# Лабораторна робота №7

## Розробка чат-бота для мобільного месенджера на Python: Використання REST API

***Мета роботи:*** Навчитись використовувати засоби Telegram API для розробки чат боту

***Література***

Telegram Bot API: <https://core.telegram.org/bots/api>

Уроки по створенню ботів для Telegram: [https://mastergroosha.github.io/telegramtutorial](https://mastergroosha.github.io/telegram-tutorial/)

Створення кінцевого автомату для боту: [https://mastergroosha.github.io/telegramtutorial/docs/lesson\_11](https://mastergroosha.github.io/telegram-tutorial/docs/lesson_11/)

**Зміст роботи**

### ***Завдання 1.*** Ознайомитись з теоретичними відомостями***.***

### ***Завдання 2.*** Створити бот в Телеграмі та програмний код для запуску стартової команди.

### ***Завдання 3.*** Розробити бот з імплементацією кінцевого автомату за власним варіантом.

|  |  |
| --- | --- |
| Варіант  | Завдання  |
| 1, 6, 11, 16  | Бот, що надає прогноз погоди на поточний день за локацією, визначеною користувачем (користувач надсилає назву населеного пункту). Додати кнопки для запиту прогнозу на завтра та наступні 5 днів. Дані про погоду отримувати з одного з ресурсів на вибір: * <https://openweathermap.org/api>- необхідно зареєструватись на сайті та отримати ключ доступу до API
* [https://sinoptik.ua](https://sinoptik.ua/) - отримувати дані засобами парсингу HTML розмітки (-10)
 |
| 2, 7, 12, 17  | Бот, що надсилає випадкове зображення або зображення за пошуковим запитом користувача (користувач надсилає пошуковий запит). До повідомлення з випадковим зображенням додати посилання на URL-адресу зображення (Inline Keyboard Markup). Додати клавіатуру з кнопками для отримання ще одного зображення за тим же запитом, або іншого випадкового зображення. В якості ресурсу зображень використовувати систему Unsplash  |
|  | <https://unsplash.com/documentation>(необхідно зареєструватись на сайті та отримати ключ доступу до API)  |
| 3, 8, 13, 18  | Гра «Вгадай прапор». Бот надсилає випадкове зображення прапору певної країни, та дає 4 варіанти відповіді. Після кожної правильної відповіді користувача, гра повторюється. Додати клавіатуру з кнопками для відображення правильної відповіді та зупинки гри. Зображення прапорів можна завантажувати з сервісу [https://www.countryflags.io.](https://www.countryflags.io/) Перелік країн та скорочених назв країн можна знайти тут: <https://en.wikipedia.org/wiki/ISO_3166-1#Current_codes>(сформувати файл в проекті боту з повними та скороченими назвами країн, цей файл використовувати для вибору випадкової країни).  |
| 4, 9, 14, 19  | Гра «Вгадай столицю». Бот надсилає випадкову назву певної країни, та дає 4 варіанти відповіді з назвами столиць. Після кожної правильної відповіді користувача, гра повторюється. Додати клавіатуру з кнопками для відображення правильної відповіді та зупинки гри. Перелік країн та скорочених назв країн можна знайти тут: <https://en.wikipedia.org/wiki/ISO_3166-1#Current_codes>(сформувати файл в проекті боту з назвами країн, цей файл використовувати для вибору випадкової країни). Для знаходження столиці країни можна робити запит на сервіс [https://restcountries.eu,](https://restcountries.eu/) або доповнити локальний файл назвами столиць (-10).  |
| 5, 10, 15, 20  | Бот для конвертації валют. Бот працює в двох режимах: підрахунок вартості гривні за курсом іноземної валюти (Долар США та Євро) та підрахунок вартості іноземної валюти (Долар США або Євро) у гривні. Для курсу валют використовувати офіційні дані НБУ (у форматі XML або JSON, на вибір): * <https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange>
* [https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange ?json](https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange?json)
 |

## Методичні рекомендації *Створення чатботу*

Перед розробкою програмного коду, необхідно зареєструвати бот в системі Telegram в боті [BotFather.](https://telegram.me/BotFather) Зробити це можна за допомогою команди

“/newbot”. Також потрібно обрати унікальну назву для бота. Після створення, [BotFather](https://telegram.me/BotFather) надає токен для доступу до API телеграму. Цей токен потрібно зберегти для подальшого використання.



