УДК 004.43 Python : 004.428 DOI: 10.31673/2412-9070.2024.025559

М. Ю. СМЕТАНА, студент;
В. Б. БІЛАВКА, аспірант,
Державний університет інформаційно-комунікаційних технологій, Київ

ВЗАЄМОДІЯ З КОРИСТУВАЧЕМ ЗА ДОПОМОГОЮ МАГІЧНИХ ФІЛЬТРІВ У AIOGRAM: ВПЛИВ PYTHON НА ЕФЕКТИВНІСТЬ ЧАТ-БОТІВ

Досліджено, як Python, зокрема через бібліотеку «Aiogram», дає можливість реалізувати магічні фільтри, та як вони можуть змінити функціональність будь-якого чат-боту. Розглянуто проблеми та обмеження, які слід брати до уваги, а також потенціал і можливості використання магічних фільтрів в aiogram для майбутніх розробок.

Ключові слова: Python; бібліотека «Aiogram»; магічні фільтри; чат-боти; взаємодія з користувачем.

Вступ

Python, будучи однією з найпопулярніших мов програмування для штучного інтелекту (ШІ) та машинного навчання, пропонує широкий спектр бібліотек і фреймворків, які спрощують упровадження магічних фільтрів у чат-ботах. Однією з таких бібліотек є «Aiogram», потужний і гнучкий фреймворк для створення ботів Telegram за допомогою Python. Завдяки aiogram розробники можуть використовувати можливості оброблення мови Python для створення складних магічних фільтрів, які поліпшують функціональність і взаємодію з користувачем їхніх чат-ботів.

Aiogram надає багатий набір інструментів і функцій, які спрощують реалізацію магічних фільтрів. Він пропонує вбудоване підтримання обробників повідомлень, які дають змогу розробникам визначати власні фільтри на основі різних критеріїв, зокрема вмісту повідомлення, ідентифікатора користувача, типу чата тощо. Це уможливлює створення чітко націлених та залежних від контексту магічних фільтрів, які здатні отримувати релевантну інформацію з повідомлень користувача та запускати певні дії чи відповіді.

Аналіз. У сучасному цифровому середовищі, що швидко розвивається, чат-боти стали невід'ємною частиною стратегії обслуговування клієнтів компаній. Одним із ключових чинників, які визначають успіх чат-боту, є якість роботи з користувачем. Щоб покращити взаємодію та зробити спілкування з користувачем ефективнішим, розробники почали впроваджувати магічні фільтри у своїх структурах чат-ботів, зокрема для наведених далі завдань.

1. *Розпізнавання сутностей.* За допомогою магічних фільтрів чат-бот може легко ідентифіковувати та витягувати конкретні сутності з повідомлень користувача. Наприклад, чат-бот, створений для служби доставляння їжі, може використовувати магічний фільтр для виявлення продуктів харчування, зазначених у повідомленні користувача, і виокремлювати відповідні деталі, зокрема кількість і вподобання. Це дає змогу чат-боту надавати персоналізовані рекомендації та спрощувати процес замовлення.

2. *Аналіз настрою.* Магічні фільтри також можна використовувати для аналізу настрою повідомлень користувачів. Застосовуючи алгоритми аналізу настроїв, чат-бот може визначити, чи є повідомлення користувача позитивним, негативним або нейтральним. Цією інформацією можна скористатися для відповідного налаштування відповідей чат-боту, надаючи щире та належне підтримання користувачам на основі їхніх емоцій.

3. *Визначення мови.* У глобалізованому світі чат-ботам здебільшого потрібно обробляти взаємодію кількома мовами. Магічні фільтри можна застосовувати для автоматичного визначення мови повідомлення користувача та надання відповідей відповідною мовою. Це позбавляє користувачів від потреби явно зазначати бажану мову, роблячи взаємодію більш зручнішою та неперервною.

Мета дослідження — розглянути можливості використання Python і магічних фільтрів для створення ефективних і зручних чат-ботів, дослідити різні функції, дії та сценарії використання магічних фільтрів, а також розробити інструкцію з використання магічних фільтрів та їхню ефективну імплементацію.

