

# Producing slides with L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>

Frank Mittelbach

2022/05/18

## 1 Introduction

With L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 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 S<sup>U</sup>T<sub>E</sub>X, in fact it is basically a copy changed to work under L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>.<sup>1</sup> Thus you have to say something like

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

and process this with L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>.

## 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 S<sup>U</sup>T<sub>E</sub>X 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.

---

<sup>1</sup>Therefore you should compare the new class with old S<sup>U</sup>T<sub>E</sub>X styles in case you have local slide classes to see what you have to change in order to use them with L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>.

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  $\text{\LaTeX} 2_{\epsilon}$  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  $\text{\LaTeX}$  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  $\text{\LaTeX} 2_{\epsilon}$  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  $\text{\LaTeX}$ .

```
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.

```
10 \DeclareOption{a4paper}
11   {\setlength\paperheight {297mm}%
12    \setlength\paperwidth  {210mm}}
13 \DeclareOption{a5paper}
14   {\setlength\paperheight {210mm}%
15    \setlength\paperwidth  {148mm}}
16 \DeclareOption{b5paper}
17   {\setlength\paperheight {250mm}%
18    \setlength\paperwidth  {176mm}}
19 \DeclareOption{letterpaper}
20   {\setlength\paperheight {11in}%
21    \setlength\paperwidth  {8.5in}}
22 \DeclareOption{legalpaper}
23   {\setlength\paperheight {14in}%
24    \setlength\paperwidth  {8.5in}}
25 \DeclareOption{executivepaper}
26   {\setlength\paperheight {10.5in}%
27    \setlength\paperwidth  {7.25in}}
```

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

```
28 \DeclareOption{landscape}
29   {\setlength\@tempdima  {\paperheight}%
30    \setlength\paperheight {\paperwidth}%
31    \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.

```
32 \newif\if@clock \@clockfalse
33 \DeclareOption{clock}{\@clocktrue
34   \AtEndDocument{\typeout{\@arabic@c@minutes\space minutes}}
35 }%
36 \newcounter{minutes}%
37 \newcounter{seconds}%
38 \newcommand*\settime[1]{\setcounter{seconds}{0}\addtime{#1}}%
39 \newcommand*\addtime[1]{\addtocounter{seconds}{#1}}%
40 \setcounter{minutes}{\value{seconds}}%
41 \global \divide \value{minutes} by 60\relax
42
```

## 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.

```
43 \@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.

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

## 6.5 Titlepage option

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

```
46 \newif\if@titlepage
47 \@titlepagetrue
48 \DeclareOption{titlepage}{\@titlepagetrue}
49 \DeclareOption{notitlepage}{\@titlepagefalse}
```

## 6.6 Twocolumn printing

Two-column printing is again forbidden.

```
50 \DeclareOption{onecolumn}{}
51 \DeclareOption{twocolumn}{%
52   \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.

```
53 \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.

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

## 7 Executing Options

Here we execute the default options to initialize certain variables.

```
55 \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.

```
56 \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

```
57 % FMi :
58 \def\rmdefault{lcms}      % no roman
59 \def\sfddefault{lcms}
60 \def\tddefault{lcmtt}
61 \def\itdefault{s1}
62 \def\sldefault{s1}
63 \def\bfdefault{bx}
```

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

```
64 \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.

L<sup>A</sup>T<sub>E</sub>X 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 L<sup>A</sup>T<sub>E</sub>X kernel, shorten the following definitions and are used throughout. They are:

<code>\@vpt</code>	5	<code>\@vipt</code>	6	<code>\@viipt</code>	7
<code>\@viipt</code>	8	<code>\@ixpt</code>	9	<code>\@xpt</code>	10
<code>\@xipt</code>	10.95	<code>\@xiipt</code>	12	<code>\@xivpt</code>	14.4
...					

`\ifourteenpt` For S<sup>L</sup>T<sub>E</sub>X, however, these are not sufficient, and we therefore need to add a few extra, larger, sizes.

```
\itwentypt 65 \def\ifourteenpt{13.82}
\itwentyfourpt 66 \def\iseventeenpt{16.59}
\itwenty-ninept 67 \def\itwentypt{19.907}
\ithirtyfourpt 68 \def\itwentyfourpt{23.89}
\ifortyonept 69 \def\itwenty-ninept{28.66}
70 \def\ithirtyfourpt{34.4}
71 \def\ifortyonept{41.28}
```

`\@setfontsize@parms` This routine is used in S<sup>L</sup>T<sub>E</sub>X 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>
```

$\langle belowdisplayskip \rangle$   
 $\langle abovedisplaysshortskip \rangle$   
 $\langle belowdisplaysshortskip \rangle$   
 $\langle strut ht \rangle$   $\langle strut dp \rangle$  (without pt)

For NFSS1 a similar style existed which did run both with a  $\text{SL}\text{T}\text{E}\text{X}$  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  $\langle size \rangle$  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 `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!

```

72 \def\@setfontsize@parms#1#2#3#4#5#6#7#8{%
73   \lineskip #1\relax%
74   \parskip #2\relax
75   \abovedisplayskip #3\relax
76   \belowdisplayskip #4\relax
77   \abovedisplayshortskip #5\relax
78   \belowdisplayshortskip #6\relax
79 %

```

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

```

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

```

Setting size relations for math scripts:

```

83 \DeclareMathSizes{13.82}{13.82}{10}{7}
84 \DeclareMathSizes{16.59}{16.59}{12}{7}
85 \DeclareMathSizes{19.907}{19.907}{16.59}{13.82}
86 \DeclareMathSizes{23.89}{23.89}{19.907}{16.59}
87 \DeclareMathSizes{28.66}{28.66}{23.89}{19.907}
88 \DeclareMathSizes{34.4}{34.4}{28.66}{23.89}
89 \DeclareMathSizes{41.28}{41.28}{34.4}{28.66}

```

`\normalsize`

```

90 \def\normalsize{%
91   \@setfontsize\normalsize\itwentypt{28\p@ plus3\p@ minus4\p@}%
92 %   {20}{30\p@ plus3\p@ minus3\p@}% made a bit shorter
93   \@setfontsize@parms
94     {2pt}%
95     {30\p@ plus18\p@ minus9\p@}%
96     {15\p@ plus3\p@ minus3\p@}%
97     {10\p@ plus3\p@ minus3\p@}%

```

```

98          {10\p@ plus3\p@}
99          \abovedisplayshortskip
100         {17}{7}}

```

We initially choose the normalsize font.

