웹으로 텍스트 및 음성 스트리밍 서비스하기

OpenAI Chat GPT

OpenAPI 활용

https://chat.openai.com/

웹으로 텍스트 및 음성 스트리밍 서비스하기

웹으로 텍스트 및 음성 스트리밍 서비스하기

- 음성 스트리밍
- 음성으로 영화 추천 웹서비스하기
- 텍스트로 스트리밍하고 음성으로 변환해서 출력하기
- FastAPI를 이용한 엔드포인트 구현과 JavaScript를 이용한 프론트엔드 구현

음성 스트리밍

11. 음성 스트리밍 서비스하기

```
import pyaudio

p = pyaudio.PyAudio()
stream = p.open(format=8,
                channels=1,
                rate=24_000,
                output=True)
```

- Speech API는 청크 전송 인코딩을 사용하여 실시간 오디오 스트리밍을 지원합니다 .
- 즉, 전체 파일이 생성되고 액세스 가능하게 되기 전에 오디오를 재생할 수 있습니다.
- pyaudio 라이브러리를 이용해서 스트리밍으로 음성을 제공할 수 있습니다.

```
with client.audio.speech.with_streaming_response.create(
        model="tts-1",
        voice="alloy",
        input="""I see skies of blue and clouds of white
             The bright blessed days, the dark sacred nights
             And I think to myself
             What a wonderful world""",
        response_format="pcm"
) as response:
    for chunk in response.iter_bytes(1024):
        stream.write(chunk)
```

- 이 코드는 Colab 환경에서는 실행되지 않습니다.
- 로컬 환경에서 실행해야 스피커를 통해 소리가 출력됩니다.

음성으로 영화 추천 웹서비스하기

11. 음성 스트리밍 서비스하기

1. 사용자 프롬프트 입력 후 음성으로 요청

2. GPT 이용해서 영화 추천
3. TTS를 이용해서 추천한 데이터를 음성으로 변환
4. pcm 파일을 wav로 변환 후 응답

5. 오디오 스트림을 받아 플레이

음성으로 영화 추천 웹서비스하기 (파이썬 코드)

11. 음성 스트리밍 서비스하기

```python
@app.get("/audio_stream")
async def audio_stream(prompt: str = Query(..., description="Text to convert to speech")):
    messages.append({"role": "user", "content": prompt})
    chat_response = client.chat.completions.create(
        model="gpt-4o-mini",
        messages=messages,
    )
    text = chat_response.choices[0].message.content

    async def audio_event_stream():
        pcm_data = BytesIO()
        with client.audio.speech.with_streaming_response.create(
                model="tts-1",
                voice="alloy",
                input=text,
                response_format="pcm"
        ) as response:
            for chunk in response.iter_bytes(1024):
                pcm_data.write(chunk)
                await asyncio.sleep(0)
        pcm_data.seek(0)
        wav_audio = pcm_to_wav(pcm_data.read())
        return wav_audio

    wav_stream = await audio_event_stream()
    headers = {
        "Content-Type": "audio/wav",
        "Cache-Control": "no-cache",
        "Connection": "keep-alive",
    }
    return StreamingResponse(wav_stream, headers=headers, media_type="audio/wav")
```

FastAPI 기반의 애플리케이션에서 사용자의 텍스트 입력을 받아 음성으로 변환하여 스트리밍하는 엔드포인트(/audio_stream)를 구현한 것입니다.

- 비동기 함수 audio_event_stream은 TTS(Text-to-Speech) 모델을 호출하여 텍스트를 음성 데이터로 변환합니다.
- OpenAI의 TTS 모델(tts-1)을 사용하며, 응답 데이터 형식은 PCM(원시 오디오 형식)입니다.
- response.iter_bytes(1024)를 사용해 스트리밍 방식으로 데이터를 청크 단위(1024 바이트)로 받아옵니다.
- 데이터를 메모리 버퍼(BytesIO)에 저장하면서 대기(await asyncio.sleep(0))합니다.

```python
# PCM 데이터를 WAV로 변환하는 함수
def pcm_to_wav(pcm_data):
    wav_io = BytesIO() # WAV 데이터를 메모리에 임시로 저장하기 위해 BytesIO 객체를 생성
    with wave.open(wav_io, "wb") as wav_file:
        wav_file.setnchannels(1)  # 모노(Mono) 채널, 스테레오로 저장하려면 2를 설정
        wav_file.setsampwidth(2)  # 16-bit PCM(일반적으로 오디오 데이터는 16비트를 많이 사용함)
        wav_file.setframerate(24000)  # 초당 24 kHz sample rate
        wav_file.writeframes(pcm_data) # WAV 파일에 프레임 단위로 기록
    wav_io.seek(0)
    return wav_io
```

인간의 청각에 적합한 고품질 오디오를 제공하면서 데이터 크기를 줄이기 위해 24kHz를 사용

- CD 품질은 44.1kHz, 전화 음질은 8kHz

- audio_event_stream에서 생성된 WAV 데이터를 StreamingResponse 객체로 반환합니다.
- 클라이언트는 HTTP 스트림을 통해 실시간으로 변환된 음성을 받을 수 있습니다.
- 응답 헤더는 audio/wav 타입을 지정하며, 캐싱을 비활성화(no-cache)하고 스트리밍 연결(keep-alive)을 유지합니다.

