to the Second Preliminary Edition
(dated December 13, 1983)
of the

L.8exaTEX Manual

Leslie Lamport

04/19/21

Appendix C

Making Color Slides With S L.32exiTEX

C.1 How S L.32exiTEX Makes Colors

S L.32exiTEX is a version of L.8exaTEX designed for making color slides, though you can use it for black-and-white slides as well. You donjt need a special printer to make color slides; S L.32exiTEX uses the same black-and-white printer as L.8exaTEX. You get color slides by copying S L.32exiTEXjs output onto colored transparencies. To see how this works, suppose you want to make the following slide:

The illustrations used here are not accurate representations of the output actually produced by S L. 32exiTEX.

where kREDl is colored red, kBLACKl is colored black, and kBLUEl is colored blue. S L.32exiTEX would generate the following three separate pages of output for this slide:

to
width 0pt height 0 depth 0

These pages are called *color layers*. Each color layer is then copied onto a special sheet that produces a transparency of the appropriate color. (Such sheets are commercially available for an assortment of colors.) The slide is produced by laying the three transparencies on top of one another.

I will refer to the text that is meant to be colored red on the slide, and is therefore printed by S L.32exiTEX on the red color layer, as kred textl. So, remember that when I write about the color of a piece of text, Ijm referring only to the color layer on which it appears; S L.32exiTEX doesnjt print anything in red ink.

Itjs hard to tell what a slide will look like from the separate color layers. Therefore, S L.32exiTEX also produces a black-and-white version of the slide, containing all the color layers properly superimposed. When first making a set of slides, you should generate only the black-and-white versions, making the color layers after youjve fixed all the problems that are visible in the black-and-white versions. If you donjt want color slides, you donjt have to make any color layers, and can copy the black-and-white versions onto transparencies.

C.2 The Root File

S L.32exiTEX is a separate program that you run the same way you run L.8exaTEX, giving it the first name of an input file whose second name is `.tex`. This file is called the *root file*. As usual, Ijll suppose that your root file is named `myfile.tex`. This file starts out with the customary `\pagelayout` and `\documentstyle` commands. The standard page layout and document styles for making slides are both named `slides`, so your file is likely to begin

```
\pagelayoutslides
\documentstyleslides
```

The commands are followed by any declarations that you may want to make, followed in turn by the `\begindocument`.

Any text that comes after the `\begindocument` is treated as kfront matterl and not as slide material. You can use it for notes to identify the slides.

[Sorry. Ignored `\beginexambox ... \endexambox`]

For S L.32exiTEX to produce color slides, you have to tell it what colors you will be using. This is done with the `\colors` command. The command

```
\colorsred,black,blue
```

states that you will be using three colors, which you have named `red`, `black`, and `blue`. S L.32exiTEX knows nothing about real colors, so you could just as well have called your three colors `puce`, `mauve`, and `fred`. If youjre making only black-and-white slides, then you donjt need a `\colors` command.

The text of your slides is contained not in `myfile.tex`, but in a separate *slide file*. This file can have any name that ends in `.tex`; I will assume that it is called `myslid.tex`. What goes into the file `myslid.tex` is explained below. Black-and white-slides are generated by placing the following command in the root file:

```
\blackandwhitemyslid
```

Color slides are generated by the command

```
\colorslidesmyslid
```

The `\colorslides` command generates a set of color layer pages for each color specified by the `\colors` command. For example, the command

```
\colorsred,black,blue
```

causes a subsequent `\colorslides` command to generate first all the red color-layer pages, then the black ones, and then the blue ones.

As usual, your root file ends with an `\enddocument` command.

C.3 The Slide File

The main purpose of the root file is to tell S L.32exiTEX what colors to use and where to find the slide file, so the root file tends to be pretty short. Itjs the slide file that actually makes the individual slides.

