Producing slides with \LaTeXe

Frank Mittelbach

2019/11/04

1 Introduction

With \LaTeXe 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 \texttt{slides} as the document class. This class is very similar to the \texttt{slides} style that came with SL\TeX, in fact it is basically a copy changed to work under \LaTeXe. Thus you have to say something like

\begin{verbatim}
\documentclass[...]{slides}
\end{verbatim}

and process this with \LaTeXe.

3 Fonts

Note, that that with NFSS you can easily produce slides with special fonts just by calling an appropriate style file (like \texttt{times}) in a \texttt{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 \texttt{pandora} won’t work because the standard \texttt{.fd} files shipped with NFSS only contain small sizes. You can, of course, produce additional sizes and change the \texttt{.fd} files accordingly so that they would be useable for slides as well.

4 Invisible text and color separation

In the original SL\TeX it was possible to produce invisible text using the \texttt{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.

\footnote{Therefore you should compare the new class with old SL\TeX styles in case you have local slide classes to see what you have to change in order to use them with \LaTeXe.}
The availability of color printers and the \texttt{color} package make color separation obsolete, so it has been removed. Although the \texttt{color} has also made \texttt{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 \texttt{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 \texttt{\DeclareFontShape} settings may not contain invisible font shape groups and thus you may be unable to use these features without adding additional \texttt{\DeclareFontShape} commands to your \texttt{.fd} files or the preamble of your document.

\section{The Implementation}

\textbf{Warning:} The implementation is still very experimental and may change internally very much. It currently basically consists of a slightly modified copy of \texttt{slides.sty} (which then forms \texttt{slides.cls}) followed by a slightly changed copy of \texttt{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 lowlevel 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.

\subsection{The class code}

At this point we input the redefinitions that are necessary for \LaTeX{}X.

\begin{verbatim}
\input{slides.def}
\end{verbatim}

Now we are ready for setting up the font tables. As usual, we first look for a local configuration file \texttt{sfons.cfg}. If there isn’t one, we fall back to the default one (\texttt{sfons.def}).

\begin{verbatim}
\InputIfFileExists{sfons.cfg}
  \typeout{**************************************^^J%
  * Local config file sfons.cfg used^^J%
  **************************************}}%
  \input{sfons.def}
\end{verbatim}

\section{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.

\begin{verbatim}
\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}}\
\end{verbatim}

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

\begin{verbatim}
\DeclareOption{landscape}{
  \setlength\@tempdima {\paperheight}\
  \setlength\paperheight {\paperwidth}\
  \setlength\paperwidth {\@tempdima}}\
\end{verbatim}

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.

\begin{verbatim}
\newif\if\clock\clockfalse\DeclareOption{clock}{\clocktrue\AtEndDocument{\typeout{\@arabic\c@minutes\space \text{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
\end{verbatim}

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.

\begin{verbatim}\@twosidefalse\end{verbatim}
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}

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

As \fontshape gets redefined we need to make sure that the default for \upshape is no longer up but again 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\langle font-size\rangle\langle baselineskip\rangle where:

\langle font-size\rangle The absolute size of the font to use from now on.

\langle baselineskip\rangle The normal value of \baselineskip for the size of the font selected. (The actual value will be \baselinestretch * \langle baselineskip\rangle.)

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

\begin{verbatim}
\@vpt 5 \@vipt 6 \@viipt 7 \@viipt 8 \@ixpt 9 \@xpt 10 \@xipt 10.95 \@xiipt 12 \@xivpt 14.4 ...
\end{verbatim}

For Sl\TeX{}, however, these are not sufficient, and we therefore need to add a few extra, larger, sizes.

\begin{verbatim}
\ifourteenpt \iseventeenpt \itwentypt \itwentyfourpt \itwentyninept \ithirtyfourpt \ifortyonept
\end{verbatim}

This routine is used in Sl\TeX{} to interface font size setting it is modeled after the settings I found in \texttt{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:

\begin{verbatim}
\@setfontsize@parms \langle lineskip\rangle \langle parskip\rangle \langle abovedisplayskip\rangle
\end{verbatim}
For NFSS1 a similar style existed which did run both with a SLT\TeX{} 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 sfons.tex the \langle size \rangle 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 sfons the command name, e.g. \itwentypt, will be used to construct the name \twentypt etc.