음성으로 영화 추천 웹서비스하기 (HTML, JavaScript 코드)

11. 음성 스트리밍 서비스하기

```
<script>
    document.getElementById('submit2').addEventListener('click', function() {
        const text = document.getElementById('prompt').value;
        if (!text) {
            alert("텍스트를 입력하세요.");
            return;
        }

        const audioElement = document.getElementById('audioContainer');
        const audioSource = document.getElementById('audioSource');
        audioSource.src = '';

        document.getElementById('response').innerHTML = '';
        document.getElementById('loader').style.display = 'block';

        const audioStreamUrl = '/audio_stream?prompt=' + encodeURIComponent(text);

        fetch(audioStreamUrl)
            .then(response => response.blob())
            .then(blob => {
                document.getElementById('loader').style.display = 'none';
                console.log(blob);
                const audioUrl = URL.createObjectURL(blob);
                console.log(audioUrl);
                audioSource.src = audioUrl;
                audioElement.load();
                audioElement.play();
            })
            .catch(error => console.error("오디오 스트림 오류:", error));
    });
</script>
```

이 JavaScript 코드는 사용자가 버튼을 클릭했을 때, 입력된 텍스트를 기반으로 오디오를 스트리밍하여 재생하는 기능을 구현합니다.

- document.getElementById('submit')로 submit ID를 가진 버튼을 가져옵니다.

아래의 HTML 코드가 자바스크립트 코드 위에 있어야 합니다.

```
<button id="submit2">음성으로 듣기</button>
<audio id="audioContainer" preload="none" controls>
    <source id="audioSource" src="" type="audio/wav">
</audio>
```

- fetch를 사용해 서버에서 반환된 데이터를 Blob 형태로 처리합니다.
- 로더를 숨기고 Blob 데이터를 URL 객체로 변환하여 오디오 소스로 설정합니다.
- 생성된 URL을 audioSource의 src 속성에 할당하고, 오디오를 로드한 뒤 재생합니다.

텍스트로 스트리밍하고 음성으로 변환해서 출력하기

11. 음성 스트리밍 서비스하기

1. 사용자 프롬프트 입력 후 SSE를 통해 스트리밍 요청 요청

2. GPT 이용해서 영화 추천
3. 텍스트를 스트리밍으로 응답(SSE 응답)

4. SSE 응답을 처리하여 텍스트 스트리밍
5. SSE 응답이 완료되면 추천 텍스트를 음성으로 변환 요청

6. TTS를 이용해서 텍스트 데이터를 음성으로 변환
7. pcm 파일을 wav로 변환 후 응답

8. 오디오 스트림을 받아 플레이

텍스트로 스트리밍하고 음성으로 변환해서 출력하기 (파이썬 코드)

11. 음성 스트리밍 서비스하기

```python
@app.post("/convert_text_to_speech")
async def convert_text_to_speech(request: Request):
    data = await request.json()
    text = data.get("text")

    # TTS 변환 코드
    pcm_data = BytesIO()
    with client.audio.speech.with_streaming_response.create(
            model="tts-1",
            voice="alloy",
            input=text,
            response_format="pcm"
    ) as response:
        for chunk in response.iter_bytes(1024):
            pcm_data.write(chunk)
            await asyncio.sleep(0)

    pcm_data.seek(0)
    wav_audio = pcm_to_wav(pcm_data.read())

    return StreamingResponse(wav_audio, media_type="audio/wav")
```

- request.json(): 요청 본문에서 JSON 데이터를 비동기적으로 읽습니다.
- data.get("text"): JSON 객체에서 text 키에 해당하는 값을 가져옵니다. 이 값이 음성으로 변환할 텍스트입니다.

텍스트를 스트리밍하는 엔드포인트는 앞에서 만든 '/chat_stream' 엔드포인트를 사용합니다.

- 이 코드는 텍스트를 받아 음성으로 변환하여 스트리밍하는 엔드포인트('/convert_text_to_speech') 입니다.

