

Создание голосового ассистента для управления компьютером

Эрдман Александр Алексеевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье рассмотрен процесс создания программы для управления компьютером при помощи голоса. Данная программа написана на языке программирования Python с использованием модулей записи и воспроизведения звука. Результатом исследования является готовая программа «Голосовой ассистент Пётр», а также подробное её описание.

Ключевые слова: голосовой ассистент, управление компьютером, python

Creating a voice assistant to control your computer

Erdman Alexander Alekseevich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article describes the process of creating a program for controlling a computer using voice. This program is written in the Python programming language using audio recording and playback modules. The result of the study is a ready-made program "Voice Assistant Peter", as well as a detailed description of it.

Keywords: voice assistant, computer management, python

1 Введение

1.1 Актуальность

В современном мире активно развиваются технологии, которые позволяют управлять техникой при помощи голоса, что в свою очередь упрощает использование различных цифровых устройств, начиная от смартфонов и компьютеров и заканчивая бытовыми устройствами, например пылесосом, микроволновой печью и прочей техникой. Активная разработка таких устройств связано прежде всего с их актуальностью, ведь гораздо удобнее сказать слово, по которому будет выполнена команда, чем использовать клавиатуру, мышку, сенсор или кнопки. Это существенно облегчает и делает более доступным использование тех же компьютеров и смартфонов, в частности это очень помогает старшему поколению. Также данные технологии предоставляют доступность использования техники для людей с ОВЗ, например, для слепого человека управление голосом становится незаменимой возможностью использовать смартфоны и компьютеры. Ярким примером таких технологий служат голосовые

ассистенты, которые имеют широкий круг применений. Например, голосовые ассистенты распространены на смартфонах в качестве управления телефоном. Помимо смартфонов на компьютеры всё чаще начинают внедрять голосовых ассистентов, примером может служить ассистент Cortana, разработанный корпорацией Microsoft для операционной системы Windows. Учитывая востребованность данных технологий, в статье будет рассмотрен процесс создания одного из видов данной технологии – голосового ассистента для компьютера под управлением популярной операционной системы Windows.

1.2 Обзор исследований

Д.А. Голуб разработал программу "Voiceman" для управления процессами организации [1]. О.В. Безотосная рассмотрела интеграцию WMS с системой голосового управления [2]. Ю.Ж. Магомедов рассмотрел влияние различных факторов влияющих на распознавание голосовых команд системой голосового оборудования таких как: высота тонов и длительность произношения команд человеком [3]. А.А. Топорин создал подсистему голосового управления интеллектуальной системы управления движением автономного мобильного робота [4]. Д.А. Киреев, Н.И. Литвинова и Н.С. Попов рассмотрели технологии управления голом через чат [5]. А.А. Рябова изучала информационную технологию «умный дом» [6].

1.3 Цель исследования

Целью исследования является создания голосового ассистента, написанного на языке программирования Python, для компьютера под управлением операционной системой Windows 10. Основной функцией голосового ассистента является выполнение базовых действий управления компьютером.

2 Материалы и методы

Для создания голосового ассистента будет использоваться язык программирования Python, среда программирования PyCharm, операционная система Windows 10, программа RHVoice для синтеза мужского голоса.

3 Результаты и обсуждения

Для создания голосового ассистента нужно сперва установить определённые модули, а именно: pyttsx3 2.71, pyriwin32, pywin32, SpeechRecognition 3.8.1, PyAudio 0.2.11, fuzzywuzzy. Модуль pyttsx3 2.71 отвечает за преобразования текста в речь, а модули pyriwin32 и pywin32 являются вспомогательными; модуль SpeechRecognition 3.8.1 отвечает за распознавание речи, PyAudio 0.2.11 позволяет считывать речь с микрофона, fuzzywuzzy отвечает за нечёткое сравнение. После установки модулей необходимо установить программу RHVoice для того, чтобы голосовой помощник имел мужской голос. Данная программа актуальна по причине того, что в OS Windows русский голос представлен одним женским голосом

и не имеет альтернатив по умолчанию. После установки модулей рекомендуется проверить их работоспособность, для этого нужно написать программу, которая проговорит заданный текст (рис. 1).