```
101 \normalsize
```

```
\small
```

```

102 \def\smallf{\@setfontsize\small\iseventeenpt{19\p@ plus3\p@ minus\p@}%
103         \@setfontsize@parms
104         {2\p@}%
105         {15\p@ plus15\p@ minus7\p@}%
106         {12\p@ plus3\p@ minus3\p@}%
107         {9\p@ plus3\p@ minus3\p@}%
108         {6\p@ plus3\p@}%
109         \abovedisplayshortskip
110         {13.5}{5.6}}

```

```
\footnotesize
```

```

\scriptsize 111 \let\footnotesize=\small
112 \let\scriptsize=\small

```

```
\tiny
```

```

113 \def\tinyf{\@setfontsize\tiny\ifourteenpt{16\p@ plus2\p@ minus\p@}%
114         \@setfontsize@parms
115         {2pt}%
116         {14\p@ plus3\p@ minus10\p@}%
117         {11\p@ plus3\p@ minus10\p@}%
118         \abovedisplayskip
119         {8\p@ plus3\p@ minus5\p@}%
120         {\z@ plus3\p@}%
121         {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 122 \def\largef{\@setfontsize\large\itwentyfourpt{42\p@ plus8\p@ minus5\p@}%
```

```
\LARGE 123         \@setfontsize@parms
```

```
\huge 124         {2\p@}%
```

```
\Huge 125         {40\p@ plus20\p@ minus4\p@}%
```

```
126         {20\p@ plus8\p@ minus3\p@}%
```

```
127         \abovedisplayskip
```

```
128         {10\p@ plus5\p@}%
```

```
129         \abovedisplayshortskip
```

```
130         {20}{8.5}}
131
```

```
132 \def\Largef{\@setfontsize\Large\itwentynept{48\p@ plus10\p@ minus6\p@}%
```

```
133         \@setfontsize@parms
```

```
134         {2\p@}%
```

```
135         {48\p@ plus30\p@ minus6\p@}%
```

```
136         {24\p@ plus10\p@ minus6\p@}%

```

```

137         \abovedisplayskip
138         {12\p@ plus8\p@}%
139         \abovedisplayshortskip
140         {27}{11}}
141
142 \def\LARGE{\@setfontsize\LARGE\ithirtyfourpt{52\p@ plus10\p@ minus6\p@}%
143         \@setfontsize@parms
144         {2\p@}%
145         {52\p@ plus30\p@ minus6\p@}%
146         {24\p@ plus10\p@ minus6\p@}%
147         \abovedisplayskip
148         {12\p@ plus8\p@}%
149         \abovedisplayshortskip
150         {27}{11}}
151
152 \def\huge{\@setfontsize\huge\ifortyonept{60\p@ plus10\p@ minus6\p@}%
153         \@setfontsize@parms
154         {2\p@}%
155         {60\p@ plus30\p@ minus6\p@}%
156         {24\p@ plus10\p@ minus6\p@}%
157         \abovedisplayskip
158         {12\p@ plus8\p@}%
159         \abovedisplayshortskip
160         {27}{11}}
161
162 \let\Huge\huge

```

## 9.2 Paragraphing

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

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

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

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

`\@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.

```
165 \@lowpenalty 51
```

```
166 \@medpenalty 151
```

```
167 \@highpenalty 301
```

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

```
168 % \clubpenalty 150
```

```
169 % \widowpenalty 150
```

`\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.

`\predisplaypenalty` Again the default values are used, therefore we only show them here.

`\postdisplaypenalty` Again the default values are used, therefore we only show them here.

```
170 % \displaywidowpenalty 50
```

```
171 % \predisplaypenalty 10000
```



```
172 % \postdisplaypenalty 0
```

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

```
173 % \interlinepenalty 0
```

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

```
174 % \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` The `\headheight` is the height of the box that will contain the running head. The `\headsep` `\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.

```
175 \setlength\headheight{14\p@}
```

```
176 \setlength\headsep {15\p@}
```

```
177 \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:

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

`\maxdepth` The `TeX` primitive register `\maxdepth` has a function that is similar to that of `\@maxdepth` `\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, therefore we set `\maxdepth` to half the value of `\topskip`.

```
179 \if@compatibility
```

```
180 \setlength\maxdepth{4\p@}
```

```
181 \else
```

```
182 \setlength\maxdepth{.5\topskip}
```

```
183 \fi
```

```
184 \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.

```
185 \if@compatibility
```

```
186 \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.

```
187 \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`.

```
188 \setlength\@tempdima{\paperwidth}
189 \addtolength\@tempdima{-2in}
190 \setbox\@tempboxa\hbox{\rmfamily im}
191 \setlength\@tempdimb{.5\wd\@tempboxa}
192 \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.

```
193 \ifdim\@tempdima>\@tempdimb\relax
194   \setlength\textwidth{\@tempdimb}
195 \else
196   \setlength\textwidth{\@tempdima}
197 \fi
198 \fi
```

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

```
199 \@settopoint\textwidth
```

```
\columnwidth
  \columnsep 200 \columnwidth \textwidth
\columnseprule 201 \columnsep 10pt
                202 \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 L<sup>A</sup>T<sub>E</sub>X2.09. 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.

```
203 \if@compatibility
204   \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.

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

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

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

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

```
208 \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.

```

209 \divide\@tempdima\baselineskip
210 \@tempcnta=\@tempdima

```

From this we can calculate the height of the text.

```

211 \setlength\textheight{\@tempcnta\baselineskip}
212 \fi

```

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

```

213 \advance\textheight by \topskip

```

### 9.3.3 Margins

`\oddsidemargin` First we give the values for the compatibility mode.

