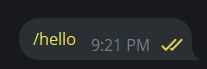
**Lesson 27. CommandBot**

Один із найпростіших в той же час найбільш надійних способів взаємодіяти із ботом – команди. Їх просто налаштувати, їх просто обробляти, вони зазвичай мають однозначну відповідь простий алгоритм.

Отже пропонуємо цього разу познайомимось із командами в Telegram, а також пригадаємо як працювати із емоджі.

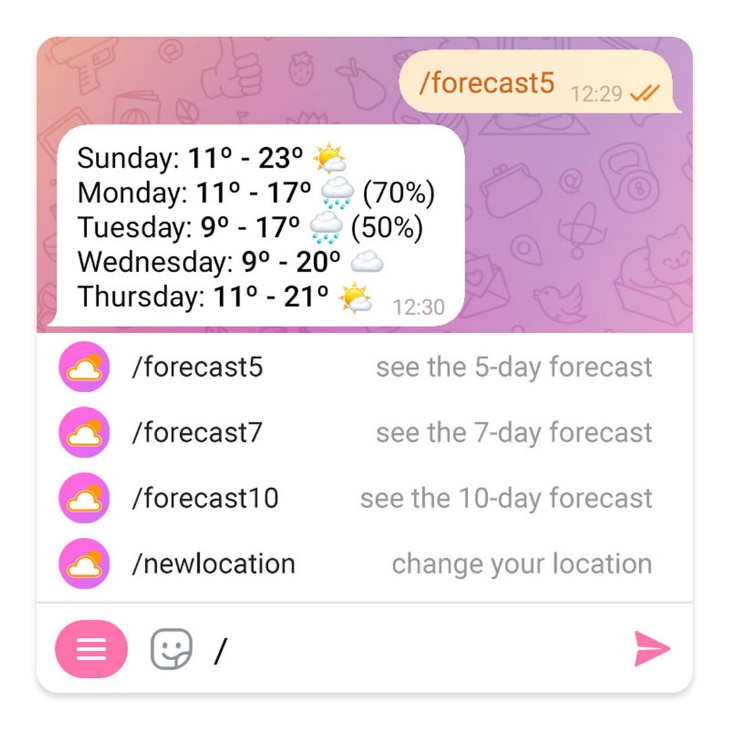
**Commands**

Користувачі можуть надсилати ботам будь-які типи повідомлень, включаючи текст, файли, геолокацію, стікери, голосові повідомлення та навіть гральні кубики для отримання випадкових значень. Однак боти Telegram пропонують також багато інших інструментів для створення гнучких інтерфейсів, адаптованих до ваших конкретних потреб.

Один із них – **команди**, які підсвічуються в повідомленнях і які можна вибрати зі списку після введення символу «**/**».

**Команда** – це всього лиш повідомлення у формі ***/keyword***, яке вказує боту, що робити.

Застосунок Telegram у роботі з командами буде:

* **Виділяти команди в повідомленнях**. Коли користувач натискає на виділену команду, ця команда негайно надсилається знову.
* **Пропонувати список підтримуваних команд** з описами, коли користувач вводить **/** (щоб це працювало, необхідно надати список команд *@BotFather* або за допомогою API). Вибір команди зі списку негайно надсилає її.
* **Показувати кнопку меню**, яка містить усі чи деякі команди бота (які визначені через @BotFather).

Команди завжди мають починатися з символу «**/**» (слеш) і містити до 32 символів. Вони можуть використовувати латинські літери, цифри та підкреслення, хоча для чіткішого вигляду рекомендується використовувати простий текст у нижньому регістрі.

Ось кілька прикладів:

Команди мають бути **якомога конкретнішими** – наприклад, /newlocation або /newrule є кращими, ніж /new команда, яка вимагає від користувача додаткового параметра, наприклад «location» або «rule».

Щоб зробити основні взаємодії більш уніфікованими, розробники телеграм просять всіх розробників ботів підтримувати кілька основних команд. Додатки Telegram матимуть інтерфейсні ярлики для цих команд.

**/start** – починає взаємодію з користувачем, як відправка вступного повідомлення.

**/help** – повертає довідкове повідомлення, наприклад короткий текст про те, що може робити ваш бот, і список команд.

**/settings** – (якщо застосовується) показує налаштування бота для цього користувача та пропонує команди для їх редагування.