This is a crude interface to the old sfons.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!

\[
def\@setfontsize@parms#1#2#3#4#5#6#7#8{%
\lineskip #1\relax%
\parskip #2\relax
\abovedisplayskip #3\relax
\belowdisplayskip #4\relax
\abovedisplayshortskip #5\relax
\belowdisplayshortskip #6\relax
%
\i@setfontsize\@tiny{12}{10}{7}\
\i@setfontsize\@small{14}{12}{8}\
\i@setfontsize\@normalsize{16}{14}{9}\
\i@setfontsize\@large{18}{16}{10}\
\i@setfontsize\@Large{20}{18}{12}\
\i@setfontsize\@LARGE{22}{20}{14}\]

Setting size relations for math scripts:

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

I don’t see a reason why the \texttt{\strutbox} has a dim different from \texttt{\baselineskip} but we will leave it for the moment

\[
def\setbox{\strutbox=\hbox{\vrule \@height\#7\p@\@depth\#8\p@\@width\z@}}%\]

\baselineskip\baselinestretch\baselineskip
\normalbaselineskip\baselineskip

\[
def\normalsize{%
\i@setfontsize\normalsize{\itwentypt}{28\p@ plus3\p@ minus4\p@}{12}{10}{7}{8}{9}{10}\]

\@setfontsize\itwentypt{28\p@ plus3\p@ minus4\p@}{12}{10}{7}{8}{9}{10}
\@setfontsize\itwentypt{28\p@ plus3\p@ minus4\p@}{12}{10}{7}{8}{9}{10}
\@setfontsize\itwentypt{28\p@ plus3\p@ minus4\p@}{12}{10}{7}{8}{9}{10}
\@setfontsize\itwentypt{28\p@ plus3\p@ minus4\p@}{12}{10}{7}{8}{9}{10}
\@setfontsize\itwentypt{28\p@ plus3\p@ minus4\p@}{12}{10}{7}{8}{9}{10}
We initially choose the normalsize font.

\normalsize

\small

\footnotesize
\scriptsize
\tiny

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.
9.2 Paragraphing

\baselineskip This is used as a multiplier for \baselineskip. The default is to not stretch the baselines.
\parindent \parindent is the width of the paragraph indentation.
\@lowpenalty 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.
\@medpenalty
\@highpenalty
\clubpenalty These penalties are use to discourage club and widow lines. Because we use their default values we only show them here, commented out.
\widowpenalty
\displaywidowpenalty 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.
Allow the breaking of a page in the middle of a paragraph.

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

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

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. The topskip is the \baselineskip for the first line on a page.

\setlength{\headheight}{14\textwidth}
\setlength{\headsep}{15\textwidth}
\setlength{\topskip}{30\textwidth}

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\textwidth} %

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\ \LaTeX\ 2.09 \maxdepth had a fixed value of 4pt; in native \LaTeX\ mode we let the value depend on the typesize. We set it so that \maxdepth + \topskip = typesize \times 1.5. As it happens, in these classes \topskip is equal to the typesize, therefore we set \maxdepth to half the value of \topskip.

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

9.3.2 The dimension of text

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\textwidth}
\else
\setlength{\textwidth}{460\textwidth}
\fi

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

Now that we have computed the width of the text, we have to take care of the height. The \texttheight 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 \texttheight minus \topskip, divided by \baselineskip for \normalsize. The old value of \texttheight was 528pt.

\if@compatibility
\setlength\texttheight{600\p@}
\else
\setlength\@tempdima{\paperheight}
\addtolength\@tempdima{-2in}
\setlength\@tempdima{\paperheight}
\addtolength\@tempdima{-1in}
\fi

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

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

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.
From this we can calculate the height of the text.