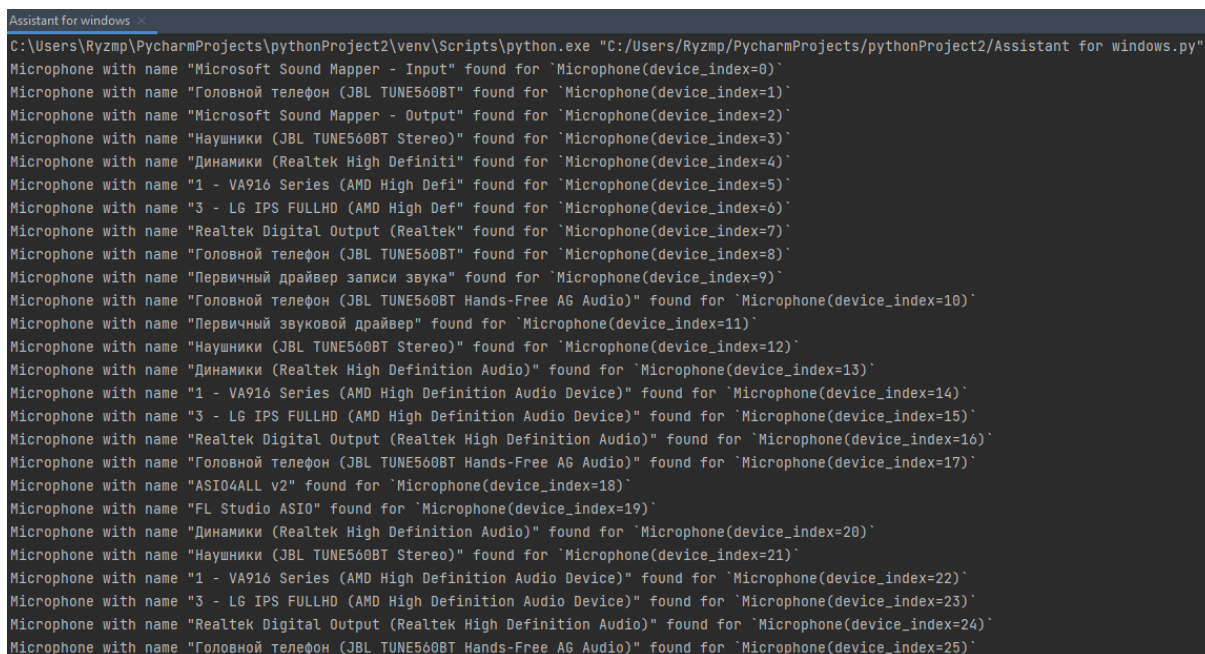
```
import pyttsx3
engine = pyttsx3.init()
engine.say("Модуль прошёл проверку и готов к использованию")
engine.runAndWait()
```

Рис. 1. Проверка работоспособности импортированного модуля pyttsx3

Для распознавание речи необходимо задействовать микрофон. На программном уровне данный микрофон имеет свой индекс в системе, который необходимо определить и использовать в коде (рис. 2 и 3).

```
import speech_recognition as sr
for index, name in enumerate(sr.Microphone.list_microphone_names()):
    print("Microphone with name \"{1}\" found for `Microphone(device_index={0})`".format(index, name))
```

Рис. 2. Код для запроса поиска устройств ввода в системе



```
Assistant for windows x
C:\Users\Ryzmp\PcharmProjects\pythonProject2\venv\Scripts\python.exe "C:/Users/Ryzmp/PcharmProjects/pythonProject2/Assistant for windows.py"
Microphone with name "Microsoft Sound Mapper - Input" found for `Microphone(device_index=0)`
Microphone with name "Головной телефон (JBL TUNE560BT)" found for `Microphone(device_index=1)`
Microphone with name "Microsoft Sound Mapper - Output" found for `Microphone(device_index=2)`
Microphone with name "Наушники (JBL TUNE560BT Stereo)" found for `Microphone(device_index=3)`
Microphone with name "Динамики (Realtek High Definition Audio)" found for `Microphone(device_index=4)`
Microphone with name "1 - VA916 Series (AMD High Definition Audio Device)" found for `Microphone(device_index=5)`
Microphone with name "3 - LG IPS FULLHD (AMD High Definition Audio Device)" found for `Microphone(device_index=6)`
Microphone with name "Realtek Digital Output (Realtek High Definition Audio)" found for `Microphone(device_index=7)`
Microphone with name "Головной телефон (JBL TUNE560BT)" found for `Microphone(device_index=8)`
Microphone with name "Первичный звуковой драйвер" found for `Microphone(device_index=9)`
Microphone with name "Головной телефон (JBL TUNE560BT Hands-Free AG Audio)" found for `Microphone(device_index=10)`
Microphone with name "Первичный звуковой драйвер" found for `Microphone(device_index=11)`
Microphone with name "Наушники (JBL TUNE560BT Stereo)" found for `Microphone(device_index=12)`
Microphone with name "Динамики (Realtek High Definition Audio)" found for `Microphone(device_index=13)`
Microphone with name "1 - VA916 Series (AMD High Definition Audio Device)" found for `Microphone(device_index=14)`
Microphone with name "3 - LG IPS FULLHD (AMD High Definition Audio Device)" found for `Microphone(device_index=15)`
Microphone with name "Realtek Digital Output (Realtek High Definition Audio)" found for `Microphone(device_index=16)`
Microphone with name "Головной телефон (JBL TUNE560BT Hands-Free AG Audio)" found for `Microphone(device_index=17)`
Microphone with name "ASIO4ALL v2" found for `Microphone(device_index=18)`
Microphone with name "FL Studio ASIO" found for `Microphone(device_index=19)`
Microphone with name "Динамики (Realtek High Definition Audio)" found for `Microphone(device_index=20)`
Microphone with name "Наушники (JBL TUNE560BT Stereo)" found for `Microphone(device_index=21)`
Microphone with name "1 - VA916 Series (AMD High Definition Audio Device)" found for `Microphone(device_index=22)`
Microphone with name "3 - LG IPS FULLHD (AMD High Definition Audio Device)" found for `Microphone(device_index=23)`
Microphone with name "Realtek Digital Output (Realtek High Definition Audio)" found for `Microphone(device_index=24)`
Microphone with name "Головной телефон (JBL TUNE560BT Hands-Free AG Audio)" found for `Microphone(device_index=25)`
```

