Producing slides with LaTeX 2ε

Frank Mittelbach

2025-12-31

1 Introduction

With LaTeX 2ε it is now no longer necessary to maintain a special format for producing overhead slides. Instead the standard format may be used and internally only different font definition files come into play.

2 Usage

For producing slides you have to use `slides` as the document class. This class is very similar to the `slides` style that came with SLiTeX, in fact it is basically a copy changed to work under LaTeX 2ε.[1] Thus you have to say something like

```
\documentclass[...]{slides}
```

and process this with LaTeX 2ε.

3 Fonts

Note, that with NFSS you can easily produce slides with special fonts just by calling an appropriate style file (like `times`) in a `\usepackage` command. This works, for example, with all fonts that are defined to be scaleable (e.g., PostScript fonts) since they can be used at any size by NFSS.

However, packages like `pandora` won't work because the standard `.fd` files shipped with NFSS only contain small sizes. You can, of course, produce additional sizes and change the `.fd` files accordingly so that they would be useable for slides as well.

4 Invisible text and color separation

In the original SLiTeX it was possible to produce invisible text using the `\invisible` command, so that one was able to put several slides on top of each other (with each slides showing additional details, etc.). It was also possible to produce 'color' slides. This was done by producing individual slides one for each color and placing them on top of each other.

[1] Therefore you should compare the new class with old SLiTeX styles in case you have local slide classes to see what you have to change in order to use them with LaTeX 2ε.

The availability of color printers and the `color` package make color separation obsolete, so it has been removed. Although the `color` has also made `\invisible` obsolete, the command is retained in the LaTeX 2ε implementation, but there are a few restrictions. Invisible fonts are implemented as special shapes where the shape names are build by prefixing the normal shape name with an uppercase `I`. For example, the 'normal invisible shape' would be `In`. When LaTeX is requested to typeset invisible it will thus change the current shape attribute in this manner. To make this work it is necessary that the resulting font shape group is defined. If not, the normal font substitution mechanism of LaTeX 2ε will change the attribute until it finds a usable font shape group with the result that the text may become visible.

As long as you use the standard fonts for slides this is not a problem because all the visible font shape groups have invisible counterparts. However, if you decide on using special fonts, e.g., PostScript fonts, your `\DeclareFontShape` settings may not contain invisible font shape groups and thus you may be unable to use these features without adding additional `\DeclareFontShape` commands to your `.fd` files or the preamble of your document.

5 The Implementation

> **Warning:** The implementation is still very experimental and may change internally very much. It currently basically consists of a slightly modified copy of `slides.sty` (which then forms `slides.cls`) followed by a slightly changed copy of `slitex.tex`. Documentation is practically non-existing. Everybody is invited to help changing this!

The code is divided into two parts, we first implement the class related functions and declarations and then define low level stuff that is necessary within every class. By placing such commands into a separate file it will be possible to share it with other slide classes.

5.1 The class code

At this point we input the redefinitions that are necessary for SliTeX.

```
1 ⟨*class⟩
2 \input{slides.def}
```

Now we are ready for setting up the font tables. As usual, we first look for a local configuration file `sfonts.cfg`. If there isn't one, we fall back to the default one (`sfonts.def`).

```
3 \InputIfFileExists{sfonts.cfg}
4           {\typeout{**************************************^^J%
5                     *^^J%
6                     * Local config file sfonts.cfg used^^J%
7                     *^^J%
8                     **************************************}}%
9           {\input{sfonts.def}}
```

6 Declaration of Options

We declare a few options as illegal.

6.1 Setting Paper Sizes

The variables \paperwidth and \paperheight should reflect the physical paper size after trimming. For desk printer output this is usually the real paper size since there is no post-processing. Classes for real book production will probably add other paper sizes and additionally the production of crop marks for trimming.

```
\DeclareOption{a4paper}
   {\setlength\paperheight {297mm}%
    \setlength\paperwidth  {210mm}}
\DeclareOption{a5paper}
   {\setlength\paperheight {210mm}%
    \setlength\paperwidth  {148mm}}
\DeclareOption{b5paper}
   {\setlength\paperheight {250mm}%
    \setlength\paperwidth  {176mm}}
\DeclareOption{letterpaper}
   {\setlength\paperheight {11in}%
    \setlength\paperwidth  {8.5in}}
\DeclareOption{legalpaper}
   {\setlength\paperheight {14in}%
    \setlength\paperwidth  {8.5in}}
\DeclareOption{executivepaper}
   {\setlength\paperheight {10.5in}%
    \setlength\paperwidth  {7.25in}}
```

The option landscape switches the values of \paperheight and \paperwidth, assuming the dimensions were given for portrait paper.

```
\DeclareOption{landscape}
   {\setlength\@tempdima   {\paperheight}%
    \setlength\paperheight {\paperwidth}%
    \setlength\paperwidth  {\@tempdima}}
```

6.2 The clock option

The option clock prints the time at the bottom of each note. We also define here the commands and counters used to keep track of time.

```
\newif\if@clock \@clockfalse
\DeclareOption{clock}{\@clocktrue
  \AtEndDocument{\typeout{\@arabic\c@minutes\space minutes}}
}%
\newcounter{minutes}%
\newcounter{seconds}%
\newcommand*{\settime}[1]{\setcounter{seconds}{0}\addtime{#1}}%
\newcommand*{\addtime}[1]{\addtocounter{seconds}{#1}%
  \setcounter{minutes}{\value{seconds}}%
  \global \divide \value{minutes} by 60\relax}

```

6.3 Two-side or one-side printing

Two-sided printing is not allowed, so don't declare an option. But it is necessary to initialize the switch.

```
\@twosidefalse
```

6.4 Draft option

If the user requests draft we show any overfull boxes. We could probably add some more interesting stuff to this option.

```
\DeclareOption{draft}{\setlength\overfullrule{5pt}}
\DeclareOption{final}{\setlength\overfullrule{0pt}}
```

6.5 Titlepage option

The default is for a `\maketitle` command to make a new page.

```
\newif\if@titlepage
\@titlepagetrue
\DeclareOption{titlepage}{\@titlepagetrue}
\DeclareOption{notitlepage}{\@titlepagefalse}
```

6.6 Twocolumn printing

Two-column printing is again forbidden.

```
\DeclareOption{onecolumn}{}
\DeclareOption{twocolumn}{%
    \ClassWarning{slides}{No 'twocolumn' layout for slides}}
```

6.7 Equation numbering on the left

The option leqno can be used to get the equation numbers on the left side of the equation.

```
\DeclareOption{leqno}{\input{leqno.clo}}
```

6.8 Flush left displays

The option fleqn redefines the displayed math environments in such a way that they come out flush left, with an indentation of `\mathindent` from the prevailing left margin.

```
\DeclareOption{fleqn}{\input{fleqn.clo}}
```

7 Executing Options

Here we execute the default options to initialize certain variables.

```
\ExecuteOptions{letterpaper,final}
```

The `\ProcessOptions` command causes the execution of the code for every option FOO which is declared and for which the user typed the FOO option in his `\documentclass` command. For every option BAR he typed, which is not declared, the option is assumed to be a global option. All options will be passed as document options to any `\usepackage` command in the document preamble.

```
\ProcessOptions
```

8 Loading Packages

The standard class files do not load additional packages.

9 Document Layout

In this section we are finally dealing with the nasty typographical details.

9.1 Fonts

```
% FMi:
\def\rmdefault{lcmss}        % no roman
\def\sfdefault{lcmss}
\def\ttdefault{lcmtt}
\def\itdefault{sl}
\def\sldefault{sl}
\def\bfdefault{bx}
```

As `\fontshape` gets redefined we need to make sure that the default for `\upshape` is no longer `up` but again `n`.

```
\def\updefault{n}
```

Since the number of parameters to set are very large it seems reasonable to set up one command `\@setfontsize@parms` which will do the work for us.

LaTeX offers the user commands to change the size of the font, relative to the 'main' size. Each relative size changing command `\size` executes the command `\@setfontsize\size`⟨*font-size*⟩⟨*baselineskip*⟩ where:

⟨***font-size***⟩ The absolute size of the font to use from now on.

⟨***baselineskip***⟩ The normal value of `\baselineskip` for the size of the font selected. (The actual value will be `\baselinestretch` * ⟨*baselineskip*⟩.)

A number of commands, defined in the LaTeX kernel, shorten the following definitions and are used throughout. They are:

`\@vpt`	5	`\@vipt`	6	`\@viipt`	7
`\@viiipt`	8	`\@ixpt`	9	`\@xpt`	10
`\@xipt`	10.95	`\@xiipt`	12	`\@xivpt`	14.4

...

`\ifourteenpt` `\iseventeenpt` `\itwentypt` `\itwentyfourpt` `\itwentyninept` `\ithirtyfourpt` `\ifortyonept`

For SliTeX, however, these are not sufficient, and we therefore need to add a few extra, larger, sizes.

```
\def\ifourteenpt{13.82}
\def\iseventeenpt{16.59}
\def\itwentypt{19.907}
\def\itwentyfourpt{23.89}
\def\itwentyninept{28.66}
\def\ithirtyfourpt{34.4}
\def\ifortyonept{41.28}
```

`\@setfontsize@parms` This routine is used in SLiTeX to interface font size setting it is modeled after the settings I found in `slides.sty`, so it probably needs an update. But any class is free to redefine it, as it is used only as an abbreviation. It's syntax is:

`\@setfontsize@parms`
⟨*lineskip*⟩
⟨*parskip*⟩
⟨*abovedisplayskip*⟩

⟨*belowdisplayskip*⟩
⟨*abovedisplayshortskip*⟩
⟨*belowdisplayshortskip*⟩
⟨*strut ht*⟩ ⟨*strut dp*⟩ (without pt)

For NFSS1 a similar style existed which did run both with a SLiTEX with old font selection or with NFSS1. But when no separate format is made this doesn't make much sense. So the following note is history and would only be true if all NFSS stuff would be removed from the file and placed into the format.

