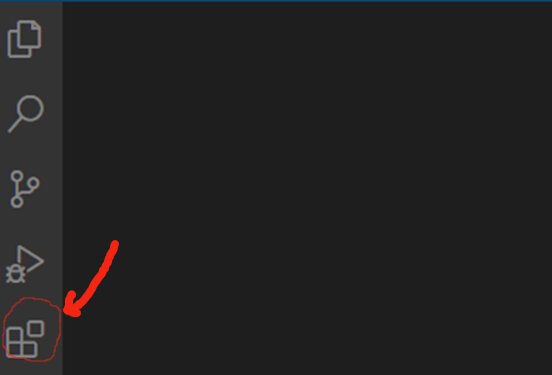
**ЗАДАНИЕ №8**

**Освоение Aiogram. Разработка простейшего Telegram-бота на Python с использованием фреймворка Aiogram. Добавление новых команд.**

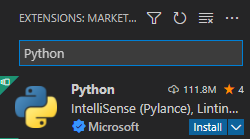
**Цель работы**: научиться использовать фреймворк *Aiogram* для разработки Telegram-бота на языке Python. Освоить основы создания команд и их обработку.

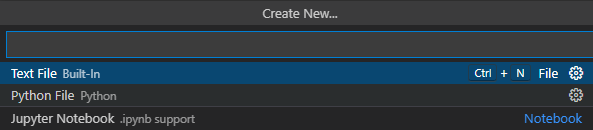
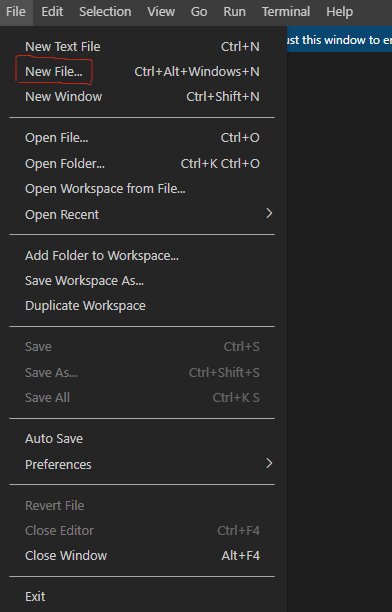
Прежде чем начать делать бота нужно сделать следующее:

1. Для начала работы с Python нужно скачать в Visual Studio Code соответствующий плагин
   1. Переходим в **«Extensions»**

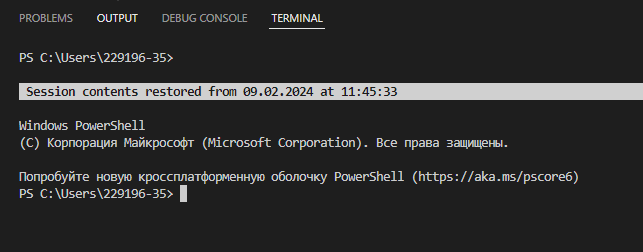


* 1. В строке поиска пишем **«Python»**, нажимаем **«Enter»** и скачиваем первый плагин, нажав **«Install»**. Ждем скачивания



* 1. Создаем новый файл **(File -> New File…)**, затем выбираем **(Python File)**

1. После установки плагина нужно настроить виртуальное окружение
   1. Открываем терминал, нажав горячие клавиши **(Ctrl + `)**

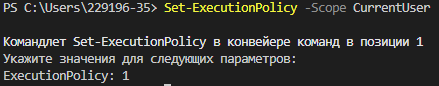


* 1. Создаем виртуальное окружение, прописав в терминале команду, и выполнив



Назовем виртуальное окружение **«venv»**

* 1. Запускаем от имени администратора и в поле **«ExexcutionPolicy»** вводим **«1»**



* 1. Активируем виртуальное окружение



* 1. Скачиваем библиотеку **«aiogram»** нужной версии



**Теоретические сведения:**

**I.** **Асинхронность в Python**

Асинхронное программирование в Python основано на использовании ключевых слов *async* и *await*. Это позволяет создавать асинхронные функции, которые могут выполняться параллельно, не блокируя выполнение других частей программы. В Python асинхронность реализуется с помощью библиотеки *asyncio*

1. Библиотека *asyncio*
   1. *asyncio* — это стандартная библиотека Python, предназначенная для асинхронного программирования. Она включает в себя ряд компонентов, таких как цикл событий (event loop), корутины, задачи (tasks), а также инструменты для ввода-вывода и сетевого взаимодействия.
2. *async/await*
   1. Ключевое слово *async* используется для определения асинхронных функций.
   2. Ключевое слово *await* используется для вызова асинхронных операций, которые могут занимать некоторое время.

