Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «**Разработка умного голосового ассистента**»

Выполнил: Юнусов Руслан Александрович

Группа: ПР-32

Преподаватель:

2024

**Задание:** Разработать умного голосового ассистента на Python

**Входные и выходные данные**

VoiceAssistant – класс, класс голосового ассистента  
commands – json, комманды  
recognizer – Recognizer, распознователь речи

Microphone – Microphone, подключение к микрофону

Voise\_input – string, строка, голосовой запрос в виде строки

**Листинг программы (если есть)**

**VoiseAssistant.py**

import random

import webbrowser

import pyttsx3

import speech\_recognition

import json

import os

from vosk import Model, KaldiRecognizer

import wave

def swap\_language(\*args):

    print("Метод swap\_language вызван!")  # Логируем вызов метода

    print(f"Текущий язык: {assistant.speech\_language}")

    """Смена языка"""

    current\_language = assistant.speech\_language

    if current\_language == "ru":

        assistant.speech\_language = "en"

        assistant.recognition\_language = "en-US"

        play\_voice\_assistant\_speech("Я меняю язык на английский.")

    else:

        assistant.speech\_language = "ru"

        assistant.recognition\_language = "ru-RU"

        play\_voice\_assistant\_speech("I am switching to Russian.")

    setup\_assistant\_voice()

def play\_greetings(\*args):

    if assistant.speech\_language == "ru":

        play\_voice\_assistant\_speech("Привет! Как я могу помочь?")

    else:

        play\_voice\_assistant\_speech("Hi! Can i help you?")

    """Функция для проигрывания приветствия"""

def play\_farewell\_and\_quit(\*args):

    """Функция для прощания и выхода"""

    if assistant.speech\_language == "ru":

        play\_voice\_assistant\_speech("До свидания! Хорошего дня")

    else:

        play\_voice\_assistant\_speech("Goodbye! Have a nice day!")

    exit()

def search\_for\_term\_on\_google(\*args):

    """Поиск в Google"""

    search\_term = " ".join(args)

    url = f"https://www.google.com/search?q={search\_term}"

    webbrowser.get().open(url)

    if assistant.speech\_language == "ru":

        play\_voice\_assistant\_speech(f"Я ищу {search\_term} в Google.")

    else:

        play\_voice\_assistant\_speech(f"I seatch on {search\_term} в Google.")

def drop\_coin(\*args):

    """Подбросить монетку"""

    result = "Орел" if random.choice([True, False]) else "Решка"

    play\_voice\_assistant\_speech(f"Выпало: {result}")

# Загрузка команд из JSON

def load\_commands\_from\_json():

    with open("commands.json", "r", encoding="utf-8") as f:

        return json.load(f)

# Глобальные переменные

commands = load\_commands\_from\_json()

recognizer = speech\_recognition.Recognizer()

microphone = speech\_recognition.Microphone()

ttsEngine = pyttsx3.init()

class VoiceAssistant:

    """Настройки голосового ассистента"""

    name = ""

    sex = ""

    speech\_language = ""

    recognition\_language = ""

# Настройка голоса

def setup\_assistant\_voice():

    """

    Установка голоса по умолчанию (индекс может меняться в зависимости от настроек операционной системы)

    """

    voices = ttsEngine.getProperty("voices")

    if assistant.speech\_language == "en":

        assistant.recognition\_language = "en-US"

        if assistant.sex == "female":

            ttsEngine.setProperty("voice", voices[1].id)  # Microsoft Zira Desktop (English)

        else:

            ttsEngine.setProperty("voice", voices[2].id)  # Microsoft David Desktop (English)

    else:

        assistant.recognition\_language = "ru-RU"

        ttsEngine.setProperty("voice", voices[0].id)  # Microsoft Irina Desktop (Russian)

# Проигрывание речи

def play\_voice\_assistant\_speech(text\_to\_speech):

    ttsEngine.say(str(text\_to\_speech))

    ttsEngine.runAndWait()

def record\_and\_recognize\_audio(\*args: tuple):

    """

    Запись и распознавание аудио с использованием выбранного языка

    """

    with microphone:

        recognized\_data = ""

        # Регулирование уровня окружающего шума

        recognizer.adjust\_for\_ambient\_noise(microphone, duration=2)

        try:

            print("Listening...")

