# Beamer Presentations with Org Mode

Randy Ridenour

June 1, 2020

## Introduction

When we moved all courses online for the semester because of the COVID-19 pandemic, I found myself having to make many more presentation slide-decks than I normally do (especially when "normal" is none). Keynote produces some beautiful slides, but it requires using the mouse more than I like. I wanted something that I could use to make slides with just a few keystrokes. I settled on exporting an Org mode document to Beamer sides. This is an explanation of the challenges that I had and an example of the solution that I found.

### Advantages of Beamer

- Plain text
- The power of LaTeX
- Produces platform-agnostic PDF

The Beamer class is a tool used to produce presentations with LaTeX. It provides the same advantages for presentations that standard LaTeXprovides for other document types, including elegant mathematical formulas, transportability between computing platforms, and the ability focus to on content when writing.

### Disadvantages of Beamer

*Writing LaTeXcan be cumbersome!*

LaTeXis not a simple markup language. After gaining a certain proficiency, it is possible to write it fairly easily. Even so, it is never as simple as using something like Markdown.

### Solution

- Emacs Org Mode
- Problem: producing presentation and article with same content
- No easy solution found online

The solution that I found was Org mode, a very powerful system that includes a simple markup language and the ability to export to LaTeX. Producing the slide deck was easy, and there were many great examples online. Producing an article was also easy, again with great examples. Producing both a slide presentation and an article, using the same contents file should have been easy, since the official guide to Beamer explains how to do that in LaTeX. Unfortunately, I couldn't find much guidance on how to do this with Org mode. The only article I could find was this on the official Org mode site. It seemed too complicated, and I was tempted to just go back to directly writing LaTeX. I decided to try anyway, but I quickly ran into a series of problems:

1. The Org Beamer export automatically inserted a title slide with `\maketitle`. Unfortunately, using the same content for the presentation and notes requires using the `ignorenonframetext` switch in the document class declaration. Since the export didn't place the command in a frame, there was no title slide.
2. I attempted to fix this by making beginning the contents file with a title-less frame that included the `\maketitle` command. That generated a title page, but it came *after* the table of contents slide.
3. The article export did not treat headings correctly, and failed to recognize the Beamer-specific commands.

This was enough to make me want to scrap the project, especially when I looked at the site linked above. The author fixed the title issue by hacking the Beamer export file, something that I certainly didn't want to do. So, as is often the case, after hours of searching for solutions, I began to wonder if the solution could be much simpler than I (or anyone else, it seems) was thinking.

### Details

I'll spare any reader the record of attempts, mistakes, other attempts, more mistakes, etc., and just get to the final workflow. I wrote a small script (in my case a function in Fish) that creates three files. One for the presentation, one for the article, and one for shared content. After creating the files, it opens a Dired buffer of the relevant folder in Emacs.[1]

For those who use the Fish shell, here is the function:

```
function lecture
    touch {$argv}.org
  echo -e '#+startup: beamer' \n'#+TITLE: ' \n'#+AUTHOR: Dr. Ridenour' \n'#+DATE: ' >>{$argv}.org
    touch {$argv}-beamer.org
  cat /Users/rlridenour/Dropbox/emacs/beamer/lecture-beamer.org >{$argv}-beamer.org
    echo -e '#+include: "'{$argv}'.org" :minlevel 1' >>{$argv}-beamer.org
    touch {$argv}-notes.org
   cat /Users/rlridenour/Dropbox/emacs/beamer/lecture-notes.org >{$argv}-notes.org
```

1. The function to open a directory in Dired is very small: `emacsclient -e "(dired \"$PWD\")"`.

```
    echo -e '#+include: "'{$argv}'.org" :minlevel 1' >>{$argv}-notes.org
    dired
end
```

So, entering "lecture kant" in the shell will open a Dired buffer containg the files kant.org, kant-beamer.org, and kant-notes.org.

### Contents File

- One org file for presentation contents
- No header information
- Choose heading level to use as slides
  - I use h3

The contents file is a standard org file. Initially, I had it containing nothing in the header, except for possibly one line containing `#+startup: beamer`, which makes it easier to insert some Beamer-specific commands. After getting tired of entering the same data twice in the other files, I wondered if shared header information could just be placed in the contents file. Occasionally things work exactly how hoped they would, so now the function adds the following lines at the top of the contents file:

```
#+startup: beamer
#+TITLE:
#+AUTHOR: Dr. Ridenour
#+DATE:
```

You will need to decide what heading level will designate a slide. Don't worry, you'll still be able to use that heading level in the article, as I'll explain later. I use h3, so slides begin with a line like this in the contents file:

```
*** Slide Title
```

To add notes, you need to structurally separate the note content, which should only be printed on the article, from the preceding slide. To do this, add another h3 heading (I creatively title it "Notes") with instructions to ignore the heading:

```
 *** Notes :B_ignoreheading:
:PROPERTIES:
:BEAMER_env: ignoreheading
:END:
```

Any text that follows will only appear in the article, not in the presentation. This does not have to be done for every successive note paragraph, it only needs to be done after a slide. So, any paragraphs that are in the scope of an h1 or h2 heading won't need that.

### Presentation File

- Specify slide level: `#+OPTIONS: H:3`
- Add `ignorenonframetext` to `#+LaTeX_CLASS_options`
- Use `#+OPTIONS: toc:nil`
- Begin with two empty slides
  - First with `\maketitle`
  - Second with `\tableofcontents`
- Add contents file with `#+include: "contents.org" :minlevel 1`

The magic happens with two small files. The first is the presentation file. At the top, put your preferred Beamer export header, but be sure to include `#+LaTeX_CLASS_options: [ignorenonframetext]` and `#+OPTIONS: toc:nil`. The latter is to ensure that Beamer export doesn't make the contents slide before the presentation title slide. Then, make the title page like this:

```
***
\maketitle
```

If you want a table of contents slide, you can do the same thing except use `\tableofcontents`. Finally, include the contents file with this line: `#+include: "contents.org" :minlevel 1`.

### Article File

- Use your normal article headers.
- Add `#+LaTeX_HEADER: \usepackage{beamerarticle}`
- Add contents file with `#+include: "contents.org" :minlevel 1`
- Be sure to process with Beamer export

For the article, use your preferred header with all of the packages declared, but be sure to add this line: `#+LaTeX_HEADER: \usepackage{beamerarticle}`. After the export header lines, include the content file, again using `#+include: "contents.org" :minlevel 1`. When exporting, be sure to export with the one of the Beamer exports. Otherwise, things just won't look right.

## Conclusion

### Lessons Learned

- Sometimes things don't have to be difficult.
- One remaining issue: verse environment won't work in beamerarticle.
- I hope this saves at least one person some time.
- Any questions or suggestions, please let me know!