Howto for TTS with Emacs-Reveal *

Jens Lechtenbörger

December 30, 2025

This presentation is an example of how emacs-reveal can be used to create presentations with automatically generated audio via text-to-speech, or TTS for short. Please check out the source file of this presentation for details. Prior knowledge of emacs-reveal is required.

Personally, the author of this software does not like learning from videos as they exhibit limited navigation capabilities (no skim reading, limited search, no document structure, no hyperlinks). His students like videos, though.

Maintaining high-quality video and audio over years is a considerable challenge. To overcome this challenge, reveal.js presentations can be run in a video mode where slides come with audio explanations and advance automatically. This presentation serves as example.

1 General thoughts

- All of this builds on emacs-reveal (Lechtenbörger 2019a, 2019b)
  - Translate Org mode source documents to HTML presentations with reveal.js
  - Check out emacs-reveal howto

  Briefly, emacs-reveal is a free and open-source software bundle to create HTML presentations from Org mode source documents as open educational resources based on the JavaScript presentation framework reveal.js. A howto document and scientific articles explain emacs-reveal in some detail.

- Text-To-Speech (TTS) reads notes (`#+begin_notes ... #+end_notes`)
  - Controlled by option `reveal-with-tts`
    - Use customization for available speakers
  - Play audio with audio slideshow plugin for Reveal.js

  This presentation demonstrates some features of emacs-reveal related to TTS.

- If slides with audio advance automatically, this is a video mode
  - Then, notes are required for every slides
  - Reveal.js "fragments" (animations) are still possible

  As you see here, animations work in a video-like mode, with slides advancing automatically.

*This PDF document is an inferior version of an OER in HTML format; free/libre Org mode source repository.

1.1 Technical Idea

- Implement TTS as two-stage process
  - First, extract notes from presentation

    Speech is generated from text in a process with two stages. First, usual presentation notes serve as text input. These notes may embed SSML `break` elements to specify a break with a given duration in seconds between sentences. See the source code of this slide for examples.

    * Generate a text file for each note
      · Its name is a hash value of the contents

      While processing the Org source code to generate a presentation, each note is extracted into a text file (with some preprocessing as revisited on a later slide). The name of such a text file is the hash value of its contents. Thus, changing contents lead to changing names.
    * Generate one index file that stores names (and other information) for all text files

      In addition, another text file serves as index, collecting the names and positions of texts in a presentation. Besides, this index file also records configuration information, such as the speaker to be used.
    * This happens during export/publication of Org files into reveal.js presentations

      This text processing happens automatically in the background.
  - Second, run TTS software on index file to generate audio
    * Implemented in Docker image emacs-reveal/tts
      · Image includes several TTS implementations, highest quality offered by Kokoro (next to SpeechT5 and SpeechBrain)
    * StyleTTS2 available in Docker image emacs-reveal/tts-styletts2
      · Activate with default voice: `#+OPTIONS: reveal_with_tts:StyleTTS2`
      · Or with target audio for voice cloning: `#+OPTIONS: reveal_with_tts:StyleTTS2:/oer/t`
    * Generated audio shares hash value of its text as part of its name, enabling caching of unchanged audio

  Second, the index serves as input for the text to speech implementation, which is available as Docker image. Here, names of generated audio again embed the hash values of their input texts, enabling caching of unchanged audio.

- Use audio slideshow plugin to play audio

In presentations, audio is played with the audio slideshow plugin.

1.2 Docker image `emacs-reveal/tts`

- Contains free/libre and open TTS implementations
  - SpeechBrain
  - Microsoft SpeechT5
  - Kokoro
- For size reasons, without GPU support
- Small wrapper package tts.py

- Sample invocation shown in .gitlab-ci.yml of this presentation
- "Production" settings in course IT Systems

This slide mentions some technical aspects of the text to speech approach. Please see for yourself if you are interested.

2 Slide with notes and fragments

Notes are transformed to audio by TTS and read by the audio plugin (if it is enabled). Org-re-reveal converts text to have each sentence on a single line, which is converted to audio by a Docker image of emacs-reveal.

Depending on the TTS model, hyphenated words and abbreviations may not be pronounced correctly. However, org-re-reveal contains a customizable set of translation rules for preprocessing.