*Рис. 1. Створення боту в BotFather* BotFather дозволяє налаштовувати власні боти: встановлювати зображення, додавати опис, інформацію про автора та перелік доступних команд в боті. Проведіть ці налаштування для свого бота.

###  Створення проекту та запуск

Для розробки бота необхідно встановити бібліотеку pytelegrambotapi: pip install pytelegrambotapi

Імпортуємо бібліотеки для роботи з ботом та проводимо його ініціалізацію, використовуючи токен, наданий BotFather: import telebot

bot = telebot.TeleBot("162118050:\*\*\*\*\*\*\*\*\*\*\*\*N7B-NMeFIn0")

Створюємо обробник запиту команди користувача. За замовчуванням, кожен бот має команду запуску – start, тому опрацюємо цю команду.

# обробка команди start

@bot.message\_handler(commands=['start']) # функція що приймає повідомлення користувача def handle\_start\_command(message):

 # відправка відповіді

 bot.send\_message(message.chat.id, "Вітаю, " + message.chat.first\_name)

Щоб надіслати повідомлення від бота, потрібно знати id чату, в який повідомлення має потрапити. Цей id можна отримати з обє’кту message, який є аргументом фунції-обробника команди. Об’єкт message містить велику кількіть інформації, детальніше про об’єкт можна дізнатися в документації Bot API.

 Аналогічно до обробки команди start, створюються і функції обробки інших команд, створених у BotFather.

### Опрацювання повідомлень

 Розглянемо обробку текстових повідомлень користувача:

@bot.message\_handler(content\_types=['text']) def handle\_text(message): if message.text.lower() == 'привіт':

 bot.send\_message(message.chat.id, "Привіт 😊") else:

 bot.send\_message(message.chat.id, "Я отримав Ваше повідомлення!")

 Важливо пам’ятати про порядок виконання функцій опрацювання подій. При певній дії користувача послідовно перевіряються всі функції message\_handler і виконується перший обробник, що задовольняє умову.

Наприклад, спочатку описаний обробник текстових повідомлень, а після цього обробник команд. В такому випадку, обробник команд не виконається, тому що команда задовольнить умову перевірки на текстове повідомлення. Після опису коду обробки команд та повідомлень обов’язково додати виклик методу pooling. Це дозволить боту слідкувати за змінами в чаті та отримувати сповіщення про ці зміни.

bot.polling()

### Додавання персоналізованої клавіатури

 Телеграм боти підтримують можливість додавання персоналізованих кнопок, так званої клавіатури. Ця клавіатура використовується для полегшення комунікації між ботом та користувачем. Кнопки клавіатури можна відображати по замовчуванню після виклику команди start, після певної команди, або певного повідомлення. Розглянемо приклад додавання клавіатури з двох кнопок.

 Для зручності роботи з повідомленнями, винесемо всі повідомлення в окремі об’єкти. Щоб розділити текст від основного коду, краще винести ці об’єкти в окремий файл.

# Об'єкт з підписами до кнопок button\_captions = {

 "hello": "Привіт",

 "plain": "Звичайна кнопка"

}

# Об'єкт з відповідями на повідомлення reply\_messages = {

 "hello\_button": "Привіт 😊",

 "plain\_button": "Ви натиснули звичайну кнопку"

}

Створюємо клавіатуру з двома кнопками та додаємо її на команду start.

# Створення та конфігурація клавіатури start\_keyboard = telebot.types.ReplyKeyboardMarkup(True, True) start\_keyboard.row(button\_captions["hello"], button\_captions["plain"])

@bot.message\_handler(commands=['start']) def handle\_start\_command(message):

 bot.send\_message(message.chat.id, "Вітаю!", reply\_markup=start\_keyboard)

Змінимо обробку отримання текстових повідомлень з урахуванням роботи кнопок.