`\evensidemargin` Values for two-sided printing:

```

\marginparwidth 214 \ifcompatibility
215 \setlength\oddsidemargin {17\p@}
216 \setlength\evensidemargin {17\p@}
217 \setlength\marginparwidth {20\p@}
218 \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.

```

219 \setlength\@tempdima {\paperwidth}
220 \addtolength\@tempdima {-2in}
221 \addtolength\@tempdima {-\textwidth}
222 \setlength\oddsidemargin {.5\@tempdima}
223 \setlength\marginparwidth {.8in}
224 \addtolength\marginparwidth {.5\@tempdima}

```

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

```

225 \setlength\evensidemargin {\paperwidth}
226 \addtolength\evensidemargin{-2in}
227 \addtolength\evensidemargin{-\textwidth}
228 \addtolength\evensidemargin{-\oddsidemargin}
229 \fi

```

`\marginparsep` The horizontal space between the main text and marginal notes is determined by

`\marginparpush` `\marginparsep`, the minimum vertical separation between two marginal notes is controlled by `\marginparpush`.

```

230 \setlength\marginparsep {5\p@}
231 \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.

```

232 \ifcompatibility
233 \setlength\topmargin{-10pt}
234 \else
235 \setlength\topmargin{\paperheight}

```

```

236 \addtolength\topmargin{-2in}
237 \addtolength\topmargin{-\headheight}
238 \addtolength\topmargin{-\headsep}
239 \addtolength\topmargin{-\textheight}
240 \addtolength\topmargin{-\footskip}      % this might be wrong!

```

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

```

241 \addtolength\topmargin{-.5\topmargin}
242 \fi
243 \@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.

```

244 \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.

```

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

```

246 \ifcompatibility
247 \def\ps@headings{%
248 \def\ps@slide{\def\@oddfoot{\@mainsize +\hfil\hb@xt@3em{\theslide
249                                     \hss}}%
250 \def\@oddhead{\@mainsize +\hfil +}%
251 \def\@evenfoot{\@mainsize +\hfil\hb@xt@3em{\theslide\hss}}%
252 \def\@evenhead{\@mainsize +\hfil +}}
253
254 \def\ps@overlay{\def\@oddfoot{\@mainsize +\hfil\hb@xt@3em{\theoverlay
255                                     \hss}}%
256 \def\@oddhead{\@mainsize +\hfil +}%
257 \def\@evenfoot{\@mainsize +\hfil\hb@xt@3em{\theoverlay\hss}}%

```

```

258 \def\@evenhead{\@mainsize +\hfil +}}
259 \def\ps@note{\def\@oddfoot{\@mainsize \hbox{\hfil\thenote}%
260 \def\@oddhead{}}%
261 \def\@evenfoot{\@mainsize \hbox{\hfil\thenote}%
262 \def\@evenhead{}}}}
263 %
264 \else %%if@compatibility
265 %
266 \def\ps@headings{%
267 \def\ps@slide{%
268 \def\@oddfoot{\@mainsize \mbox{\hfil\hb@xt@3em{\theslide\hss}}}%
269 \def\@oddhead{}}%
270 \def\@evenfoot{\@mainsize \mbox{\hfil\hb@xt@3em{\theslide\hss}}}%
271 \def\@evenhead{}}}}
272
273 \def\ps@overlay{%
274 \def\@oddfoot{\@mainsize \mbox{\hfil\hb@xt@3em{\theoverlay\hss}}}%
275 \def\@oddhead{}}%
276 \def\@evenfoot{\@mainsize \mbox{\hfil\hb@xt@3em{\theoverlay\hss}}}%
277 \def\@evenhead{}}}}
278
279 \def\ps@note{%
280 \def\@oddfoot{%
281 \@mainsize
282 \if@clock
283 \fbox{\large \@arabic\c@minutes\space min}%
284 \else
285 \null
286 \fi
287 \hfil\thenote}%
288 \def\@oddhead{}}%
289 \def\@evenfoot{%
290 \@mainsize
291 \if@clock
292 \fbox{\large \@arabic\c@minutes\space min}%
293 \else
294 \null
295 \fi
296 \hfil\thenote}%
297 \def\@evenhead{}}}}
298 \fi %% if@compatibility

```

\ps@plain

```

299 \def\ps@plain{\def\ps@slide{%
300 \def\@oddfoot{\@mainsize \mbox{\hfil\hb@xt@3em{\theslide\hss}}}%
301 \def\@oddhead{}}%
302 \def\@evenfoot{\@mainsize \mbox{\hfil\hb@xt@3em{\theslide\hss}}}%
303 \def\@evenhead{}}}}
304 \def\ps@overlay{\def\@oddfoot{\@mainsize
305 \mbox{\hfil\hb@xt@3em{\theoverlay\hss}}}%
306 \def\@oddhead{}}%
307 \def\@evenfoot{\@mainsize \mbox{\hfil\hb@xt@3em{\theoverlay\hss}}}%
308 \def\@evenhead{}}}}
309 \def\ps@note{\def\@oddfoot{\@mainsize \hbox{\hfil\thenote}%

```

```

310 \def\@oddhead{}%
311 \def\@evenfoot{\@mainsize \hbox{}\hfil\thenote}%
312 \def\@evenhead{}}

```

\ps@empty

```

313 \def\ps@empty{%
314 \def\ps@slide{\def\@oddhead{}\def\@oddfoot{}%
315 \def\@evenhead{}\def\@evenfoot{}%
316 \def\ps@overlay{\def\@oddhead{}\def\@oddfoot{}%
317 \def\@evenhead{}\def\@evenfoot{}%
318 \def\ps@note{\def\@oddhead{}\def\@oddfoot{}%
319 \def\@evenhead{}\def\@evenfoot{}}

```

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

```

320 \ps@headings

```

Set ordinary page style to 'empty'

```

321 \let\@oddhead\@empty\let\@oddfoot\@empty
322 \let\@evenhead\@empty\let\@evenfoot\@empty

```

## 9.5 Providing math *versions*

L<sup>A</sup>T<sub>E</sub>X provides two *versions*. We call them normal and bold, respectively. S<sup>U</sup>T<sub>E</sub>X 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.