Основна частина

Знайомство з магічними фільтрами в aiogram. Magic filters — це зовнішній пакет, який підтримує основна бібліотека «Aiogram». Його інсталюють як усталене налаштування разом із aiogram, він також доступний у PyPi — magic-filter (https://pypi.org/project/magic-filter/). Тобто ви маєте можливість інсталювати та використовувати магічний фільтр з іншими бібліотеками та у власних проєктах, незалежно від того, чи інстальовано у вас aiogram. Проте слід зауважити, що aiogram має невелике розширення над magic filter. Якщо ви хочете використовувати це розширення, вам слід імпортувати його з aiogram замість пакету magic_filter, тобто

```
from aiogram import F
```

замість

```
from magic_filter import F
```

Вивчення можливих дій за допомогою магічних фільтрів. Об'єкт magic filter підтримує кілька основних логічних операцій над атрибутами об'єкта.

Розглянемо докладніше деякі з можливих дій.

• *Існує або не існує*. Дія за замовчуванням перевіряє, існує чи ні певний атрибут.

Наприклад:

F.photo перевіряє наявність атрибута photo в об'єкті.

• *Дорівнює*. Оператор == перевіряє, чи дорівнює атрибут певному значенню.

Наприклад:

F.text == 'hello' перевіряє, чи текст атрибута дорівнює рядку 'hello'.

• *Є одним із*. Метод in_ або оператор @ можна використовувати, щоб перевірити, чи є атрибут одним із значень в ітерації.

Наприклад:

F.from_user.id.in_({42, 1000, 123123}) перевіряє, чи дорівнює атрибут from_user.id будь-якому зі значень у наборі {42, 1000, 123123}.

• *Містить*. Метод contains перевіряє, чи містить атрибут рядка певний підрядок.

Наприклад:

F.text.contains('foo') перевіряє, чи текст атрибута містить підрядок 'foo'.

• *Startswith/Endswith*. Методи startswith і endswith можна застосовувати лише до текстових атрибутів.

Наприклад:

F.text.startswith('foo') перевіряє, чи починається текст атрибута з рядка 'foo'.

• *Регулярний вираз*. Метод regexp дає змогу використовувати регулярні вирази для зіставлення значень атрибутів.

Наприклад:

F.text.regexp(r'Hello, .+') перевіряє, чи текст атрибута відповідає шаблону регулярного виразу 'Hello, .+'.

• *Власні функції*. Ви також можете використовувати власну функцію як дію.

Наприклад:

F.chat.func(lambda chat: chat.id == -42) перевіряє, чи дорівнює атрибут chat.id -42 за допомогою власної функції.

• *Інвертування результату*. Будь-яку доступну операцію можна інвертувати за допомогою оператора побітової інверсії ~.

Наприклад:

~(F.text == 'spam') інвертує результат операції F.text =='spam'.

• *Комбінування дій*. Усі операції можна комбінувати за допомогою побітових операторів і (&) та або (|).

Наприклад:

(F.from_user.id == 42) & (F.text == 'admin') перевіряє, чи атрибути from_user.id і text задовольняють заданим умовам.

• *Модифікатори атрибутів — маніпуляції рядками*. Магічні фільтри також дають можливість використовувати модифікатори атрибутів, зокрема для рядкових атрибутів. Ці модифікатори можна застосовувати для керування регістром атрибута рядка.

Наприклад:

F.text.lower() == 'test' перевіряє, чи текст атрибута дорівнює рядку 'test' у нижньому регістрі.

F.text.upper().in_({'FOO', 'BAR'}) перевіряє, чи версія тексту атрибута у верхньому регістрі дорівнює будь-якому з рядків у наборі {'FOO', 'BAR'}.

F.text.len() == 5 перевіряє, чи довжина тексту атрибута дорівнює 5.

• *Отримати результат фільтра як аргумент обробника*. Хоча ця функція недоступна безпосередньо у magic filter, її можна використовувати в поєднанні з aiogram. Імпортувавши *F* з aiogram, ви можете використовувати цю функцію.

Наприклад:

```
from aiogram import F
...
@router.message(F.text.regexp(r"^(\d+)$").as_("digits"))
async def any_digits_handler(message: Message, digits: Match[str]):
    await message.answer(html.quote(str(digits)))
```

У цьому прикладі фільтр F.text.regexp(r"^(\d+)$").as_("digits") використовується для отримання цифр із текстового атрибута повідомлення. Здобуті цифри потім передаються як аргумент функції any_digits_handler.

