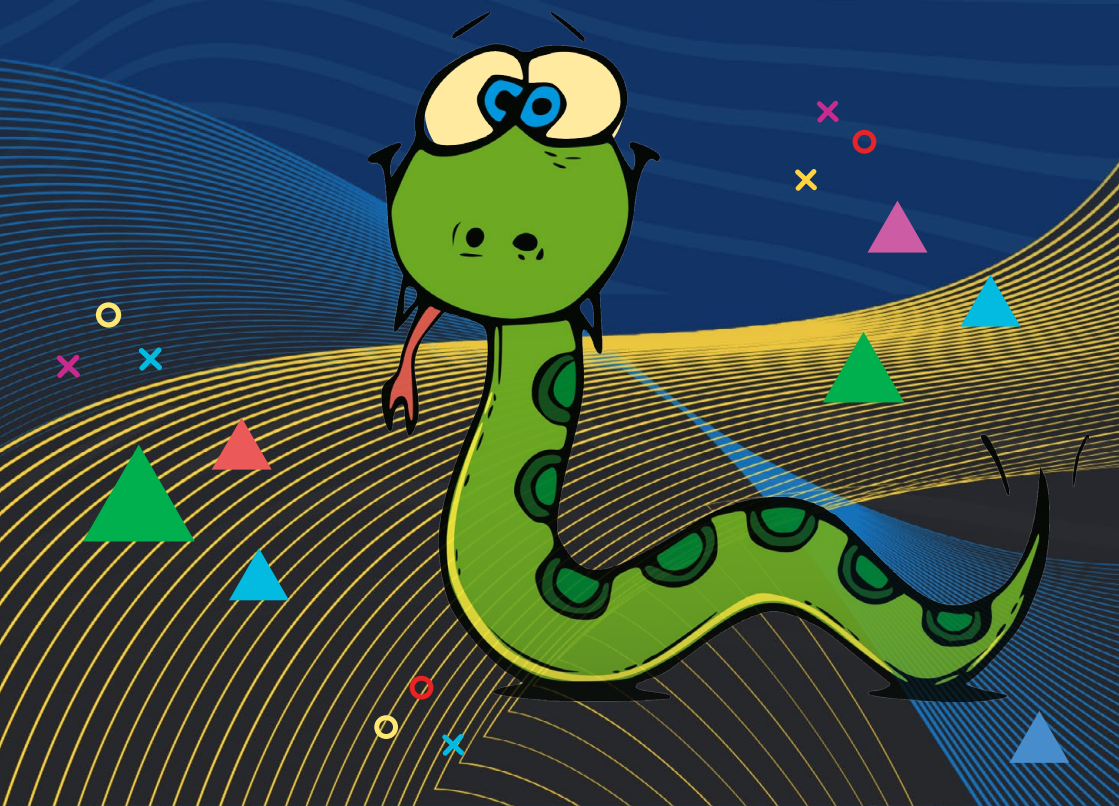


# РОЗРОБКА ДОДАТКІВ НА PYTHON middle




# Урок 11

## БОТ-ПОМІЧНИК

### Зміст

Чат-боти.....	3
Створення і запуск чат-бота у Telegram .....	4

У цьому уроці використовуються відеоматеріали, позначені **PLAY**  поверх ілюстрації. Клацніть на значок, щоб переглянути відео. Для коректного перегляду відео рекомендуємо відкрити урок у програмі [Foxit PDF Reader](#).

## Чат-боти

Автоматизація з кожним роком набирає обертів, що не дивує, адже дедалі більше ІТ-спеціалістів працюють над нею.

Така тенденція не лише здешевлює продукт, але й робить користування ним зручнішим, спрощуючи життя. Одним із найяскравіших проявів таких мотивів є **чат-боти**, які дають змогу, наприклад, з легкістю дістати інформацію про прогноз погоди, здійснити бронювання квитків або проконсультуватися щодо роботи з продуктом тощо (рис. 1).

