

A L^AT_EX Package for changing the page grid and MVL ^{*†}

Arthur Ogawa [‡]

October 9, 2009

This file embodies the `ltxgrid` package, the implementation and its user documentation.

The distribution point for this work is publish.aps.org/revtex, which contains the REVTeX package, and includes source and documentation for this package.

The `ltxgrid` package was commissioned by the American Physical Society and is distributed under the terms of the L^AT_EX Project Public License, the same license under which all the portions of L^AT_EX itself is distributed. Please see <http://ctan.tug.org/macros/latex/base/lppl.txt> for details.

To use this document class, you must have a working T_EX installation equipped with L^AT_EX 2_ε and possibly pdf_{te}x and Adobe Acrobat Reader or equivalent.

To install, retrieve the distribution, unpack it into a directory on the target computer, into a location in your filesystem where it will be found by L^AT_EX; in a TDS-compliant installation this would be: `texmf/tex/macros/latex/revtex/`.

To use, read the user documentation `src/ltxgrid.pdf`.

Contents

1	Processing Instructions	3
1.1	Build Instructions	3
1.2	Change Log	4
1.3	Bill of Materials	4
1.3.1	Primary Source	4
1.3.2	Generated by <code>latex ltxgrid.dtx</code>	4
1.3.3	Generated by <code>tex ltxgrid.ins</code>	4
1.3.4	Documentation	4
1.3.5	Auxiliary	4
2	Code common to all modules	5

*This file has version number 4.1h, last revised 2009/10/09.

†Version 4.1h © 2009 The American Physical Society

‡mailto:arthur_ogawa@sbcglobal.net

3	The driver module doc	5
3.1	The Preamble	5
3.1.1	Docstrip and info directives	6
3.2	The “Read Me” File	6
3.3	The Document Body	9
4	Using this package	9
4.1	Invoking the package	9
4.2	Changing the page grid	10
4.3	Changing the MVL	10
5	Compatibility with L^AT_EX’s Required Packages	12
5.1	ftnright	12
5.2	longtable	13
5.3	multicol	13
5.4	ltxgrid	14
6	How ltxgrid places footnotes	14
7	Limitations in ltxgrid’s default column balancing method	14
8	Implementation of package	15
8.1	Beginning of the ltxgrid DOCSTRIP module	15
8.2	Banner	15
8.3	Sundry	16
8.4	Mark Components	16
8.4.1	Procedures that expose the component data structure	17
8.4.2	Procedures that do not expose the component data structure	17
8.4.3	Using mark components	18
8.5	Output Super-routine	19
8.6	Further thoughts about inserts	24
8.7	The difference between inserts and floats	25
8.8	The natural output routine	25
8.9	Natural output routine	26
8.10	Float placement	35
8.11	Clearing pages	42
8.12	Other interfaces to L ^A T _E X	46
8.13	One-off output routines	54
8.14	Output messages	57
8.15	Messages to alter the page grid	61
8.16	Application Note: implementing a page grid	63
8.16.1	One-column page grid	64
8.16.2	Two-column page grid	66
8.16.3	Page grid utility procedures	69
8.17	Patches for the longtable package	80
8.18	Patches for index processing	87

8.19	Checking the auxiliary file	87
8.20	Dealing with stuck floats and stalled float dequeuing	87
9	Support for legacy L^AT_EX commands	91
9.0.1	Building the page for shipout	92
9.0.2	Warning message	93
10	Line-wise processing	93
11	Patching the lineno.sty package	101
12	End of the ltxgrid DOCSTRIP module	106
	Index	107

1 Processing Instructions

The package file `ltxgrid.sty` is generated from this file, `ltxgrid.dtx`, using the DOCSTRIP facility of L^AT_EX via `tex ltxgrid.dtx`. The typeset documentation that you are now reading is generated from this same file by typesetting it with L^AT_EX or pdf_TE_X via `latex ltxgrid.dtx` or `pdflatex ltxgrid.dtx`.

1.1 Build Instructions

You may bootstrap this suite of files solely from `ltxgrid.dtx`. Prepare by installing L^AT_EX 2_ε (and either `tex` or `pdfTEX`) on your computer, then carry out the following steps:

1. Within an otherwise empty directory, typeset `ltxgrid.dtx` with T_EX or pdf_TE_X; thereby generating the package file `ltxgrid.sty`.
2. Now typeset `ltxgrid.dtx` with L^AT_EX or pdf_TE_X; you will obtain the typeset documentation you are now reading, along with the file `00readme`.