            audio = recognizer.listen(microphone, 5, 5)

            with open("microphone-results.wav", "wb") as file:

                file.write(audio.get\_wav\_data())

        except speech\_recognition.WaitTimeoutError:

            print("Can you check if your microphone is on, please?")

            return ""

        try:

            print("Started recognition...")

            recognized\_data = recognizer.recognize\_google(audio, language=assistant.recognition\_language).lower()

            print(f"Распознано: {recognized\_data}")  # Выводим распознанный текст

        except speech\_recognition.UnknownValueError:

            pass

        except speech\_recognition.RequestError:

            print("Trying to use offline recognition...")

            recognized\_data = use\_offline\_recognition()

        return recognized\_data.strip()

def use\_offline\_recognition():

    recognized\_data = ""

    try:

        if not os.path.exists("models/vosk-model-small-ru-0.4"):

            print("Please download the model from the specified URL and unpack it in the 'models' folder.")

            exit(1)

        wave\_audio\_file = wave.open("microphone-results.wav", "rb")

        model = Model("models/vosk-model-small-ru-0.4")

        offline\_recognizer = KaldiRecognizer(model, wave\_audio\_file.getframerate())

        data = wave\_audio\_file.readframes(wave\_audio\_file.getnframes())

        if len(data) > 0:

            if offline\_recognizer.AcceptWaveform(data):

                recognized\_data = offline\_recognizer.Result()

                recognized\_data = json.loads(recognized\_data)

                recognized\_data = recognized\_data["text"]

    except:

        print("Error with offline speech recognition.")

    return recognized\_data

def execute\_command\_with\_name(command\_name: str, \*args: list):

    print(f"Ищем команду: {command\_name}")  # Логирование полученной команды

    for key, command in commands.items():

        print(f"Проверка команд: {command\_name} в {command['examples']}")  # Логирование примеров команды

        # Проверяем, содержит ли примеры команд введенную команду

        if any(command\_name in example for example in command["examples"]):

            print(f"Команда {command\_name} найдена, выполняем {command['responses']}")  # Логируем, что команда найдена

            function\_name = command["responses"]

            if function\_name in globals():

                globals()[function\_name](\*args)

            if "responses\_voice" in command:

                # play\_voice\_assistant\_speech(command["responses\_voice"])

                break

    else:

        print("Команда не найдена.")

        play\_voice\_assistant\_speech("Извините, я вас не понял")

if \_\_name\_\_ == "\_\_main\_\_":

    assistant = VoiceAssistant()

    assistant.name = "Alice"

    assistant.sex = "female"

    assistant.speech\_language = "ru"

    setup\_assistant\_voice()

    while True:

        voice\_input = record\_and\_recognize\_audio()

        os.remove("microphone-results.wav")

        print(f"Вы сказали: {voice\_input}")

        # Если команда пуста, пропускаем итерацию

        if not voice\_input:

            continue

        # Разделение команды и дополнительных параметров

        voice\_input = voice\_input.split(" ")

        command = voice\_input[0]

        command\_options = voice\_input[1:]

        # Выполнение команды с голосовым ответом

        execute\_command\_with\_name(command, \*command\_options)

**commands.json**

{

  "greeting": {

    "examples": ["привет", "здравствуй", "добрый день", "hello", "good morning"],

    "responses": "play\_greetings",

    "responses\_voice": "Привет! Как я могу помочь?"

  },

  "farewell": {

    "examples": ["пока", "до свидания", "увидимся", "до встречи", "goodbye", "bye", "see you soon"],

    "responses": "play\_farewell\_and\_quit",

    "responses\_voice": "До свидания! Хорошего дня!"

  },

  "google\_search": {

    "examples": ["найди в гугл", "search on google", "google", "find on google"],

    "responses": "search\_for\_term\_on\_google",

    "responses\_voice": "Я ищу это в Google."

  },

  "language": {

    "examples": ["смени язык", "switch language", "language", "switch", "change", "смени", "поменяй",  "язык", "change language"],

    "responses": "swap\_language",

    "responses\_voice": "Я меняю язык."