> Note: To interface the old `sfonts.tex` the ⟨*size*⟩ must be hidden in commands denoting the size by its name prefixed with 'i', i.e. 20pt size is called `\itwentypt` at this point. The NFSS interface will define those sizes to expand to the internal size, e.g. 20 but for the old sfonts the command name, e.g. `\itwentypt`, will be used to construct the name `\twentypt` etc.
>
> This is a crude interface to the old `sfonts.tex`. It will be a bit slower than the old one because it must define `\@tiny` etc. every time a size changes.

If classes are set up that are only for use with NFSS then the second argument may be an ordinary font size!

```
\def\@setfontsize@parms#1#2#3#4#5#6#7#8{%
   \lineskip #1\relax%
   \parskip #2\relax
   \abovedisplayskip #3\relax
   \belowdisplayskip #4\relax
   \abovedisplayshortskip #5\relax
   \belowdisplayshortskip #6\relax
%
```

I don't see a reason why the `\strutbox` has a dim different from `\baselineskip` but we will leave it for the moment

```
  \setbox\strutbox=\hbox{\vrule \@height#7\p@\@depth#8\p@\@width\z@}%
  \baselineskip\baselinestretch\baselineskip
  \normalbaselineskip\baselineskip}
```

Setting size relations for math scripts:

```
\DeclareMathSizes{13.82}{13.82}{10}{7}
\DeclareMathSizes{16.59}{16.59}{12}{7}
\DeclareMathSizes{19.907}{19.907}{16.59}{13.82}
\DeclareMathSizes{23.89}{23.89}{19.907}{16.59}
\DeclareMathSizes{28.66}{28.66}{23.89}{19.907}
\DeclareMathSizes{34.4}{34.4}{28.66}{23.89}
\DeclareMathSizes{41.28}{41.28}{34.4}{28.66}
```

`\normalsize`

```
\def\normalsize{%
      \@setfontsize\normalsize\itwentypt{28\p@ plus3\p@ minus4\p@}%
%            {20}{30\p@ plus3\p@ minus3\p@}%  made a bit shorter
      \@setfontsize@parms
            {2pt}%
            {30\p@ plus18\p@ minus9\p@}%
            {15\p@ plus3\p@ minus3\p@}%
            {10\p@ plus3\p@ minus3\p@}%
```

```
            {10\p@ plus3\p@}
            \abovedisplayshortskip
            {17}{7}}
```

We initially choose the normalsize font.

```
\normalsize
```

`\small`

```
\def\small{\@setfontsize\small\iseventeenpt{19\p@ plus3\p@ minus\p@}%
           \@setfontsize@parms
            {2\p@}%
            {15\p@ plus15\p@ minus7\p@}%
            {12\p@ plus3\p@ minus3\p@}%
            {9\p@ plus3\p@ minus3\p@}%
            {6\p@ plus3\p@}%
            \abovedisplayshortskip
            {13.5}{5.6}}
```

`\footnotesize` `\scriptsize`

```
\let\footnotesize=\small
\let\scriptsize=\small
```

`\tiny`

```
\def\tiny{\@setfontsize\tiny\ifourteenpt{16\p@ plus2\p@ minus\p@}%
          \@setfontsize@parms
            {2pt}%
            {14\p@ plus3\p@ minus10\p@}%
            {11\p@ plus3\p@ minus10\p@}%
            \abovedisplayskip
            {8\p@ plus3\p@ minus5\p@}%
            {\z@ plus3\p@}%
            {10}{4}}
```

Actually copying the code above would be better because this would correct the error message. Maybe one should remove the first argument of `\set@font@size@parms`.

`\large` `\Large` `\LARGE` `\huge` `\Huge`

```
\def\large{\@setfontsize\large\itwentyfourpt{42\p@ plus8\p@ minus5\p@}%
           \@setfontsize@parms
            {2\p@}%
            {40\p@ plus20\p@ minus4\p@}%
            {20\p@ plus8\p@ minus3\p@}%
            \abovedisplayskip
            {10\p@ plus5\p@}%
            \abovedisplayshortskip
            {20}{8.5}}

\def\Large{\@setfontsize\Large\itwentyninept{48\p@ plus10\p@ minus6\p@}%
           \@setfontsize@parms
            {2\p@}%
            {48\p@ plus30\p@ minus6\p@}%
            {24\p@ plus10\p@ minus6\p@}%
```

```
            \abovedisplayskip
            {12\p@ plus8\p@}%
            \abovedisplayshortskip
            {27}{11}}

\def\LARGE{\@setfontsize\LARGE\ithirtyfourpt{52\p@ plus10\p@ minus6\p@}%
           \@setfontsize@parms
            {2\p@}%
            {52\p@ plus30\p@ minus6\p@}%
            {24\p@ plus10\p@ minus6\p@}%
            \abovedisplayskip
            {12\p@ plus8\p@}%
            \abovedisplayshortskip
            {27}{11}}

\def\huge{\@setfontsize\huge\ifortyonept{60\p@ plus10\p@ minus6\p@}%
           \@setfontsize@parms
            {2\p@}%
            {60\p@ plus30\p@ minus6\p@}%
            {24\p@ plus10\p@ minus6\p@}%
            \abovedisplayskip
            {12\p@ plus8\p@}%
            \abovedisplayshortskip
            {27}{11}}

\let\Huge\huge
```

9.2 Paragraphing

`\baselinestretch` This is used as a multiplier for `\baselineskip`. The default is to *not* stretch the baselines.

```
\renewcommand\baselinestretch{}
```

`\parindent` `\parindent` is the width of the paragraph indentation.

```
\setlength\parindent{\z@}
```

`\@lowpenalty` `\@medpenalty` `\@highpenalty` The commands `\nopagebreak` and `\nolinebreak` put in penalties to discourage these breaks at the point they are put in. They use `\@lowpenalty`, `\@medpenalty` or `\@highpenalty`, dependent on their argument.

```
\@lowpenalty   51
\@medpenalty  151
\@highpenalty 301
```

`\clubpenalty` `\widowpenalty` These penalties are use to discourage club and widow lines. Because we use their default values we only show them here, commented out.

```
% \clubpenalty  150
% \widowpenalty 150
```

`\displaywidowpenalty` `\predisplaypenalty` `\postdisplaypenalty` Discourage (but not so much) widows in front of a math display and forbid breaking directly in front of a display. Allow break after a display without a penalty. Again the default values are used, therefore we only show them here.

```
% \displaywidowpenalty 50
% \predisplaypenalty   10000
```

```
% \postdisplaypenalty  0
```

\interlinepenalty Allow the breaking of a page in the middle of a paragraph.

```
% \interlinepenalty 0
```

\brokenpenalty We allow the breaking of a page after a hyphenated line.

```
% \brokenpenalty 0
```

9.3 Page Layout

All margin dimensions are measured from a point one inch from the top and lefthand side of the page.

9.3.1 Vertical spacing

\headheight \headsep \topskip The \headheight is the height of the box that will contain the running head. The \headsep is the distance between the bottom of the running head and the top of the text. \topskip is the \baselineskip for the first line on a page.

```
\setlength\headheight{14\p@}
\setlength\headsep   {15\p@}
\setlength\topskip   {30\p@}
```

\footskip The distance from the baseline of the box which contains the running footer to the baseline of last line of text is controlled by the \footskip. Bottom of page:

```
\setlength\footskip{25\p@}   %
```

\maxdepth \@maxdepth The TEX primitive register \maxdepth has a function that is similar to that of \topskip. The register \@maxdepth should always contain a copy of \maxdepth. In both plain TEX and LATEX 2.09 \maxdepth had a fixed value of 4pt; in native LATEX2e mode we let the value depend on the typesize. We set it so that \maxdepth + \topskip = typesize ×1.5. As it happens, in these classes \topskip is equal to the typesize, therefor we set \maxdepth to half the value of \topskip.

```
\if@compatibility
  \setlength\maxdepth{4\p@}
\else
  \setlength\maxdepth{.5\topskip}
\fi
\setlength\@maxdepth\maxdepth
```

9.3.2 The dimension of text

\textwidth When we are in compatibility mode we have to make sure that the dimensions of the printed area are not different from what the user was used to see.

```
\if@compatibility
  \setlength\textwidth{460\p@}
```

When we are not in compatibility mode we can set some of the dimensions differently, taking into account the paper size for instance.

```
\else
```

First, we calculate the maximum textwidth, which depends on the papersize. Then we calculate the approximate length of 65 characters, which should be the maximum length of a line of text. The calculated values are stored in `\@tempdima` and `\@tempdimb`.

```
\setlength\@tempdima{\paperwidth}
\addtolength\@tempdima{-2in}
\setbox\@tempboxa\hbox{\rmfamily im}
\setlength\@tempdimb{.5\wd\@tempboxa}
\setlength\@tempdimb{65\@tempdimb}
```

Now we can set the `\textwidth`, depending on whether we will be setting one or two columns.

The text should not be wider than the minimum of the paperwidth (minus 2 inches for the margins) and the maximum length of a line as defined by the number of characters.

```
\ifdim\@tempdima>\@tempdimb\relax
  \setlength\textwidth{\@tempdimb}
\else
  \setlength\textwidth{\@tempdima}
\fi
\fi
```

Here we modify the width of the text a little to be a whole number of points.

```
\@settopoint\textwidth
```

`\columnwidth`
`\columnsep`
`\columnseprule`

```
\columnwidth \textwidth
\columnsep 10pt
\columnseprule \z@
```

`\textheight` Now that we have computed the width of the text, we have to take care of the height. The `\textheight` is the height of text (including footnotes and figures, excluding running head and foot).

