

Разработка графического интерфейса для синтеза речи с настройкой скорости и пола голоса

Малик Александр Игоревич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье представлена разработка приложения для преобразования текста в речь с графическим интерфейсом, использующего библиотеку pytsx3 для синтеза речи. Программа предоставляет пользователю возможность выбирать язык, пол голоса, а также регулировать скорость воспроизведения. Это решение направлено на улучшение доступности технологий синтеза речи, предоставляя гибкость настроек для различных пользователей. В статье рассмотрены архитектура приложения, реализация функций выбора параметров речи, а также возможности дальнейшего улучшения программы. Исследование включает использование Python и популярных библиотек для обработки текста и голосовых данных.

Ключевые слова: Текст в речь, синтез речи, pytsx3, графический интерфейс, Python, голосовые технологии, регулировка скорости речи, выбор пола голоса, многоязычное приложение.

Development of a graphical interface for speech synthesis with voice speed and gender settings

Malik Aleksandr Igorevich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article presents the development of a text-to-speech application with a graphical interface using the pytsx3 library for speech synthesis. The program provides the user with the ability to select the language, gender of the voice, and adjust the playback speed. This solution aims to improve the availability of speech synthesis technologies by providing flexibility in settings for different users. The article discusses the architecture of the application, the implementation of speech parameter selection functions, as well as the possibilities for further improvement of the program. The research includes the use of Python and popular libraries for processing text and voice data.

Keywords: Text-to-speech, speech synthesis, pytsx3, graphical interface, Python, voice technology, speech speed control, voice gender selection, multilingual application.

1 Введение

1.1 Актуальность

В последние годы наблюдается рост интереса к технологиям синтеза речи, которые активно используются в различных областях, таких как образование, медицина, разработки для людей с ограниченными возможностями и интерфейсы для устройств с голосовым управлением. Существующие решения часто ограничены функционально, не предоставляя пользователю гибких настроек. Разработка приложения с возможностью выбора языка, пола голоса и регулировки скорости делает эти технологии более доступными и удобными для широкого круга пользователей.

1.2 Обзор исследований

В.И. Расулов в своём исследовании провёл сравнительный анализ популярных библиотек для синтеза речи в языке программирования Python, включая pyttsx3, gTTS и другие, с акцентом на их возможности и ограничения для различных приложений [1]. Р.А. Кривонос и В.В. Перевозников в своей работе предложили подходы к разработке голосового ассистента для улучшения информационного обеспечения служебно-боевой деятельности, акцентируя внимание на функциональных требованиях и этапах разработки голосовых систем в контексте военной сферы [2]. Ж. Жумашева, М.Сулейменов, Ж. Айтуганова и Д. Ертаева в своём исследовании сосредоточились на создании голосового помощника для управления персональным компьютером на базе операционной системы Windows с использованием Python, уделив внимание алгоритмам распознавания речи и улучшению взаимодействия с операционными системами [3]. К.Т. Айдашева и Д.И. Шигабетдинова в своей работе подробно рассматривают принципы работы голосовых ассистентов, а также подходы к их реализации на Python, включая использование различных технологий для распознавания команд и синтеза речи [4]. С. Манджу, Дж. Анита и Дж.Д. Сужита в своём исследовании предлагают инновационный метод преобразования рукописных изображений в речь с применением глубокого обучения, фокусируясь на алгоритмах распознавания текста и последующего его озвучивания с использованием библиотеки pyttsx3, что может быть полезно для создания доступных технологий для людей с нарушениями зрения [5].

1.3 Цель исследования

Целью исследования является разработка многофункционального приложения с графическим интерфейсом для синтеза речи, позволяющего пользователю настраивать параметры речи, такие как скорость и пол голоса, с использованием библиотеки pyttsx3. Программа также должна поддерживать несколько языков, что расширяет ее потенциальное применение.

2 Материалы и методы

Для реализации приложения использовались следующие инструменты и методы:

- **Язык программирования:** Python, благодаря его доступности и мощным библиотекам для работы с текстом и аудио.
- **Библиотека для синтеза речи:** pyttsx3 — кросс-платформенная библиотека для синтеза речи, поддерживающая работу на различных операционных системах.
- **Графический интерфейс:** Tkinter, стандартная библиотека Python для создания простых графических интерфейсов.
- **Методы:** Разработка пользовательского интерфейса с возможностью выбора языка, пола голоса и регулировки скорости, а также обработка текстовых данных для их последующего синтеза в речь.