```

323 \DeclareMathVersion{invisible}

```

Now we define the basic *math groups* used by L<sup>A</sup>T<sub>E</sub>X. 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.

```

324 \SetSymbolFont{operators}{normal}
325           {OT1}{lcmss}{m}{n}
326
327 \SetSymbolFont{letters}{normal}
328           {OML}{lcm}{m}{it}
329 \SetSymbolFont{symbols}{normal}
330           {OMS}{lcm}{m}{n}
331 \SetSymbolFont{largesymbols}{normal}
332           {OMX}{lcmex}{m}{n}
333
334 \SetSymbolFont{operators}{invisible}
335           {OT1}{lcmss}{m}{In}
336 \SetSymbolFont{letters}{invisible}
337           {OML}{lcm}{m}{Iit}
338 \SetSymbolFont{symbols}{invisible}
339           {OMS}{lcm}{m}{In}
340 \SetSymbolFont{largesymbols}{invisible}
341           {OMX}{lcmex}{m}{In}
342
343
344 \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.

```
345 \newenvironment{titlepage}
346     {\newpage
347      \thispagestyle{empty}%
348      \setcounter{page}{\z@}}
349     {\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 L<sup>A</sup>T<sub>E</sub>X manual for an explanation of the meaning of the parameters.

```
\leftmargini
\leftmarginii 350 \setlength\leftmargini   {38\p@}
\leftmarginiii 351 \setlength\leftmarginii  {30\p@}
\leftmarginiv  352 \setlength\leftmarginiii {20\p@}
\leftmarginv   353 \setlength\leftmarginiv  {15\p@}
\leftmarginvi  354 \setlength\leftmarginv   {15\p@}
\leftmarginvii 355 \setlength\leftmarginvi   {10\p@}
```

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

```
\@listv 356 \def\@listi{\leftmargin\leftmargini
\@listvi 357     \parsep .5\parskip
358     \topsep \parsep
359     \itemsep\parskip
360     \partopsep \z@}
361
362 \def\@listii{\leftmargin\leftmarginii
363     \labelwidth\leftmarginii
364     \advance\labelwidth-\labelsep
365     \parsep .5\parskip
366     \topsep \parsep
367     \itemsep\parskip}
368 \def\@listiii{\leftmargin\leftmarginiii
369     \labelwidth\leftmarginiii
370     \advance\labelwidth-\labelsep}
371 \def\@listiv{\leftmargin\leftmarginiv
372     \labelwidth\leftmarginiv
373     \advance\labelwidth-\labelsep}
374 \def\@listv{\leftmargin\leftmarginv
375     \labelwidth\leftmarginv
376     \advance\labelwidth-\labelsep}
377 \def\@listvi{\leftmargin\leftmarginvi
378     \labelwidth\leftmarginvi
379     \advance\labelwidth-\labelsep}
```

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

```
380 \leftmargin\leftmargini
381 \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.

```
382 \newenvironment{verse}{\let\=\@centercr
383 \list{}{\itemsep \z@
384 \itemindent -15\p@
385 \listparindent \itemindent
386 \rightmargin \leftmargin
387 \advance\leftmargin 15\p@}%
388 \item[]}
389 {\endlist}
```

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

```
390 \newenvironment{quotation}{\list{}{\listparindent 20\p@
391 \itemindent\listparindent
392 \rightmargin\leftmargin}%
393 \item[]}
394 {\endlist}
```

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

```
395 \newenvironment{quote}{\list{}{\rightmargin\leftmargin}\item[]}
396 {\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  $\text{\LaTeX}$  format.

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

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

```
401 \newcommand*{\descriptionlabel}[1]{\hspace\labelsep
402 \normalfont\bfseries #1}
403
```

### 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` The counters are already defined in the  $\text{\LaTeX}$  format, but their representation is changed here.

```
\theenumii
\theenumiii 404 \renewcommand\theenumi{\@arabic\c@enumi}
\theenumiv 405 \renewcommand\theenumii{\@alph\c@enumii}
```



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

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

`\labelenumiii` 408 `\newcommand\labelenumi{\theenumi.}`  
`\labelenumiv` 409 `\newcommand\labelenumii{\theenumii}`  
 410 `\newcommand\labelenumiii{\theenumiii.}`  
 411 `\newcommand\labelenumiv{\theenumiv.}`

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

`\p@enumiv` 412 `\renewcommand\p@enumii{\theenumi}`  
 413 `\renewcommand\p@enumiii{\theenumi(\theenumii)}`  
 414 `\renewcommand\p@enumiv{\p@enumiii\theenumiii}`

### 9.6.5 Itemize

`\labelitemi` Itemization is controlled by four commands: `\labelitemi`, `\labelitemii`,  
`\labelitemiii` `\labelitemiii`, and `\labelitemiv`, which define the labels of the various item-  
`\labelitemiiii` ization levels.

`\labelitemiv` 415 `\newcommand\labelitemi{\m@th\bullet}`  
 416 `\newcommand\labelitemii{\normalfont\bfseries \textendash}`  
 417 `\newcommand\labelitemiii{\m@th\ast}`  
 418 `\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

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

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

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

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

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

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

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

### 9.7.2 Tabbing

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

423 `\labelsep 10pt`

424 `\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.

```
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  $\text{\LaTeX}$  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,  $\text{\LaTeX}$  commands to execute in text mode and  $\text{\LaTeX}$  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  $\text{\LaTeX}$ 2.09 behaviour.

```
\sf 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}
```

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

`\cal` The commands `\cal` and `\mit` should only be used in math mode, outside math `\mit` 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` Footnotes are numbered within slides, overlays, and notes and numbered with `*`, `\thefootnote` †, etc.

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

`\@makefnmark` The footnote mechanism of L<sup>A</sup>T<sub>E</sub>X calls the macro `\@makefnmark` 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 `\@makefnmark` 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 T<sub>E</sub>X code.

```
\long\def\@makefnmark#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.