**Час і нам попрацювати над своїм чат-ботом!**



Рисунок 1

# Створення і запуск чат-бота у Telegram

Одним із найзручніших інструментів створення чат-ботів є **Telegram**, тому наша програма буде працювати саме з ним.

Тоді розроблення помічника можна розбити на два етапи. На першому ми працюватимемо із самим Telegram, де зареєструємо та базово налаштуємо бота. На другому – реалізуємо логіку його роботи. Отже, почнемо.

Зайдіть до Telegram та авторизуйтесь. У рядку пошуку введіть **botfather** (рис. 2):

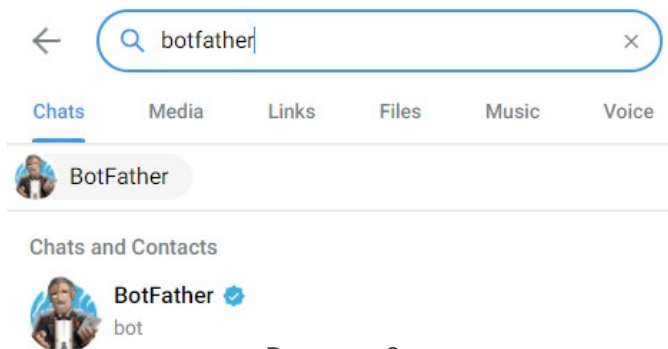


Рисунок 2

Перейдіть у вказаний чат та пропишіть **/start** (див. рис. 3 на стор. 5).

Наступною надішліть команду **/newbot** (рис. 4).