Note: you will have to run L^AT_EX twice, then `makeindex`, then L^AT_EX again in order to obtain a valid index and table of contents.

3. Install the following files into indicated locations within your TDS-compliant `texmf` tree (you may need root access):

- `$TEXMF/tex/latex/revtex/ltxgrid.sty`
- `$TEXMF/source/latex/revtex/ltxgrid.dtx`
- `$TEXMF/doc/latex/revtex/ltxgrid.pdf`

where `TEXMF/` stands for `texmf-local/`, or some other `texmf` tree in your installation.

4. Run `mktexlsr` on directory `$TEXMF/` (you may need root access).

5. Build and installation are now complete; now put a `\usepackage{ltxgrid}` in your document preamble! (Note: `ltxgrid` requires package `ltxutil`.)

1.2 Change Log

1.3 Bill of Materials

Following is a list of the files in this distribution arranged according to provenance.

1.3.1 Primary Source

One single file generates all.

```
%ltxgrid.dtx
%
```

1.3.2 Generated by latex ltxgrid.dtx

Typesetting the source file under \LaTeX generates the readme and the installer.

```
%00readme ltxgrid.ins
%
```

1.3.3 Generated by tex ltxgrid.ins

Typesetting the installer generates the package files.

```
%ltxgrid.sty
%
```

1.3.4 Documentation

The following are the online documentation:

```
%ltxgrid.pdf
%
```

1.3.5 Auxiliary

The following are auxiliary files generated in the course of running \LaTeX :

```
%ltxgrid.aux ltxgrid.idx ltxgrid.ind ltxgrid.log ltxgrid.toc
%
```

2 Code common to all modules

The following may look a bit kloohty, but we want to require only one place in this file where the version number is stated, and we also want to ensure that the version number is embedded into every generated file.

Now we declare that these files can only be used with L^AT_EX 2_ε. An appropriate message is displayed if a different T_EX format is used.

```
1 %<*driver|package>
2 \NeedsTeXFormat{LaTeX2e}[1995/12/01]%
3 %</driver|package>
```

As desired, the following modules all take common version information:

```
4 %<package>\ProvidesFile{ltxgrid.sty}%
5 %<*driver>
6 \expandafter\ProvidesFile\expandafter{\jobname.dtx}%
7 %</driver>
```

The following line contains, for once and for all, the version and date information. By various means, this information is reproduced consistently in all generated files and in the typeset documentation.

```
8 %<*driver|package>
9 [2009/10/09 4.1h page grid package]% \fileversion
10 %</driver|package>
```

3 The driver module doc

This module, consisting of the present section, typesets the programmer's documentation, generating the `.ins` installer and `00readme` as required.

Because the only uncommented-out lines of code at the beginning of this file constitute the `doc` module itself, we can simply typeset the `.dtx` file directly, and there is thus rarely any need to generate the “doc” DOCSTRIP module. Module delimiters are nonetheless required so that this code does not find its way into the other modules.

The `\end{document}` command concludes the typesetting run.

```
11 %<*driver>
```

3.1 The Preamble

The programmers documentation is formatted with the `ltxdoc` class with local customizations, and with the usual code line indexing.

```
12 \documentclass{ltxdoc}
13 \RequirePackage{ltxdocext}%
14 \RequirePackage[colorlinks=true,linkcolor=blue]{hyperref}%
15 \ifx\package@font\@undefined\else
16 \expandafter\expandafter
17 \expandafter\RequirePackage
18 \expandafter\expandafter
```

```

19 \expandafter{%
20         \csname package@font\endcsname
21     }%
22 \fi
23 \CodelineIndex\EnableCrossrefs % makeindex -s gind.ist ltxgrid
24 \RecordChanges % makeindex -s gglo.ist -o ltxgrid.gls ltxgrid.glo

```

3.1.1 Docstrip and info directives

We use so many DOCSTRIP modules that we set the `StandardModuleDepth` counter to 1.

```
25 \setcounter{StandardModuleDepth}{1}
```

The following command retrieves the date and version information from this file.

```
26 \expandafter\GetFileInfo\expandafter{\jobname.dtx}%

```

3.2 The “Read Me” File

As promised above, here is the contents of the “Read Me” file. That file serves a double purpose, since it also constitutes the beginning of the programmer’s documentation. What better thing, after all, to have appear at the beginning of the typeset documentation?

