символов. Команды могут состоять из букв латинского алфавита, цифр и подчёркивания. Несколько примеров:
/get_messages_stats
/set_timer 10min Alarm!
/get_timezone London, UK

Сообщения, начинающиеся с косой черты, будут всегда доставляться боту (точно также, как и при ответе на его сообщения и на @упоминания бота в чате). Приложения Telegram будут:

- Предлагать список поддерживаемых команд с их описанием, когда пользователь введёт символ косой черты «/» (чтобы этот пункт работал, вам необходимо задать описание команд у @BotFather). Нажатие на описание приведёт к отправке этой команды.
- Показывать кнопку (/) в поле ввода текста во всех чатах с ботами. Нажатие на эту кнопку отобразит список доступных команд.
- Подсвечивать /команды в сообщениях. При нажатии на такую подсвеченную команду, она будет сразу же отправлена боту.

Если в группе есть несколько ботов, вы можете дописать после команды имя бота, чтобы избежать коллизий в общих командах:

/start@TriviaBot
/start@ApocalypseBot
Это происходит автоматически, если вы выбираете команду из списка доступных.

Чтобы пользователям было проще работать с ботами, желательно реализовывать поддержку нескольких простых команд. В интерфейсе приложений Telegram будут ярлыки (быстрые ссылки) для этих команд.

- /start — начинает общение с пользователем (например, отправляет приветственное сообщение). В эту команду также можно передавать дополнительные аргументы.

- /help — отображает сообщение с помощью по командам. Оно может представлять собой короткое сообщение о вашем боте и список доступных команд.
- /settings — (по возможности) возвращает список возможных настроек и команды для их изменения.

При попытке начать общение с роботом, пользователь увидит кнопку СТАРТ. На странице профиля бота также будут доступны ссылки Помощь и Настройки

Ботов часто добавляют в группы, чтобы получать различную информацию — новости, уведомления и т.д. Именно поэтому у роботов есть режимы приватности.
Робот с включенным режимом приватности не будет получать всех сообщений, а только сообщения, удовлетворяющие этим условиям:

- Сообщения, начинающиеся с символа косой черты "/"
- Сообщения, содержащие @упоминание бота
- Ответы на сообщения бота
- Служебные сообщения (о добавлении пользователя, смены изображения группы и т.д.)

Это хорошо со всех сторон: во первых, некоторые люди будут спать спокойно, не опасаясь, что их будут прослушивать. Во-вторых, режим приватности избавляет разработчиков от необходимости обрабатывать сотни ненужных сообщений из групповых чатов.
Режим приватности включен по умолчанию во всех ботах. Он может быть выключен - тогда бот начнёт получать все сообщения, как и обычный пользователь. Всем участникам конференции виден текущий статус режима приватности в списке участников группы.
Рекомендуется отключать режим приватности только в случаях крайней необходимости. В подавляющем большинстве случаев, запроса принудительного ответа на сообщение бота будет достаточно.

Давайте поговорим более подробно о регистрации нашего первого бота. Как мы помним для это нам необходимо обратиться к BotFather.

BotFather — один бот, чтобы править всеми. При помощи него меняются настройки у существующих ботов и создаются новые.
Находим в телеграм BotFather и пишем команду /newbot, чтобы создать нового робота.
BotFather спросит у вас имя нового бота и предложит придумать username.
Имя (name) будет отображаться в контактах и чатах.
Username — короткое имя на латинице, которое используется для упоминаний бота и в ссылках на профиль в telegram.me. Username должен состоять из букв латинского алфавита, подчеркиваний и цифр и быть длиной от 5 до 32 символов. Также имя пользователя обязательно должно заканчиваться на «bot», например: «tetris_bot» или «TetrisBot».

Ключ (токен) это набор символов вида 110201543:AAHdqTcvCH1vGWJxfSeofSAs0K5PALDsaw, который нужен, чтобы получать и отправлять сообщения с помощью Bot API.

Если вы потеряли или утратили доступ к токену, отправьте команду /token, чтобы сгенерировать новый.

Настройки

- /setname — Изменить имя робота.
- /setdescription — Изменить описание робота, представляющее собой короткий текст с описанием бота. Пользователи увидят его в самом начале, под заголовком «Что умеет этот робот?».
- /setabouttext — Изменить информацию о боте, ещё более короткий текст, отображающийся в профиле бота. Ещё, если кто-то поделится вашим ботом, то вместе со ссылкой на него отправится этот текст.
- /setuserpic — Изменить аватарку бота.
- /setcommands — Изменить список команд бота. Каждая команда состоит из собственно командного слова, начинающегося с символа косой черты («/») и короткого описания. Пользователи увидят список команд при вводе символа «/».
- /setjoingroups — Определяет, можно ли добавлять вашего бота в группы.
- /setprivacy — Определяет, все ли сообщения видит ваш бот в группах. В выключенном состоянии роботу будут отправляться все сообщения.
- /deletebot — Удалить бота и его имя пользователя.