Рис. 3. Вывод результата запроса поиска устройств ввода в системе

После определения индекса микрофона можно приступать к непосредственному написанию голосового ассистента. Сперва импортируем все необходимые модули (рис. 4). Далее необходимо прописать словарь с настройками ассистента (рис. 5). Ячейка `alias` хранит всевозможные вариации имени ассистента, благодаря которым можно обращаться к голосовому помощнику. Ячейка `tbr` содержит слова, которые нужно удалять из речевой команды. Ячейка `cmd` хранит всевозможные команды ассистента.

```
import os
import time
import speech_recognition as sr
from fuzzywuzzy import fuzz
import pyttsx3
import datetime
```

Рис. 4. Импорт модулей

```
opts = {
    "alias": ('ассистент', 'помощник', 'петя', 'пётр',),
    "tbr": ('расскажи', 'скажи', 'поведуй',),
    "cmd": {
        "Time": ('время', 'который час'),
        "Audio": ('включи музыку', 'воспроизведи музыку', 'музыки'),
        "Story": ('история', 'рассказ', 'можешь что-нибудь рассказать')
    }
}
```

Рис. 5. Словарь обращения и команд голосового помощника

После подключения модулей и определения словаря ассистента необходимо прописать запуск ассистента (рис. 6). Метод «`r.adjust_for_ambient_noise(source)`» в течение одной секунды слушает фон, чтобы впоследствии не путать шум фона с речью человека. При помощи метода «`setProperty`» указывается мужской голос, которым будет говорить ассистент. После этого голосовой помощник произносит приветственные фразы и начинает слушать микрофон в фоне. Данный алгоритм должен соблюдаться, в противном же случае если сначала дать команду прослушивания микрофона, а потом команду приветствия, то возможно, что ассистент начнёт слушать сам себя, что приведёт к нежелательным результатам.

```
r = sr.Recognizer()
m = sr.Microphone(device_index=1)

with m as source:
    r.adjust_for_ambient_noise(source)

speak_engine = pyttsx3.init()

voices = speak_engine.getProperty('voices')
speak_engine.setProperty('voice', voices[4].id)

speak("Здравствуйте. Чем я могу вам помочь")

stop_listening = r.listen_in_background(m, callback)
while True: time.sleep(0.1)
```

Рис. 6. Написание параметров запуска ассистента

Далее необходимо прописать три функции (рис. 7). Функция «`callback`» вызывается каждый раз, когда будет произнесена команда в микрофон. Функция «`recognize_cmd`» осуществляет нечёткий поиск команд, которые получил голосовой помощник. Функция «`execute_cmd`»

осуществляет преобразование команды в конкретное действие. После написания функций ассистент готов к использованию (рис. 8).

```
# Функции
def speak(what):
    print(what)
    speak_engine.say(what)
    speak_engine.runAndWait()
    speak_engine.stop()

def callback(recognizer, audio):
    try:
        voice = recognizer.recognize_google(audio, language="ru-RU").lower()
        print("[log] Распознано: " + voice)

        if voice.startswith(opts["alias"]):
            # Обращение к голосовому помощнику
            cmd = voice

            for x in opts['alias']:
                cmd = cmd.replace(x, "").strip()

            for x in opts['tbr']:
                cmd = cmd.replace(x, "").strip()

            # Распознавание и выполнение команды
            cmd = recognize_cmd(cmd)
            execute_cmd(cmd['cmd'])

    except sr.UnknownValueError:
        print("[log] Голос не распознан!")
    except sr.RequestError as e:
        print("[log] Неизвестная ошибка, проверьте интернет!")

def recognize_cmd(cmd):
    RC = {'cmd': '', 'percent': 0}
    for c, v in opts['cmds'].items():
        for x in v:
            vrt = fuzz.ratio(cmd, x)
            if vrt > RC['percent']:
                RC['cmd'] = c
                RC['percent'] = vrt

    return RC

def execute_cmd(cmd):
    if cmd == 'Time':
        # Сказать текущее время компьютера
        now = datetime.datetime.now()
        speak("Сейчас " + str(now.hour) + ":" + str(now.minute))

    elif cmd == 'Audio':
        # Воспроизвести аудиофайл
        os.system("D:\\User\\Desktop\\audio.m3u")

    elif cmd == 'Story':
        # Рассказать историю
        speak("Я не очень хороший рассказчик, но я могу порекомендовать вам почитать Достовского")

    else:
        print('Команда не распознана, повторите')
```