Пример создания асинхронной функции (рис. 1):

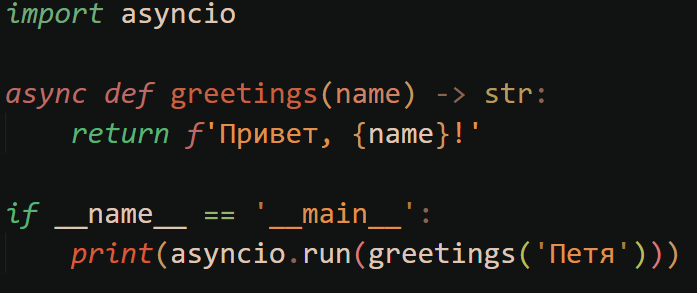


Рисунок 1 - пример асинхронной функции c использованием модуля asyncio

Этот пример демонстрирует асинхронную функцию, которая выводит приветственное сообщение по полученному имени.

**II. Модуль aiogram**

1. Краткое описание

*Aiogram* — это асинхронный фреймворк для разработки Telegram-ботов на языке Python. Он предоставляет удобные инструменты для взаимодействия с API Telegram, обработки событий и команд бота. *Aiogram* построен поверх библиотеки *asyncio* и облегчает создание эффективных и отзывчивых ботов. Благодаря асинхронности *Aiogram* может выдерживать большую нагрузку и огромное количество пользователей одновременно.

1. Установка

Для создания бота на основе фреймворка *Aiogram*, вам необходимо установить соответствующую библиотеку с использованием инструмента управления пакетами Python - *pip*

* 1. Откройте среду разработки Visual Studio Code
  2. Нажмите кнопку «Терминал» на верхней панели, далее – «Создать терминал»
  3. Введите в терминале команду *pip install aiogram==2.25.1*
  4. Ожидайте завершения установки. После завершения установки фреймворк готов к использованию.

1. Основная информация об обработчиках и типах сообщений
   1. Обработчики

Обработчики в *Aiogram* - это функции, которые вызываются при определенных событиях, таких как получение сообщений, команд, коллбэков и др. Обработчики регистрируются с использованием декораторов, таких как *@dp.message\_handler() (для текстовых сообщений)*, *@dp.callback\_query\_handler() (для клавиатур)*, и другие. В *Aiogram*, в аргументы декоратора *@dp.message\_handler()* можно передать параметр commands, который определяет, на какие команды должен реагировать обработчик сообщений. Параметр *commands* принимает список строк, представляющих собой названия команд (рис. 2)

**Так как в *Aiogram* последовательная регистрация обработчиков, то если в коде есть обработчик всех текстовых сообщений - обработчики команд ставятся строго выше него!**

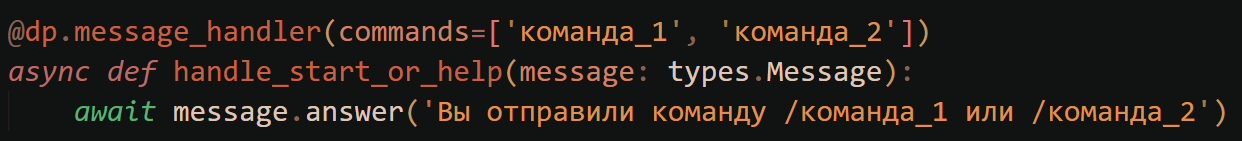


Рисунок 2 - пример декоратора с командами в аргументе

* 1. Типы сообщений

В *Aiogram* есть множество типов сообщений, но мы рассмотрим только *types.Message.* Он представляет собой объект, содержащий всю информацию о сообщении, полученном от пользователя в Telegram. В контексте *Aiogram*, это является одним из ключевых типов, который позволяет боту взаимодействовать с пользователями. Рассмотрим подробнее атрибут *message.from\_user*, предоставляющий информацию о пользователе, отправившем сообщение.