```
447 \long\def\@makefntext#1{
448     \noindent
449     \hangindent 10\p@
450     \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.

```
451 %\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`.

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

## 10 Initialization

### 10.1 Date

`\today` This macro uses the T<sub>E</sub>X primitives `\month`, `\day` and `\year` to provide the date of the L<sup>A</sup>T<sub>E</sub>X-run.

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

Default initializations

```
459 \pagenumbering{arabic}
460 \onecolumn
461 \</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

```
462 \*cmd)
463 \message{hacks,}
464
```

```

465 \outer\def\newifG#1{\count@\escapechar \escapechar\m@ne
466 \expandafter\expandafter\expandafter
467 \edef\@ifG#1{true}{\global\let\noexpand#1\noexpand\iftrue}%
468 \expandafter\expandafter\expandafter
469 \edef\@ifG#1{false}{\global\let\noexpand#1\noexpand\iffalse}%
470 \@ifG#1{false}\escapechar\count@} % the condition starts out false
471 \def\@ifG#1#2{\csname\expandafter\ifG@\string#1#2\endcsname}
472 {\uccode'1='i \uccode'2='f \uccode'3='G \uppercase{\gdef\ifG@123{G}}}
473 % 'ifG' is required
474
475 \def\@gobbletoend#1{\def\@argend{#1}\@gobtoend}
476
477 \long\def\@ggobtoend#1\end#2{\fi\def\reserved@a{#2}%
478 \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}`

```
479 \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

```

```

480 \newif\if@bw
481 \newif\if@visible
482 \newif\if@onlyslidesw \@onlyslideswfalse
483 \newif\if@onlynotesw \@onlynoteswfalse
484 \newif\if@makingslides

```

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

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

Counters

```

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

```

```

486 \countdef\c@slide=0 \c@slide=0
487 \def\cl@slide{}
488 \countdef\c@overlay=1 \c@overlay=0
489 \def\cl@overlay{}
490 \countdef\c@note=2 \c@note=0
491 \def\cl@note{}

```

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

```

492 \g@addto@macro\cl@ckpt{\@elt{slide}\@elt{overlay}\@elt{note}}
493 \@addtoreset{overlay}{slide}
494 \@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.

```

495 \@definecounter{page}
496 \@addtoreset{page}{slide}
497 \@addtoreset{page}{note}

```

```

498 \@addtoreset{page}{overlay}
499
500 \def\theslide{\@arabic\c@slide}
501 \def\theoverlay{\theslide-\@alph\c@overlay}
502 \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.

503 \def\@sl@getargs#1-#2-#3\relax#4#5{\xdef#4{#1}\xdef#5{#2}}
504 \def\@sl@ccdr#1,#2\relax#3#4{\xdef#3{#1-#1-}\xdef#4{#2}}
505
506 \def\@setlimits #1#2#3{\expandafter\@sl@ccdr#1\relax\@sl@gtmp #1%
507 \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

508 \def\onlyslides#1{\@onlyslideswtrue
509 \gdef\@doglslidelist{#1,999999,999999}%
510 \if@onlynotesw \else
511 \@onlynoteswtrue\gdef\@doglnotelist{999999,999999}\fi
512 \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

513 \def\onlynotes#1{\@onlynoteswtrue
514 \gdef\@doglnotelist{#1,999999,999999}%
515 \if@onlyslidesw \else
516 \@onlyslideswtrue\gdef\@doglslidelist{999999,999999}\fi
517 \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

518 \if@compatibility
519 % In compatibility mode, need to define \verb+\blackandwhite+,
520 % \verb+\colors+, \verb+\colorslides+, etc.
521 \def\blackandwhite#1{\newpage\setcounter{page}{0}\@bwtrue\@visibletrue
522 \if@onlyslidesw \xdef\@doslidelist{\@doglslidelist}%
523 \@setlimits\@doslidelist\@doslidelow\@doslidehigh\fi
524 \if@onlynotesw \xdef\@donotelist{\@doglnotelist}%
525 \@setlimits\@donotelist\@donotelow\@donotehigh\fi
526 \normalsize\setcounter{slide}{0}\setcounter{overlay}{0}%
527 \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

528 \def\colors#1{\@for\@colortemp:=#1\do{\expandafter
529 \xdef\csname\@colortemp\endcsname{\noexpand\@color{\@colortemp}}}\ifx
530 \@colorlist@empty \gdef\@colorlist{#1}%
531 \else \xdef\@colorlist{\@colorlist,#1}\fi}
532
533 \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

534 \def\colorslides#1{\newpage\setcounter{page}{0}\@bwfalse
535 \@for\@currcolor:=\@colorlist\do
536 {\@visibletrue
537 \if@onlyslidesw \xdef\@doslidelist{\@doglslidelist}%
538 \@setlimits\@doslidelist\@doslidelow\@doslidehigh\fi
539 \if@onlynotesw \xdef\@donotelist{\@doglnotelist}%
540 \@setlimits\@donotelist\@donotelow\@donotehigh\fi
541 \normalsize\setcounter{slide}{0}\setcounter{overlay}{0}%
542 \setcounter{note}{0}\typeout{color \@currcolor}%
543 \newpage
544 \begin{huge}%
545 \begin{center}%
546 COLOR LAYER\ [.75in]%
547 \@currcolor
548 \end{center}%
549 \end{huge}%
550 \newpage
551 \@makingslidestrue
552 \input #1
553 \@makingslidesfalse}}
554 %
555 \else %% if@compatibility
556 %
557 \def\setupcounters{\newpage\setcounter{page}{0}\@bwtrue\@visibletrue
558 \if@onlyslidesw \xdef\@doslidelist{\@doglslidelist}%
559 \@setlimits\@doslidelist\@doslidelow\@doslidehigh\fi
560 \if@onlynotesw \xdef\@donotelist{\@doglnotelist}%
561 \@setlimits\@donotelist\@donotelow\@donotehigh\fi
562 \normalsize\setcounter{slide}{0}\setcounter{overlay}{0}%
563 \setcounter{note}{0}\@makingslidesfalse}
564
565 \AtBeginDocument{\setupcounters}
566 \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
\beginingroup
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 \@whilenum \c@slide >\@doslidehigh\relax
570 \do{\@setlimits\@doslidelist\@doslidelow\@doslidehigh}\ifnum
571 \c@slide <\@doslidelow\relax\G@slideswfalse\fi\fi
572 \ifG@slidesw
573 \G@slideswfalse
574 % FMi this is only a hack at the moment to get things running.
575 % \beginingroup
576 \if@bw\G@slideswtrue\else
577 \@color{#1}\if@visible \G@slideswtrue \fi
578 \fi
579 % \endgroup
580 \fi
581 \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).

