**Урок 4**

**Бот отправляет информацию о пользователе**

На предыдущем уроке мы создали диалог с ботом. Добавили новую команду и научили бота реагировать на конкретные фразы пользователя. Также познакомились с функцией **register\_next\_step\_handler.**

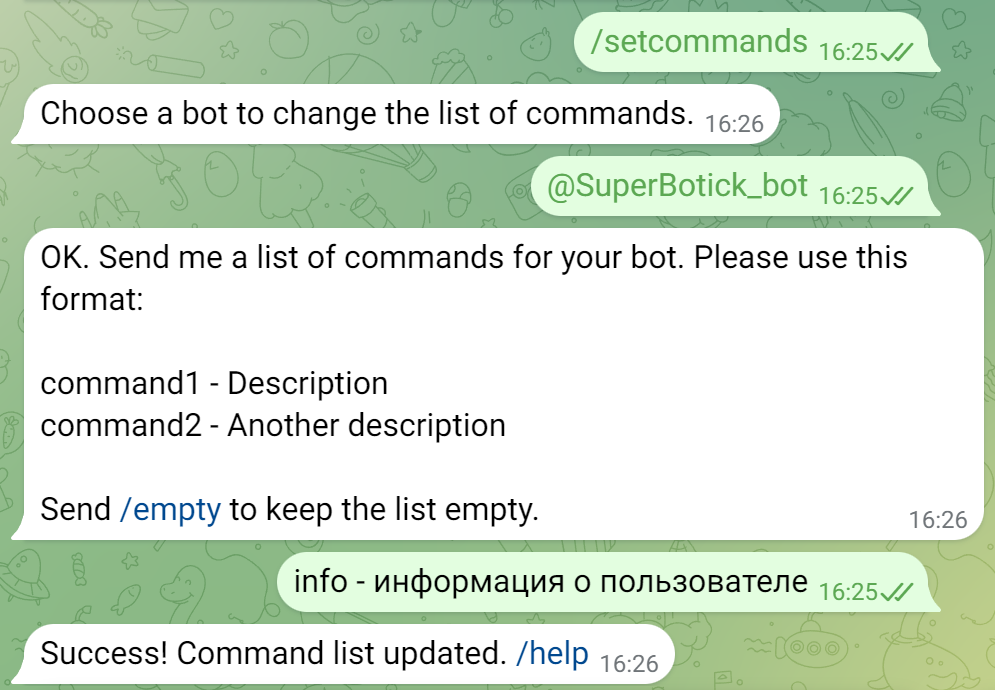
Сегодня мы сразу перейдём к чему-то более полезному — бот, отправляющий пользователю его имя, ID и другую информацию. Но сначала разберёмся откуда бот может взять эти данные.

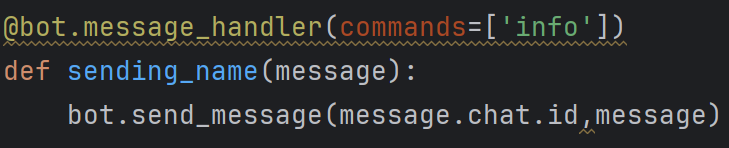
На втором уроке мы создавали «Эхо-бота», который повторял за пользователем текст сообщения. С помощью какой команды мы смогли это реализовать?