Объект *message.from\_user* предоставляет различные атрибуты с информацией о пользователе, включая:

* *id*: Уникальный идентификатор пользователя в Telegram.
* *is\_bot*: Является ли пользователь ботом (True/False).
* *first\_name*: Имя пользователя.
* *last\_name*: Фамилия пользователя (может быть None, если отсутствует).
* *full\_name*: Полное имя пользователя (Имя + Фамилия)
* *username*: Имя пользователя (никнейм) без символа "@" (может быть None, если отсутствует).
* *language\_code*: Код языка, предпочтительного для данного пользователя.

Использование объекта *message.from\_user* в Python производится с помощью F-строк (рис. 3)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3 - использование объекта message.from\_user с помощью F-строки

1. Структура эхо-бота
   1. Импорты библиотек и создание объектов (рис. 4)

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Рисунок 4 - импорты библиотек и создание объектов

* Импорты (executor, types, Bot, Dispatcher), необходимые для корректной работы бота
* Переменная *token*: переменная с токеном бота от @BotFather (получение описано в главе III)
* *Bot*: Создание объекта бота, который предоставляет интерфейс для взаимодействия с API Telegram.
* *Dispatcher*: Создание объекта диспетчера, который отслеживает и обрабатывает входящие обновления от пользователя.
  1. Обработчик команды */start* (рис. 5)

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

Рисунок 5 - обработчик команды /start

Когда пользователь отправляет боту команду */start*, вызывается асинхронная функция *start\_handler*, которая отправляет ответное сообщение с приветственным текстом. Это типичный сценарий для начальной команды, когда бот предоставляет пользователю информацию о своих возможностях или просто приветствует его. *Обработчик команды строго до обработчика всех текстовых сообщений!*

* 1. Обработчик для любых текстовых сообщений (рис. 6)

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

Рисунок 6 - обработчик для любых текстовых сообщений

Когда бот получает текстовое сообщение (не команду), вызывается асинхронная функция *echo\_handler*. Эта функция отвечает пользователю, повторяя текст его сообщения. Такой сценарий часто используется для эхо-ботов, которые просто повторяют введенный текст обратно пользователю.

* 1. Запуск бота (рис. 7)

Изображение выглядит как текст, Шрифт, снимок экрана, часы

Автоматически созданное описание

Рисунок 7 - запуск бота

Этот блок кода проверяет, запущен ли скрипт напрямую (а не импортирован в другой скрипт) с помощью условия, и, если условие выполняется, запускает цикл опроса бота *start\_polling(dp)*. Цикл опроса отвечает за постоянное отслеживание обновлений от серверов Telegram и вызов соответствующих обработчиков для их обработки. В данном случае бот начинает слушать входящие сообщения и реагировать на них согласно установленным обработчикам.

**III. Создание Telegram-бота в @BotFather**

1. Регистрация в *Telegram* и поиск *@BotFather*:
   1. Загрузите приложение Telegram на ваше устройство или воспользуйтесь веб-версией (<https://web.telegram.org>), войдите в аккаунт.
   2. В поисковой строке Telegram найдите аккаунт *@BotFather* или перейдите по ссылке (https://t.me/BotFather)
2. Создание нового бота с помощью *@BotFather*:
   1. Нажмите на кнопку "Начать" или напишите */start*, чтобы начать взаимодействие с *@BotFather*.
   2. Введите команду /newbot для создания нового бота.
   3. *@BotFather* запросит имя вашего бота. Придумайте уникальное имя для бота, и отправьте в чат.
   4. *@BotFather* запросит уникальный юзернейм для вашего бота. Придумайте юзернейм, который должен заканчиваться на «*\_bot*» или «*bot*» (например, *YourBotUsername\_bot* или *YourBotUsernamebot*).
   5. После успешного создания бота *@BotFather* предоставит вам токен. Этот токен уникален и используется для взаимодействия с API Telegram от имени вашего бота. Сохраните полученный токен.