```

582 \csname \last@color \endcsname
583 \else\end{slide}\@gobbletoend{slide}\fi}
584 %
585 \else %% if@compatibility
586 %
587 \def\slide{\par\break
588 \stepcounter{slide}\setcounter{page}{0}\G@slideswtrue\if@onlyslidesw
589 \@whilenum \c@slide >\doslidehigh\relax
590 \do{\@setlimits\@doslidelist\@doslidelow\@doslidehigh}\ifnum
591 \c@slide <\@doslidelow\relax\G@slideswfalse\fi\fi
592 \ifG@slidesw
593 \G@slideswfalse
594 % FMI this is only a hack at the moment to get things running.
595 % \begingroup
596 \if@bw\G@slideswtrue\else
597 \if@visible \G@slideswtrue \fi
598 \fi
599 % \endgroup
600 \fi
601 \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).

```

602 \csname \last@color \endcsname
603 \else\end{slide}\@gobbletoend{slide}\fi}
604 \fi %% if@compatibility
605
606 \let\last@color\@empty
607
608 \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
fi
END

\endoverlay ::=
BEGIN
    \par\break
END

609 \if@compatibility
610 \def\overlay#1{\stepcounter{overlay}\G@slideswtrue%
611 \if@onlyslidesw\ifnum \c@slide <\@doslidelow\relax
612 \G@slideswfalse\fi\fi
613 \ifG@slidesw \G@slideswfalse\begin\group\if@bw\G@slideswtrue%
614 \else@color{#1}\if@visible \G@slideswtrue\fi\fi\end\group\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
621 \stepcounter{overlay}%
622 \setcounter{page}{0}%
623 \G@slideswtrue%
624 \if@onlyslidesw\ifnum \c@slide <\@doslidelow\relax
625 \G@slideswfalse\fi\fi
626 \ifG@slidesw \G@slideswfalse
627 \begin\group\if@bw\G@slideswtrue%
628 \else\if@visible \G@slideswtrue\fi\fi
629 \end\group\fi
630 \ifG@slidesw \@makingslides\thispagestyle{overlay}%
631 \else\end{overlay}\@gobbletoend{overlay}\fi}
632 \fi %%if@compatibility
633
634 \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
  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     \end{note}\@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     \end{note}\@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

```

```

        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\@tempwafalse` would produce the same effect I imagine.

```

660 \def\@color#1{\@mmodetest
661  {\if@bw \@tempwatrue \else \@tempwafalse
662   \@for \reserved@a :=#1\do{\ifx\reserved@a\currcolor\@tempwatrue\fi
663    \let\last@color\reserved@a}\fi
664   \if@tempwa \visible \else \invisible \fi
665   \ignorespaces}}
666
667 \def\@mmodetest#1{\ifmode\ClassWarning{slides}{Color-changing command
668   in math mode has been ignored}\else #1\fi}
669
670 \def\invisible{\@mmodetest
671  {\if@visible
672   \@visiblefalse
673   \fontshape\f@shape\selectfont
674   \mathversion{invisible}%
675   \fi
676   \ignorespaces}}
677
678 \def\visible{\@mmodetest
679  {\if@visible
680   \else
681    \@visibletrue

```

Here is the  $\LaTeX 2_{\epsilon}$  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.

```

682   \fontshape{\expandafter\@gobble\f@shape}\selectfont
683   \mathversion{normal}%
684   \fi
685   \ignorespaces}}
686
687 \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.

```

688 \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`.

```

689 \DeclareFontFamily{U}{lasy}{}{}
690 \DeclareFontShape{U}{lasy}{m}{n}{%
691     <12><13.82><16.59><19.907><23.89><28.66><34.4><41.28>lasy8
692 }{}
693 \DeclareFontShape{U}{lasy}{m}{In}{%
694     <13.82><16.59><19.907><23.89><28.66><34.4><41.28>ilasy8
695 }{}
696 \message{picture,}

```

### 10.3.1 Modifications to the picture environment

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

```

697 \def\line(#1,#2)#3{\if@visible\@xarg #1\relax \@yarg #2\relax
698 \@linelen #3\unitlength
699 \ifnum\@xarg =\z@ \@vline
700 \else \ifnum\@yarg =\z@ \@hline \else \@sline\fi
701 \fi\fi}
702
703 \def\vector(#1,#2)#3{\if@visible\@xarg #1\relax \@yarg #2\relax
704 \@linelen #3\unitlength
705 \ifnum\@xarg =\z@ \@vvector
706 \else \ifnum\@yarg =\z@ \@hvector \else \@svector\fi
707 \fi\fi}
708
709 \def\dashbox#1(#2,#3){%
710 \leavevmode\if@visible\hb@xt@\z@{\@baselineskip \z@
711 \lineskip \z@
712 \@dashdim #2\unitlength
713 \@dashcnt \@dashdim \advance\@dashcnt 200
714 \@dashdim #1\unitlength\divide\@dashcnt \@dashdim
715 \ifodd\@dashcnt\@dashdim\z@
716 \advance\@dashcnt \@one \divide\@dashcnt \tw@
717 \else \divide\@dashdim \tw@ \divide\@dashcnt \tw@
718 \advance\@dashcnt \m@ne
719 \setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth
720 \@width \@dashdim}\put(0,0){\copy\@dashbox}%
721 \put(0,#3){\copy\@dashbox}%
722 \put(#2,0){\hskip-\@dashdim\copy\@dashbox}%
723 \put(#2,#3){\hskip-\@dashdim\box\@dashbox}%
724 \multiply\@dashdim \thr@@
725 \fi
726 \setbox\@dashbox \hbox{\vrule \@height \@halfwidth \@depth \@halfwidth
727 \@width #1\unitlength\hskip #1\unitlength}\@tempcnta\z@
728 \put(0,0){\hskip\@dashdim \@whilenum \@tempcnta <\@dashcnt
729 \do{\copy\@dashbox\advance\@tempcnta \@one }}\@tempcnta\z@
730 \put(0,#3){\hskip\@dashdim \@whilenum \@tempcnta <\@dashcnt
731 \do{\copy\@dashbox\advance\@tempcnta \@one }}%
732 \@dashdim #3\unitlength
733 \@dashcnt=\@dashdim \advance\@dashcnt 200
734 \@dashdim #1\unitlength\divide\@dashcnt \@dashdim
735 \ifodd\@dashcnt \@dashdim=\z@

```