Використання магічних фільтрів в aiogram. Тепер, після розгляду різних дій і функцій magic filter, наведемо деякі типові сценарії використання в aiogram:

• @router.message(F.text == 'hello'): цей фільтр запускає обробник, коли надходить повідомлення з точним текстом 'hello';

• @router.inline_query(F.data == 'button:1'): цей фільтр запускає обробник, коли отримано вбудований запит із даними 'button:1';

• @router.message(F.text.startswith('foo')): цей фільтр запускає обробник, коли надходить повідомлення з текстом, який починається з 'foo';

• @router.message(F.content_type.in_({'text', 'sticker'})): цей фільтр запускає обробник, коли надходить повідомлення з типом вмісту тексту або наклейки;

• @router.message(F.text.regexp(r'\d+')): цей фільтр запускає обробник, коли надходить повідомлення з текстом, що містить одну чи більше цифр.

Це лише кілька прикладів того, як магічні фільтри можна використовувати в aiogram. Існує багато інших випадків, коли може знадобитися перевірити будь-який із доступних атрибутів події за допомогою магічних фільтрів.

Обмеження та проблеми використання магічних фільтрів у чат-ботах. Хоча магічні фільтри пропонують численні переваги, важливо визнати їх обмеження та проблеми. Ось кілька ключових міркувань.

1. *Складність реалізації.* Упровадження магічних фільтрів може бути складним завданням, особливо для розробників, які не знайомі з обробленням природної мови та машинним навчанням. Це потребує глибокого розуміння алгоритмів, попереднього оброблення даних і методів навчання моделі.

2. *Якість і доступність даних.* Точність і ефективність магічних фільтрів переважно залежать від якості та доступності навчальних даних. Збір і оброблення відповідних даних може бути трудомістким і ресурсомістким процесом.

3. *Мовні та культурні нюанси.* Магічним фільтрам може бути важко зрозуміти мовні та культурні нюанси, що призводить до неправильного тлумачення або неправильних відповідей. З огляду на ці нюанси важливо постійно вдосконалювати магічні фільтри для підвищення їх точності.

Найкращі практики впровадження магічних фільтрів в aiogram. Хоча магічні фільтри можуть значно підвищити функціональність вашого чат-боту, слід враховувати найкращі практики, щоб забезпечити їх ефективне впровадження. Ось кілька порад, які варто взяти до уваги.

1. *Зрозумійте своїх користувачів.* Перш ніж використовувати магічні фільтри, дуже важливо мати повне розуміння вашої цільової аудиторії та її потреби. Це допоможе вам розробити правильні шаблони та програми для захоплення за допомогою магічних фільтрів, гарантуючи, що вони очікують від користувачів.

2. *Тестуйте та повторюйте.* Магічні фільтри можуть використовувати точні налаштування для досягнення оптимальної продуктивності. Перевірте свої магічні фільтри та зберіть відгуки користувачів, щоб удосконалити програму. Повторюйте свої фільтри на основі інформації, здобутої під час взаємодії користувачів, постійно підвищуючи їх точність і ефективність.

3. *Підтримуйте баланс.* Хоча магічні фільтри можуть автоматизувати багато завдань, важливо знайти баланс між автоматизацією та втручанням людини. Для деяких взаємодій може знадобитися особистий підхід або людський досвід, тому для вирішення усіх потреб обов'язково надайте резервні варіанти.

Обговорення результатів. Упровадження магічних фільтрів у чат-ботах може принести багато переваг як бізнесу, так і клієнтам. Наведемо деякі з них.

1. *Покращення взаємодії з користувачем.* Магічні фільтри дають можливість чат-ботам більше розуміти та інтерпретувати введені користувачем дані, що забезпечує персоналізованішу та ефективнішу взаємодію. Це сприяє загальній взаємодії з користувачем, зумовлюючи підвищення рівня задоволеності та лояльності клієнтів.

2. *Економія часу та коштів*. Завдяки автоматизації таких завдань, як розпізнавання об'єктів і аналіз настроїв, магічні фільтри можуть значно скоротити час і зусилля, потрібні для оброблення повідомлень користувачів. Це не тільки заощаджує цінні ресурси, а й дає змогу чат-ботам надавати швидші та точніші відповіді.