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

\advance\textheight by \topskip

9.3.3 Margins

First we give the values for the compatibility mode.

\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{-.5\@tempdima}
\fi

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

\setlength\marginparpush{5\p@}
\setlength\marginparwidth{5\p@}

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

\addtolength\topmargin{-\topmargin}
\fi
\@settopoint\topmargin

9.3.4 Footnotes

\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\footnotesp{20\p@}

\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).

\ps@headings
\ps@overlay
The \ps@... command defines the macros \@oddhead, \@oddfoot, \@evenhead, \@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
\def\@evenhead{\@mainsize +\hfil +}
\def\@oddfoot{\@mainsize \hbox{}\hfil\thenote}
\def\@oddhead{}
\def\@evenfoot{\@mainsize \hbox{}\hfil\thenote}
\def\@evenhead{}}

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

%}
def\ps@plain{
\def\ps@slide{}
\def\@oddfoot{\@mainsize \hbox{}\hfil\hb@xt@3em{\theslide\hss}}
\def\@oddhead{}
\def\@evenfoot{\@mainsize \hbox{}\hfil\hb@xt@3em{\theslide\hss}}
\def\@evenhead{}}

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

%}
def\ps@note{\def\@oddfoot{\@mainsize \hbox{}\hfil\thenote}
\def\@oddhead{}}
% if@compatibility
% 
\def\ps@headings{%
\def\ps@note{%
%}
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{}}
Providing math versions

\LaTeX provides two versions. We call them normal and bold, respectively. \LaTeX{} 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.

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}{lcmsa}{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}{lcmsa}{m}{In}
\SetSymbolFont{letters}{invisible}{OML}{lcmm}{m}{Iit}
\SetSymbolFont{symbols}{invisible}{OMS}{lcmsy}{m}{In}
\SetSymbolFont{largesymbols}{invisible}{OMX}{lcmex}{m}{In}
9.6 Environments

\textbf{titlepage} This environment starts a new page, with pagestyle \texttt{empty} and sets the page counter to 0.

\begin{verbatim}
\newenvironment{titlepage}{
\newpage
\thispagestyle{empty}\
\setcounter{page}{\z@}}{\newpage}
\end{verbatim}

\subsection{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.

\begin{verbatim}
\leftmargini 38\p@  
\leftmarginii 30\p@  
\leftmarginiii 20\p@  
\leftmarginiv 15\p@  
\leftmarginv 15\p@  
\leftmarginvi 10\p@  
\setlength{\leftmargini} 38\p@  
\setlength{\leftmarginii} 30\p@  
\setlength{\leftmarginiii} 20\p@  
\setlength{\leftmarginiv} 15\p@  
\setlength{\leftmarginv} 15\p@  
\setlength{\leftmarginvi} 10\p@  
\def\@listi{
\leftmargin\leftmargini  
\parsep .5\parskip  
\topsep \parsep  
\itemsep \parskip}
\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}
\end{verbatim}

\verb|\@listi| 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 \verb|\@listi|, i.e. for a level one list, as a list environment may appear inside a \texttt{trivlist}, for example inside a \texttt{theorem} environment.
Here we initialize \texttt{\leftmargin} and \texttt{\labelwidth}.

\begin{verbatim}
9.6.2 Paragraph-formatting environments

verse Inside a \texttt{verse} environment, \texttt{\textbackslash} ends a line, and line continuations are indented further. A blank line makes new paragraph with \texttt{\parskip} space.

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

\end{verbatim}

quotation The \texttt{quotation} environment fills lines, indents paragraphs.

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

quote The \texttt{quote} environment is the same as the \texttt{quotation} environment, except that there is no paragraph indentation.

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

9.6.3 List-making environments

description The description environment is defined here – while the itemize and enumerate environments are defined in \texttt{latex.dtx}.