3 Результаты и обсуждения

1. Импорт библиотек (рис.1).

```
import os
import tkinter as tk
from tkinter import messagebox, filedialog
import pyttsx3
```

Рисунок 1. Импорт библиотек

Для разработки приложения были использованы следующие библиотеки:

- os — для работы с операционной системой (например, для работы с путями файлов);
- tkinter — для создания графического интерфейса пользователя (GUI);
- pyttsx3 — для синтеза речи из текста.

2. Определение поддерживаемых языков (рис.2).

```
LANGUAGES = {
    'Русский': 'ru',
    'Английский': 'en',
    'Испанский': 'es',
    'Немецкий': 'de',
    'Французский': 'fr',
    'Итальянский': 'it',
    'Португальский': 'pt',
}
```

Рисунок 2. Поддерживаемые языки

Словарь LANGUAGES используется для определения доступных языков для синтеза речи. Каждый язык связан с его кодом, который будет передаваться в функцию pyttsx3.

3. Функция для синтеза речи (рис.3).

```
def text_to_speech(text, lang='ru', slow=False, voice_gender='male'): 1 usage
    try:
        # Инициализация pyttsx3
        engine = pyttsx3.init()

        # Получаем список доступных голосов
        voices = engine.getProperty('voices')

        # Выбираем голос в зависимости от пола
        selected_voice = None
        for voice in voices:
            if voice_gender == 'male' and 'male' in voice.name.lower():
                selected_voice = voice
                break
            elif voice_gender == 'female' and 'female' in voice.name.lower():
                selected_voice = voice
                break

        # Если голос найден, устанавливаем его
        if selected_voice:
            engine.setProperty(name='voice', selected_voice.id)
        else:
            engine.setProperty(name='voice', voices[0].id) # По умолчанию выбираем первый голос

        # Настройка скорости речи
        rate = engine.getProperty('rate')
        engine.setProperty(name='rate', rate - 50 if slow else rate) # Замедление, если slow=True

        # Воспроизведение текста
        engine.say(text)
        engine.runAndWait()

        messagebox.showinfo(title="Готово", message="Текст успешно преобразован в речь и воспроизведен.")

    except Exception as e:
        messagebox.showerror(title="Ошибка", message=f"Произошла ошибка: {e}")
```

Рисунок 3. Код функции синтеза речи

Функция `text_to_speech` выполняет основные действия по преобразованию текста в речь:

- Инициализация движка `pyttsx3`.
- Получение списка доступных голосов и выбор нужного голоса в зависимости от пола.
- Настройка скорости речи в зависимости от параметра `slow`.
- Воспроизведение текста с использованием метода `say()`, после чего звук воспроизводится с помощью `runAndWait()`.
- В случае успеха показывается всплывающее окно с информацией о завершении процесса.

4. Обработчик для кнопки преобразования текста (рис.4).

```
def on_convert_button_click(): 1 usage
    text = text_entry.get(index1: "1.0", index2: "end-1c").strip()
    if not text:
        messagebox.showwarning(title: "Предупреждение", message: "Пожалуйста, введите текст для преобразования.")
        return

    language = language_var.get() # Получаем язык из выпадающего списка
    slow = slow_var.get() # Получаем состояние флажка для замедленной речи
    voice_gender = voice_gender_var.get() # Получаем выбранный пол голоса

    lang_code = LANGUAGES.get(language, 'ru') # Получаем код языка
    text_to_speech(text, lang=lang_code, slow=slow, voice_gender=voice_gender)
```

Рисунок 4. Код обработчика для кнопки преобразования текста

Этот обработчик отвечает за:

- Получение текста из текстового поля text_entry.
- Проверку, что текст не пуст.
- Получение выбранных пользователем параметров (язык, пол голоса, замедленная речь).
- Вызов функции text_to_speech, передавая в нее введенный текст и выбранные настройки.