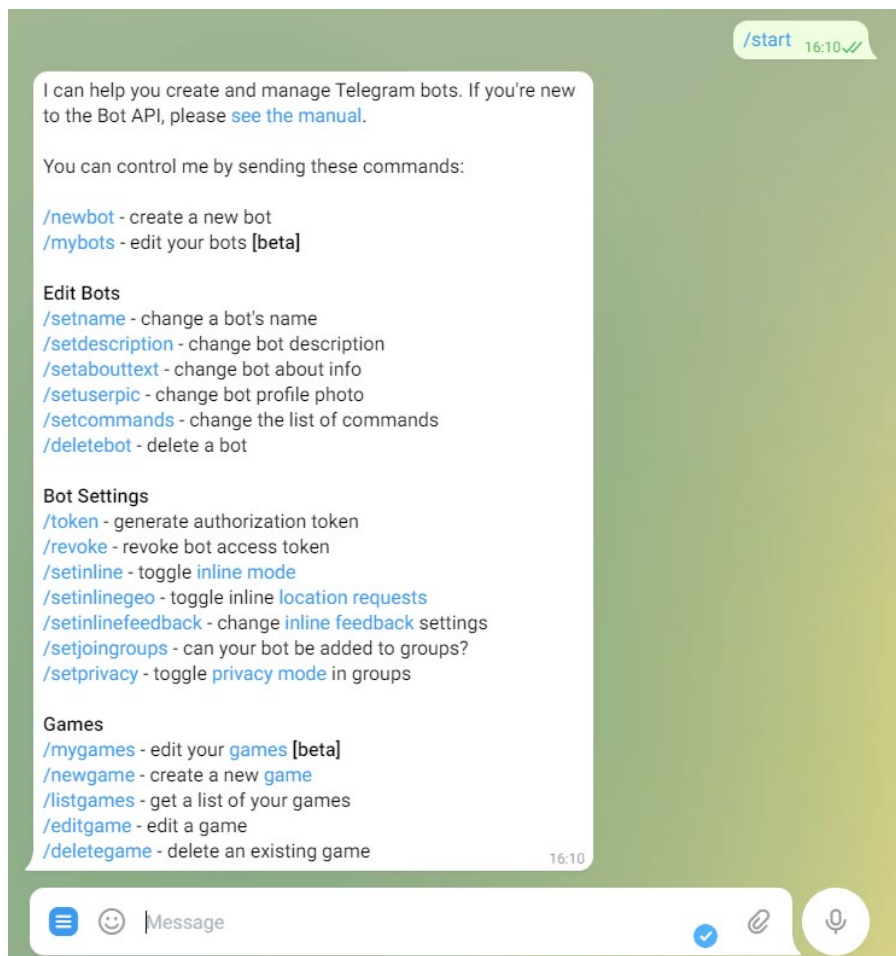


Рисунок 3

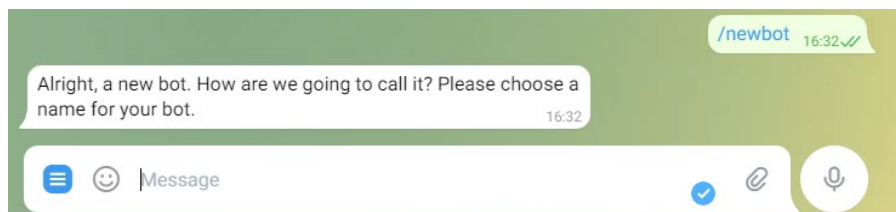


Рисунок 4

Відправте назву для свого бота (рис. 5):

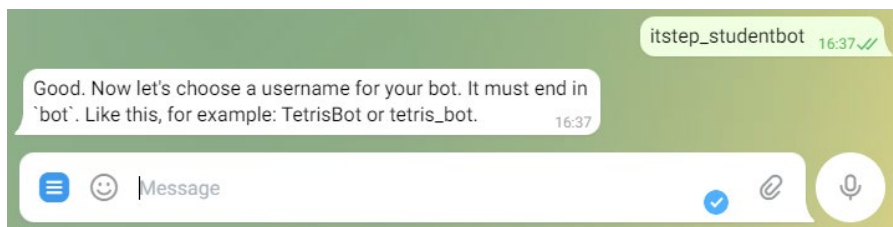


Рисунок 5

Останнім кроком надішліть користувацьке ім'я бота. Воно обов'язково має закінчуватися часткою **bot** (рис. 6):

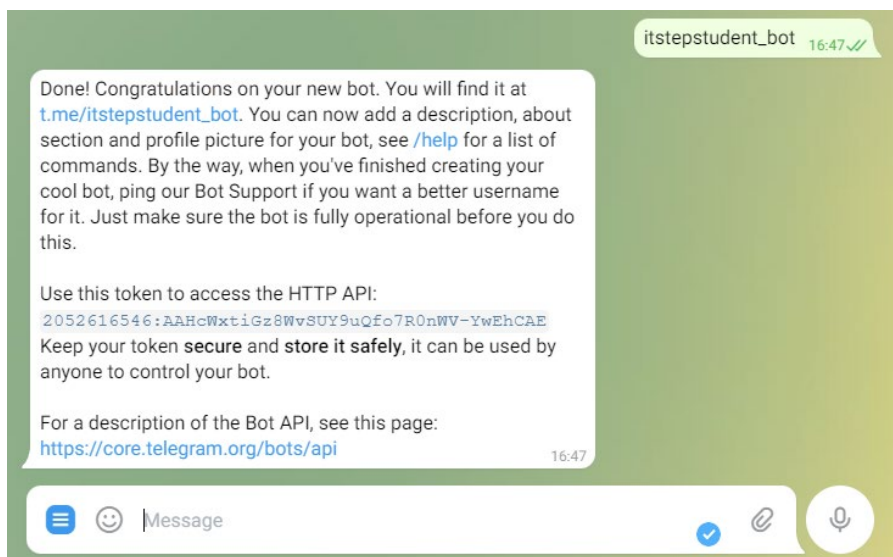


Рисунок 6

Для подальшої роботи знадобляться:

- посилання на бота – в нашому випадку: [t.me/itstepstudent\\_bot](https://t.me/itstepstudent_bot));

- та його **токен**:

**2052616546:AAHcWxtiGz8WvSUY9uQfo7R0nWV-YwEhCAE.**

А щоби зробити його більш особливим, встановимо аватар. Для цього надішліть команду **/mybots** та оберіть свого помічника (рис. 7):

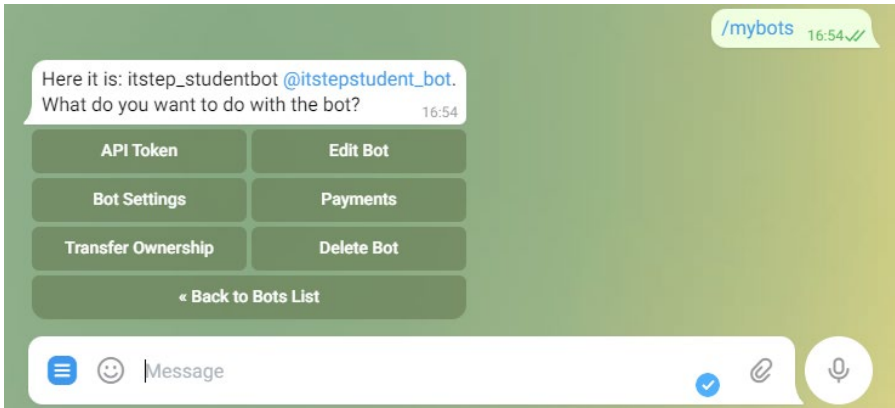


Рисунок 7

Натисніть клавішу **Edit Bot** (рис. 8):

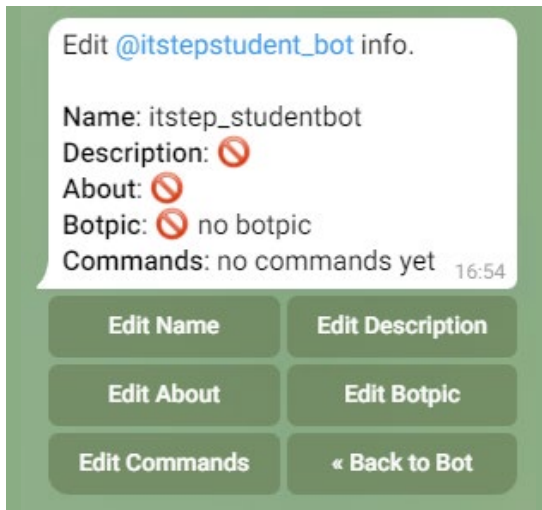


Рисунок 8

Оберіть **Edit Botpic**. У відповідь одержимо (рис. 9):

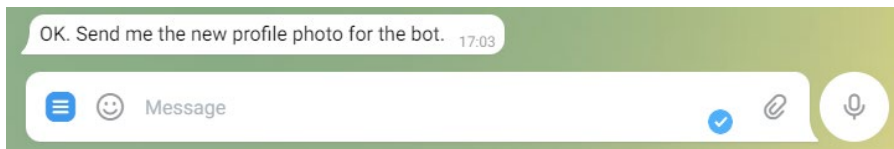


Рисунок 9

Тепер надішліть вподобане зображення для вашого бота (рис. 10):

