

## 작업 프로세스

- 시간제한이 있는 과제의 특성상 빠르고 간편하게 작업할 수 있는 구글 코랩과 뤼튼AI를 활용하였습니다.
- 아래는 과제에 적합한 TTS 라이브러리를 찾기 위해 뤼튼 AI 에게 질문한 내용입니다.

TTS 라이브러리 각각의 장단점은 뭐야?

🦋 답변

각 TTS 라이브러리의 장단점은 다음과 같습니다:

### 1. gTTS (Google Text-to-Speech)

- 장점:
  - 사용이 간편하고 설치가 쉬움.
  - 다양한 언어 지원.
  - 무료로 제공되며, 품질이 좋음.
- 단점:
  - 인터넷 연결이 필요함 (온라인 서비스).
  - 커스터마이징 옵션이 제한적.

### 2. pyttsx3

- 장점:
  - 오프라인에서 작동, 인터넷 필요 없음.
  - 다양한 TTS 엔진을 지원하여 유연성 있음.
  - 플랫폼 독립적.
- 단점:
  - 음성 품질이 gTTS보다 떨어질 수 있음.
  - 설치 및 설정이 gTTS보다 복잡할 수 있음.

### 3. Microsoft Azure TTS

- 장점:
  - 고품질의 자연스러운 음성 제공.
  - 다양한 음성 스타일과 감정 표현 가능.
  - 클라우드 기반으로 확장성이 좋음.
- 단점:
  - 유료 서비스로 비용이 발생함.
  - 설정이 복잡할 수 있음.

#### 4. Amazon Polly

- 장점:
  - 높은 품질의 음성을 제공하며, 다양한 언어 지원.
  - 실시간 음성 합성과 스트리밍 가능.
  - 다양한 음성 옵션 제공.
- 단점:
  - 유료 서비스이며, 사용량에 따라 비용이 발생.
  - AWS 계정 및 설정이 필요.

#### 5. IBM Watson Text to Speech

- 장점:
  - 다양한 언어 지원과 고품질 음성 제공.
  - 음성의 톤과 스타일 조정 가능.
  - API를 통한 손쉬운 통합.
- 단점:
  - 유료 서비스로 비용 발생.
  - 설정이 복잡할 수 있음.

#### 6. Coqui TTS

- 장점:
  - 오픈소스 프로젝트로 무료로 사용 가능.
  - 다양한 모델을 제공하여 커스터마이징 가능.
  - 커뮤니티 지원이 활발함.
- 단점:
  - 설치 및 설정이 복잡할 수 있음.
  - 음성 품질은 사용되는 모델에 따라 다름.

- 위의 답변을 통해 간편하면서도 성능이 좋은 gTTS 로 작업을 하기로 결정했습니다.
- 그리고 적합한 코드 내용을 도출하기 위해 아래와 같은 질문을 하여 코드를 도출하였습니다.

구글 코랩 환경에서 gTTS 라이브러리를 사용해 한국어 텍스트를 음성으로 변환하는 파이썬 코드를 작성할거야.  
사용자가 텍스트를 직접 작성하거나 텍스트 파일을 업로드 하면 그 내용을 음성으로 변환하는 프로그램이야.  
그리고 생성된 음성을 재생할 수 있는 방법도 포함해 줘

## 🦋 답변

아래는 구글 코랩 환경에서 gTTS 라이브러리를 사용하여 한국어 텍스트를 음성으로 변환하는 프로그램의 코드입니다. 사용자가 직접 텍스트를 입력하거나 텍스트 파일을 업로드하면 그 내용을 음성으로 변환하고, 생성된 음성을 재생할 수 있는 기능을 포함하고 있습니다.

```
python

import os
from google.colab import files
from gtts import gTTS
import IPython.display as ipd

# 음성 변환 함수
def speak_text(text):
    tts = gTTS(text=text, lang='ko')
    filename = 'output.mp3'
    tts.save(filename)
    return filename

# 텍스트 입력 받기
def get_text_input():
    text_input = input("음성으로 변환할 텍스트를 입력하세요: ")
    if text_input:
        audio_file = speak_text(text_input)
        return audio_file # 음성 파일 경로 반환
    else:
        print("텍스트를 입력하지 않았습니다.")
        return None
```

```

# 파일 업로드 및 내용 읽기
def upload_file():
    uploaded = files.upload()
    for filename in uploaded.keys():
        with open(filename, 'r', encoding='utf-8') as file:
            file_content = file.read()
            audio_file = speak_text(file_content)
            return audio_file # 음성 파일 경로 반환
    return None

# 선택지 제공
audio_file = None # 변수 초기화
choice = input("1. 텍스트 입력\n2. 파일 업로드\n원하는 옵션의 번호를 입력하세요: ")
if choice == '1':
    audio_file = get_text_input()
elif choice == '2':
    audio_file = upload_file()
else:
    print("잘못된 선택입니다.")

# 음성 재생
if audio_file:
    audio = ipd.Audio(audio_file) # 음성 객체 생성
    display(audio) # 음성 재생