A good discussion of how to write a ReadMe file can be found in Engst, Tonya, “Writing a ReadMe File? Read This” *MacTech* October 1998, p. 58.

Note the appearance of the `\StopEventually` command, which marks the dividing line between the user documentation and the programmer documentation.

The usual user will not be asked to do a full build, not to speak of the bootstrap. Instructions for carrying these processes begin the programmer’s manual.

```

27 \begin{filecontents*}{00readme}
28 \title{%
29 A \LaTeX\ Package for changing the page grid and MVL%
30 \thanks{%
31 This file has version number \fileversion,
32 last revised \filedate.%
33 }%
34 \thanks{%
35 Version \fileversion\ \copyright\ 2009 The American Physical Society
36 }%
37 }%
38 \author{%
39 Arthur Ogawa%
40 \thanks{\texttt{mailto:arthur\_ogawa at sbcglobal.net}}%
41 }%
42 %\iffalse
43 % For version number and date,
44 % search on "\fileversion" in the .dtx file,
45 % or see the end of the 00readme file.
46 %\fi

```

47 \maketitle
48
49 This file embodies the \classname{ltxgrid} package,
50 the implementation and its user documentation.
51
52 The distribution point for this work is
53 \url{publish.aps.org/revtex},
54 which contains the REV\TeX\ package, and includes source and documentation for this package.
55
56 The \classname{ltxgrid} package was commissioned by the American Physical Society
57 and is distributed under the terms of the \LaTeX\ Project Public License,
58 the same license under which all the portions of \LaTeX\ itself is distributed.
59 Please see \url{http://ctan.tug.org/macros/latex/base/lppl.txt} for details.
60
61 To use this document class, you must have a working
62 \TeX\ installation equipped with \LaTeXe\
63 and possibly pdftex and Adobe Acrobat Reader or equivalent.
64
65 To install, retrieve the distribution,
66 unpack it into a directory on the target computer,
67 into a location in your filesystem where it will be found by \LaTeX;
68 in a TDS-compliant installation this would be:
69 \file{texmf/tex/macros/latex/revtex/.}
70
71 To use, read the user documentation \file{src/ltxgrid.pdf}.
72
73 \tableofcontents
74
75 \section{Processing Instructions}
76
77 The package file \file{ltxgrid.sty}
78 is generated from this file, \file{ltxgrid.dtx},
79 using the {\sc docstrip} facility of \LaTeX
80 via |tex ltxgrid.dtx|.
81 The typeset documentation that you are now reading is generated from
82 this same file by typesetting it with \LaTeX\ or pdftex
83 via |latex ltxgrid.dtx| or |pdflatex ltxgrid.dtx|.
84
85 \subsection{Build Instructions}
86
87 You may bootstrap this suite of files solely from \file{ltxgrid.dtx}.
88 Prepare by installing \LaTeXe\ (and either tex or pdftex) on your computer,
89 then carry out the following steps:
90 \begin{enumerate}
91 \item
92 Within an otherwise empty directory,
93 typeset \file{ltxgrid.dtx} with \TeX\ or pdftex;
94 thereby generating the package file \file{ltxgrid.sty}.
95
96 \item

```

97 Now typeset \file{ltxgrid.dtx} with \LaTeX\ or pdflatex;
98 you will obtain the typeset documentation you are now reading,
99 along with the file \file{00readme}.
100
101 Note: you will have to run \LaTeX\ twice, then \file{makeindex}, then
102 \LaTeX\ again in order to obtain a valid index and table of contents.
103
104 \item
105 Install the following files into indicated locations within your
106 TDS-compliant \texttt{texmf} tree (you may need root access):
107 \begin{itemize}
108 \item
109 \file{$TEXMF/}\file{tex/}\file{latex/}\file{revtex/}\classname{ltxgrid.sty}
110 \item
111 \file{$TEXMF/}\file{source/}\file{latex/}\file{revtex/}\classname{ltxgrid.dtx}
112 \item
113 \file{$TEXMF/}\file{doc/}\file{latex/}\file{revtex/}\classname{ltxgrid.pdf}