First make sure that the compatibility mode gets the same dimensions as we had with LaTeX2.09. The number of lines was calculated as the floor of the old `\textheight` minus `\topskip`, divided by `\baselineskip` for `\normalsize`. The old value of `\textheight` was 528pt.

```
\if@compatibility
  \setlength\textheight{600\p@}
```

Again we compute this, depending on the papersize and depending on the baselineskip that is used, in order to have a whole number of lines on the page.

```
\else
  \setlength\@tempdima{\paperheight}
```

We leave at least a 1 inch margin on the top and the bottom of the page.

```
  \addtolength\@tempdima{-2in}
```

We also have to leave room for the running headers and footers.

```
  \addtolength\@tempdima{-1in}
```

Then we divide the result by the current `\baselineskip` and store this in the count register `\@tempcnta`, which then contains the number of lines that fit on this page.

```
  \divide\@tempdima\baselineskip
  \@tempcnta=\@tempdima
```

From this we can calculate the height of the text.

```
  \setlength\textheight{\@tempcnta\baselineskip}
\fi
```

The first line on the page has a height of `\topskip`.

```
\advance\textheight by \topskip
```

9.3.3 Margins

`\oddsidemargin` `\evensidemargin` `\marginparwidth`

First we give the values for the compatibility mode.

Values for two-sided printing:

```
\if@compatibility
   \setlength\oddsidemargin  {17\p@}
   \setlength\evensidemargin {17\p@}
   \setlength\marginparwidth {20\p@}
\else
```

When we are not in compatibility mode we can take the dimensions of the selected paper into account.

We center the text on the page, by calculating the difference between `textwidth` and `\paperwidth−2in`. Half of that difference is then used for the margin. The amount of space that can be used for marginal notes is at least 0.8 inch, to which we add any 'leftover' space.

```
  \setlength\@tempdima        {\paperwidth}
  \addtolength\@tempdima      {-2in}
  \addtolength\@tempdima      {-\textwidth}
  \setlength\oddsidemargin    {.5\@tempdima}
  \setlength\marginparwidth   {.8in}
  \addtolength\marginparwidth {.5\@tempdima}
```

The `\evensidemargin` can now be computed from the values set above.

```
\setlength\evensidemargin  {\paperwidth}
\addtolength\evensidemargin{-2in}
\addtolength\evensidemargin{-\textwidth}
\addtolength\evensidemargin{-\oddsidemargin}
\fi
```

`\marginparsep` `\marginparpush`

The horizontal space between the main text and marginal notes is determined by `\marginparsep`, the minimum vertical separation between two marginal notes is controlled by `\marginparpush`.

```
\setlength\marginparsep {5\p@}
\setlength\marginparpush{5\p@}
```

`\topmargin`

The `\topmargin` is the distance between the top of 'the printable area' –which is 1 inch below the top of the paper– and the top of the box which contains the running head.

It can now be computed from the values set above.

```
\if@compatibility
  \setlength\topmargin{-10pt}
\else
  \setlength\topmargin{\paperheight}
```

```
  \addtolength\topmargin{-2in}
  \addtolength\topmargin{-\headheight}
  \addtolength\topmargin{-\headsep}
  \addtolength\topmargin{-\textheight}
  \addtolength\topmargin{-\footskip}     % this might be wrong!
```

By changing the factor in the next line the complete page can be shifted vertically.

```
  \addtolength\topmargin{-.5\topmargin}
\fi
\@settopoint\topmargin
```

9.3.4 Footnotes

\footnotesep `\footnotesep` is the height of the strut placed at the beginning of every footnote. It equals the height of a normal `\footnotesize` strut in this class, thus no extra space occurs between footnotes.

```
\setlength\footnotesep{20\p@}
```

\footins `\skip\footins` is the space between the last line of the main text and the top of the first footnote.

```
\setlength{\skip\footins}{10\p@ \@plus 2\p@ \@minus 4\p@}
```

9.4 Page Styles

The page style *foo* is defined by defining the command `\ps@foo`. This command should make only local definitions. There should be no stray spaces in the definition, since they could lead to mysterious extra spaces in the output (well, that's something that should be always avoided).

\@evenhead \@oddhead \@evenfoot \@oddfoot The `\ps@...` command defines the macros `\@oddhead`, `\@oddfoot`, `\@evenhead`, and `\@evenfoot` to define the running heads and feet—e.g., `\@oddhead` is the macro to produce the contents of the heading box for odd-numbered pages. It is called inside an `\hbox` of width `\textwidth`.

The page styles of slides is determined by the 'slide' page style, the slide environment executing a `\thispagestyle{slide}` command. The page styles of overlays and notes are similarly determined by 'overlay' and 'note' page styles. The command standard 'headings', 'plain' and 'empty' page styles work by redefining the 'slide', 'overlay', and 'note' styles.

\ps@headings

```
\if@compatibility
\def\ps@headings{%
\def\ps@slide{\def\@oddfoot{\@mainsize +\hfil\hb@xt@3em{\theslide
                                                        \hss}}%
 \def\@oddhead{\@mainsize +\hfil +}%
 \def\@evenfoot{\@mainsize +\hfil\hb@xt@3em{\theslide\hss}}%
 \def\@evenhead{\@mainsize +\hfil +}}

\def\ps@overlay{\def\@oddfoot{\@mainsize +\hfil\hb@xt@3em{\theoverlay
                                                         \hss}}%
 \def\@oddhead{\@mainsize +\hfil +}%
 \def\@evenfoot{\@mainsize +\hfil\hb@xt@3em{\theoverlay\hss}}%
```

```
 \def\@evenhead{\@mainsize +\hfil +}}
\def\ps@note{\def\@oddfoot{\@mainsize \hbox{}\hfil\thenote}%
 \def\@oddhead{}%
 \def\@evenfoot{\@mainsize \hbox{}\hfil\thenote}%
 \def\@evenhead{}}}
%
\else %%if@compatibility
%
\def\ps@headings{%
  \def\ps@slide{%
    \def\@oddfoot{\@mainsize \mbox{}\hfil\hb@xt@3em{\theslide\hss}}%
    \def\@oddhead{}%
    \def\@evenfoot{\@mainsize \mbox{}\hfil\hb@xt@3em{\theslide\hss}}%
    \def\@evenhead{}}

  \def\ps@overlay{%
    \def\@oddfoot{\@mainsize \mbox{}\hfil\hb@xt@3em{\theoverlay\hss}}%
    \def\@oddhead{}%
    \def\@evenfoot{\@mainsize \mbox{}\hfil\hb@xt@3em{\theoverlay\hss}}%
    \def\@evenhead{}}

  \def\ps@note{%
    \def\@oddfoot{%
      \@mainsize
      \if@clock
        \fbox{\large \@arabic\c@minutes\space min}%
      \else
        \null
      \fi
      \hfil\thenote}%
    \def\@oddhead{}%
    \def\@evenfoot{%
      \@mainsize
      \if@clock
        \fbox{\large \@arabic\c@minutes\space min}%
      \else
        \null
      \fi
      \hfil\thenote}%
    \def\@evenhead{}}}
\fi  %% if@compatibility
```

\ps@plain

```
\def\ps@plain{\def\ps@slide{%
 \def\@oddfoot{\@mainsize \mbox{}\hfil\hb@xt@3em{\theslide\hss}}%
 \def\@oddhead{}%
 \def\@evenfoot{\@mainsize \mbox{}\hfil\hb@xt@3em{\theslide\hss}}%
 \def\@evenhead{}}
\def\ps@overlay{\def\@oddfoot{\@mainsize
   \mbox{}\hfil\hb@xt@3em{\theoverlay\hss}}%
 \def\@oddhead{}%
 \def\@evenfoot{\@mainsize \mbox{}\hfil\hb@xt@3em{\theoverlay\hss}}%
 \def\@evenhead{}}
\def\ps@note{\def\@oddfoot{\@mainsize \hbox{}\hfil\thenote}%
```

```
\def\@oddhead{}%
\def\@evenfoot{\@mainsize \hbox{}\hfil\thenote}%
\def\@evenhead{}}}
```

\ps@empty

```
\def\ps@empty{%
\def\ps@slide{\def\@oddhead{}\def\@oddfoot{}%
\def\@evenhead{}\def\@evenfoot{}}%
\def\ps@overlay{\def\@oddhead{}\def\@oddfoot{}%
\def\@evenhead{}\def\@evenfoot{}}%
\def\ps@note{\def\@oddhead{}\def\@oddfoot{}%
\def\@evenhead{}\def\@evenfoot{}}}
```

Default definition the 'slide', 'overlay', and 'note' page styles.

```
\ps@headings
```

Set ordinary page style to 'empty'

```
\let\@oddhead\@empty\let\@oddfoot\@empty
\let\@evenhead\@empty\let\@evenfoot\@empty
```

9.5 Providing math *versions*

LaTeX provides two *versions*. We call them normal and bold, respectively. SLiTeX does not have a bold version. But we treat the invisible characters as a version. The only thing we have to take care of is to ensure that we have exactly the same fonts in both versions available.

```
\DeclareMathVersion{invisible}
```

Now we define the basic *math groups* used by LaTeX. Later on, in packages some other *math groups*, e.g., the AMS symbol fonts, will be defined.