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

\texttt{\descriptionlabel} To change the formatting of the label, you must redefine \texttt{\descriptionlabel}.

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

9.6.4 Enumerate

The enumerate environment uses four counters: \texttt{enumi}, \texttt{enumii}, \texttt{enumiii} and \texttt{enumiv}, where \texttt{enumN} controls the numbering of the \texttt{N}th level enumeration.

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

\begin{verbatim}
\renewcommand{\theenumiii}{\@arabic\c@enumiii}
\renewcommand{\theenumiv}{\@alph\c@enumiv}
```
```
\end{verbatim}

\end{verbatim}
The label for each item is generated by the four commands \labelenumi ... \labelenumiv.

\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.

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.

9.7 Setting parameters for existing environments

9.7.1 Array and tabular

\arraycolsep
\tabcolsep

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

\setlength\arraycolsep{8\p@}
\setlength\tabcolsep{10\p@}

\arrayrulewidth
\doublerulesep

The width of rules in the array and tabular environments is given by the length parameter\arrayrulewidth.
\setlength\arrayrulewidth{.6\p@}
\setlength\doublerulesep{3\p@}

9.7.2 Tabbing

\tabbingsep

This controls the space that the \' command puts in. (See \LaTeX\ manual for an explanation.)
\setlength\tabbingsep{10pt}
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.

425 \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.

426 \setlength\fboxsep{5\p@}
427 \setlength\fboxrule{.6\p@}

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

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

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

429 % \setlength\jot{3pt}

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

430 % \def\@eqnnum{({\theequation})}

9.8 Font changing

Here we supply the declarative font changing commands that were common in \TeX{} 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, \TeX{} commands to execute in text mode and \TeX{} commands to execute in math mode.

\rm The commands to change the family. When in compatibility mode we select the ‘default’ font first, to get \TeX{}2.09 behaviour.

431 \DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
432 \DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
433 \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.

434 \DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}
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 \textit{upshape} to explicitly change back to the upright shape.

435 \DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
436 \DeclareOldFontCommand{\sl}{\normalfont\slshape}{\relax}
437 \DeclareOldFontCommand{\sc}{\normalfont\scshape}{\relax}

\textit{cal} The commands \textit{cal} and \textit{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’.

438 \DeclareRobustCommand*{\cal}{\@fontswitch{\relax}{\mathcal}}
439 \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.

440 \renewcommand\footnoterule{}
441 % \let \footnoterule = \relax

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

442 % \newcounter{footnote}
443 \def\thefootnote{\fnsymbol{footnote}}
444 %\@addtoreset{footnote}{slide}
445 %\@addtoreset{footnote}{overlay}
446 %\@addtoreset{footnote}{note}

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

An example of what can be achieved is given by the following piece of \LaTeX{} code.

\long\def\@makefntext{11\%
 \@setpar{@par
 \@tempdimb = \hsize
 \advance\@tempdimb - 10pt
 \parshape \one 10pt \@tempdimb}
 \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.
\long\def\@makefnmark{$^\@thefnmark\m@th$}

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 Initialisation
10.1 Date
\today This macro uses the \TeX{} primitives \month, \day and \year to provide the date of the \TeX{}-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 \texttt{slitex.tex} with some changes.
Global changes so far:

10.2.1 Hacks for slide macros
\message{hacks,}
\outer\def\newifG#1{%\count@\escapechar \escapechar\m@ne
 \expandafter\expandafter\expandafter
 \def\@ifG#1{true}{\global\let\noexpand#1\noexpand\iftrue}%
 \expandafter\expandafter\expandafter
 \def\@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\@bw
\newif\@visible
\newif\@onlyslidew \@onlyslidewfalse
\newif\@onlynotesw \@onlynoteswfalse
\newif\@makingslides
FMi: \newifG replaces \gdef\@slidew{T} stuff
\newifG\@slidew

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\c@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.
\@definecounter{page}
\@addtoreset{page}{slide}
\@addtoreset{page}{note}
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.