114 \end{itemize}
115 where \file{TEXMF/} stands for \file{texmf-local/}, or some other \texttt{texmf} tree in your i
116 \item
117 Run \texttt{mktexlsr} on directory \file{$TEXMF/} (you may need root access).
118 \item
119 Build and installation are now complete;
120 now put a \cmd\usepackage\texttt{\{ltxgrid\}} in your document preamble!
121 (Note: \texttt{ltxgrid} requires package \texttt{ltxutil}.)
122 \end{enumerate}
123
124 \subsection{Change Log}
125 \changes{4.0a}{2001/06/18}{Introduce \cs{marry@height} }
126 \changes{4.0a}{2001/06/18}{Introduce \cs{set@marry@height} }
127 \changes{4.0a}{2008/06/26 }{\cs{@yfloat}: de-fang \cs{set@footnotewidth} (see ltxutil.dtx): we
128 \changes{4.1a}{2008/06/29}{Change \cs{LT@array@new}: restore \cs{@tabularcr} and \cs{@tabularc
129 \changes{4.1a}{2008/06/29}{Change \cs{LT@array@new}: set \cs{LT@LL@FM@cr} to \cs{@arraycr@array
130 \changes{4.1a}{2008/06/29}{Repair error in \cs{endlongtable@new} involving \cs{@ifx}: argument
131 \changes{4.1b}{2008/08/04}{Get rid of the \cs{reserved@a} idiom}
132 \changes{4.1b}{2008/08/04}{Turn off the \cs{set@footnotewidth} mechanism; a float 'knows' its p
133 \changes{4.1b}{2008/08/04}{(A0, 452) Support length checking: show size of shipped out text.}
134 \changes{4.1b}{2008/08/04}{(A0, 456) Compatibility with other packages that override the output
135 \changes{4.1b}{2008/08/04}{}
136 \changes{4.1b}{2008/08/04}{Box \cs{footbox} changed to box \cs{footsofar}}
137 \changes{4.1b}{2008/08/04}{Change \cs{@combinepage} to \cs{@combinepage} with argument}
138 \changes{4.1b}{2008/08/04}{Change \cs{@makecol} to \cs{@makecolumn} with argument}
139 \changes{4.1b}{2008/08/04}{Change \cs{set@colroom} to \cs{set@colht}}
140 \changes{4.1b}{2008/08/04}{New procedure \cs{@iffpsbit} replaces \cs{@getfpsbit}}
141 \changes{4.1b}{2008/08/04}{New procedure \cs{@output@combined@page}}
142 \changes{4.1b}{2008/08/04}{New procedure for showing a box contents, \cs{trace@box}}
143 \changes{4.1b}{2008/08/04}{Procedure \cs{@outputpage@head} headpatches \cs{@outputpage}}%
144 \changes{4.1b}{2008/08/04}{Procedure \cs{@outputpage@tail} tailpatches \cs{@outputpage}}%
145 \changes{4.1b}{2008/08/04}{Procedure \cs{balance@2} defined more transparently}%
146 \changes{4.1b}{2008/08/04}{Procedure \cs{set@adj@footins} to adjust for footnotes and other ins

```



```

147 \changes{4.1b}{2008/08/04}{Tally the height of the float}
148 \changes{4.1b}{2008/08/04}{Use \cs{document@inithook} instead of \cs{AtBeginDocument}}
149 \changes{4.1b}{2008/08/04}{Use \cs{trace@box} instead of \cs{showbox}}
150 \changes{4.1f}{2009/07/07}{(AO, 515) Prevent line numbering within a footnote}
151 \changes{4.1f}{2009/07/10}{(AO, 518) Tally register overflow when locument is long}
152 \changes{4.1f}{2009/07/14}{(AO, 519) \cs{footins} content must be preserved and reintegrated}
153 \changes{4.1f}{2009/07/14}{(AO, 519) Do not use \cs{dimen@} register as a scratch register in \
154 \changes{4.1f}{2009/07/15}{(AO, 519) Preserve footnotes that are in \cs{footsofar} across a pag
155 \changes{4.1g}{2009/10/06}{(AO, 531) Fix package \classname{float} }
156
157 \end{filecontents*}

```

3.3 The Document Body

Here is the document body, containing only a `\DocInput` directive—referring to this very file. This very cute self-reference is a common `ltxdoc` idiom.

```

158 \begin{document}%
159 \def\revtex{REV\TeX}%
160 \expandafter\DocInput\expandafter{\jobname.dtx}%
161 \end{document}
162 %</driver>

```

4 Using this package

Once this package is installed on your filesystem, you can employ it in adding functionality to \LaTeX by invoking it in your document or document class.