As a default I used serif fonts for mathgroup 0 to get things like `\log` look right.

```
\SetSymbolFont{operators}{normal}
                  {OT1}{lcmss}{m}{n}

\SetSymbolFont{letters}{normal}
                  {OML}{lcmm}{m}{it}
\SetSymbolFont{symbols}{normal}
                  {OMS}{lcmsy}{m}{n}
\SetSymbolFont{largesymbols}{normal}
                  {OMX}{lcmex}{m}{n}

\SetSymbolFont{operators}{invisible}
                  {OT1}{lcmss}{m}{In}
\SetSymbolFont{letters}{invisible}
                  {OML}{lcmm}{m}{Iit}
\SetSymbolFont{symbols}{invisible}
                  {OMS}{lcmsy}{m}{In}
\SetSymbolFont{largesymbols}{invisible}
                   {OMX}{lcmex}{m}{In}


\def\@mainsize{\visible\tiny}
```

9.6 Environments

`titlepage` (*env.*) This environment starts a new page, with pagestyle *empty* and sets the page counter to 0.

```
\newenvironment{titlepage}
    {\newpage
     \thispagestyle{empty}%
     \setcounter{page}{\z@}}
    {\newpage}
```

9.6.1 General List Parameters

The following commands are used to set the default values for the list environment's parameters. See the LaTeX manual for an explanation of the meaning of the parameters.

`\leftmargini`
`\leftmarginii`
`\leftmarginiii`
`\leftmarginiv`
`\leftmarginv`
`\leftmarginvi`

```
\setlength\leftmargini   {38\p@}
\setlength\leftmarginii  {30\p@}
\setlength\leftmarginiii {20\p@}
\setlength\leftmarginiv  {15\p@}
\setlength\leftmarginv   {15\p@}
\setlength\leftmarginvi  {10\p@}
```

`\@listi`
`\@listii`
`\@listiii`
`\@listiv`
`\@listv`
`\@listvi`

These commands set the values of `\leftmargin`, `\parsep`, `\topsep`, and `\itemsep` for the various levels of lists. It is even necessary to initialize `\leftmargin` in `\@listi`, i.e. for a level one list, as a list environment may appear inside a `trivlist`, for example inside a `theorem` environment.

```
\def\@listi{\leftmargin\leftmargini
            \parsep .5\parskip
            \topsep \parsep
            \itemsep\parskip
            \partopsep \z@}

\def\@listii{\leftmargin\leftmarginii
             \labelwidth\leftmarginii
             \advance\labelwidth-\labelsep
             \parsep .5\parskip
             \topsep \parsep
             \itemsep\parskip}
\def\@listiii{\leftmargin\leftmarginiii
              \labelwidth\leftmarginiii
              \advance\labelwidth-\labelsep}
\def\@listiv{\leftmargin\leftmarginiv
             \labelwidth\leftmarginiv
             \advance\labelwidth-\labelsep}
\def\@listv{\leftmargin\leftmarginv
            \labelwidth\leftmarginv
            \advance\labelwidth-\labelsep}
\def\@listvi{\leftmargin\leftmarginvi
             \labelwidth\leftmarginvi
             \advance\labelwidth-\labelsep}
```

Here we initialize `\leftmargin` and `\labelwidth`.

```
\leftmargin\leftmargini
\labelwidth\leftmargini\advance\labelwidth-\labelsep
```

9.6.2 Paragraph-formatting environments

`verse` (*env.*) Inside a `verse` environment, `\\` ends a line, and line continuations are indented further. A blank line makes new paragraph with `\parskip` space.

```
\newenvironment{verse}{\let\\=\@centercr
                       \list{}{\itemsep       \z@
                               \itemindent    -15\p@
                               \listparindent \itemindent
                               \rightmargin   \leftmargin
                               \advance\leftmargin 15\p@}%
                       \item[]}
                      {\endlist}
```

`quotation` (*env.*) The `quotation` environment fills lines, indents paragraphs.

```
\newenvironment{quotation}{\list{}{\listparindent 20\p@
                                   \itemindent\listparindent
                                   \rightmargin\leftmargin}%
                           \item[]}
                          {\endlist}
```

`quote` (*env.*) The `quote` environment is the same as the `quotation` environment, except that there is no paragraph indentation.

```
\newenvironment{quote}{\list{}{\rightmargin\leftmargin}\item[]}
                      {\endlist}
```

9.6.3 List-making environments

`description` (*env.*) The description environment is defined here – while the itemize and enumerate environments are defined in the LaTeX format.

```
\newenvironment{description}{\list{}{\labelwidth\z@
                                      \itemindent-\leftmargin
                                      \let\makelabel\descriptionlabel}}
                            {\endlist}
```

`\descriptionlabel` To change the formatting of the label, you must redefine `\descriptionlabel`.

```
\newcommand*{\descriptionlabel}[1]{\hspace\labelsep
                                   \normalfont\bfseries #1}

```

9.6.4 Enumerate

The enumerate environment uses four counters: *enumi*, *enumii*, *enumiii* and *enumiv*, where *enumN* controls the numbering of the Nth level enumeration.

`\theenumi` `\theenumii` `\theenumiii` `\theenumiv` The counters are already defined in the LaTeX format, but their representation is changed here.

```
\renewcommand\theenumi{\@arabic\c@enumi}
\renewcommand\theenumii{\@alph\c@enumii}
```

```
\renewcommand\theenumiii{\@roman\c@enumiii}
\renewcommand\theenumiv{\@Alph\c@enumiv}
```

\labelenumi
\labelenumii
\labelenumiii
\labelenumiv

The label for each item is generated by the four commands \labelenumi ... \labelenumiv.

```
\newcommand\labelenumi{\theenumi.}
\newcommand\labelenumii{(\theenumii)}
\newcommand\labelenumiii{\theenumiii.}
\newcommand\labelenumiv{\theenumiv.}
```

\p@enumii
\p@enumiii
\p@enumiv

The expansion of \p@enumN\theenumN defines the output of a \ref command when referencing an item of the Nth level of an enumerated list.

```
\renewcommand\p@enumii{\theenumi}
\renewcommand\p@enumiii{\theenumi(\theenumii)}
\renewcommand\p@enumiv{\p@enumiii\theenumiii}
```

9.6.5 Itemize

\labelitemi
\labelitemii
\labelitemiii
\labelitemiv

Itemization is controlled by four commands: \labelitemi, \labelitemii, \labelitemiii, and \labelitemiv, which define the labels of the various itemization levels.

```
\newcommand\labelitemi{$\m@th\bullet$}
\newcommand\labelitemii{\normalfont\bfseries \textendash}
\newcommand\labelitemiii{$\m@th\ast$}
\newcommand\labelitemiv{$\m@th\cdot$}
```

9.7 Setting parameters for existing environments

9.7.1 Array and tabular

\arraycolsep

The columns in an array environment are separated by 2\arraycolsep. Array and tabular environment parameters

```
\setlength\arraycolsep{8\p@}
```

\tabcolsep

The columns in an tabular environment are separated by 2\tabcolsep.

```
\setlength\tabcolsep{10\p@}
```

\arrayrulewidth

The width of rules in the array and tabular environments is given by the length parameter\arrayrulewidth.

```
\setlength\arrayrulewidth{.6\p@}
```

\doublerulesep

The space between adjacent rules in the array and tabular environments is given by \doublerulesep.

```
\setlength\doublerulesep{3\p@}
```

9.7.2 Tabbing

\tabbingsep

This controls the space that the \' command puts in. (See LaTeX manual for an explanation.)

```
\labelsep 10pt
\setlength\tabbingsep{\labelsep}
```

9.7.3 Minipage

\@minipagerestore The macro \@minipagerestore is called upon entry to a minipage environment to set up things that are to be handled differently inside a minipage environment. In the current styles, it does nothing.

\@mpfootins Minipages have their own footnotes; \skip\@mpfootins plays same rôle for footnotes in a minipage as \skip\footins does for ordinary footnotes.

```
\skip\@mpfootins = \skip\footins
```

9.7.4 Framed boxes

\fboxsep The space left by \fbox and \framebox between the box and the text in it.

\fboxrule The width of the rules in the box made by \fbox and \framebox.

```
\setlength\fboxsep{5\p@}
\setlength\fboxrule{.6\p@}
```

\theequation The equation number will be typeset as arabic numerals.

```
\def\theequation{\@arabic\c@equation}
```

\jot \jot is the extra space added between lines of an eqnarray environment. The default value is used.

```
% \setlength\jot{3pt}
```

\@eqnnum The macro \@eqnnum defines how equation numbers are to appear in equations. Again the default is used.

```
% \def\@eqnnum{(\theequation)}
```

9.8 Font changing

Here we supply the declarative font changing commands that were common in LaTeX version 2.09 and earlier. These commands work in text mode *and* in math mode. They are provided for compatibility, but one should start using the \text... and \math... commands instead. These commands are redefined using \DeclareOldFontCommand, a command with three arguments: the user command to be defined, LaTeX commands to execute in text mode and LaTeX commands to execute in math mode.

\rm The commands to change the family. When in compatibility mode we select the
\tt 'default' font first, to get LaTeX2.09 behaviour.
\sf

```
\DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}
```

\bf The command to change to the bold series. One should use \mdseries to explicitly switch back to medium series.

```
\DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}
```

\sl \it \sc And the commands to change the shape of the font. The slanted and small caps shapes are not available by default as math alphabets, so those changes do nothing in math mode. One should use \upshape to explicitly change back to the upright shape.

```
\DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
\DeclareOldFontCommand{\sl}{\normalfont\slshape}{\relax}
\DeclareOldFontCommand{\sc}{\normalfont\scshape}{\relax}
```

\cal \mit The commands \cal and \mit should only be used in math mode, outside math mode they have no effect. Currently the New Font Selection Scheme defines these commands to generate warning messages. Therefore we have to define them 'by hand'.

```
\DeclareRobustCommand*{\cal}{\@fontswitch{\relax}{\mathcal}}
\DeclareRobustCommand*{\mit}{\@fontswitch{\relax}{\mathnormal}}
```

9.9 Footnotes

\footnoterule Usually, footnotes are separated from the main body of the text by a small rule. This rule is drawn by the macro \footnoterule. We have to make sure that the rule takes no vertical space (see plain.tex). The resulting rule will appear on all color layers, so it's best not to draw a rule.

```
\renewcommand\footnoterule{}
% \let \footnoterule = \relax
```

\c@footnote \thefootnote Footnotes are numbered within slides, overlays, and notes and numbered with ∗, †, etc.

```
% \newcounter{footnote}
\def\thefootnote{\fnsymbol{footnote}}
\@addtoreset{footnote}{slide}
\@addtoreset{footnote}{overlay}
\@addtoreset{footnote}{note}
```

\@makefntext The footnote mechanism of LaTeX calls the macro \@makefntext to produce the actual footnote. The macro gets the text of the footnote as its argument and should use \@makefnmark to produce the mark of the footnote. The macro \@makefntext is called when effectively inside a \parbox of width \columnwidth (i.e., with \hsize = \columnwidth).