5. Обработчик для кнопки загрузки текста из файла (рис.5).

```
def on_file_open_button_click(): 1 usage
    """Обработчик для кнопки загрузки текста из файла"""
    file_path = filedialog.askopenfilename(title="Выберите текстовый файл", filetypes=[("Text files", "*.txt")])
    if file_path:
        with open(file_path, 'r', encoding='utf-8') as file:
            text = file.read()
            text_entry.delete(index1: "1.0", index2: "end-1c") # Очищаем текстовое поле
            text_entry.insert(index: "1.0", text) # Вставляем текст из файла
```

Рисунок 5. Код обработчика для кнопки загрузки текста из файла

Этот обработчик позволяет загружать текст из текстового файла:

- Открытие окна выбора файла с помощью filedialog.askopenfilename().
- Чтение содержимого файла и вставка его в текстовое поле text_entry.

6. Создание графического интерфейса (рис.6).

```
root = tk.Tk()
root.title("Приложение для преобразования текста в речь")

# Настройка размеров окна
root.geometry("450x550")
```

Рисунок 6 Код создания основного окна

Здесь создается окно приложения с помощью Tkinter (рис.6):

- root — основное окно приложения.
- title() — задает название окна.
- geometry() — задает размеры окна (ширина 450 пикселей, высота 550 пикселей).

7. Размещение элементов интерфейса (рис.7).

```
# Метка для ввода текста
label = tk.Label(root, text="Введите текст для преобразования в речь:")
label.pack(pady=10)

# Текстовое поле для ввода текста
text_entry = tk.Text(root, height=6, width=40)
text_entry.pack(pady=10)

# Метка для выбора языка
label_language = tk.Label(root, text="Выберите язык:")
label_language.pack(pady=10)

# Выпадающий список для выбора языка
language_var = tk.StringVar()
language_dropdown = tk.OptionMenu(root, language_var, *LANGUAGES.keys())
language_dropdown.pack(pady=10)
language_var.set('Русский') # Значение по умолчанию

# Чекбокс для замедленной речи (для изменения тональности)
slow_var = tk.BooleanVar()
slow_checkbox = tk.Checkbutton(root, text="Замедлить речь", variable=slow_var)
slow_checkbox.pack(pady=10)

# Выпадающий список для выбора пола голоса
voice_gender_var = tk.StringVar()
voice_gender_dropdown = tk.OptionMenu(root, voice_gender_var, value='male', *values: 'female')
voice_gender_dropdown.pack(pady=10)
voice_gender_var.set('male') # Значение по умолчанию

# Кнопка для преобразования текста в речь
convert_button = tk.Button(root, text="Преобразовать в речь", command=on_convert_button_click)
convert_button.pack(pady=20)

# Кнопка для загрузки текста из файла
file_button = tk.Button(root, text="Загрузить текст из файла", command=on_file_open_button_click)
file_button.pack(pady=10)
```

Рисунок 7. Код создания и размещения элементов программы

В этом блоке создаются и размещаются различные элементы пользовательского интерфейса:

- Метки (Label) для ввода текста и выбора языка.
- Текстовое поле (Text) для ввода текста.
- Выпадающие списки (OptionMenu) для выбора языка и пола голоса.
- Чекбокс (Checkbutton) для замедления речи.
- Кнопки (Button) для преобразования текста в речь и загрузки текста из файла.

Каждому элементу интерфейса присваиваются необходимые параметры, такие как текст, команду для обработки событий и начальное значение.

8. Запуск главного цикла (рис.8).

```
root.mainloop()
```

Рисунок 8. Код запуска главного цикла

Запуск главного цикла Tkinter, который позволяет интерфейсу быть интерактивным и ожидать действия пользователя. Когда пользователь взаимодействует с приложением, например, нажимает кнопки, происходят вызовы соответствующих обработчиков, которые выполняют нужные действия.

9. Внешний вид программы (рис.9).