**message.text**

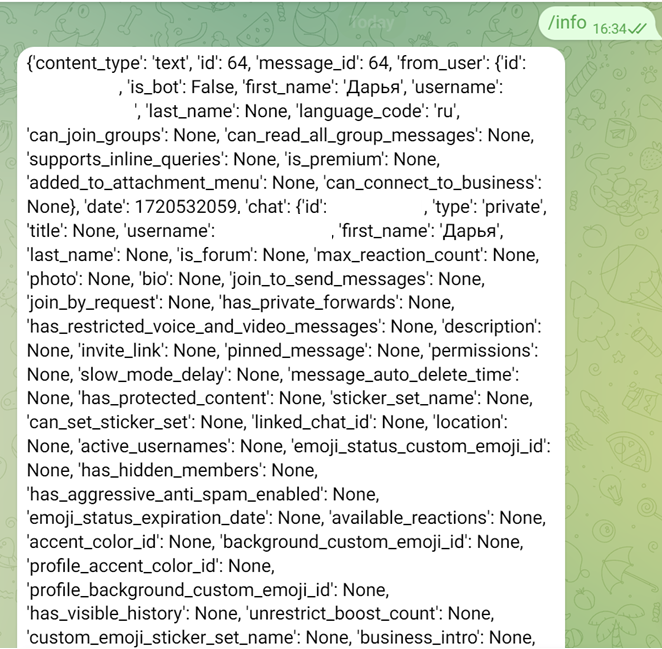
Мы использовали метод **message**, указывая, что возвращаем исходный текст, принятый в **message.** Скорее всего вы уже догадались, что с помощью этого метода можно возвращать не только текст, но и всю информацию про сообщение и самого пользователя, который работает с нашим ботом и информацию про чат. Если мы вместо текста будем выводить параметр **message**, то получим объект, где записаны различные ключи и значения.

**Самостоятельная работа**

Добавьте нашему боту ещё одну команду **info**

И напишите программу для бота, чтобы он вместо текста выводил параметр **message:**

Запустите бота и посмотрите, что он вам отправит.

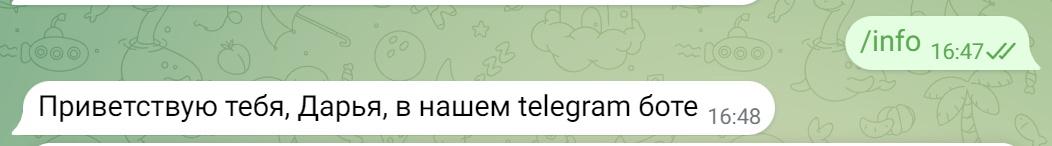


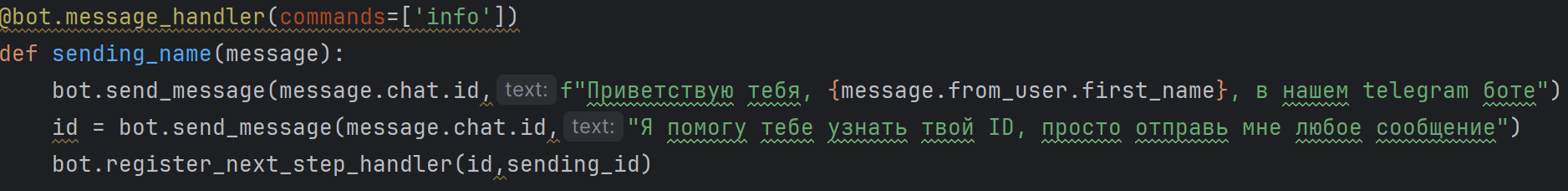
Мы видим всю доступную информацию о пользователе и чате с ним. Подробное описание можно найти в официальной документации (<https://core.telegram.org/bots/api#message>) вместе с описанием всех вложенных объектов типа chat, user и т. д.

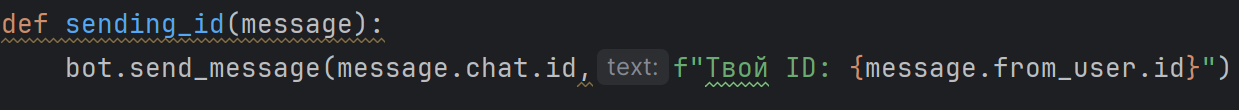
**Начинаем писать код**

Теперь сделаем так, чтобы выводилось имя пользователя, указанное в профиле. Для этого сначала надо обратиться к объекту **message**, к ключу **first\_name** и тогда мы получим имя пользователя (также мы можем получить его фамилию, логин, на каком языке он общается и многое другое. Но сейчас остановимся только на его имени).

Теперь давайте в нашем тексте будем выводить имя пользователя — то есть «Приветствую тебя, **имя пользователя**, в нашем telegram боте». Чтобы это сделать в самом тексте мы обратимся в фигурных скобках к параметру **message**, далее внутри этого объекта необходимо обратиться к значению **from\_user**, т.к. это еще один отдельный объект, а уже там мы обратимся к **first\_name**.