An example of what can be achieved is given by the following piece of TeX code.

```
\long\def\@makefntext#1{%
   \@setpar{\@@par
            \@tempdima = \hsize
            \advance\@tempdima-10pt
            \parshape \@ne 10pt \@tempdima}%
   \par
   \parindent 1em\noindent
   \hbox to \z@{\hss\@makefnmark}#1}
```

The effect of this definition is that all lines of the footnote are indented by 10pt, while the first line of a new paragraph is indented by 1em. To change these

dimensions, just substitute the desired value for '10pt' (in both places) or '1em'. The mark is flushright against the footnote.

In these document classes we use a simpler macro, in which the footnote text is set like an ordinary text paragraph, with no indentation except on the first line of a paragraph, and the first line of the footnote. Thus, all the macro must do is set `\parindent` to the appropriate value for succeeding paragraphs and put the proper indentation before the mark.

```
\long\def\@makefntext#1{
    \noindent
    \hangindent 10\p@
    \hb@xt@10\p@{\hss\@makefnmark}#1}
```

`\@makefnmark` The footnote markers that are printed in the text to point to the footnotes should be produced by the macro `\@makefnmark`. We use the default definition for it.

```
%\def\@makefnmark{\hbox{$^{\@thefnmark}\m@th$}}
```

9.10 The title

The commands `\title`, `\author`, and `\date` are already defined, so here we just define `\maketitle`.

```
\newcommand\maketitle{{\centering {\Large \@title \par}%
     \@author \par \@date\par}%
     \if@titlepage \break \fi}
```

10 Initialization

10.1 Date

`\today` This macro uses the TEX primitives `\month`, `\day` and `\year` to provide the date of the LATEX-run.

```
\newcommand\today{\ifcase\month\or
  January\or February\or March\or April\or May\or June\or
  July\or August\or September\or October\or November\or December\fi
  \space\number\day, \number\year}
```

Default initializations

```
\pagenumbering{arabic}
\onecolumn
⟨/class⟩
```

10.2 Basic code

The code below is basically a copy of `slitex.tex` with some changes.

Global changes so far:

10.2.1 Hacks for slide macros

```
⟨*cmd⟩
\message{hacks,}

```

```
\outer\def\newifG#1{\count@\escapechar \escapechar\m@ne
  \expandafter\expandafter\expandafter
   \edef\@ifG#1{true}{\global\let\noexpand#1\noexpand\iftrue}%
  \expandafter\expandafter\expandafter
   \edef\@ifG#1{false}{\global\let\noexpand#1\noexpand\iffalse}%
  \@ifG#1{false}\escapechar\count@} % the condition starts out false
\def\@ifG#1#2{\csname\expandafter\ifG@\string#1#2\endcsname}
{\uccode`1=`i \uccode`2=`f \uccode`3=`G \uppercase{\gdef\ifG@123{G}}}
 % `ifG' is required

\def\@gobbletoend#1{\def\@argend{#1}\@ggobtoend}

\long\def\@ggobtoend#1\end#2{\fi\def\reserved@a{#2}%
\ifx\reserved@a\@argend\else\@ggobtoend\fi}
```

FMi: I don't see any reason for this command since `\fi` is hidden anyway in the replacement text `\def\@xfi{\fi}`

```
 \message{slides,}
```

10.2.2 Slide macros

Switches:

`@bw`	true if making black and white slides
`@visible`	true if visible output to be produced.
`@makingslides`	true if making a slide/overlay/note

```
\newif\if@bw
\newif\if@visible
\newif\if@onlyslidesw \@onlyslideswfalse
\newif\if@onlynotesw  \@onlynoteswfalse
\newif\if@makingslides
```

FMi: `\newifG` replaces `\gdef\@slidesw{T}` stuff

```
\newifG\ifG@slidesw
```

Counters

slide	slide number
overlay	overlay number for a slide
note	note number for a slide

```
\countdef\c@slide=0 \c@slide=0
\def\cl@slide{}
\countdef\c@overlay=1 \c@overlay=0
\def\cl@overlay{}
\countdef\c@note=2 \c@note=0
\def\cl@note{}
```

Add these counters explicitly to the 'ckpt list' so that the `\include` mechanism works.

```
\g@addto@macro\cl@@ckpt{\@elt{slide}\@elt{overlay}\@elt{note}}
\@addtoreset{overlay}{slide}
\@addtoreset{note}{slide}
```

Redefine page counter to some other number. The page counter will always be zero except when putting out an extra page for a slide, note or overlay.

```
\newcount\c@page
\@definecounter{page}
\@addtoreset{page}{slide}
```

```
\@addtoreset{page}{note}
\@addtoreset{page}{overlay}

\def\theslide{\@arabic\c@slide}
\def\theoverlay{\theslide-\@alph\c@overlay}
\def\thenote{\theslide-\@arabic\c@note}
```

```
 \@setlimits \LIST \LOW \HIGH

    Assumes that \LIST = RANGE1,RANGE2,...,RANGEn  (n>0)
    Where RANGEi = j or j-k.

    Then \@setlimits  globally sets
        (i) \LIST := RANGE2, ... , RANGEn
       (ii) \LOW  := p
      (iii) \HIGH := q
   where either RANGE1 = p-q   or  RANGE1 = p  and  q=p.
```

```
\def\@sl@getargs#1-#2-#3\relax#4#5{\xdef#4{#1}\xdef#5{#2}}
\def\@sl@ccdr#1,#2\relax#3#4{\xdef#3{#1-#1-}\xdef#4{#2}}

\def\@setlimits #1#2#3{\expandafter\@sl@ccdr#1\relax\@sl@gtmp #1%
\expandafter\@sl@getargs\@sl@gtmp\relax#2#3}
```

```
 \onlyslides{LIST} ::=
  BEGIN
    @onlyslidesw := true
    \@doglslidelist :=G LIST,999999,999999
   if @onlynotesw = true
     else @onlynotesw := true
          \@doglnotelist :=G LIST,999999,999999
   fi
   message: Only Slides LIST
  END
```

```
\def\onlyslides#1{\@onlyslideswtrue
   \gdef\@doglslidelist{#1,999999,999999}%
   \if@onlynotesw \else
      \@onlynoteswtrue\gdef\@doglnotelist{999999,999999}\fi
   \typeout{Only Slides #1}}
```

```
 \onlynotes{LIST} ::=
  BEGIN
    @onlynotesw := true
    \@doglnotelist :=G LIST,999999,999999
   if @onlyslidesw = true
     else \@onlyslidesw := true
          \@doglslidelist{999999,999999}
   fi
   message: Only Notes LIST
  END
```

```
\def\onlynotes#1{\@onlynoteswtrue
   \gdef\@doglnotelist{#1,999999,999999}%
   \if@onlyslidesw \else
      \@onlyslideswtrue\gdef\@doglslidelist{999999,999999}\fi
   \typeout{Only Notes #1}}
```

```
\setupcounters ::=    (similar to old \blackandwhite #1  ::= )
   \newpage
   page counter := 0
   @bw := T
   @visible := T
   if @onlyslidesw = true
     then  \@doslidelist := \@doglslidelist
           \@setlimits\@doslidelist\@doslidelow\@doslidehigh
   fi
   if @onlynotesw = true
     then  \@donotelist := \@doglnotelist
           \@setlimits\@donotelist\@donotelow\@donotehigh
   fi
   \normalsize    % Note, this sets font to \rmfamily , which sets
                    % \@currfont to \rmfamily
   counter slidenumber := 0
   counter note        := 0
   counter overlay     := 0
   @makingslides       := F  %% \blackandwhite: @makingslides  := T
                             %%                 input #1
                             %%                 @makingslides  := F
```

```
\if@compatibility
% In compatibility mode, need to define \verb+\blackandwhite+,
% \verb+\colors+, \verb+\colorslides+, etc.
\def\blackandwhite#1{\newpage\setcounter{page}{0}\@bwtrue\@visibletrue
\if@onlyslidesw \xdef\@doslidelist{\@doglslidelist}%
\@setlimits\@doslidelist\@doslidelow\@doslidehigh\fi
\if@onlynotesw \xdef\@donotelist{\@doglnotelist}%
\@setlimits\@donotelist\@donotelow\@donotehigh\fi
\normalsize\setcounter{slide}{0}\setcounter{overlay}{0}%
\setcounter{note}{0}\@makingslidestrue\input #1\@makingslidesfalse}
```

```
\colors{COLORS} ::=
 for \@colortemp := COLORS
    do \csname \@colortemp \endcsname == \@color{\@colortemp} od
 if \@colorlist = empty
    then \@colorlist := COLORS
    else \@colorlist := \@colorlist , COLORS
 fi
```

```
\def\colors#1{\@for\@colortemp:=#1\do{\expandafter
  \xdef\csname\@colortemp\endcsname{\noexpand\@color{\@colortemp}}}\ifx
  \@colorlist\@empty \gdef\@colorlist{#1}%
    \else \xdef\@colorlist{\@colorlist,#1}\fi}

\def\@colorlist{}
```

```
\colorslides{FILE} ::=
   \newpage
   page counter := 0
   @bw := F
   for \@currcolor := \@colorlist
     do  @visible := T
         if @onlyslidesw = true
```

```
          then  \@doslidelist := \@doglslidelist
                \@setlimits\@doslidelist\@doslidelow\@doslidehigh
        fi
        if @onlynotesw = true
          then  \@donotelist := \@doglnotelist
                \@setlimits\@donotelist\@donotelow\@donotehigh
        fi
        \normalsize
        counter slide := 0
        counter overlay := 0
        counter note    := 0
        type message
        generate color layer output page
        @makingslides := T
        input #1
        @makingslides := F
    od
```

```
\def\colorslides#1{\newpage\setcounter{page}{0}\@bwfalse
\@for\@currcolor:=\@colorlist\do
{\@visibletrue
\if@onlyslidesw \xdef\@doslidelist{\@doglslidelist}%
\@setlimits\@doslidelist\@doslidelow\@doslidehigh\fi
\if@onlynotesw \xdef\@donotelist{\@doglnotelist}%
\@setlimits\@donotelist\@donotelow\@donotehigh\fi
\normalsize\setcounter{slide}{0}\setcounter{overlay}{0}%
\setcounter{note}{0}\typeout{color \@currcolor}%
\newpage
\begin{huge}%
\begin{center}%
COLOR LAYER\\[.75in]%
\@currcolor
\end{center}%
\end{huge}%
\newpage
\@makingslidestrue
\input #1
\@makingslidesfalse}}
%
\else  %% if@compatibility
%
\def\setupcounters{\newpage\setcounter{page}{0}\@bwtrue\@visibletrue
\if@onlyslidesw \xdef\@doslidelist{\@doglslidelist}%
\@setlimits\@doslidelist\@doslidelow\@doslidehigh\fi
\if@onlynotesw \xdef\@donotelist{\@doglnotelist}%
\@setlimits\@donotelist\@donotelow\@donotehigh\fi
\normalsize\setcounter{slide}{0}\setcounter{overlay}{0}%
\setcounter{note}{0}\@makingslidesfalse}

\AtBeginDocument{\setupcounters}
\fi %% if@compatibility
```

```
\slide COLORS ::=
 BEGIN
\changes{v2.3}{1994/03/16}{Moved \cs{newpage} up front, here and in
```

```
   \cs{note} and \cs{overlay}}
  \par\break
  \stepcounter{slide}
  \setcounter{page}{0}                    % in case of non-slide pages
  \@slidesw :=G T
  if @onlyslidesw = true                  % set \@slidesw = T iff
    then                                  % page to be output
      while \c@slide > \@doslidehigh
         do  \@setlimits\@doslidelist\@doslidelow\@doslidehigh  od
      if \c@slide < \@doslidelow
        then \@slidesw := F
      fi
  fi
  if \@slidesw = T
     then \@slidesw :=G F
          \begingroup
             if @bw = true
               then  \@slidesw :=G T
               else \@color{COLORS}
                    \if@visible then \@slidesw :=G T \fi
             fi
           \endgroup
 fi
 if \@slidesw = T
   then @makingslides  := T
        \thispagestyle{slide}
   else \end{slide}
         \@gobbletoend{slide}
 fi
END

\endslide ::=
 BEGIN
   \par\break
 END
```

```
\if@compatibility
\def\slide#1{\stepcounter{slide}\G@slideswtrue\if@onlyslidesw
\@whilenum \c@slide >\@doslidehigh\relax
\do{\@setlimits\@doslidelist\@doslidelow\@doslidehigh}\ifnum
\c@slide <\@doslidelow\relax\G@slideswfalse\fi\fi
\ifG@slidesw
  \G@slideswfalse
% FMi this is only a hack at the moment to get things running.
%  \begingroup
  \if@bw\G@slideswtrue\else
    \@color{#1}\if@visible \G@slideswtrue \fi
  \fi
%  \endgroup
\fi
\ifG@slidesw \newpage\thispagestyle{slide}%
```