- 외부 TTS API나 라이브러리를 사용해 스트리밍 방식으로 음성을 생성합니다.
- 모델은 tts-1 모델을 사용합니다.
- voice="alloy"는 음성 스타일을 지정합니다.
- 입력: 변환할 텍스트(input=text).
- response_format="pcm"은 음성 데이터를 PCM 형식으로 반환하도록 지정합니다.

- TTS 응답은 큰 데이터일 수 있으므로, 1024바이트씩 청크 단위로 읽습니다.
- pcm_data.write(chunk): 각 청크를 pcm_data에 추가로 저장합니다.
- await asyncio.sleep(0): 비동기 루프에서 다른 작업들이 중단되지 않도록 컨텍스트를 양보합니다.

- pcm_data.seek(0): BytesIO 객체의 읽기 포인터를 처음으로 되돌립니다.
- pcm_to_wav 함수: PCM 데이터를 WAV 형식으로 변환하는 함수입니다

- StreamingResponse: 변환된 WAV 데이터를 스트리밍 방식으로 반환합니다.
- media_type="audio/wav": 반환 데이터의 MIME 타입을 audio/wav로 지정하여 클라이언트가 올바르게 해석하도록 합니다.

텍스트로 스트리밍하고 음성으로 변환해서 출력하기 (JavaScript 코드)

11. 음성 스트리밍 서비스하기

```
<script>
    document.getElementById('submit3').addEventListener('click', function() {
        const prompt = document.getElementById('prompt').value;
        if (!prompt) {
            alert('프롬프트를 입력해주세요.');
            return;
        }

        document.getElementById('response').innerHTML = '';
        document.getElementById('loader').style.display = 'block';

        // 기존 EventSource가 있다면 닫기
        if (window.eventSource) {
            window.eventSource.close();
        }

        // 새로운 EventSource 생성
        window.eventSource = new EventSource('/chat_stream?prompt=' + encodeURIComponent(prompt));

        let fullText = "";  // 전체 텍스트를 저장할 변수

        window.eventSource.onmessage = function(event) {
            const data = JSON.parse(event.data);
            if (data.status === 'processing') {
                document.getElementById('response').innerHTML += data.data;
                fullText += data.data;  // 스트리밍된 텍스트를 이어서 저장
            } else if (data.status === 'complete') {
                document.getElementById('loader').style.display = 'none';
                window.eventSource.close();
```

이 코드의 주요 흐름

1. 사용자가 프롬프트를 입력하고 버튼을 클릭합니다.
2. 서버와 스트리밍 연결을 열어 텍스트 응답을 실시간으로 받습니다.
3. 스트리밍 완료 후, 응답 텍스트를 음성으로 변환 요청을 보냅니다.
4. 음성 데이터를 받아와 브라우저에서 재생합니다.

여기 추가되는 코드를 제외하고 나머지는 submit 버튼을 누를 때 실행되는 코드와 같은 코드입니다.

텍스트로 스트리밍하고 음성으로 변환해서 출력하기 (JavaScript 코드)

11. 음성 스트리밍 서비스하기

```
                document.getElementById('loader').innerHTML = '음성으로 변환하는 중...
                document.getElementById('loader').style.display = 'block';

                // TTS 변환을 위한 POST 요청
                fetch('/convert_text_to_speech', {
                    method: 'POST',
                    headers: {
                        'Content-Type': 'application/json',
                    },
                    body: JSON.stringify({ text: fullText })
                })
                .then(response => response.blob())
                .then(blob => {
                    document.getElementById('loader').innerHTML = '불러오는 중...'
                    document.getElementById('loader').style.display = 'none';
                    const audioUrl = URL.createObjectURL(blob);
                    const audioElement = document.getElementById('audioContainer');
                    const audioSource = document.getElementById('audioSource');
                    audioSource.src = audioUrl;
                    audioElement.load();
                    audioElement.play();
                })
                .catch(error => console.error("오디오 변환 오류:", error));
            }
        };

        window.eventSource.onerror = function(err) {
            console.error('EventSource 오류:', err);
            document.getElementById('loader').style.display = 'none';
            window.eventSource.close();
        };
    });
</script>
```

추가되는 코드입니다.

- 서버의 /convert_text_to_speech 엔드포인트에 POST 요청을 보냅니다.
- 요청 본문에는 누적된 전체 텍스트(fullText)를 JSON 형식으로 전달합니다.

- response.blob(): TTS 서버의 응답 데이터를 Blob(바이너리 데이터)으로 변환합니다.
- URL.createObjectURL: Blob 데이터를 브라우저에서 사용할 수 있는 URL로 변환합니다.
- HTML 오디오 재생:
- audioSource 요소의 src 속성에 Blob URL을 설정합니다.
- 오디오 요소(audioElement)를 로드하고 바로 재생(play())합니다.