4.1 Invoking the package

In your document, you can simply call it up in your preamble:

```

%\documentclass{book}%
%\usepackage{ltxgrid}%
%\begin{document}
%your document here
%\end{document}
%

```

However, the preferred way is to invoke this package from within your customized document class:

```

%\NeedsTeXFormat{LaTeX2e}[1995/12/01]%
%\ProvidesClass{myclass}%
%\LoadClass{book}%
%\RequirePackage{ltxgrid}%
%class customization commands
%\endinput
%

```

Note that this package requires the features of the `ltxutil` package, available at publish.aps.org/revtex.

Once loaded, the package gives you access to certain procedures, usually to be invoked by a \LaTeX command or environment, but not at the document level.

4.2 Changing the page grid

This package provides two procedures, `\onecolumngrid`, `\twocolumngrid`, that change the page grid (it can be extended to more columns and to other page grids).

They differ from standard \LaTeX 's `\onecolumn` and `\twocolumn` commands in that they do not force a page break. Also, upon leaving a multiple-column grid, the columns are balanced. In other respects they work same.

They differ from the grid-changing commands of Frank Mittelbach's `multicol` package in that they allow floats of all types (single- and double column floats, that is) and preserve compatibility with the `longtable` package.

These commands must be issued in vertical mode (conceivably via a `\vadjust`) such that they are ultimately present in the MVL, where they can do their work. Because they do not work in \LaTeX 's left-right mode, they are unsuitable at the document level. Furthermore, packaging a grid command in a `\vadjust`, although possible, will probably not achieve satisfactory page layout.

Page grid commands are not intended to be issued unnecessarily: only the first of two successive `\onecolumngrid` commands is effective; the second will be silently ignored.

`\onecolumngrid` You command \LaTeX to return to the one-column grid with the `\onecolumngrid` command. If you are already in the one-column grid, this is a no-op. The one-column grid is considered special of all page grids, in that no portion of the page is held back (in `\pagesofar`); all items that might go on the current page (with the exception of floats and footnotes) are on the MVL.

`\twocolumngrid` You command \LaTeX to return to the two-column grid with the `\twocolumngrid` command. If you are already in the two-column grid, this is a no-op.

These two commands should be issued by a macro procedure that can ensure that \TeX is in outer vertical mode.

4.3 Changing the MVL

This package also provides commands to modify the main vertical list (MVL) in a safe way. The scheme here is to structure, insofar possible, \TeX 's MVL as follows:

```
    box or boxes
    penalty
    glue
```

This should be a familiar sequence. It is the prototype sequence for a vertical list, and is followed when \TeX breaks paragraphs into lines, and when \TeX generates a display math equation.

If you (as a macro programmer) wish to modify the value of the penalty or glue item, you can use one of the MVL-altering commands to do so. Certain operations are implemented here; you can make up your own.

Note that these commands must be issued in vertical mode, perhaps via a `\vadjust` or a `\noalign`. They can work directly if you are in inner mode (say within a parbox or a minipage).

`\removestuff` You instruct L^AT_EX to remove both the penalty and the glue item with this command.

`\addstuff` You issue the `\addstuff{<penalty>}{<glue>}` command to add a penalty, glue, or both. If you do not wish to add one or the other, the corresponding argument should be nil. Note that the effect of `\addstuff` is to stack the penalties and glue items. Therefore, the lesser of the two penalties takes effect, and the two glue items add together.

`\addstuff` is limited because once applied, it cannot be applied again with correct results.

`\replacestuff` The `\replacestuff` command is syntactically the same as `\addstuff`, but works differently: the existing penalty and glue are replaced or modified.

The specified penalty is not inserted if the existing penalty is greater than 10000 (that is, in case of a `\nobreak`), otherwise, the lower (non-zero) of the two penalties is inserted.

If the specified glue has a larger natural component than the existing glue, we replace the glue. However, if the specified glue's natural component is negative, then the existing glue's natural component is changed by that amount.