This will set up the last color specified in the argument to `\slide` as the current color. If only back and white slides are prepared `\last@color` will be empty and effectively `\relax` will be generated (hopefully).

We need to reset to a default font at the beginning of a slide. (not done yet).

```
\csname \last@color \endcsname
\else\end{slide}\@gobbletoend{slide}\fi}
%
\else  %% if@compatibility
%
\def\slide{\par\break
\stepcounter{slide}\setcounter{page}{0}\G@slideswtrue\if@onlyslidesw
\@whilenum \c@slide >\@doslidehigh\relax
\do{\@setlimits\@doslidelist\@doslidelow\@doslidehigh}\ifnum
\c@slide <\@doslidelow\relax\G@slideswfalse\fi\fi
\ifG@slidesw
  \G@slideswfalse
% FMi this is only a hack at the moment to get things running.
%  \begingroup
  \if@bw\G@slideswtrue\else
    \if@visible \G@slideswtrue \fi
  \fi
%  \endgroup
\fi
\ifG@slidesw \@makingslidestrue\thispagestyle{slide}%
```

This will set up the last color specified in the argument to `\slide` as the current color. If only back and white slides are prepared `\last@color` will be empty and effectively `\relax` will be generated (hopefully).

We need to reset to a default font at the beginning of a slide. (not done yet).

```
\csname \last@color \endcsname
\else\end{slide}\@gobbletoend{slide}\fi}
\fi %% if@compatibility

\let\last@color\@empty

\def\endslide{\par\break}
```

```
\overlay COLORS ::=
 BEGIN
  \par\break
  \stepcounter{overlay}
  \setcounter{page}{0}                    % in case of non-slide pages
  \@slidesw :=G T
  if @onlyslidesw = T                     % set \@slidesw = T iff
    then                                  % page to be output
      if \c@slide < \@doslidelow
        then \@slidesw :=G F
      fi
 fi
 if \@slidesw = T
   \@slidesw :=G F
   \begingroup
     if @bw = true
         then  \@slidesw :=G T
         else  \@color{COLORS}
               \if@visible then \@slidesw :=G T \fi
```

```
     fi
   \endgroup
 fi
 if \@slidesw = T
    then @makingslides  := T
         \thispagestyle{overlay}
    else \end{overlay}
         \@gobbletoend{overlay}
 fi
END

\endoverlay ::=
 BEGIN
   \par\break
 END
```

```
\if@compatibility
\def\overlay#1{\stepcounter{overlay}\G@slideswtrue%
\if@onlyslidesw\ifnum \c@slide <\@doslidelow\relax
\G@slideswfalse\fi\fi
\ifG@slidesw \G@slideswfalse\begingroup\if@bw\G@slideswtrue%
\else\@color{#1}\if@visible \G@slideswtrue\fi\fi\endgroup\fi
\ifG@slidesw \newpage\thispagestyle{overlay}%
\else\end{overlay}\@gobbletoend{overlay}\fi}
%
\else %%if@compatibility
%
\def\overlay{\par\break
  \stepcounter{overlay}%
  \setcounter{page}{0}%
  \G@slideswtrue%
  \if@onlyslidesw\ifnum \c@slide <\@doslidelow\relax
    \G@slideswfalse\fi\fi
  \ifG@slidesw \G@slideswfalse
    \begingroup\if@bw\G@slideswtrue%
                \else\if@visible \G@slideswtrue\fi\fi
    \endgroup\fi
  \ifG@slidesw \@makingslidestrue\thispagestyle{overlay}%
  \else\end{overlay}\@gobbletoend{overlay}\fi}
\fi %%if@compatibility

\def\endoverlay{\par\break}
```

```
\note ::=
 BEGIN
  \par\break
  \stepcounter{note}
  \setcounter{page}{0}                    % in case of non-slide pages
  if @bw = T
    then
      \@slidesw :=G T
      if @onlynotesw = true                  % set \@notesw = T iff
        then                                 % page to be output
          while \c@slide > \@donotehigh
             do  \@setlimits\@donotelist\@donotelow\@donotehigh  od
```

```
          if \c@slide < \@donotelow
            then \@slidesw :=G F
          fi
       fi
    else \@slidesw :=G F
 fi
 if \@slidesw = T
    then @makingslides   := T
         \thispagestyle{note}
    else \end{note}
         \@gobbletoend{note}
 fi
END

\endnote ::=
 BEGIN
   \par\break
 END
```

```
\if@compatibility
\def\note{\stepcounter{note}%
   \if@bw
      \G@slideswtrue
      \if@onlynotesw\@whilenum \c@slide >\@donotehigh\relax
      \do{\@setlimits\@donotelist\@donotelow\@donotehigh}\ifnum
        \c@slide <\@donotelow\relax \G@slideswfalse\fi\fi
      \else\G@slideswfalse\fi
      \ifG@slidesw \newpage\thispagestyle{note}\else
      \end{note}\@gobbletoend{note}\fi}
%
\else %%if@compatibility
%
\def\note{\par\break\stepcounter{note}\setcounter{page}{0}%
   \if@bw
      \G@slideswtrue
      \if@onlynotesw\@whilenum \c@slide >\@donotehigh\relax
      \do{\@setlimits\@donotelist\@donotelow\@donotehigh}\ifnum
        \c@slide <\@donotelow\relax \G@slideswfalse\fi\fi
      \else\G@slideswfalse\fi
      \ifG@slidesw \@makingslidestrue\thispagestyle{note}\else
      \end{note}\@gobbletoend{note}\fi}
\fi %%if@compatibility

\def\endnote{\par\break}
```

```
\@color{COLORS} ::=
 BEGIN
  if math mode
    then type warning
  fi
  if @bw
    then \visible
    else \invisible
        for \last@color := COLORS
         do if \last@color = \@currcolor
```

```
              then \visible
            fi
          od
   fi
   \ignorespaces
 END
```