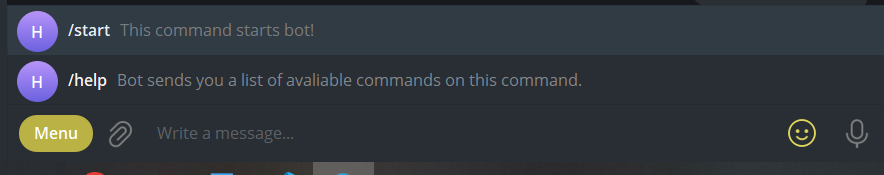
Користувачі побачать кнопку **«Start»**, коли вперше відкриють чат із вашим ботом. Посилання на /help та /settings будуть доступні в меню на сторінці профілю бота, якщо додати їх у @BotFather.

**Register Commands**

Зареєструвати команди через Батька всіх ботів (<https://t.me/BotFather>) не складно (за допомогою коду це теж можна зробити, але з цим познайомимось трохи згодом):

1. У списку ваших ботів обираємо потрібного;
2. Тиснемо Edit Bot;
3. Тиснемо Edit Commands;
4. Надаємо список у вигляді, який просить бот.

Після цього в меню бота повинні з’явитись додані команди. Наприклад:



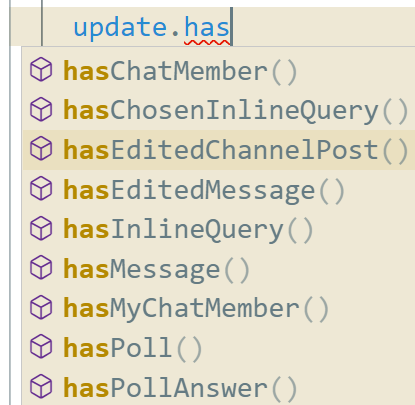
**IsCommand**

У цьому невеличкому пункті розглянемо, яким чином можна найпростіше розпізнати у повідомленні команду, надіслану телеграм-боту.

Розглянемо простеньку, але дуже корисну для нашої цілі реалізацію методу OnUpdateReceived на початку наступної сторінки. Прочитавши це шматочок коду просто як текст англійською мовою можем зрозуміти:

«*Якщо в апдейті бота міститься повідомлення, то перевіряємо, чи це повідомлення є командою. Якщо так і бот отримав команду, то обробляємо повідомлення як команду, а інакше – як усі інші повідомлення.*» Як саме в



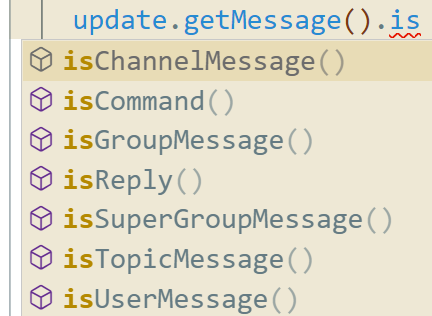
Зупинимось детальніше на методах, які ми використали у даному фрагменті:

1. hasMessage() – метод, що належить класу Update. Він один із багатьох інших методів даного класу, що дозволяє встановити наявність в апдейті того чи іншого об’єкту (повідомлення, голосування, посту, тощо). Якщо скористатись опцією автозаповнення почавши писати в коді update.has… можна побачити і інші подібні методи (на скріншоті не всі очевидно):

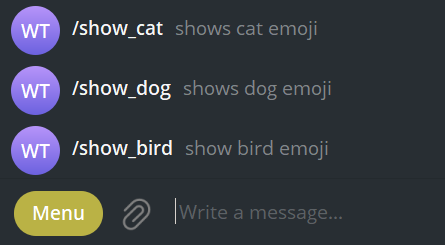
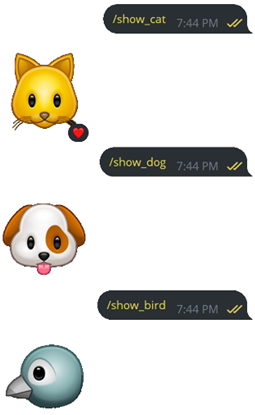


1. getMessage() – метод, що належить класу Update, який дозволяє отримати об’єкт повідомлення класу Message із об’єкту Update. Інші методи, які також починаються зі слова get і дозволяють отримати із апдейту інші об’єкти можна також переглянути за допомогою функціоналу автозаповнення:
2. isCommand() – метод класу Message, що дозволяє визначити, чи отримане повідомлення можна вважати командою за всіма правилами надсилання команд у телеграм (метод повертає очевидно значення boolean).