Прежде чем мы начнем изучать более детально разработку ботов давайте сравним в написании простого эхо бота без использования сторонних модулей и с использованием aiogram.

Начнем с бота без модулей. Для начала создадим переменные в которые запишем адрес, по которому мы будем отправлять запросы, а также переменную с токеном бота

```
import asyncio
import requests
BASE_URL = 'https://api.telegram.org/bot'

TOKEN = 'Ваш токен'
```

После этого нам необходимо проверять есть ли у бота новые сообщения и в ответ отправлять это же сообщение пользователю, который отправил сообщение.

```
import asyncio
import requests
```

```
BASE_URL = 'https://api.telegram.org/bot'

TOKEN = 'Ваш токен'


def get_updates():
   r = requests.get(f'{BASE_URL}{TOKEN}/getUpdates')
   message = r.json()['result'][-1]['message']['text']
   user_id = r.json()['result'][-1]['message']['chat']['id']

requests.get(f'{BASE_URL}{TOKEN}/sendMessage?chat_id={user_id}&text={messag
e}')


get_updates()
```

В данном коде мы отправляем запрос по ссылке согласно документации, состоящей из адреса api, токена и названия метода. Для получения сообщений мы используем getUpdates. В переменную r мы сохраняем ответ сервера и в дальнейшем разбираем на необходимые нам составляющие: сообщение и пользователя отправившего его.
После получения необходимых данных мы отправляем запрос используя в адресе метод sendMessage, который имеет обязательные параметры chat_id и text.

Данный код срабатывает один раз, так как у нас не реализованы, какие либо циклы позволяющие проверять данные постоянно.

Теперь попробуем сделать тоже самое, с помощью модуля aiogram.
pip install aiogram

```
from aiogram import Bot, types
from aiogram.dispatcher import Dispatcher
from aiogram.utils import executor

bot = Bot(token='Ваш токен')
dp = Dispatcher(bot)


@dp.message_handler()
async def echo_message(msg: types.Message):
   await bot.send_message(msg.from_user.id, msg.text)


if __name__ == '__main__':
   executor.start_polling(dp)
```

Как мы видим с использованием модуля aiogram код стал более простым для понимания. Нам нет надобности разбирать словарь вернувшийся нам в ответ, для нахождения необходимых данных.

Первое, на что нужно обратить внимание: aiogram — асинхронная библиотека, поэтому ваши функции тоже должны быть асинхронными, а перед вызовами методов API нужно ставить ключевое слово await, т.к. эти вызовы возвращают корутины.

Переменная bot, является экземпляром класса Bot и при инициализации мы передаем в качестве атрибута токен бота.

Диспетчер регистрирует функции-обработчики, дополнительно ограничивая перечень вызывающих их событий через фильтры. После получения очередного апдейта (события от Telegram), диспетчер выберет нужную функцию обработки, подходящую по всем фильтрам, например, «обработка сообщений, являющихся изображениями, в чате с ID икс и с длиной подписи игрек». Если две функции имеют одинаковые по логике фильтры, то будет вызвана та, что зарегистрирована раньше.

Чтобы зарегистрировать функцию как обработчик сообщений, нужно сделать одно из двух действий:
1. Навесить на неё декоратор, как в примере.
2. Напрямую вызвать метод регистрации у диспетчера.

Декоратор `@dp.message_handler()` в данном случае говорит нашему боту, что необходимо перехватывать все сообщения. Мы также можем использовать фильтры, чтобы ограничить выполнение той или иной функции, только по определенной команде.

Например давайте укажем, что при вводе команды test1 будет выводиться соответствующее сообщение.

```
from aiogram import Bot, types
from aiogram.dispatcher import Dispatcher
from aiogram.utils import executor

bot = Bot(token='Ваш токен')
dp = Dispatcher(bot)


@dp.message_handler(commands="test1")
async def cmd_test1(message: types.Message):
   await message.reply("Test 1")


@dp.message_handler()
async def echo_message(msg: types.Message):
   await bot.send_message(msg.from_user.id, msg.text)


if __name__ == '__main__':
   executor.start_polling(dp)
```

В данном случае стоит учитывать, что если указать хендлер, который перехватывает все сообщения, выше чем остальные, то остальные хендлеры не будут работать.