Notes can contain Org markup, such as hyperlinks, **bold**, *emphasis*, `code`, `verbatim`.
Such markup is removed for TTS in org-re-reveal.
Lists can be used in notes as well:

1. This is a first item in a list.
2. Second item.

As we aim for text to speech, notes should consist of full sentences, including full stops, question marks etc. Warnings are shown upon export if the code detects this not to be the case.

Notes on this slide clarify some aspects of the text generated by org-re-reveal as basis for TTS. To pronounce numbers, abbreviations, and "complicated" word, see variable `org-re-reveal-tts-normalize-table`.

Besides, for demonstration purposes, this slide contains fragments with separate notes:

- First appearing point, with notes

  Each fragment has its own notes.

  These ones are meant for the first bullet point.

- Second appearing point

  Explanations continue with this second bullet point.

3 A real example

- Next slide is part of a course on IT Systems
- Some evaluation results regarding TTS quality and presentation features are provided towards the end of the Readme of emacs-reveal

The next slide embeds a real slide from a course on IT Systems. Thus, its references to surrounding contents need to be understood in the context of that course. See there if you are interested in the concept of virtual memory in operating systems.

As technical detail, note that the figure on that slide embeds names of audio files, which are defined by `audio-name` properties on the notes.

On a general note, I publish some evaluation results regarding presentation features and TTS quality in the Readme of emacs-reveal.

3.0.1 Offset as Pointer into Range

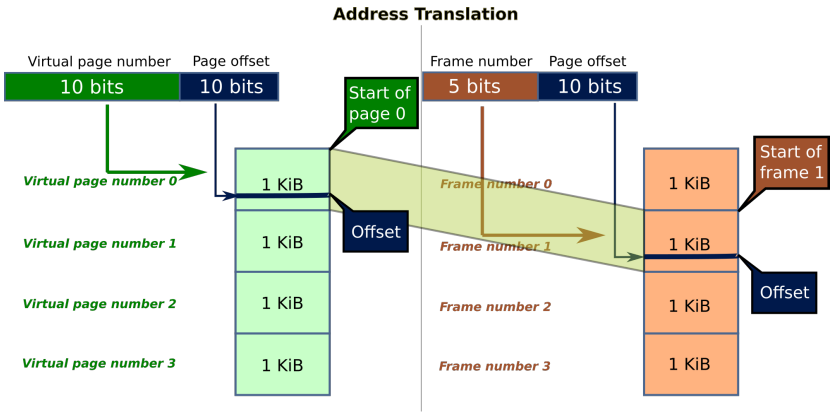

Figure 1: "Address translation with offset in covered address range" by Max Lütkemeyer and Jens Lechtenbörger under CC BY-SA 4.0; from GitLab

For a different view on the hierarchical nature of virtual addresses, let us continue the previous scenario of virtual addresses of 20 bits, to be translated to physical addresses of 15 bits, with a page size of 1 KiB.

Out of the $2^{10} = 1024$ possible pages and $2^5 = 32$ possible frames, only the first four of each type are shown.

As before, suppose that page 0 is located in frame 1 as recorded in the page table. Thus, for translation of addresses falling into page 0, the 0 encoded in the first 10 bits of the virtual address is replaced by a 1 encoded in the first 5 bits of the physical address. Importantly, the 10 offset bits do **not** change under address translation.

Note how, given 10 bits for the offset, each page and each frame cover a range of 1024 addresses. The offset identifies a single byte in that range.

Subsequent slides provide sample calculations for address translation.

4 The End

Figure 2: The road ahead . . . ("Figure" under CC0 1.0; converted from Pixabay)

`https://gitlab.com/oer/`

The end is near. Or the beginning?

4.1 Bibliography

Lechtenbörger, Jens. 2019a. "Emacs-reveal: A software bundle to create OER presentations." *Journal of Open Source Education (Jose)* 2 (18). https://doi.org/10.21105/jose.00050.

———. 2019b. "Simplifying license attribution for OER with emacs-reveal." In *17. Fachtagung Bildungstechnologien (DELFI 2019)*, edited by Niels Pinkwart and Johannes Konert, 205–16. Bonn: Gesellschaft für Informatik e.V. https://doi.org/10.18420/delfi2019_280.

My presentations usually contain a bibliography like this one. Did you notice the references on an earlier slide? Those are hyperlinks into this slide.

License Information