FMi: `\last@color` will be used in `\slide` to set up first color if no color is given. I suppose that this is much too complicated. `\else\@tempswafalse` would produce the same effect I imagine.

```
\def\@color#1{\@mmodetest
 {\if@bw \@tempswatrue \else \@tempswafalse
   \@for \reserved@a :=#1\do{\ifx\reserved@a\@currcolor\@tempswatrue\fi
                          \let\last@color\reserved@a}\fi
  \if@tempswa \visible \else \invisible \fi
  \ignorespaces}}

\def\@mmodetest#1{\ifmmode\ClassWarning{slides}{Color-changing command
       in math mode has been ignored}\else #1\fi}

\def\invisible{\@mmodetest
  {\if@visible
     \@visiblefalse
     \fontshape\f@shape\selectfont
     \mathversion{invisible}%
   \fi
   \ignorespaces}}

\def\visible{\@mmodetest
  {\if@visible
   \else
     \@visibletrue
```

Here is the LaTeX 2_ε interface hidden. We use a trick to provide ourselves with a sort of additional attribute without making the current mechanism even larger. The trick is that we denote invisible by putting an uppercase `I` in front of the shape name for invisible shapes and remove it again if we want to become visible.

```
     \fontshape{\expandafter\@gobble\f@shape}\selectfont
     \mathversion{normal}%
   \fi
   \ignorespaces}}

\def\fontshape#1{\edef\f@shape{\if@visible \else I\fi #1}}
```

10.3 Macros for font handling

We let `\familydefault` point at `\sfdefault`, to make it easier to use the document class slides with packages that set up other fonts.

```
\renewcommand{\familydefault}{\sfdefault}
```

The `latexsym` package, which is needed to be able to access the LaTeX symbol fonts (lasy), sets things up so that for sizes larger then 10 point magnifications of `lasy10` are used. For slides we want to use magnifications of `lasy8`, so we set up the lasy family here to prevent LaTeX from loading `Ulasy.fd`.

```
\DeclareFontFamily{U}{lasy}{}{}
\DeclareFontShape{U}{lasy}{m}{n}{%
      <12><13.82><16.59><19.907><23.89><28.66><34.4><41.28>lasy8
}{}
\DeclareFontShape{U}{lasy}{m}{In}{%
      <13.82><16.59><19.907><23.89><28.66><34.4><41.28>ilasy8
}{}
\message{picture,}
```

10.3.1 Modifications to the picture environment

Below are the new definitions of the picture-drawing macros required for SLiTeX. Only those commands that actually draw something must be changed so that they do not produce any output when the `@visible` switch is false.

```
\def\line(#1,#2)#3{\if@visible\@xarg #1\relax \@yarg #2\relax
\@linelen #3\unitlength
\ifnum\@xarg =\z@ \@vline
  \else \ifnum\@yarg =\z@ \@hline \else \@sline\fi
\fi\fi}

\def\vector(#1,#2)#3{\if@visible\@xarg #1\relax \@yarg #2\relax
\@linelen #3\unitlength
\ifnum\@xarg =\z@ \@vvector
  \else \ifnum\@yarg =\z@ \@hvector \else \@svector\fi
\fi\fi}

\def\dashbox#1(#2,#3){%
\leavevmode\if@visible\hb@xt@\z@{\baselineskip \z@
\lineskip \z@
\@dashdim #2\unitlength
\@dashcnt \@dashdim \advance\@dashcnt 200
\@dashdim #1\unitlength\divide\@dashcnt \@dashdim
\ifodd\@dashcnt\@dashdim\z@
\advance\@dashcnt \@ne \divide\@dashcnt \tw@
\else \divide\@dashdim \tw@ \divide\@dashcnt \tw@
\advance\@dashcnt \m@ne
\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth
\@width \@dashdim}\put(0,0){\copy\@dashbox}%
\put(0,#3){\copy\@dashbox}%
\put(#2,0){\hskip-\@dashdim\copy\@dashbox}%
\put(#2,#3){\hskip-\@dashdim\box\@dashbox}%
\multiply\@dashdim \thr@@
\fi
\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth
\@width #1\unitlength\hskip #1\unitlength}\@tempcnta\z@
\put(0,0){\hskip\@dashdim \@whilenum \@tempcnta <\@dashcnt
\do{\copy\@dashbox\advance\@tempcnta \@ne }}\@tempcnta\z@
\put(0,#3){\hskip\@dashdim \@whilenum \@tempcnta <\@dashcnt
\do{\copy\@dashbox\advance\@tempcnta \@ne }}%
\@dashdim #3\unitlength
\@dashcnt=\@dashdim \advance\@dashcnt 200
\@dashdim #1\unitlength\divide\@dashcnt \@dashdim
\ifodd\@dashcnt \@dashdim=\z@
```

```
\advance\@dashcnt \@ne \divide\@dashcnt \tw@
\else
\divide\@dashdim \tw@ \divide\@dashcnt \tw@
\advance\@dashcnt \m@ne
\setbox\@dashbox\hbox{\hskip -\@halfwidth
\vrule \@width \@wholewidth
\@height \@dashdim}\put(0,0){\copy\@dashbox}%
\put(#2,0){\copy\@dashbox}%
\put(0,#3){\lower\@dashdim\copy\@dashbox}%
\put(#2,#3){\lower\@dashdim\copy\@dashbox}%
\multiply\@dashdim \thr@@
\fi
\setbox\@dashbox\hbox{\vrule \@width \@wholewidth
\@height #1\unitlength}\@tempcnta\z@
\put(0,0){\hskip -\@halfwidth \vbox{\@whilenum \@tempcnta <\@dashcnt
\do{\vskip #1\unitlength\copy\@dashbox\advance\@tempcnta \@ne }%
\vskip\@dashdim}}\@tempcnta\z@
\put(#2,0){\hskip -\@halfwidth \vbox{\@whilenum \@tempcnta <\@dashcnt
\relax\do{\vskip #1\unitlength\copy\@dashbox\advance\@tempcnta \@ne }%
\vskip\@dashdim}}\fi\@makepicbox(#2,#3)}
```

(re)declare these booleans as they not defined in old format (or with latexrelease package)

```
\newif\if@ovvline \@ovvlinetrue
\newif\if@ovhline \@ovhlinetrue

\def\@oval(#1,#2)[#3]{\if@visible\begingroup \boxmaxdepth \maxdimen
  \@ovttrue \@ovbtrue \@ovltrue \@ovrtrue

  \@ovvlinefalse \@ovhlinefalse

  \@tfor\reserved@a :=#3\do
     {\csname @ov\reserved@a false\endcsname}%
  \@ovxx#1\unitlength \@ovyy #2\unitlength

  \@tempdimb \ifdim \@ovyy >\@ovxx \@ovxx \@ovvlinetrue
  \else \@ovyy \ifdim \@ovyy =\@ovxx \else \@ovhlinetrue \fi\fi

  \advance \@tempdimb -2\p@
  \@getcirc \@tempdimb
  \@ovro \ht\@tempboxa \@ovri \dp\@tempboxa
  \@ovdx\@ovxx \advance\@ovdx -\@tempdima \divide\@ovdx \tw@
  \@ovdy\@ovyy \advance\@ovdy -\@tempdima \divide\@ovdy \tw@

  \ifdim \@ovdx >\z@ \@ovhlinetrue \fi
  \ifdim \@ovdy >\z@ \@ovvlinetrue \fi

  \@circlefnt \setbox\@tempboxa
  \hbox{\if@ovr \@ovvert32\kern -\@tempdima \fi
  \if@ovl \kern \@ovxx \@ovvert01\kern -\@tempdima \kern -\@ovxx \fi
  \if@ovt \@ovhorz \kern -\@ovxx \fi
  \if@ovb \raise \@ovyy \@ovhorz \fi}\advance\@ovdx\@ovro
  \advance\@ovdy\@ovro \ht\@tempboxa\z@ \dp\@tempboxa\z@
  \@put{-\@ovdx}{-\@ovdy}{\box\@tempboxa}%
  \endgroup\fi}

\def\@circle#1{\if@visible \begingroup \boxmaxdepth \maxdimen
   \@tempdimb #1\unitlength
```

```
   \ifdim \@tempdimb >15.5\p@\relax \@getcirc\@tempdimb
      \@ovro\ht\@tempboxa
     \setbox\@tempboxa\hbox{\@circlefnt
      \advance\@tempcnta\tw@ \char \@tempcnta
      \advance\@tempcnta\m@ne \char \@tempcnta \kern -2\@tempdima
      \advance\@tempcnta\tw@
      \raise \@tempdima \hbox{\char\@tempcnta}\raise \@tempdima
        \box\@tempboxa}\ht\@tempboxa\z@ \dp\@tempboxa\z@
      \@put{-\@ovro}{-\@ovro}{\box\@tempboxa}%
   \else  \@circ\@tempdimb{96}\fi\endgroup\fi}

\def\@dot#1{%
  \if@visible\@tempdimb #1\unitlength \@circ\@tempdimb{112}\fi}

\def\@frameb@x#1{%
  \@tempdima\fboxrule
  \advance\@tempdima\fboxsep
  \advance\@tempdima\dp\@tempboxa
  \leavevmode
  \hbox{%
    \lower\@tempdima\hbox{%
      \vbox{%
        \if@visible\hrule\@height\else\vskip\fi\fboxrule
        \hbox{%
          \if@visible\vrule\@width\else\hskip\fi\fboxrule
          #1%
          \vbox{%
            \vskip\fboxsep
            \box\@tempboxa
            \vskip\fboxsep}%
          #1%
          \if@visible\vrule\@width\else\hskip\fi\fboxrule}%
        \if@visible\hrule\@height\else\vskip\fi\fboxrule}}}}

\long\def\frame#1{\if@visible\leavevmode
\vbox{\vskip-\@halfwidth\hrule \@height\@halfwidth \@depth \@halfwidth
  \vskip-\@halfwidth\hbox{\hskip-\@halfwidth \vrule \@width\@wholewidth
  \hskip-\@halfwidth #1\hskip-\@halfwidth \vrule \@width \@wholewidth
  \hskip -\@halfwidth}\vskip -\@halfwidth\hrule \@height \@halfwidth
  \@depth \@halfwidth\vskip -\@halfwidth}\else #1\fi}

\message{mods,}
```