\@setlimits \LIST \LOW \HIGH

\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 

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

\setlimits \LIST \LOW \HIGH

\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 

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

\setlimits \LIST \LOW \HIGH

\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
\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
518 \if@compatibility
519 % In compatibility mode, need to define \verb+\blackandwhite+, \verb+\colors+, \verb+\colorslides+, etc.
520 \def\blackandwhite#1\{
521 \if@onelyslidesw \xdef@doslidelist\doslidelist\doslidelow\doslidehigh\fi
522 \if@onelnotesw \xdef@donotelist\donotelist\donotelow\donotehigh\fi
523 \normalsize\setcounter{slide}{0}\setcounter{overlay}{0}%% \makingslidestrue\input #1\makingslidesfalse
524 \def\colors#1\{
525 \def\@colorlist{}
526 \def\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 % Note, this sets font to \rmfamily , which sets % \@currfont to \rmfamily
      counter slidenumber := 0
      counter note := 0
      counter overlay := 0
      @makingslides := T
      \blackandwhite: @makingslides := T
      \input #1
      @makingslides := F
   fi
528 \def\colors#1\{
529 \for\colortemp=#1\do\expandafter
530 \endcsname\endcsname\noexpand\color{\@colortemp}}\ifx
531 @colorlist\empty \def\colorlist{1}%%
532 \else \def\colorlist{\@colorlist,1}fi
533 \def\colorlist{}
\colors{COLORS} ::= \for\colortemp::COLORS
   do \csname \colortemp \endcsname \color{\@colortemp} od
   if \@colorlist = empty
      then \@colorlist := COLORS
      else \@colorlist := \@colorlist , COLORS
   fi
538 \def\colors#1\{
539 \for\colortemp=#1\do\expandafter
540 \endcsname\endcsname\noexpand\color{\@colortemp}}\ifx
541 \colorlist\empty \def\colorlist{1}%%
542 \else \colorlist{\@colorlist,1}fi
543 \def\colorlist{}
\colorslides{FILE} ::= \newpage
page counter := 0
@bw := F
for \currcolor := \@colorlist
   do @visible := T
      if @onlyslidesw = true
23
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}\fi
\if@onlynotesw \xdef\donotelist{\doglnotelist}\fi
\if@onlyslidesw \xdef\doslidelist{\doglslidelist}\fi
\if@onlynotesw \xdef\donotelist{\doglnotelist}\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
\end{verbatim}
\begin{verbatim}
\def\setupcounters{\newpage\setcounter{page}{0}\@bvue\@visibletrue
\if@onlyslidesw \xdef\doslidelist{\doglslidelist}\fi
\if@onlynotesw \xdef\donotelist{\doglnotelist}\fi
\if@onlyslidesw \xdef\doslidelist{\doglslidelist}\fi
\if@onlynotesw \xdef\donotelist{\doglnotelist}\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
\node{note} and \node{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\@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

567 \if@compatibility
568 \def\slide#1{\stepcounter{slide}\G@slideswtrue\if@onlyslidesw
569 \if\@whilenum \c@slide >\@doslidehigh\relax
570 \do\@setlimits\@doslidelist\@doslidehigh\@doslidehigh\ifnum
571 \c@slide <\@doslidelow\relax\G@slideswfalse\fi\fi
572 \if\G@slidesw
573 \G@slideswfalse
574 \% FMi this is only a hack at the moment to get things running.
575 \% \begingroup
576 \if\@bw\G@slideswtrue\else
577 \@color{#1}\if@visible \G@slideswtrue \fi
578 \fi
579 \% \endgroup
580 \fi
581 \if\G@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\ifonlyslidesw
\@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
\ifG@slidesw \makeslidestrue \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
\if@slidesw = T
then @makingslides := T
\thispagestyle{overlay}
else \end{overlay}
\@gobbletoend{overlay}
fi