`\replacestuff` can be applied multiple times because it retains the list structure in the canonical form.

Note that we treat two penalties specially (as does T_EX): a penalty of 10000 is considered a garbage value, to be replaced if found. This is the signal value that T_EX inserts on the MVL replacing the penalty that caused the page break (if the page break occurred at a penalty). Also, a penalty of zero is indistinguishable from no penalty at all, so it will always be replaced by the given value.

Therefore, it is highly recommended to never set any of T_EX's penalty parameters to zero (a value of, say, 1, is practically the same), nor should a skip parameter be set to zero (instead, use, say, 1sp). Also, to prevent a pagebreak, do not use a penalty of 10000, use, say 10001 instead.

You can define your own construct that modifies the MVL: Define a command, say, `\myadjust`, as follows:

```
%\def\myadjust#1{\noexpand\do@main@vlist{\noexpand\@myadjust{#1}}\@tempa}%
%
```

that is, `\myadjust` invokes `\do@main@vlist`, passing it the procedure name `\@myadjust` along with the arguments thereof pre-expanded. Next, define the procedure `\@myadjust`:

```
%\def\@myadjust#1{<meddle with the MVL>}%
%
```

when `\@myadjust` executes, you will be in the output routine (in inner vertical mode) and the MVL will be that very vertical list.

5 Compatability with L^AT_EX's Required Packages

Certain packages, usually ones written by members of the L^AT_EX Project itself, have been designated “required” and are distributed as part of standard L^AT_EX. These packages have been placed in a privileged position vis á vis the L^AT_EX kernel in that they override the definitions of certain kernel macros.

Compatability between `ltxgrid` and these packages is complicated by a number of factors. First is that `ltxgrid` alters the meaning of some of the same kernel macros as certain of the “required” packages. Second is that fact that certain of the “required” packages of L^AT_EX are incompatible with each other.

Examples of the first kind are the `ftnright`, `multicol`, and `longtable` packages. The `ltxgrid` package is not compatible with `multicol`, but if you are using `ltxgrid`, you do not need to use `ftnright` or `multicol` anyway. The `ltxgrid` package does however attempt to be compatible with `longtable`.

Among the “required” packages that are mutually incompatible are `multicol` and `longtable`, the incompatibility arising because both packages replace L^AT_EX's output routine: if one package is active, the other must not be so. This state of affairs has remained essentially unchanged since the introduction of the two as L^AT_EX2.09 packages in the late 1980s.

The reason that `ltxgrid` can remain compatible with `longtable` is due to the introduction of a more modern architecture, the “output routine dispatcher”, which allows all macro packages access to the safe processing environment of the output routine, on an equal footing. The relevant portions of the `longtable` package are reimplemented in `ltxgrid` to take advantage of this mechanism.

Timing is critical: the `ltxgrid` package will be incompatible with any package that redefines any of the kernel macros that `ltxgrid` patches—if that package is loaded *after* `ltxgrid`.

Hereinafter follows some notes on specific L^AT_EX packages.

5.1 `ftnright`

Frank Mittelbach's `ftnright` package effects a change to L^AT_EX's `\twocolumn` mode such that footnotes are set at the bottom of the right-hand column instead of at the foot of each of the two columns.

Note that it overwrites three L^AT_EX kernel macros: `\@outputdblcol`, `\@startcolumn`, and `\@makecolumn`. Fortunately none of the three are patched by `ltxgrid`, so that compatability is not excluded on this basis.

At the same time, it changes the meaning of `\footnotesize`, the macro that is automatically invoked when setting a document's footnote into type. One might well argue that it is an error for the meaning of `\footnotesize` to be determined by a package such as `ftnright`, that indeed such a choice should be made in the document class, or in a file such as `bk10.c1o`.

To avoid being tripped up by this misfeature in `ftnright`, it is only necessary to reassert our meaning for `\footnotesize` later on, after `ftnright` has been loaded.

Note that `ftnright` inserts code that demands that L^AT_EX's flag `\if@twocolumn` is true, that is, it will complain if deployed in a `\onecolumn` document. It is therefore necessary for any other multicolumn package to assert that flag in order to avoid this package's complaint. It is an interesting question exactly why this package has this limitation. After all, a one-column page grid is just a degenerate case of the two column.