10.3.2 Other modifications to TEX and LATEX commands

\rule

```
\def\@rule[#1]#2#3{\@tempdima#3\advance\@tempdima #1\leavevmode
 \hbox{\if@visible\vrule
  \@width#2 \@height\@tempdima \@depth-#1\else
\vrule \@width \z@ \@height\@tempdima \@depth-#1\vrule
 \@width#2 \@height\z@\fi}}

% \_  (Added 10 Nov 86)
```

```
\def\_{\leavevmode \kern.06em \if@visible\vbox{\hrule \@width.3em}\else
   \vbox{\hrule \@height \z@ \@width.3em}\vbox{\hrule \@width \z@}\fi}
```

```
\overline, \underline, \frac and \sqrt

\@mathbox{STYLE}{BOX}{MTEXT} : Called in math mode, typesets MTEXT and
  stores result in BOX, using STYLE.

\@bphant{BOX}    : Creates a phantom with dimensions BOX.
\@vbphant{BOX}   : Creates a phantom with ht of BOX and zero width.
\@hbphant{BOX}   : Creates a phantom with width of BOX
                   and zero ht & dp.
\@hvsmash{STYLE}{MTEXT} : Creates a copy of MTEXT with zero height and
                          width in STYLE.
```

```
\def\@mathbox#1#2#3{\setbox#2\hbox{$\m@th#1{#3}$}}

\def\@vbphantom#1{\setbox\tw@\null \ht\tw@\ht #1\dp\tw@\dp #1%
   \box\tw@}

\def\@bphantom#1{\setbox\tw@\null
   \wd\tw@\wd #1\ht\tw@\ht #1\dp\tw@\dp #1%
   \box\tw@}

\def\@hbphantom#1{\setbox\tw@\null \wd\tw@\wd #1\ht\tw@\z@ \dp\tw@\z@
   \box\tw@}

\def\@hvsmash#1#2{\@mathbox#1\z@{#2}\ht\z@\z@ \dp\z@\z@ \wd\z@\z@
   \box\z@}

\def\underline#1{\relax\ifmmode
  \@xunderline{#1}\else $\m@th\@xunderline{\hbox{#1}}$\relax\fi}

\def\@xunderline#1{\mathchoice{\@xyunderline\displaystyle{#1}}%
   {\@xyunderline
    \textstyle{#1}}{\@xyunderline\scriptstyle{#1}}{\@xyunderline
      \scriptscriptstyle{#1}}}

\def\@xyunderline#1#2{%
   \@mathbox#1\@smashboxa{#2}\@hvsmash#1{\copy\@smashboxa}%
   \if@visible \@hvsmash#1{\@@underline{\@bphantom\@smashboxa}}\fi
  \@mathbox#1\@smashboxb{\@@underline{\box\@smashboxa}}%
   \@bphantom\@smashboxb}

\let\@@overline=\overline

\def\overline#1{\mathchoice{\@xoverline\displaystyle{#1}}{\@xoverline
    \textstyle{#1}}{\@xoverline\scriptstyle{#1}}{\@xoverline
      \scriptscriptstyle{#1}}}

\def\@xoverline#1#2{%
   \@mathbox#1\@smashboxa{#2}\@hvsmash#1{\copy\@smashboxa}%
   \if@visible \@hvsmash#1{\@@overline{\@bphantom\@smashboxa}}\fi
   \@mathbox#1\@smashboxb{\@@overline{\box\@smashboxa}}%
   \@bphantom\@smashboxb}
```

```
\@frac {STYLE}{DENOMSTYLE}{NUM}{DEN}{FONTSIZE} :
  Creates \frac{NUM}{DENOM}
  in style STYLE with NUM and DENOM in style DENOMSTYLE
  FONTSIZE should be \textfont \scriptfont or \scriptscriptfont
```

Added a group around the first argument of `\frac` to prevent changes (for example font changes) to modify the contents of the second argument.

```
\def\frac#1#2{\mathchoice
   {\@frac\displaystyle\textstyle{#1}{#2}\textfont}{\@frac
         \textstyle\scriptstyle{#1}{#2}\textfont}{\@frac
         \scriptstyle\scriptscriptstyle{#1}{#2}\scriptfont}{\@frac
         \scriptscriptstyle\scriptscriptstyle{#1}{#2}\scriptscriptfont}}

\def\@frac#1#2#3#4#5{%
   \@mathbox#1\@smashboxc{{\begingroup#3\endgroup\over#4}}%
   \setbox\tw@\null
   \ht\tw@ \ht\@smashboxc
   \dp\tw@ \dp\@smashboxc
   \wd\tw@ \wd\@smashboxc
   \box\if@visible\@smashboxc\else\tw@\fi}

\def\r@@t#1#2{\setbox\z@\hbox{$\m@th#1\@xysqrt#1{#2}$}%
  \dimen@\ht\z@ \advance\dimen@-\dp\z@
  \mskip5mu\raise.6\dimen@\copy\rootbox \mskip-10mu\box\z@}

\def\sqrt{\@ifnextchar[{\@sqrt}{\@xsqrt}}
\def\@sqrt[#1]{\root #1\of}
\def\@xsqrt#1{\mathchoice{\@xysqrt\displaystyle{#1}}{\@xysqrt
     \textstyle{#1}}{\@xysqrt\scriptstyle{#1}}{\@xysqrt
    \scriptscriptstyle{#1}}}
\def\@xysqrt#1#2{\@mathbox#1\@smashboxa{#2}\if@visible
    \@hvsmash#1{\sqrtsign{\@bphantom\@smashboxa}}\fi
    \phantom{\sqrtsign{\@vbphantom\@smashboxa}}\box\@smashboxa}

\newbox\@smashboxa
\newbox\@smashboxb
\newbox\@smashboxc
```

array and tabular environments: changes to '`|`', `\hline`, `\cline`, and `\vline`, added 8 Jun 88

```
\def\@arrayrule{\if@visible\@addtopreamble{\hskip -.5\arrayrulewidth
   \vrule \@width \arrayrulewidth\hskip -.5\arrayrulewidth}\fi}

\def\cline#1{\if@visible\@cline#1\@nil\fi}

\def\hline{\noalign{\ifnum0=`}\fi
    \if@visible \hrule \@height \arrayrulewidth
      \else \hrule \@width \z@
    \fi
    \futurelet \reserved@a\@xhline}

\def\vline{\if@visible \vrule \@width \arrayrulewidth
            \else \vrule \@width \arrayrulewidth \@height \z@
           \@depth \z@ \fi}

\message{output,}
```

10.3.3 Changes to LaTeX output routine

```
 \@makecol ==
   BEGIN
% Following test added for slides to check if extra page
    if @makingslides = T
    then if \c@page > 0
            then if \c@note > 0
                    then type 'Note \thenote too long.'
                    else if \c@overlay > 0
                           then type 'Overlay \theoverlay too long.'
                           else type 'Slide \theslide too long'
    fi     fi     fi     fi
    ifvoid \insert\footins
       then  \@outputbox := \box255
       else  \@outputbox := \vbox {\unvbox255
                                   \vskip \skip\footins
                                   \footnoterule
                                   \unvbox\@footinsert
                                  }
   fi
   \@freelist :=G \@freelist * \@midlist
   \@midlist  :=G empty
   \@combinefloats
   \@outputbox := \vbox to \@colht{\boxmaxdepth := \maxdepth
                                    \vfil     %%\vfil added for slides
                                    \unvbox\@outputbox
                                    \vfil }   %%\vfil added for slides
   \maxdepth :=G \@maxdepth
   END
```

FMi simple hack to allow none centered slides Should be revised of course.

```
\let\@topfil\vfil

\def\@makecol{\if@makingslides\ifnum\c@page>\z@ \@extraslide\fi\fi
\ifvoid\footins \setbox\@outputbox\box\@cclv \let\@botfil\vfil
   \else\let\@botfil\relax\setbox\@outputbox
     \vbox{\unvbox\@cclv\vfil
           \vskip\skip\footins\footnoterule\unvbox\footins\vskip
            \z@ plus.1fil\relax}\fi
  \xdef\@freelist{\@freelist\@midlist}\gdef\@midlist{}\@combinefloats
     \setbox\@outputbox\vbox to\@colht{\boxmaxdepth\maxdepth
        \@topfil\unvbox\@outputbox\@botfil}\global\maxdepth\@maxdepth}

\def\@extraslide{\ifnum\c@note>\z@
    \ClassWarning{slides}{Note \thenote\space too long}\else
     \ifnum\c@overlay>\z@
        \ClassWarning{slides}{Overlay \theoverlay\space too long}\else
        \ClassWarning{slides}{Slide \theslide\space too long}\fi\fi}

\message{init}
```

10.3.4 Special SliTeX initializations

FMi why not allow for ref's ?

```
%   \nofiles

\@visibletrue
⟨/cmd⟩
```