END

\endoverlay ::= BEGIN \par\break END

609 \if@compatibility
610 \def\overlay#1{\stepcounter{overlay}\G@slideswtrue%
611 \if@onlyslidesw\ifnum \c@slide < \@doslide\relax
612 \G@slideswfalse\fi\fi
613 \ifG@slidesw \G@slideswfalse\begingroup\if@bw\G@slideswtrue%
614 \else\@color{#1}\if@visible \G@slideswtrue\fi\fi\endgroup\fi
615 \ifG@slidesw \newpage\thispagestyle{overlay}\%
616 \else\end{overlay}\@gobbletoend{overlay}\fi}
617 \%
618 \else \%if@compatibility
619 \%
620 \def\overlay{\par\break \stepcounter{overlay}\
621 \setcounter{page}{0} % in case of non-slide pages
622 \if@bw = T
623 \else \%if@compatibility
624 \else \%if@compatibility
625 \end{overlay}{\par\break}
626 \note ::= BEGIN \par\break \stepcounter{note}
627 \setcounter{page}{0}
628 if \@notesw = T
629 then \@setlimits\@notelist\@notelow\@notehigh od
630 \%
631 \if@notesw = T
632 then % set \@notesw = T iff
633 \end{overlay}{\par\break}
634 \%
635 \if@notesw = T
636 then % page to be output
637 while \c@slide > \@notehigh
638 do \@setlimits\@notelist\@notelow\@notehigh od
if \c@slide < \@donotelow  
then \@slidesw :=G F  
fi  
else \@slidesw :=G F  
fi  
if \@slidesw = T  
then @makingslides := T  
\thispagestyle{note}  
else \endnote{}  
\@gobbletoend{note}  
fi  
END  

\endnote ::=  
BEGIN  
\par\break  
END  

635 \if@compatibility  
636 \def\note{\stepcounter{note}\%  
637 \if@bw  
638 \G@slideswtrue  
639 \if@onlynotesw\@whilenum \c@slide >\@donotehigh\relax  
640 \do{\@setlimits\@donotelist\@donotelow\@donotehigh}\ifnum  
641 \c@slide <\@donotelow\relax \G@slideswfalse\fi\fi  
642 \else\G@slideswfalse\fi  
643 \ifG@slidesw \newpage\thispagestyle{note}\else  
644 \endnote{}\@gobbletoend{note}\fi}  
645 \%  
646 \else \%\if@compatibility  
647 \%  
648 \def\note{\par\break\stepcounter{note}\setcounter{page}{0}\%  
649 \if@bw  
650 \G@slideswtrue  
651 \if@onlynotesw\@whilenum \c@slide >\@donotehigh\relax  
652 \do{\@setlimits\@donotelist\@donotelow\@donotehigh}\ifnum  
653 \c@slide <\@donotelow\relax \G@slideswfalse\fi\fi  
654 \else\G@slideswfalse\fi  
655 \ifG@slidesw \@makingslidestrue\thispagestyle{note}\else  
656 \endnote{}\@gobbletoend{note}\fi}  
657 \fi \%\if@compatibility  
658  
659 \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 

28
then \visible
   \fi
\od
\\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 \@tempswafalse \else \@tempswatrue
   \@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}\i
   \\ignorespaces}}
\def\visible{\@mmodetest
   \if@visible \else \@visibletrue
   \fontshape{\expandafter\@gobble\f@shape}\selectfont\mathversion{normal}\i
   \\ignorespaces}}
\def\fontshape#1{\edef\f@shape{\if@visible \else I\fi #1}}

Here is the \LaTeX\2e 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.
\def\fontshape\f@shape\selectfont\mathversion{invisible}\i
\\ignorespaces}
\def\fontshape\f@shape\selectfont\mathversion{normal}\i
\\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 than 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.
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\}

\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\}

\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\@dashdim \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\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(0,\@height){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,\@height){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,\@height){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi\setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth\@width #1\unitlength\@height \@halfwidth\@depth \@halfwidth\@width \@dashdim\put(0,0){\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\do\@dashdim\@whilenum \@tempcnta <\@dashcnt\put(0,0){\hskip-\@dashdim\copy\@dashbox}\%\put(#2,0){\hskip-\@dashdim\copy\@dashbox}\%\multiply\@dashdim \thr@@\fi
10.3.2 Other modifications to TeX and \LaTeX \ commands

\rule

% _ (Added 10 Nov 86)
\@mathbox{STYLE}{BOX}{MTEXT} : Called in math mode, typesets MTEXT and stores result in BOX, using style STYLE.