Рис. 7. Функции ассистента

```

import os
import time
import speech_recognition as sr
from fuzzywuzzy import fuzz
import pyttsx3
import datetime

opts = {
    "alias": ('ассистент', 'помощник', 'петя', 'пётр',),
    "tbr": ('расскажи', 'скажи', 'поведуй',),
    "cmd": {
        "Time": ('время', 'который час'),
        "Audio": ('включи музыку', 'воспроизведи музыку', 'музыки'),
        "Story": ('история', 'рассказ', 'можешь что-нибудь рассказать')
    }
}

# Функции
def speak(what):
    print(what)
    speak_engine.say(what)
    speak_engine.runAndWait()
    speak_engine.stop()

def callback(recognizer, audio):
    try:
        voice = recognizer.recognize_google(audio, language="ru-RU").lower()
        print("[Log] Распознано: " + voice)

        if voice.startswith(opts["alias"]):
            # Обращение к голосовому помощнику
            cmd = voice

            for x in opts['alias']:
                cmd = cmd.replace(x, "").strip()

            for x in opts['tbr']:
                cmd = cmd.replace(x, "").strip()

            # Распознавание и выполнение команды
            cmd = recognize_cmd(cmd)
            execute_cmd(cmd['cmd'])

    except sr.UnknownValueError:
        print("[Log] Голос не распознан!")
    except sr.RequestError as e:
        print("[Log] Неизвестная ошибка, проверьте интернет!")

def recognize_cmd(cmd):
    RC = {'cmd': '', 'percent': 0}
    for c, v in opts['cmds'].items():
        for x in v:
            vrt = fuzz.ratio(cmd, x)
            if vrt > RC['percent']:
                RC['cmd'] = c
                RC['percent'] = vrt

    return RC

def execute_cmd(cmd):
    if cmd == 'Time':
        # Сказать текущее время компьютера
        now = datetime.datetime.now()
        speak("Сейчас " + str(now.hour) + ":" + str(now.minute))

    elif cmd == 'Audio':
        # Воспроизвести аудиофайл
        os.system("D:\\User\\Desktop\\audio.m3u")

    elif cmd == 'Story':
        # Рассказать историю
        speak("Я не очень хороший рассказчик, но я могу порекомендовать вам почитать Достовского")

    else:
        print('Команда не распознана, повторите')

r = sr.Recognizer()
m = sr.Microphone(device_index=1)
with m as source:
    r.adjust_for_ambient_noise(source)
    speak_engine = pyttsx3.init()
    voices = speak_engine.getProperty('voices')
    speak_engine.setProperty('voice', voices[4].id)
    speak("Здравствуйте. Чем я могу вам помочь")
    stop_listening = r.listen_in_background(m, callback)
    while True: time.sleep(0.1)

```

Рис. 8. Код голосового помощника «Пётр»

Таким образом, была разработана программа «Голосовой ассистент Пётр», благодаря которой можно осуществлять управление компьютером при помощи голоса.

Библиографический список

1. Голуб Д.А. "Voiceman". Система управления процессами с помощью голосового управления // Свидетельство о регистрации программы для ЭВМ 2020665194, 24.11.2020. Заявка № 2020664715 от 24.11.2020.
2. Безотосная О.В. Как голосовое управление дало толчок интенсивному развитию: проект создания системы управления складом // Логистика сегодня. 2013. № 3. С. 130-133.
3. Магомедов Ю.Ж. Исследование возможностей применения голосового управления для системы управления оборудованием // Актуальные научные исследования в современном мире. 2021. № 6-1 (74). С. 76-81.
4. Топорин А.А. Подсистема голосового управления системы интеллектуального управления мобильным роботом // Вестник науки и образования. 2020. № 14-4 (92). С. 9-13.
5. Киреев Д.А., Литвинова Н.И., Попов Н.С. UX в новых каналах взаимодействия с приложением: голосовое управление и управление через чат // В книге: МНСК-2020. Материалы 58-й Международной научной студенческой конференции. 2020. С. 66.
6. Рябова А.А. Голосовое управление как элемент информационной системы "Умный дом" // Вестник молодых ученых Санкт-Петербургского государственного университета технологии и дизайна. 2018. № 4. С. 93-99.