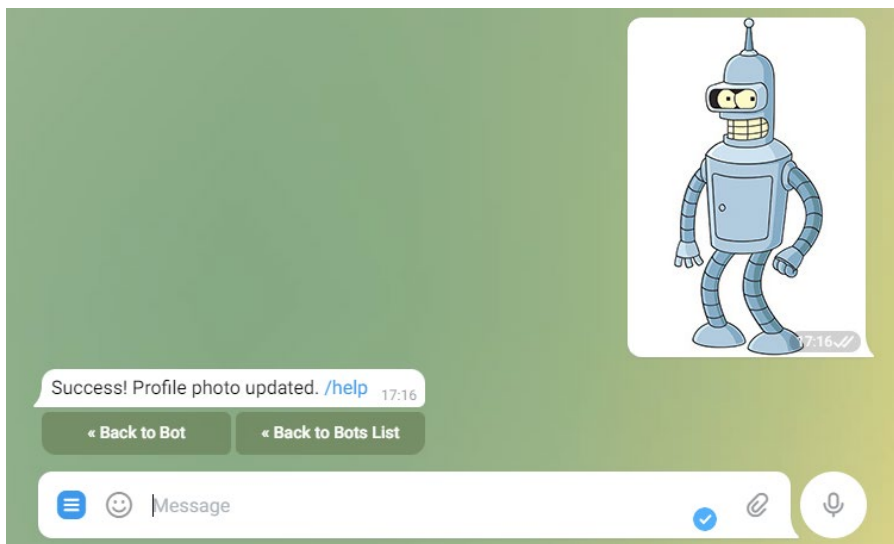


Рисунок 10

**Базове налаштування завершено.**

**Перейдемо до проєкту в PyCharm, щоби попрацювати над змістом бота.**



Оскільки будемо працювати з Telegram, а вбудованої бібліотеки для роботи з ним не передбачено, то треба її встановити. Натисніть клавішу **Terminal** на нижній панелі середовища розробки (рис. 11):

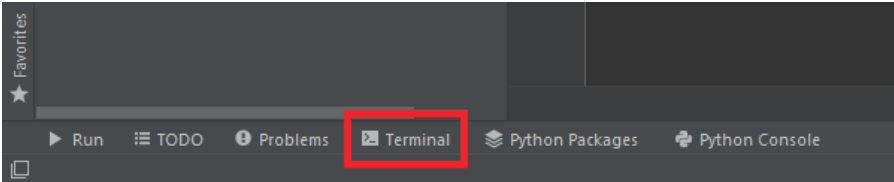


Рисунок 11

Тепер будемо працювати у вікні, що відкрилось.  
**Впевніться, що шлях веде до директорії проекту**  
(рис. 12):

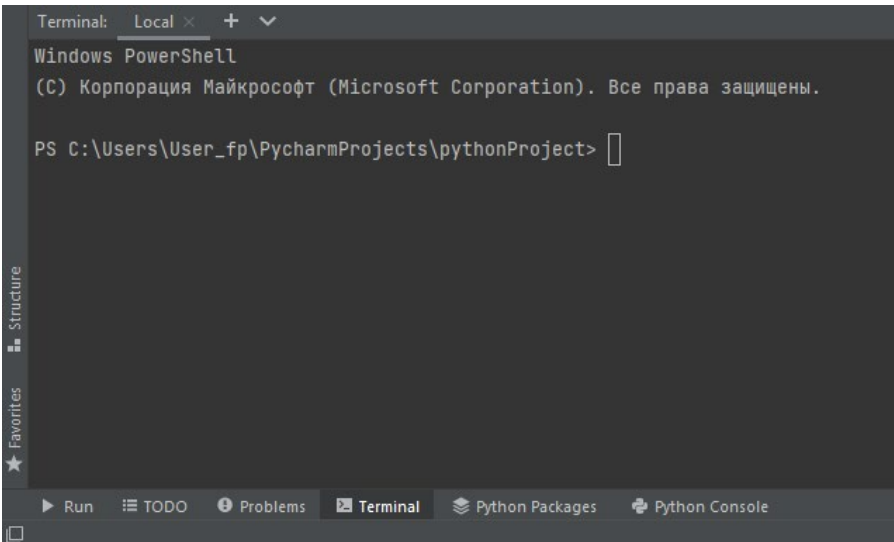
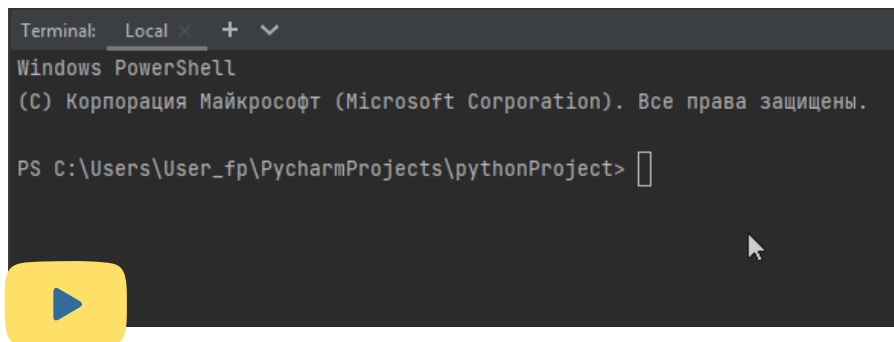


