Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

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

Выполнил: Жеребцев Виталий Алексеевич

Группа: ПР-32

Преподаватель: Шевцова Вера Дмитриевна

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

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

**Листинг программы (если есть)**

[assist.py](https://github.com/morsals1/YP_assist/blob/main/YP/assist.py)

import random

import webbrowser

import pyttsx3

import speech\_recognition as sr

import json

import os

from vosk import Model, KaldiRecognizer

import wave

# Глобальные переменные для хранения текущих языков

current\_recognition\_language = "ru-RU"  # По умолчанию русский

current\_speech\_language = "ru"  # По умолчанию русский

# Глобальные переменные

recognizer = sr.Recognizer()

microphone = sr.Microphone()

ttsEngine = pyttsx3.init()

# Загрузка команд из JSON

def load\_commands\_from\_json():

    try:

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

            commands = json.load(f)

            print("Команды успешно загружены из файла.")

            return commands

    except FileNotFoundError:

        print("Ошибка: файл 'commands.json' не найден.")

        return {}

    except json.JSONDecodeError:

        print("Ошибка: не удается декодировать JSON в файле 'commands.json'.")

        return {}

    except Exception as e:

        print(f"Неизвестная ошибка при загрузке команд: {e}")

        return {}

# Загрузка команд

commands = load\_commands\_from\_json()

# Настройка голоса

def setup\_assistant\_voice(language="ru"):

    voices = ttsEngine.getProperty("voices")

    print("Доступные голоса:")

    for voice in voices:

        print(f"ID: {voice.id}, Name: {voice.name}")

    # Меняем голос в зависимости от языка

    if language == "en":

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

    else:

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

# Проигрывание речи

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

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

    ttsEngine.runAndWait()

# Смена языка распознавания и синтеза речи

def change\_language(new\_language="ru"):

    global current\_recognition\_language, current\_speech\_language

    if new\_language == "en":

        current\_recognition\_language = "en-US"

        current\_speech\_language = "en"

    else:

        current\_recognition\_language = "ru-RU"

        current\_speech\_language = "ru"

    # Настройка голоса на новый язык

    setup\_assistant\_voice(language=current\_speech\_language)

    # Приветствие на новом языке

    if new\_language == "en":

        play\_voice\_assistant\_speech("I am changing the language to English.")

    else:

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

    print(f"Язык изменен на: {new\_language}")

# Приветствие

def play\_greetings(\*args):

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

# Прощание и выход

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

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

    exit()

# Поиск в Google

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

    search\_term = " ".join(args)

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

    webbrowser.get().open(url)

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

# Подбросить монетку

def drop\_coin(\*args):

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

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

# Запись и распознавание аудио

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

    with microphone:

        recognized\_data = ""

        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 sr.WaitTimeoutError:

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

            return ""

        try:

            print("Started recognition...")

            recognized\_data = recognizer.recognize\_google(audio, language=current\_recognition\_language).lower()

            print(f"Распознано: {recognized\_data}")

        except sr.UnknownValueError:

            pass

        except sr.RequestError:

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

            recognized\_data = use\_offline\_recognition()

        return recognized\_data.strip()

# Оффлайн распознавание

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 Exception as e:

        print(f"Error with offline speech recognition: {e}")

    return recognized\_data

# Выполнение команды по имени

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

    print(f"Ищем команду: {command\_name}")

    for key, command in commands.items():

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

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

            function\_name = command["responses"]

            if function\_name in globals():

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

            break

    else:

        print("Команда не найдена.")

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

# Главная часть программы

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

    setup\_assistant\_voice(language="ru")  # По умолчанию русский язык

    while True:

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

        # Check if the file exists before deleting it

        if os.path.exists("microphone-results.wav"):

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

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

        if not voice\_input:

            continue

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

        command = voice\_input[0]

        command\_options = voice\_input[1:]

        # Обработка команды смены языка

        if "сменить" in voice\_input or "поменяй" in voice\_input or "переключить" in voice\_input:

            if "английский" in voice\_input:

                change\_language("en")

            elif "русский" in voice\_input:

                change\_language("ru")

            else:

                play\_voice\_assistant\_speech("Не могу понять, на какой язык нужно переключиться.")

        else:

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

commands.json

{

  "greeting": {

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

    "responses": "play\_greetings",

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

    "responses\_voice\_en": "Hello! How can I help?"

  },

  "farewell": {

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

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

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

    "responses\_voice\_en": "Goodbye! Have a nice day!"

  },

  "google\_search": {

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

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

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

    "responses\_voice\_en": "I'm searching for this on Google."

  },

  "language": {

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

    "responses": "change\_language",

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

    "responses\_voice\_en": "I am changing the language."

  },

  "coin": {

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

    "responses": "drop\_coin",

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

    "responses\_voice\_en": "I flipped a coin!"

  }

}

**Вывод**

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