C.3.1 Slides

Each slide is produced by a `slide` environment. This environment has a single argument, which is a list of all the colors contained on the slide. For example, a slide that has the colors `red` and `blue` is created by an environment

```
\beginslidered,blue
...
\endslide
```

The colors in the argument must have been declared by a `\colors` command in the root file. They tell S L.32exiTEX which color layers to produce for this particular slide. If there is green text in the slide, that text will appear in the black-and-white version, but no green color layer will be generated unless `green` is included in the `slide` environmentjs argument. If you want only black-and-white slides, then you can use a null argument:

```
\beginslide
...
```

The text that appears on a slide is produced using ordinary L.8exaTEX commands. You can use any commands that make sense for slides. Commands that *donjt* make sense include sectioning commands, `figure` and `table` environments, indexing commands, commands for generating a bibliography, and page-breaking commands. The latter make no sense in a slide because each slide must fit on a single page. You can use an `\input` command, but not an `\include` command. Commands for producing only some of the slides in your slide file are described below.

There are two major differences between the text generated by S L.32exiTEX and that generated by L.8exaTEX. First of all, text is automatically centered vertically on the slide. Secondly, and most noticable, S L.32exiTEX uses a set of type faces especially chosen for slides. The characters in these type faces are much larger than the ones in the corresponding L.8exaTEX type faces. S L.32exiTEXjs `\normalsize` produces roughly the same size characters as L.8exaTEXjs `\LARGE`. Also, S L.32exiTEXjs ordinary Roman type style is similar to L.8exaTEXjs sans serif style. Besides Roman, the only other type styles generally available are italic (`\it`), bold (`\bf`), and typewriter (`\tt`).

The only commands you need inside a slide that arenjt present in ordinary L.8exaTEX input are ones to tell S L.

32exiTEX what color the text is. The `\colors` command in your root file defines the declarations for doing this. For example, if the root file contains the command `\colorsred, black,blue`, then `\red`, `\black`, and `\blue` are declarations that specify the color. They work just like any other declaration, such as `\bf`, having the same scoping rules. This is illustrated below, where only the red color layer is shown.

[Sorry. Ignored `\beginexambox ... \endexambox`]

A color declaration does not affect the type style, as illustrated by the following example, where again only the red color layer is shown.

[Sorry. Ignored `\beginexambox ... \endexambox`]

The command `\invisible` is a special color declaration for invisible text. Invisible text is not only colorless, appearing in no color layer, but does not appear in the black-and-white version either. The use of invisible text is explained below.

Warning: Donjt use a color declaration or an `?invisible` *command in math mode.*

Warning: Certain horizontal lines drawn by TEX *will appear in color layers where they shouldnjt. The offending lines are the ones produced by* `?underline`, `?overline`, *and* `?frac`.

C.3.2 Overlays

The `overlay` environment is exactly the same as the `slide` environment except for how the page is numbered. The first `overlay` following slide number 9 is numbered k9al, the second one is numbered k9bl, and so forth. To make an overlay that perfectly overlays a slide, the slide and the overlay should be absolutely identical except that text visible in one should be invisible in the other. This is illustrated by the following example.

[Sorry. Ignored `\beginexambox ... \endexambox`]

[Sorry. Ignored `\beginexambox ... \endexambox`]

The slide and the overlay will match up to read

An overlay goes here.

when placed atop one another.

C.3.3 Notes

It is sometimes convenient to put notes to yourself in with the slides. The `note` environment produces a one-page note that appears only in the black-and-white versions of the slides. For example,

[Sorry. Ignored `\beginexambox ... \endexambox`]

Notes that follow slide number 9 are numbered k9-1l, k9-2l, etc.

C.4 Making Some of the Slides

For making corrections, itjs handy to be able to produce a subset of the slides in your file. The command

```
\onlyslides4,7-13,23
```

in the root file will cause the following `\blackandwhite` and `\colorslides` commands to generate only slides numbered 4, 7-13 (inclusive) and 23, plus all of their overlays. The slide numbers in the argument must be in ascending order, and can include nonexistent slidesnfor example, you can type

```
\onlyslides10-9999
```

to produce all but the first nine slides. The argument of the `\onlyslides` command must be nonempty.

There is also an analogous `\onlynotes` command to generate a subset of the notes. Notes numbered 11-1, 11-2, etc. will all be generated by specifying page 11 in the argument of the `\onlynotes` command.

If your input has an `\onlyslides` command and no `\onlynotes` command, then notes will be produced for the specified slides. If there is an `\onlynotes` command but no `\onlyslide` command, then no slides will be produced. Including both an `\onlyslides` and an `\onlynotes` command has the expected effect of producing only the specified slides and notes.