5.2 `longtable`

David Carlisle's `longtable` package sets tables that can be so long as to break over pages. According to its author, it uses the same override of L^AT_EX's output routine as Frank Mittelbach's `multicol` package. By implication, then, it has a hard incompatibility with the latter.

The `longtable` package also performs a check of whether the document is in `\twocolumn` mode, and declines to work if this is the case. It is not clear, however, that there is any true incompatibility present if so. It's just that David did not see any reason anyone would want to set such long tables in a multicolumn document, hence the check.

There does not appear to be any indication that `longtable` would work less well under `ltxgrid` than under standard L^AT_EX's `\twocolumn` mode. Therefore, this `ltxgrid` patches `longtable` (if loaded) so as to provide compatibility. In the course of which, `longtable` becomes more robust (`longtable` has numerous bugs and incompatibilities of long standing, some of which are repaired by `ltxgrid`).

One problem remains, namely that, if a `longtable` environment breaks over columns and thereby inserts its special headers and footers at that break, and those columns are then balanced (due to a return to the one-column page grid), then those inserted rows will remain, and may no longer fall at the column break. This will, of course look wrong.

The only way to fix this problem is to avoid doing column balancing in the way I have implemented here; such an enhancement to this package is possible.

5.3 `multicol`

Frank Mittelbach's `multicol` package provides a page grid with many columns, albeit denies the placement of floats in individual columns.

It establishes its own `\output` routine, which is the reason it runs afoul of the `longtable` package. On the other hand, `ltxgrid` specifically allows for the case where a package installs its own `\output` routine, so there is no incompatibility on that basis.

Still, it is pointless to use `multicol` if you are using `ltxgrid`, since both packages provide multicolumn page layouts. Therefore, `multicol` is not supported by `ltxgrid`.

5.4 ltxgrid

It has been pointed out that one of the disadvantages of adopting the `ltxgrid` package is that it does alter the \LaTeX kernel. Any package that itself alters the \LaTeX kernel may be incompatible with `ltxgrid`, and new packages (destined perhaps to become part of the successor to $\text{\LaTeX} 2_{\epsilon}$) may break `ltxgrid`.

The consequence is that packages introduced in future, and future changes to \LaTeX may be incompatible with `ltxgrid`. This is, of course, true. The development plan for `ltxgrid` is that when such packages and \LaTeX kernel changes come about, the burden will be on `ltxgrid` to change in a way that provides for continued compatibility with those packages and \LaTeX kernel changes.

6 How ltxgrid places footnotes

In conventional multicolumn layouts, a footnote will appear at the bottom of the column in which it is called out. The `ltxgrid` package implements this conventional layout choice by default. However, other choices are possible (a la `ftnright`, whose compatibility with `ltxgrid` has not been tested).

One unusual feature of `ltxgrid`'s default implementation must be mentioned, though, namely the case in a two-column page grid, where a footnote is followed by a temporary change to the one-column page grid (e.g., for a wide equation). In such a case, the material above the wide material is split into two columns, and a footnote whose callout appears in the right-hand column will nonetheless be set at the base of the left column.

This arrangement was chosen because it ensures that the footnotes at the bottom of any page will appear in numerical order. It can be argued that this choice is “incorrect”, but be that as it may, the `ltxgrid` package does not foreclose on other arrangements for the footnotes. The package can be adapted to accomodate any page design desired.

7 Limitations in ltxgrid's default column balancing method

In a multicolumn page grid, when encountering a page that is not completely full, it is customary to set the material in balanced columns (typically with the last column no longer than any of the others). Such a case also crops up when temporarily interrupting the multicolumn grid to set material on the full width of the page: the material on the page above the break is customarily set in balanced columns.

An awkward case arises when we have already set one or more complete columns of type before encountering the need to balance columns. In this subset of cases, the default in `ltxgrid` is to do an operation I call “re-balancing”: the material on the page so far is pasted back together into a single column, and new, balanced column breaks are calculated.

This scheme typically works fine, but it has a significant vulnerability: any discardable items trimmed at the original column break are lost, never to be retrieved. Consequently, after re-balancing, an element like, say, a section head can fail to have the correct amount of whitespace above.