Рисунок 12

Далі треба встановити бібліотеку, для чого пропишемо команду `pip install python-telegram-bot` (відео 1):



Відео 1

Бібліотеку встановлено, тож почнемо роботу з нею. Імпортуємо обробника оновлень:

```
from telegram.ext import Updater
```

Оголосимо функцію, що реалізує запуск бота, а першим рядком створимо об'єкт обробника, куди передамо токен бота:

```
def main():
    updater =
    Updater("2052616546:AAHcWxtiGz8WvSUY9uQfo7R0nWV-
    YwEhCAE")
```

Лишається викликати методи `start_polling()` та `idle()`. Перший запустить процес слухання сервера та оновлення даних, другий – завершить цей процес після

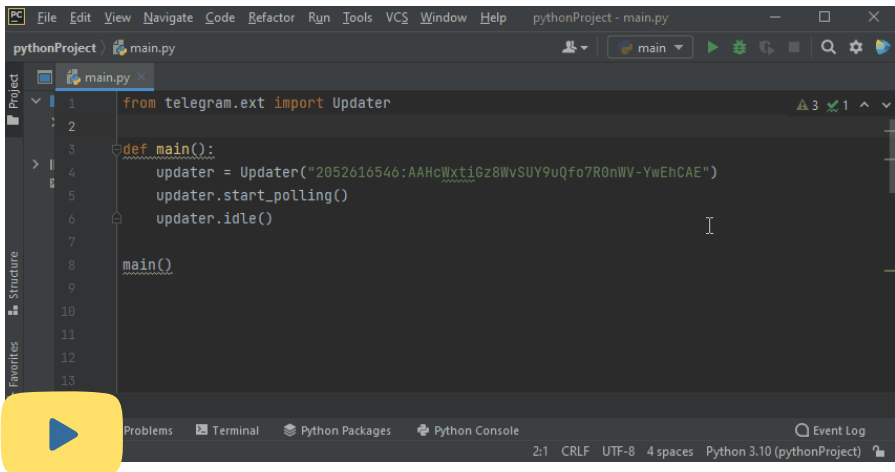
зупинки програми. Якщо його не прописати та знову запусити програму, виникне конфлікт, адже працювати з ботом на сервері Telegram будуть одразу дві програми. Лишається викликати функцію:

```
from telegram.ext import Updater

def main():
    updater =
    Updater("2052616546:AAHcWxtiGz8WvSUY9uQfo7R0nWV-
    YwEhCAE")
    updater.start_polling()
    updater.idle()

main()
```

Запустимо код (відео 2):



Відео 2

Тепер бот приймає запити. Але якщо почати роботу з ним, нічого не відбуватиметься, адже бракує логіки обробки даних. Отже, попрацюємо над нею. Додамо імпорти:

- **Update** – об'єкт оновлень.
- **CommandHandler** – обробник команд.
- **CallbackContext** – контекстний об'єкт. Саме він надає інформацію про помилки.