  },

  "coin": {

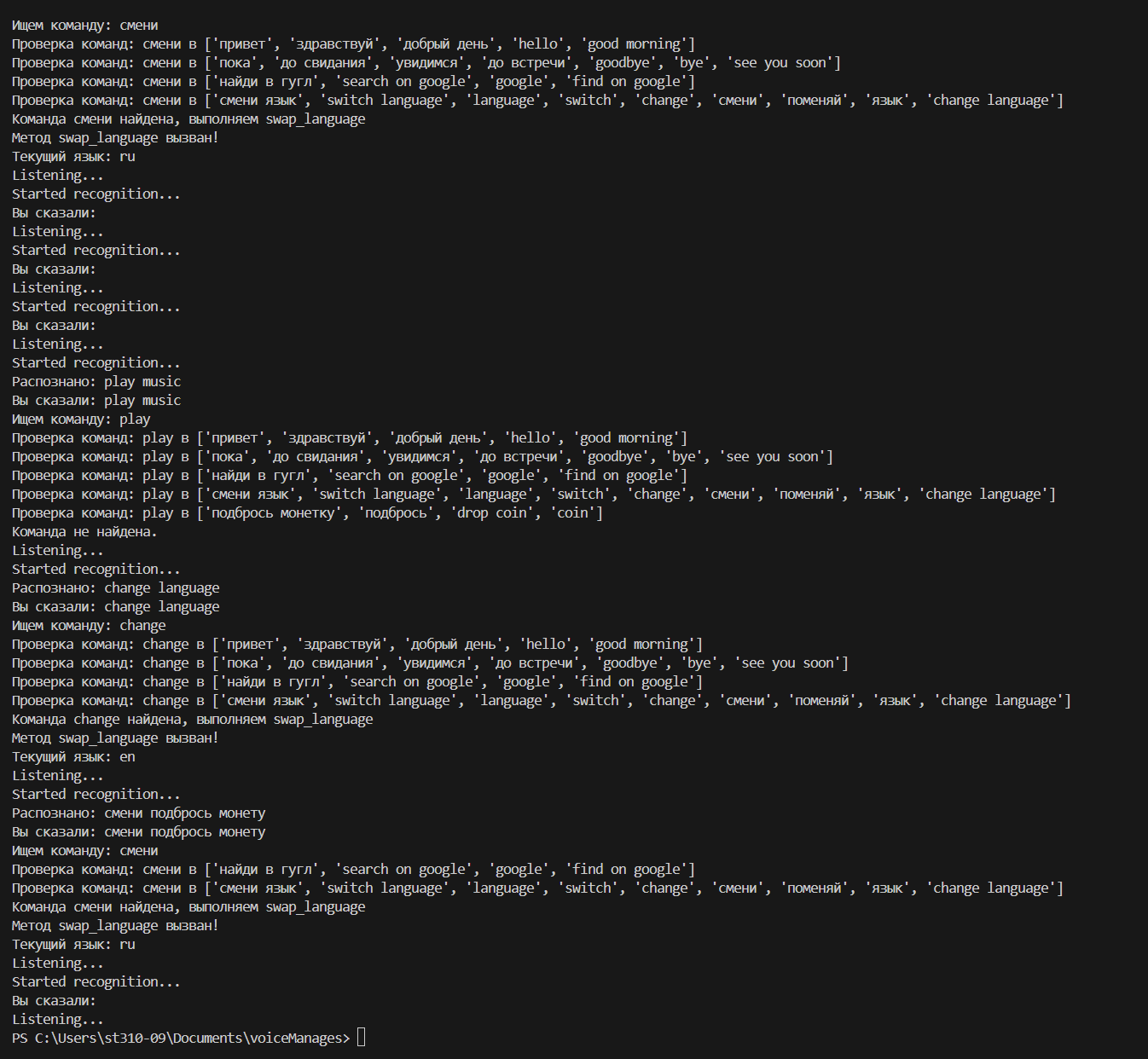
    "examples": ["подбрось монетку", "подбрось", "drop coin", "coin"],

    "responses": "drop\_coin",

    "responses\_voice": "Я подбросил монетку!"

  }

}

**Скрин-шот э**

**Вывод**

Я укрепил знания по языку программирования Python, и изучил несколько полезных для меня библиотек.