**Порядок выполнения работы:**

1. Создать своего собственного бота посредством *@BotFather* и сохранить токен для взаимодействия с Telegram API.
2. Написать код собственного эхо-бота и доработать его в соответствии с вариантом (№ в журнале) (Таблица 1). Вместо «{}» для нечётных номеров в журнале бот должен выводить полное имя пользователя, для чётных – только юзернейм.
3. Сделать скриншоты взаимодействия с ботом на основе добавленных команд. Составить отчёт о проделанной работе.

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| **№ в журнале** | **Ответ на /start** | **Дополнительная команда** | **Текст ответа на дополнительную команду** |
| 1 | Привет, {}! Я ваш эхо-бот! | /info | Это информационная команда. Выберите /info для получения дополнительной информации. |
| 2 | Добро пожаловать, {}! | /help | Помощь по использованию бота. Выберите /help для получения справки. |
| 3 | Приветствую, {}! Это мой бот. | /about | Описание вашего бота. Выберите /about, чтобы узнать больше. |
| 4 | Здравствуй, {}! Я готов слушать. | /commands | Список доступных команд. Выберите /commands, чтобы увидеть все команды. |
| 5 | Привет, {}! | /settings | Настройки вашего бота. Выберите /settings для настройки параметров. |
| 6 | Добрый день, {}! Я ваш ассистент. | /feedback | Обратная связь с ботом. Выберите /feedback, чтобы оставить свой отзыв. |
| 7 | Рад видеть вас тут, {}! | /news | Последние новости. Выберите /news, чтобы узнать последние события. |
| 8 | Привет, {}! Я твой персональный бот. | /random | Генерация случайного числа. Выберите /random, чтобы получить случайное число. |
| 9 | Здравствуйте, {}! Как я могу помочь? | /search | Поиск информации. Выберите /search, чтобы начать поиск. |
| 10 | Добро пожаловать в мир ботов, {}! | /weather | Информация о погоде. Выберите /weather, чтобы узнать текущую погоду. |
| 11 | Привет, {}! Я тут для вас. | /quote | Цитаты и высказывания. Выберите /quote, чтобы получить вдохновение. |
| 12 | Доброго времени суток, {}! | /translate | Перевод текста. Выберите /translate, чтобы перевести текст. |
| 13 | Рад приветствовать вас, {}! | /calculator | Калькулятор. Выберите /calculator, чтобы решить математические задачи. |
| 14 | Привет, {}! Как я могу быть полезен? | /joke | Шутки для хорошего настроения. Выберите /joke, чтобы посмеяться. |
| 15 | Добро пожаловать, {}! Чем могу помочь? | /music | Музыкальные рекомендации. Выберите /music, чтобы получить плейлист. |
| 16 | Привет, {}! | /movies | Рекомендации по фильмам. Выберите /movies, чтобы узнать, что посмотреть. |
| 17 | Здравствуйте, {}! Готов ответить на ваши вопросы. | /gallery | Галерея изображений. Выберите /gallery, чтобы просмотреть картинки. |
| 18 | Добрый день, {}! Что я могу сделать для вас? | /poll | Опросы и голосования. Выберите /poll, чтобы участвовать в опросах. |
| 19 | Привет, {}! Как я могу вам помочь?" | /quiz | Викторины и вопросы. Выберите /quiz, чтобы проверить свои знания. |
| 20 | Рад видеть вас снова, {}! | /remind | Напоминания. Выберите /remind, чтобы установить напоминание. |
| 21 | Приветствую, {}! Что нового? | /tasks | Задачи и список дел. Выберите /tasks, чтобы управлять своими задачами. |
| 22 | Привет, {}! Готов к ваших командам. | /countdown | Обратный отсчет. Выберите /countdown, чтобы запустить обратный отсчет. |
| 23 | Доброго времени суток, {}! Как я могу вам помочь? | /links | Ссылки и ресурсы. Выберите /links, чтобы получить полезные ссылки." |
| 24 | Здравствуйте, {}! Чем могу помочь? | /memes | Мемы и смешные картинки. Выберите /memes, чтобы поднять настроение. |
| 25 | Привет, {}! Что нового? | /riddles | Загадки и головоломки. Выберите /riddles, чтобы разгадать загадки. |