```

736 \advance\@dashcnt \@ne \divide\@dashcnt \tw@
737 \else
738 \divide\@dashdim \tw@ \divide\@dashcnt \tw@
739 \advance\@dashcnt \m@ne
740 \setbox\@dashbox\hbox{\hskip -\@halfwidth
741 \vrule \@width \@wholewidth
742 \@height \@dashdim}\put(0,0){\copy\@dashbox}%
743 \put(#2,0){\copy\@dashbox}%
744 \put(0,#3){\lower\@dashdim\copy\@dashbox}%
745 \put(#2,#3){\lower\@dashdim\copy\@dashbox}%
746 \multiply\@dashdim \thr@@
747 \fi
748 \setbox\@dashbox\hbox{\vrule \@width \@wholewidth
749 \@height #1\unitlength}\@tempcnta\z@
750 \put(0,0){\hskip -\@halfwidth \vbox{\@whilenum \@tempcnta <\@dashcnt
751 \do{\vskip #1\unitlength\copy\@dashbox\advance\@tempcnta \@ne }%
752 \vskip\@dashdim}}\@tempcnta\z@
753 \put(#2,0){\hskip -\@halfwidth \vbox{\@whilenum \@tempcnta <\@dashcnt
754 \relax\do{\vskip #1\unitlength\copy\@dashbox\advance\@tempcnta \@ne }%
755 \vskip\@dashdim}}\fi\@makepicbox(#2,#3)}

```

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

```

756 \newif\if@ovvline \@ovvlinetrue
757 \newif\if@ovhline \@ovhlinetrue

758 \def\@oval(#1,#2)[#3]{\if@visible\begin@group \boxmaxdepth \maxdimen
759 \@ovttrue \@ovbtrue \@ovltrue \@ovrtrue
760 \@ovvlinefalse \@ovhlinefalse

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

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

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

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

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

781
782 \def\@circle#1{\if@visible \begin@group \boxmaxdepth \maxdimen
783   \@tempdimb #1\unitlength

```

```

784 \ifdim \@tempdimb >15.5\p@\relax \@getcirc\@tempdimb
785 \ovro\ht\@tempboxa
786 \setbox\@tempboxa\hbox{\@circlefnt
787 \advance\@tempcnta\tw@ \char \@tempcnta
788 \advance\@tempcnta\m@ne \char \@tempcnta \kern -2\@tempdima
789 \advance\@tempcnta\tw@
790 \raise \@tempdima \hbox{\char\@tempcnta}\raise \@tempdima
791 \box\@tempboxa}\ht\@tempboxa\z@ \dp\@tempboxa\z@
792 \@put{-\@ovro}{-\@ovro}{\box\@tempboxa}%
793 \else \@circ\@tempdimb{96}\fi\endgroup\fi}
794
795 \def\@dot#1{%
796 \if@visible\@tempdimb #1\unitlength \@circ\@tempdimb{112}\fi}
797 \def\@frameb#x#1{%
798 \@tempdima\fbboxrule
799 \advance\@tempdima\fbboxsep
800 \advance\@tempdima\dp\@tempboxa
801 \leavevmode
802 \hbox{%
803 \lower\@tempdima\hbox{%
804 \vbox{%
805 \if@visible\hrule\@height\else\vskip\fi\fbboxrule
806 \hbox{%
807 \if@visible\vrule\@width\else\hskip\fi\fbboxrule
808 #1%
809 \vbox{%
810 \vskip\fbboxsep
811 \box\@tempboxa
812 \vskip\fbboxsep}%
813 #1%
814 \if@visible\vrule\@width\else\hskip\fi\fbboxrule}%
815 \if@visible\hrule\@height\else\vskip\fi\fbboxrule}}}}
816
817 \long\def\frame#1{\if@visible\leavevmode
818 \vbox{\vskip-\@halfwidth\hrule \@height\@halfwidth \@depth \@halfwidth
819 \vskip-\@halfwidth\hbox{\hskip-\@halfwidth \vrule \@width\@wholewidth
820 \hskip-\@halfwidth #1\hskip-\@halfwidth \vrule \@width \@wholewidth
821 \hskip -\@halfwidth}\vskip -\@halfwidth\hrule \@height \@halfwidth
822 \@depth \@halfwidth\vskip -\@halfwidth}\else #1\fi}
823 \message{mods,}

```

### 10.3.2 Other modifications to T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X commands

```

\rule
824 \def\@rule[#1]#2#3{\@tempdima#3\advance\@tempdima #1\leavevmode
825 \hbox{\if@visible\vrule
826 \@width#2 \@height\@tempdima \@depth-#1\else
827 \vrule \@width \z@ \@height\@tempdima \@depth-#1\vrule
828 \@width#2 \@height\z@\fi}}
829
830 % \_ (Added 10 Nov 86)
831

```



```

832 \def\_f\leavevmode \kern.06em \if@visible\ vbox{\hrule \@width.3em}\else
833   \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.