Дістанемо такий код:

```
from telegram.ext import Updater, CommandHandler,
CallbackContext
from telegram import Update
```

Оголосимо функцію для команди **start**, яка викличеться разом зі стартом бота. Як вхідні параметри за замовчуванням передамо **Update** та **CallbackContext**:

```
def start(update=Update, context=CallbackContext):
    pass
```

Наповнимо функцію змістом. Надіславши стартову команду, бот відповість повідомленням з привітанням. Для реалізації такого алгоритму всередині функції звернемося до об'єкта оновлень, у якого викличемо метод **message.reply\_text()**, до якого й передамо привітання:

```
def start(update=Update, context=CallbackContext):
    update.message.reply_text("Hi User")
```

Функція готова, але треба передати цей механізм обробнику оновлень. Для цього всередині `main()` створимо обробника команд, який передамо обробнику оновлень. Саму назву функції та текст команди передамо до обробника команд:

```
def main():
    updater =
    Updater("2052616546:AAHcWxtiGz8WvSUY9uQfo7R0nWV-
    YwEhCAE")
    dispatcher = updater.dispatcher
    dispatcher.add_handler(CommandHandler("start",
                                         start))

    updater.start_polling()
    updater.idle()
```

Зараз код бота має такий вигляд:

```
from telegram.ext import Updater, CommandHandler,
CallbackContext
from telegram import Update

def start(update=Update, context=CallbackContext):
    update.message.reply_text("Hi User")

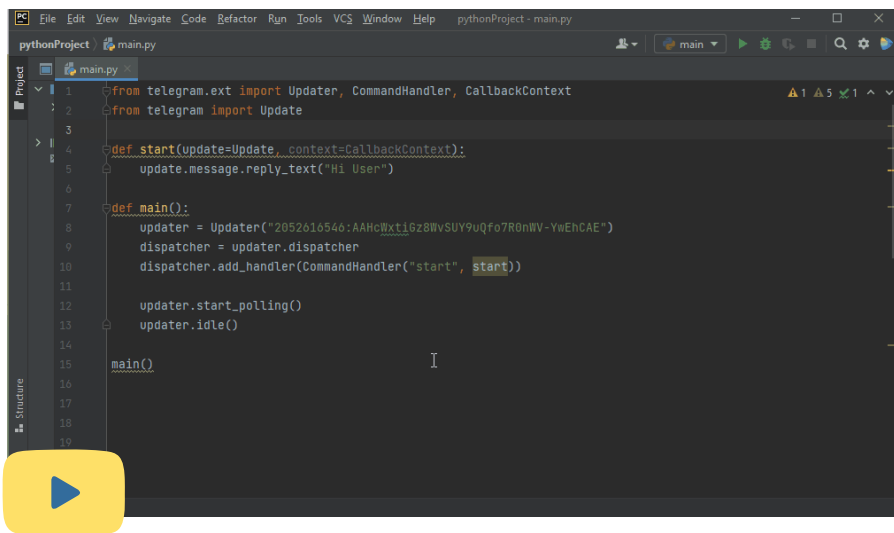
def main():
    updater =
    Updater("2052616546:AAHcWxtiGz8WvSUY9uQfo7R0nWV-
    YwEhCAE")
```

```
dispatcher = updater.dispatcher
dispatcher.add_handler(CommandHandler("start",
                                     start))
```

```
updater.start_polling()
updater.idle()
```

```
main()
```

Тепер запусимо код і відкриємо Telegram, де перейдемо до чату з ботом та надішлемо команду `/start` (відео 3):



Відео 3

Бот запрацював. Аналогічним способом можна додати ще кілька команд. Але як навчити бота працювати з текстом?

Створимо функцію, завдяки якій будь-який текст, що надсилається боту, буде надіслано користувачеві у відповідь. Нам уже відомо, як надсилати текст, але де ж узяти дані, що надсилає користувач? Знайти їх можна так само в об'єкті оновлень, звернувшись до `message.text`. У результаті дістанемо:

```
def echo(update=Update, context=CallbackContext):
    update.message.reply_text(update.message.
                              text)
```