3. *Покращена масштабованість і гнучкість*. Магічні фільтри здатні легко адаптуватися до мінливих потреб і вподобань користувачів. Відповідно до того, як чат-бот розвивається, можливо оновлювати та вдосконалювати магічні фільтри для оброблення нових шаблонів або намірів, забезпечуючи актуальність та ефективність чат-боту.

4. *Статистика на основі даних.* Магічні фільтри можуть генерувати цінну інформацію про поведінку та вподобання користувачів. Аналізуючи здобуті дані, можна глибше зрозуміти своїх клієнтів і ухвалювати рішення на основі даних, щоб підвищити продуктивність і ефективність чат-боту.

Висновки

Отже, магічні фільтри на основі Python, реалізовані в aiogram, пропонують потужний набір інструментів для поліпшення взаємодії з чат-ботами. Використовуючи методи оброблення природної мови, магічні фільтри дають можливість чат-ботам розуміти та інтерпретувати введені користувачем дані більш розумно та контекстно. Це сприяє більш ефективній та персоналізованій взаємодії, що зрештою зумовлює підвищення рівня задоволеності та лояльності клієнтів. Хоча існують проблеми та обмеження, на які слід зважати, потенціал магічних фільтрів в аіограмі величезний із цікавими можливостями для майбутніх розробок. Оскільки компанії продовжують використовувати чат-боти як невід'ємну частину своїх стратегій щодо обслуговування клієнтів, застосування Python і магічних фільтрів матиме вирішальне значення для створення ефективних і зручних чат-ботів.

Список використаної літератури

1. ***aiogram*** *Documentation Release 3.2.0. URL: https://readthedocs.org/projects/aiogram/downloads/pdf/latest/.*

*2. **Документація** та приклади, надані командою aiogram. URL: https://docs.aiogram.dev/en/dev-3.x/dispatcher/filters/magic_filters.html.*

M. Smetana, V. Bilavka

HOW PYTHON BRINGS EFFICIENCY TO CHATBOTS: ENHANCING USER EXPERIENCE WITH MAGIC FILTERS IN AIOGRAM

User experience is of immense importance, especially when it comes to chatbots. Continuous and intuitive interaction with a chatbot can significantly elevate customer satisfaction and loyalty. Conversely, a poor user experience can lead to disappointment and abandonment, ultimately harming the reputation and effectiveness of the chatbot. To ensure a positive user experience, chatbot developers need to focus on designing conversations that are natural, context-aware, and efficient.

Chatbots have become a key element of customer service in the digital environment, and this article explores how the Python programming language, particularly through the aiogram library, allows the implementation of magic filters and how they can alter the functionality of any chatbot. The article discusses issues and limitations to consider, as well as the potential and possibilities of using magic filters in aiogram for future developments. Magic filters enhance user interaction, recognize entities, analyze moods, and determine the language of messages.

The research aims to explore the possibilities of using Python and magic filters to enhance chatbots using aiogram. The goal is to create a guide on using magic filters and their effective implementation.

It's essential to consider the complexity of implementing magic filters, the need for quality data, and understanding linguistic and cultural nuances. These aspects can pose challenges for developers. Recommendations include understanding the target audience, testing, and maintaining a balance between automation and human intervention.

Developing chatbots using Python and aiogram with magic filters opens up broad possibilities for improving interaction and functionality. Understanding limitations and implementing best practices are key to success.

The implementation of magic filters in chatbots brings numerous benefits for businesses and users. Personalized interaction, time and cost savings, improved scalability, and data-driven statistics collection are key advantages that can significantly enhance chatbot functionality and user satisfaction.

Keywords: Python; library aiogram; magic filters; chatbots; user experience.

ЗВ'ЯЗОК

Наукове фахове видання

Редакційна обробка та коректура
Т. В. Ількевич

Комп'ютерна верстка та дизайн
Г. С. Тимченко

Відповідальний за випуск
І. І. Тищенко

Формат 60×84/8. Папір друкарський.
Гарнітура SchoolBookC, EuropeCond. Зам. 270
Наклад 300 прим.

Державний університет інформаційно-комунікаційних технологій
03110, м. Київ, вул. Солом'янська, 7
Тел. (044) 249-25-75
E-mail: *zviaz-ok@ukr.net*