834 \def\@mathbox#1#2#3{\setbox#2\hbox{\m@th#1{#3}}$}
835
836 \def\@vbphantom#1{\setbox\tw@ \null \ht\tw@\ht #1\dp\tw@\dp #1%
837   \box\tw@}
838
839 \def\@bphantom#1{\setbox\tw@ \null
840   \wd\tw@\wd #1\ht\tw@\ht #1\dp\tw@\dp #1%
841   \box\tw@}
842
843 \def\@hbphantom#1{\setbox\tw@ \null \wd\tw@\wd #1\ht\tw@\z@ \dp\tw@\z@
844   \box\tw@}
845
846 \def\@hvsmash#1#2{\@mathbox#1\z@{#2}\ht\z@\z@ \dp\z@\z@ \wd\z@\z@
847   \box\z@}
848
849 \def\underline#1{\relax\ifmmode
850   \@xunderline{#1}\else $\m@th\@xunderline{\hbox{#1}}$\relax\fi}
851
852 \def\@xunderline#1{\mathchoice{\@xyunderline\displaystyle{#1}}%
853   {\@xyunderline
854     \textstyle{#1}}{\@xyunderline\scriptstyle{#1}}{\@xyunderline
855     \scriptscriptstyle{#1}}}
856
857 \def\@xyunderline#1#2{%
858   \@mathbox#1\@smashboxa{#2}\@hvsmash#1{\copy\@smashboxa}%
859   \if@visible \@hvsmash#1{\@underline{\@bphantom\@smashboxa}}\fi
860   \@mathbox#1\@smashboxb{\@underline{\box\@smashboxa}}%
861   \@bphantom\@smashboxb}
862
863 \let\@overline=\overline
864
865 \def\overline#1{\mathchoice{\@xoverline\displaystyle{#1}}{\@xoverline
866   \textstyle{#1}}{\@xoverline\scriptstyle{#1}}{\@xoverline
867   \scriptscriptstyle{#1}}}
868
869 \def\@xoverline#1#2{%
870   \@mathbox#1\@smashboxa{#2}\@hvsmash#1{\copy\@smashboxa}%
871   \if@visible \@hvsmash#1{\@overline{\@bphantom\@smashboxa}}\fi
872   \@mathbox#1\@smashboxb{\@overline{\box\@smashboxa}}%
873   \@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.

```

874 \def\frac#1#2{\mathchoice
875   {\@frac\displaystyle\textstyle{#1}{#2}\textfont}{\@frac
876     \textstyle\scriptstyle{#1}{#2}\textfont}{\@frac
877       \scriptstyle\scriptscriptstyle{#1}{#2}\scriptfont}{\@frac
878         \scriptscriptstyle\scriptscriptstyle{#1}{#2}\scriptscriptfont}}
879
880 \def\@frac#1#2#3#4#5{%
881   \@mathbox#1\@smashboxc{\@begingroup#3\endgroup\over#4}}%
882   \setbox\tw@null
883   \ht\tw@ \ht\@smashboxc
884   \dp\tw@ \dp\@smashboxc
885   \wd\tw@ \wd\@smashboxc
886   \box\if@visible\@smashboxc\else\tw@\fi}
887
888 \def\r@t#1#2{\setbox\z@\hbox{\$m@th#1\@xysqrt#1{#2}$}%
889   \dimen@ht\z@ \advance\dimen@-\dp\z@
890   \mskip5mu\raise.6\dimen@\copy\rootbox \mskip-10mu\box\z@}
891 \def\sqrt{\@ifnextchar[{\@sqrt}{\@xsqrt}}
892 \def\@sqrt[#1]{\root #1\of}
893 \def\@xsqrt#1{\mathchoice{\@xysqrt\displaystyle{#1}}{\@xysqrt
894   \textstyle{#1}}{\@xysqrt\scriptstyle{#1}}{\@xysqrt
895   \scriptscriptstyle{#1}}}
896 \def\@xysqrt#1#2{\@mathbox#1\@smashboxa{#2}\if@visible
897   \@hvsplash#1{\sqrtsign{\@bphantom\@smashboxa}}\fi
898   \phantom{\sqrtsign{\@vbphantom\@smashboxa}}\box\@smashboxa}
899
900 \newbox\@smashboxa
901 \newbox\@smashboxb
902 \newbox\@smashboxc

array and tabular environments: changes to '|', \hline, \cline, and \vline,
added 8 Jun 88
903 \def\@arrayrule{\if@visible\@addtopreamble{\hskip -.5\arrayrulewidth
904   \vrule \@width \arrayrulewidth\hskip -.5\arrayrulewidth}\fi}
905 \def\@cline#1{\if@visible\@cline#1\@nil\fi}
906
907 \def\@hline{\noalign{\ifnum0='}\fi
908   \if@visible \hrule \@height \arrayrulewidth
909   \else \hrule \@width \z@
910   \fi
911   \futurelet \reserved@a\@xhline}
912
913 \def\@vline{\if@visible \vrule \@width \arrayrulewidth
914   \else \vrule \@width \arrayrulewidth \@height \z@
915   \@depth \z@ \fi}
916 \message{output,}

```

### 10.3.3 Changes to L<sup>A</sup>T<sub>E</sub>X 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.

```

917 \let\@topfil\vfil
918
919 \def\@makecol{\if@makingslides\ifnum\c@page>\z@ \@extraside\fi\fi
920 \ifvoid\footins \setbox\@outputbox\box\@cclv \let\@botfil\vfil
921 \else\let\@botfil\relax\setbox\@outputbox
922 \vbox{\unvbox\@cclv\vfil
923 \vskip\skip\footins\footnoterule\unvbox\footins\vskip
924 \z@ plus.1fil\relax}\fi
925 \xdef\@freelist{\@freelist\@midlist}\gdef\@midlist{}\@combinefloats
926 \setbox\@outputbox\vbox to\@colht{\boxmaxdepth\maxdepth
927 \@topfil\unvbox\@outputbox\@botfil}\global\maxdepth\@maxdepth}
928
929 \def\@extraside{\ifnum\c@note>\z@
930 \ClassWarning{slides}{Note \thenote\space too long}\else
931 \ifnum\c@overlay>\z@
932 \ClassWarning{slides}{Overlay \theoverlay\space too long}\else
933 \ClassWarning{slides}{Slide \theslide\space too long}\fi\fi}
934 \message{init}

```

### 10.3.4 Special L<sup>A</sup>T<sub>E</sub>X initializations

FMi why not allow for ref's ?

```
935 % \nofiles
936
937 \@visibletrue
938 </cmd>
```