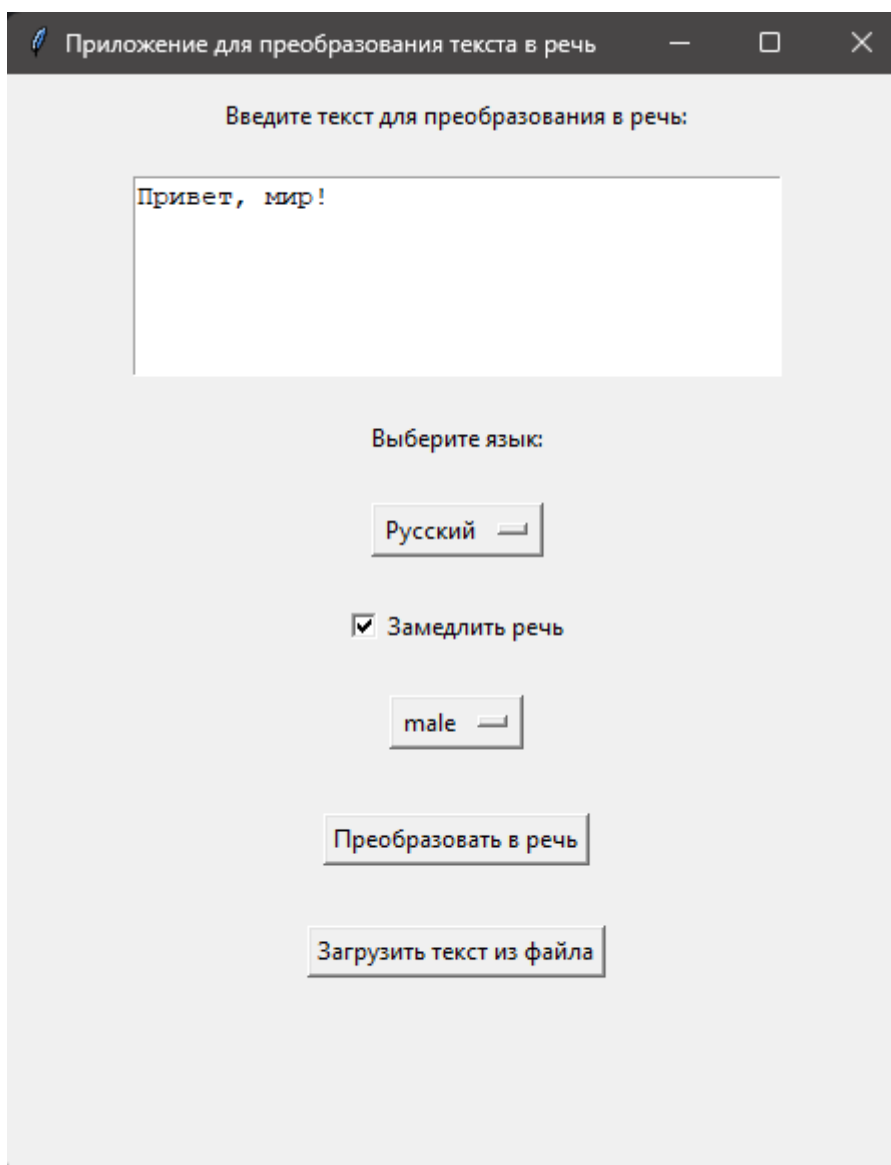


Рисунок 9. Внешний вид программы

4 Выводы

В результате исследования было создано приложение, позволяющее преобразовывать текст в речь с возможностью выбора параметров, таких как язык, пол голоса и скорость. Это приложение может быть использовано в различных сферах, таких как создание обучающих программ, помощь людям с ограниченными возможностями, а также для улучшения взаимодействия с устройствами на базе искусственного интеллекта. В будущем возможно расширение функционала программы, например, путем добавления новых языков, улучшения качества синтеза речи и интеграции с другими голосовыми технологиями.

Библиографический список

1. Расулов В.И. Сравнительный анализ библиотек text-to-speech языка программирования python // Международная научно-техническая конференция молодых ученых БГТУ им. В.Г. Шухова, посвященная 170-летию со дня рождения В.Г. Шухова. 2023. С. 401-405.
2. Кривоносов Р.А., Перевозников В.В. Разработка голосового ассистента на языке программирования python в целях совершенствования информационного обеспечения служебно-боевой деятельности войск // Перспективы совершенствования технической подготовки военнослужащих и сотрудников войск национальной гвардии Российской Федерации. 2021. С. 267-273.
3. Жумашева Ж., Сулейменов М., Айтуганова Ж., Ертаева Д. Разработка голосового помощника на базе Windows для управления персональным компьютером // Вестник КазАТК. 2024. Т. 134. №. 5. С. 206-214.
4. Айдашева К.Т., Шигабетдинова Д.И. Разработка голосового помощника на языке программирования python // XXV Туполевские чтения (школа молодых ученых). 2021. С. 10-13.
5. Manju S., Anitha J., Sujitha J.D. Investigation of Handwritten Image-To-Speech Using Deep Learning // 2024 International conference on advances in modern age technologies for health and engineering science (amathe). 2024. С. 1-5.