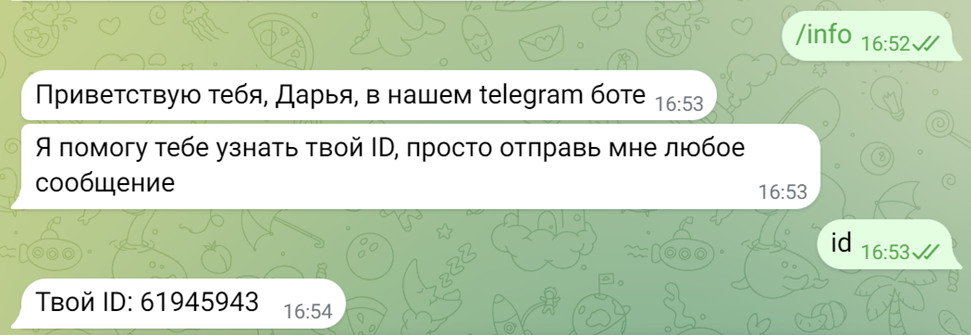
Теперь давайте перезапустим бота и проверим его.

Всё получилось! Дополним нашего бота, чтобы он отправлял id после того, как пользователь напишет любое сообщение. Для этого используем уже знакомую нам функцию **register\_next\_step\_handler:**

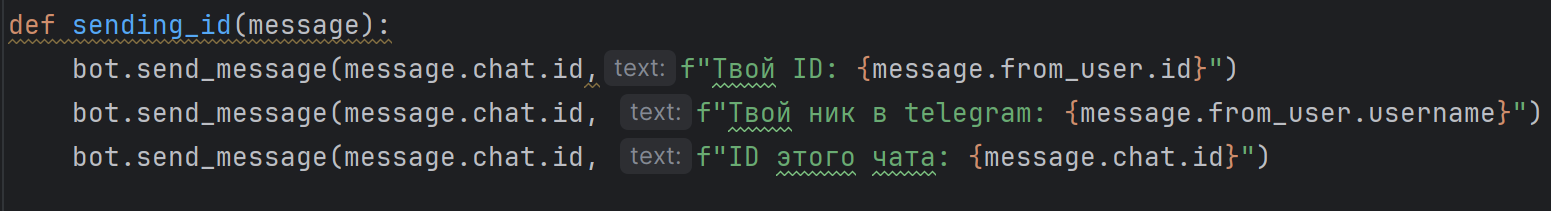
Создаём вторую функцию **sending\_id**:

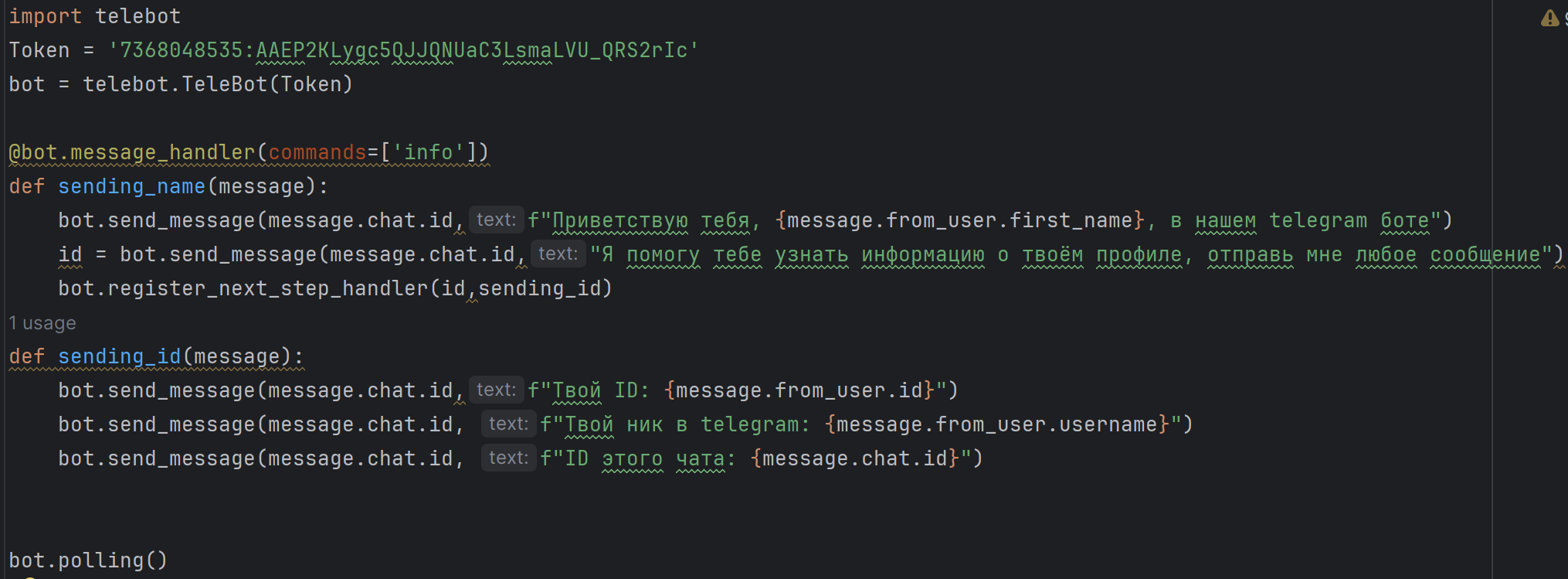
Как и в предыдущей функции мы обращаемся к фигурным скобкам, к параметру **message,** к значению **from\_user.** Но указываем не **first\_name,** а объект **id.**

Проверим нашего бота ещё раз:

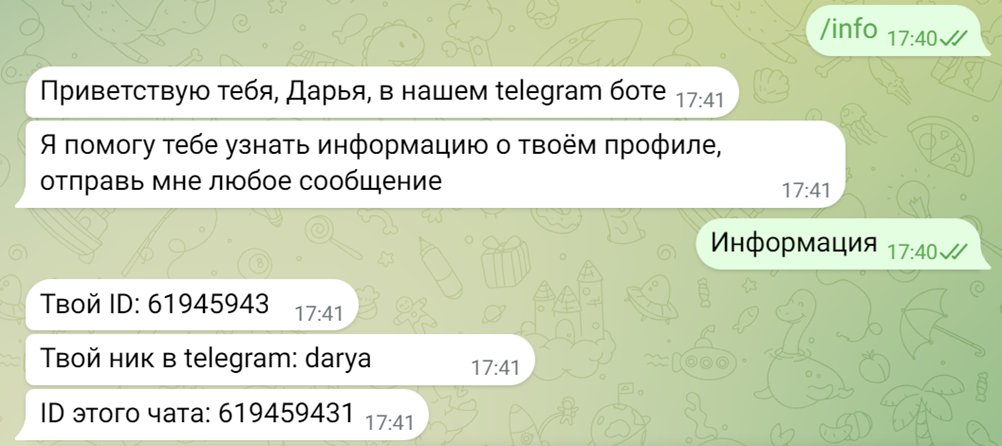


**Самостоятельная работа**

Сделайте так, чтобы бот отправлял также ник пользователя в telegram и id чата с ним:

Теперь вся программа выглядит так:

Протестируем нашего бота



Получилось! Теперь вы можете добавить другую информацию о пользователе, которая также будет отправляться ботом.

**Рефлексия**

* Сегодня мы создали бота, отправляющего пользователю его имя, ID и другую информацию
* Разобрались откуда бот может взять эти данные
* Научились использовать метод message

***Тайминг.***

|  |  |  |
| --- | --- | --- |
|  | Раздел урока | Время с начала урока, мин. |
|  | Повторение материала предыдущего урока. Постановка цели урока | 0 |
|  | Использование метода message | 10 |
|  | Самостоятельная работа | 20 |
|  | Информация о пользователе и чате с ним | 40 |
|  | Начинаем писать код | 60 |
|  | Бескомпьютерная деятельность | 65 |
|  | Самостоятельная работа | 70 |
|  | Рефлексия | 85 |