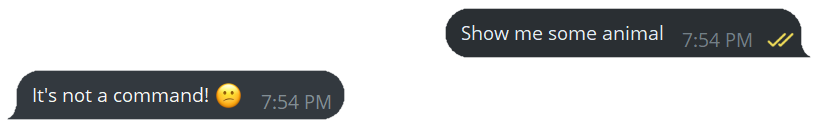
Зверніть увагу на наступний запис:

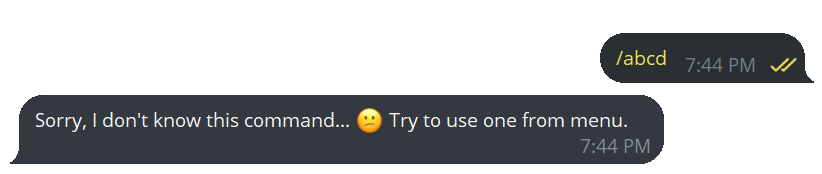
У даному випадку ми отримавши доступ до об’єкта Message у Update, одразу ж виконуємо вже його власний метод (метод повідомлення). Таким чином може бути побудований ланцюжок із кількох об’єктів, які містяться один в одному. Це дуже зручно і ефективно, оскільки виключає створення зайвих змінних-об’єктів.

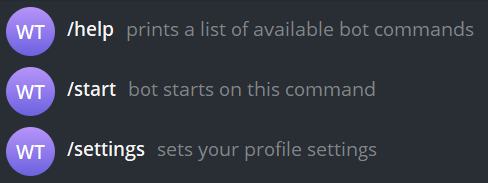
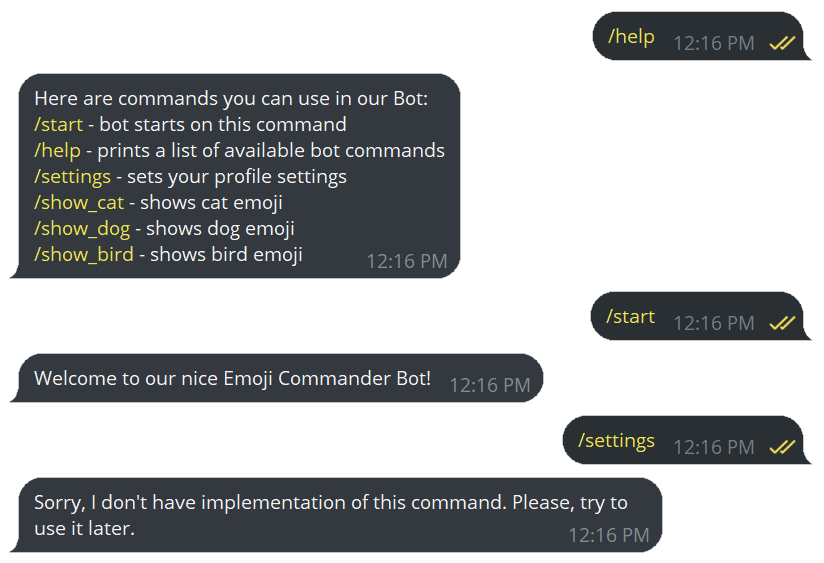
За допомогою інших подібних методів об’єкта Message можемо отримати також іншу додаткову інформацію про отримане повідомлення:

**Завдання 1**. Зареєструйте для свого бота три команди як показано на рисунку нижче, за допомогою функціоналу бота BotFather. При отриманні одної із трьох команд, бот у відповідь повинен відправляти відповідний емоджі.

**CommandBot**

Якщо ж бот отримує невідому команду, то відповідає ось таким повідомленням:

Отримавши текстове повідомлення, яку не є командою бот відповідає:

**Завдання 2**. Додайте боту із попереднього завдання ще три команди (ту самі що розробники телеграм рекомендують додавати всім телеграм-ботам). Приклад відповідей на дані команди показано на рисунку нижче:

**Homework**

**Завдання 1.** Створіть бота із трьома командами: /start (бот вітається), /help (бот надсилає перелік доступних команд), /chat\_id (надсилає id чату із якого було отримано команду).