Функцію ми написали, тож час попрацювати над її обробником. Імпортуємо `MessageHandler` та `Filters`. Перший – обробник повідомлень, другий – фільтр, який дасть змогу відрізнити команду від звичайного тексту. Зауважимо, що обробник повідомлень має бути лише один. Додамо його до обробника оновлень. Паралельно з цим передамо необхідні фільтри й розроблену функцію до обробника повідомлень:

```
from telegram.ext import Updater, CommandHandler,
CallbackContext, MessageHandler, Filters
from telegram import Update

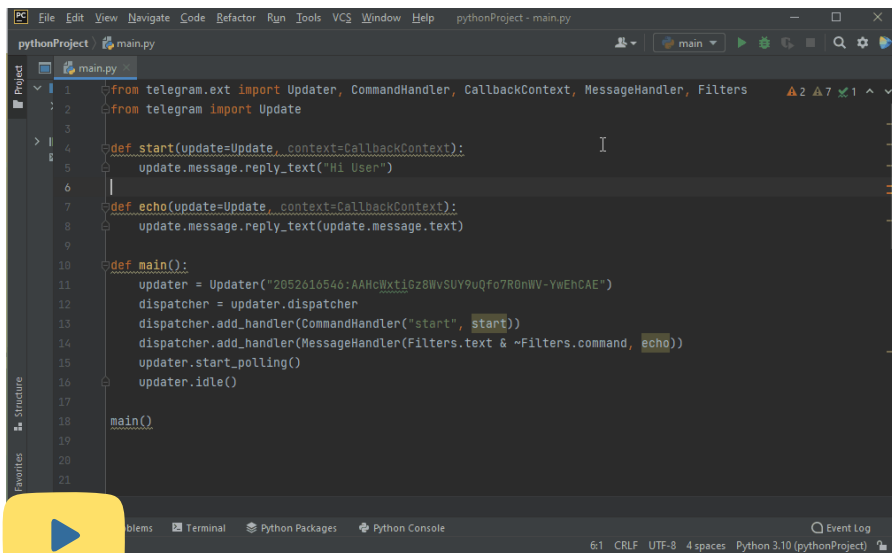
def start(update=Update, context=CallbackContext):
    update.message.reply_text("Hi User")

def echo(update=Update, context=CallbackContext):
    update.message.reply_text(update.message.text)
```

```
def main():
    updater =
    Updater("2052616546:AAHcWxtiGz8WvSUY9uQfo7R0nWV-
    YwEhCAE")
    dispatcher = updater.dispatcher
    dispatcher.add_handler(CommandHandler("start",
                                         start))
    dispatcher.add_handler(MessageHandler(Filters.
                                         text & ~Filters.command, echo))
    updater.start_polling()
    updater.idle()

main()
```

Запустимо код і поглянемо на результат (відео 4):



Відео 4



Сьогодні ми навчилися встановлювати зовнішні бібліотеки та працювати з ними, попрацювали з Telegram та створили власного бота-помічника.



## Урок 11

# БОТ-ПОМІЧНИК

© Комп'ютерна Академія ШАГ  
[www.itstep.org](http://www.itstep.org)

Усі права на фото-, аудіо- і відеотвори, що охороняються авторським правом і фрагменти яких використані в матеріалі, належать їх законним власникам. Фрагменти творів використовуються в ілюстративних цілях в обсязі, виправданому поставленим завданням, у рамках учбового процесу і в учбових цілях, відповідно до законодавства про вільне використання твору без згоди його автора (або іншої особи, яка має авторське право на цей твір). Обсяг і спосіб цитованих творів відповідає прийнятим нормам, не завдає збитку нормальному використанню об'єктів авторського права і не обмежує законні інтереси автора і правовласників. Цитовані фрагменти творів на момент використання не можуть бути замінені альтернативними аналогами, що не охороняються авторським правом, і відповідають критеріям добросовісного використання і чесного використання.

Усі права захищені. Повне або часткове копіювання матеріалів заборонене. Узгодження використання творів або їх фрагментів здійснюється з авторами і правовласниками. Погоджене використання матеріалів можливе тільки якщо вказано джерело.

Відповідальність за несанкціоноване копіювання і комерційне використання матеріалів визначається чинним законодавством України.