\@vbphantom{BOX} : Creates a phantom with dimensions BOX.
\@hbphantom{BOX} : Creates a phantom with ht of BOX and zero width.
\@hvsmash{STYLE}{MTEXT} : Creates a copy of MTEXT with zero height and width in style STYLE.

\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}

\let\@overline=\overline
\let\@underline=\underline
\frac{STYLE}{DENOMSTYLE}{NUM}{DEN}{FONTSIZE}:

Creates $\frac{NUM}{DEN}$ 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.

\begin{verbatim}
864 \def\frac#1#2#3#4#5{
865 \mathchoice
866 \{\@frac\displaystyle\textstyle{#1}{#2}\textfont\}{\@frac
867 \textstyle\scriptstyle{#1}{#2}\textfont\}{\@frac
868 \scriptstyle\scriptscriptstyle{#1}{#2}\scriptfont\}{\@frac
869 \scriptscriptstyle\scriptscriptstyle{#1}{#2}\scriptscriptfont\}}
870 \def\@frac#1#2#3#4#5{\%
871 \boxif\@visible\@smashboxcelse\tw@0fi}
872 \def\r@@t#1#2{
873 \setbox\z@hbox{$\m@th#1\@xysqrt#1{#2}$}\
874 \dimen@\ht\z@ \advance\dimen@-\dp\z@
875 \mskip5mu\raise.6\dimen@\copy\rootbox \mskip-10mu\box\z@}
876 \def\sqrt{\@ifnextchar[\@sqrt}{\@xsqrt}
877 \def\@sqrt[#1]{\root #1\of}
878 \def\@xsqrt#1{\mathchoice{\@xysqrt\displaystyle{#1}}{\@xysqrt
879 \textstyle{#1}}{\@xysqrt\scriptstyle{#1}}{\@xysqrt
880 \scriptscriptstyle{#1}}}\def\@xysqrt#1#2{\@mathbox#1\@smashboxa{#2}\if@visible
881 \@hvsmash#1{\sqrtsign{\@bphantom\@smashboxa}}\fi
882 \phantom{\sqrtsign{\@vbphantom\@smashboxa}}\box\@smashboxa}
883 \newbox\@smashboxa
884 \newbox\@smashboxb
885 \newbox\@smashboxc
886 \array and tabular environments: changes to ‘|’, \hline, \cline, and \vline,
adDED 8 Jun 88
887 \def\@arrayrule{\if@visible\addtopreamble{\hskip -.5\arrayrulewidth
888 \vrule \@width \arrayrulewidth\hskip -.5\arrayrulewidth}fi}
889 \def\@cline{\if@visible\@cheart\ifnum0='\fi}
890 \def\hline{\if@visible \hrule \@height \arrayrulewidth
891 \vrule \@width \arrayrulewidth \hskip -.5\arrayrulewidth}fi}
892 \def\vline{\if@visible \vrule \@height \arrayrulewidth
893 \else \@cheart\ifnum0='\fi}
894 \futurelet \reserved@a@reserved@b@reserved@c@reserved@d@reserved@e
895 \message{output,}
\end{verbatim}
10.3.3 Changes to \LaTeX output routine

\texttt{\@makecol == BEGIN}
% Following test added for slides to check if extra page
if \@makingslides = T
then if \c@page > 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 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.

\message{init}

10.3.4 Special \LaTeX initializations

FMi why not allow for ref's?
\% \nofiles
\@visibletrue
\cmd