@bot.message\_handler(content\_types=['text']) def handle\_text(message): if message.text == button\_captions["hello"]:

 bot.send\_message(message.chat.id, reply\_messages["hello\_button"]) elif message.text == button\_captions["plain"]:

 bot.send\_message(message.chat.id, reply\_messages["plain\_button"])

Щоб прибрати клавіатуру, необхідно використати наступний код:

# створюємо порожню клавіатуру empty\_keybord = telebot.types.ReplyKeyboardRemove()

# надсилаємо повідомлення без клавіатури

bot.send\_message(message.chat.id, reply\_messages["removed\_buttons"], reply

\_markup=empty\_keybord)

На рисунку 2 зображена персоналізована клавіатура бота.



*Рис. 2. Персоналізована клавіатура*

Детальніше про створення клавіатури можна почитати на сторінці API: [ReplyKeyboardMarkup](https://core.telegram.org/bots/api#replykeyboardmarkup) та [ReplyKeyboardRemove.](https://core.telegram.org/bots/api#replykeyboardremove)

### Додавання кнопок до повідомлення

Окрім клавіатури, можна додавати кнопки до певного повідомлення, що надсилає бот. Це може бути кнопка-посилання на ресурс (рис. 3), або кнопка, що викликає подію (рис. 4).



*Рис. 3. Кнопка-посилання* Розглянемо створення кнопки-посилання:

# створюємо контейнер для кнопок keyboard = InlineKeyboardMarkup()

# створюємо кнопку-посилання

url\_button = InlineKeyboardButton(text="Gooole", url="https://google.com")

# додаємо кнопку до контейнеру keyboard.add(url\_button)

# додаємо контейнер з кнопками до повідомлення bot.send\_message(message.chat.id, reply\_messages["google\_button"], reply\_m arkup=keyboard)

Розглянемо створення кнопки, що викликає подію:

Створюємо клавіатуру

# створюємо контейнер для кнопок callback\_keybord = InlineKeyboardMarkup() # створюємо кнопку, що викликає подію callback\_button = InlineKeyboardButton(text="Натисни на мене", callback\_da ta="test")

# додаємо кнопку до контейнеру callback\_keybord.add(callback\_button)

Додаємо клавіатуру

# обробка текстового повідомлення

@bot.message\_handler(content\_types=['text']) def handle\_text(message): if message.text == "test":

 # надсилаємо повідомлення з новою клавіатурою bot.send\_message(message.chat.id, reply\_messages["test\_button"], rep ly\_markup=callback\_keybord)

Зазвичай, при виведенні результату на клік кнопки, що викликає подію, повідомлення з кнопкою замінюють. Для цього використовується функція edit\_message\_text:

@bot.callback\_query\_handler(func=lambda call: True)

def callback\_inline(call):

 # відповідаємо на певну кнопку з даними "test" if call.data == "test": # змінюємо повідомлення bot.edit\_message\_text(chat\_id=call.message.chat.id, message\_id=call.

message.message\_id, text="Повідомлення з тестовою кнопкою змінилось")


## Рис. 4. Робота кнопки, що викликає подію Надсилання зображення

# імпортуємо необхідні бібліотеки import requests import io

# створюємо функцію яка отримує зображення def get\_random\_photo():

 response = requests.get("https://picsum.photos/600") picture = io.BytesIO(response.content) picture.seek(0) return picture

# надсилаємо зображення photo = get\_random\_photo() bot.send\_photo(message.chat.id, photo=photo, caption="Random photo")

Розглянуті в методичних рекомендаціях теми, допоможуть розробити програмний код для роботи простого боту. Для того, щоб бот умів виконувати більш складні логічні операції та слідкувати за послідовністю запитів користувача, необхідно імплементувати простий кінцевий автомат.

## Контрольні запитання

1. Для чого потрібен токен боту?
2. Яким чином можна сформувати відповідь на відправку користувачем тексту, відео чи аудіо?
3. В чому різниця між InlineKeyboardMarkup та

ReplyKeyboardMarkup?

1. Як можна видалити персоналізовану клавіатуру?
2. Що таке polling і для чого він потрібен?
3. Для чого потрібні кінцеві автомати?