```

- 위와 같은 코드를 복붙하여 코랩에서 테스트를 하였습니다.
- 코드를 실행하면 아래와 같은 디스플레이가 실행됩니다.

1. 텍스트 입력  
2. 파일 업로드  
원하는 옵션의 번호를 입력하세요:

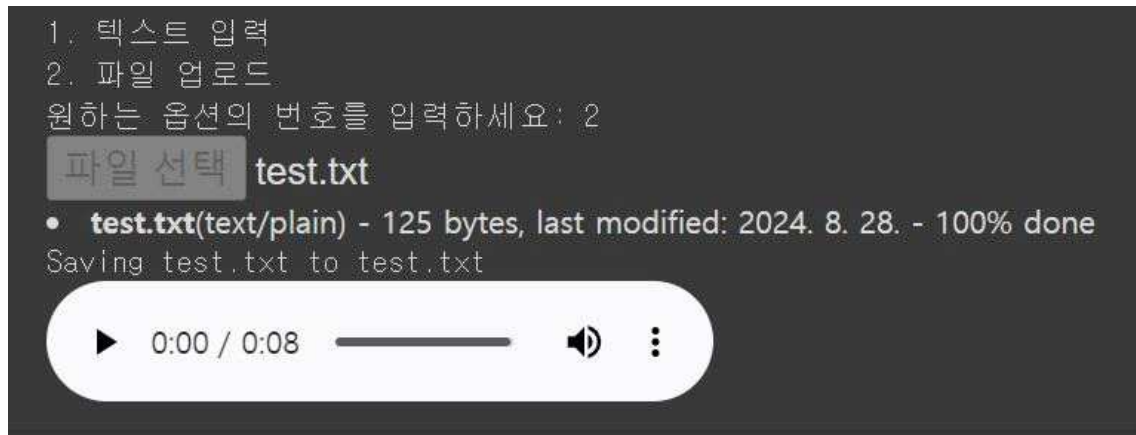
- 1번을 선택하면 텍스트를 입력할 수 있는 창이 나오고, 텍스트를 입력하면

1. 텍스트 입력  
2. 파일 업로드  
원하는 옵션의 번호를 입력하세요: 1  
음성으로 변환할 텍스트를 입력하세요: 안녕하세요 김하늘입니다

▶ 0:00 / 0:02 ————— 🔊 ⋮

- 이렇게 음성을 들어볼 수 있습니다.

- 코드를 재실행 하여 이번에는 2번 파일 업로드를 재생하는 것을 테스트 했습니다.



- 미리 준비한 텍스트 파일을 업로드 해 주면 역시 음성을 들어볼 수 있습니다.

<과제를 마무리하며>

부트캠프에서 부가서비스로 음성서비스를 만든 적이 있는데 그때는 pyttsx3 라이브러리를 사용한 적이 있습니다.

pyttsx3 역시 기본적인 반응 등 쓸만한 라이브러리라고 생각했지만, 띄어쓰기에 대한 반응, 음성 크기, 성우의 발음이나 느낌 등이 아쉽다고 느꼈던 기억이 있습니다.

그래도 사용했던 것을 하는 것이 빠르게 할 수 있지 않을까 라는 고민을 잠시 하기도 했지만 이번 테스트를 통해 사용해 보지 않았던 TTS 라이브러리를 테스트 해보고 싶었고 마침 AI 검색을 통해 과제의 요구 사항과 잘 맞는 라이브러리 인 것을 확인해서 적용할 수 있었습니다.

시간이 더욱 많았다면 streamlit 등을 활용해 조금 더 사용자에게 친화적인 서비스를 만들 수 있었으면 좋았겠다라는 마음이 듭니다. 그렇지만 제한된 시간을 지키는 것도 중요하기에 이렇게 마무리 짓도록 하겠습니다.

(\* 코드 출처 뽀튼 AI)