This problem is due to an unfortunate optimization in \TeX , wherein a certain class of nodes is trimmed from the top of main vertical list upon returning from the output routine: any penalty, glue, or leader node falls in to this class of discardable nodes, and trimming proceeds until a non-discardable node (such as a box, or rule) is encountered. It gets better: a third class of nodes is transparent to this trimming process; they are neither discarded nor do they halt the process of trimming: mark nodes and all whatsits fall into this class of transparent nodes; they are quietly passed over during trimming.

An alternative approach for \TeX to take would have been, rather than discarding the node entirely, to simply *mark* it as discarded. (Implementors of extended \TeX , please note!) Then, upon shipping out, such nodes would not make it into the DVI. \TeX 's optimization, driven by the small computer architectures current when it was developed, does save mem, but at the cost of revisiting page breaks in a reliable way.

FIXME: how to fix a column break in the above case? Widetext?

8 Implementation of package

Special acknowledgment: this package uses concepts pioneered and first realized by William Baxter (mailto:web at superscript.com) in his SuperScript line of commercial typesetting tools, and which are used here with his permission. His thorough understanding of \TeX 's output routine underpins the entire `ltxgrid` package.

8.1 Beginning of the `ltxgrid` DOCSTRIP module

Requires the underpinnings of the `ltxkrnext` package.

```

163 %<*package>
164 \def\package@name{ltxgrid}%
165 \expandafter\PackageInfo\expandafter{\package@name}{%
166 Page grid for \protect\LaTeXe,
167 by A. Ugawa (arthur_ogawa at sbcglobal.net)%
168 }%
169 \RequirePackage{ltxutil}%
170 %</package>

```

8.2 Banner

Credit where due.

```

171 %<*kernel>
172 \typeout{%
173 ltxgrid [2009/07/07 4.1f]: portions licensed from W. E. Baxter (web at superscript.com)%
174 }%

```

8.3 Sundry

Here are assorted macro definitions.

`\lineloop` The document-level command `\lineloop` sets numbered lines until the specified count is reached. This command is mainly used to construct test documents.

Because the counter is globally advanced and never reset, successive calls to `\lineloop` should have an argument ever larger. The formatted output will have each line labeled with its ordinal number.

```
175 \newcounter{linecount}
176 \def\loop@line#1#2{%
177 \par
178 \hb@xt@\hszize{%
179 \global\advance#1\@ne
180 \edef\@tempa{\@ifnum{100>#1}{0}{}\@ifnum{10>#1}{0}{}\number#1}%
181 \@tempa\edef\@tempa{\special{trace:\@tempa}}\@tempa
182 \vrule depth2.5\p@#2\leaders\hrule\hfil
183 }%
184 }%
185 \def\lineloop#1{%
186 \loopwhile{\loop@line\c@linecount}\@ifnum{#1>\c@linecount}}%
187 }%
```

8.4 Mark Components

Override LaTeX's mark macros to allow more components.

We remain bound by the weakness of LaTeX's scheme in that one cannot emulate the action of `TEX` whereby material with marks can be inserted in the middle of a vertical list such that the marks are reliably calculated. If we did that, `\themark` would no longer be utilized.

A more robust scheme involves placing all marks (component and value) into a list (using global scoping, i.e., `\gdef`), and using `\@@markto` to place an index on that list into the MVL. Then, e.g., `\@@botmark` signifies the place where that list is to be cut, and the `\botmark` of any component is the value of the last element of the cut list having the given component. The `\firstmark` and `\topmark` can likewise be defined relative to `\@@firstmark` and `\@@topmark`, except in the latter case, we want the first following the cut instead of the last preceding the cut.

The limitation of this scheme is its demands upon `TEX`'s mem. The list of marks would need to be trimmed back to, effectively, `\topmark` at the beginning of every page.

This approach is not yet part of the extended LaTeX kernel.

```
\@@mark Remember primitives under a new set of names.
\@@topmark 188 \let\@@mark\mark
\@@firstmark 189 \let\@@topmark\topmark
\@@botmark 190 \let\@@firstmark\firstmark
\@@splitfirstmark 191 \let\@@botmark\botmark
\@@splitbotmark 192 \let\@@splitfirstmark\splitfirstmark
```



```
193 \let\@@splitbotmark\splitbotmark
```

8.4.1 Procedures that expose the component data structure

This portion of the code exposes the internal representation of the mark components. If we wish to add more components, we will have to revise these macro definitions: `\@themark`, `\nul@mark`, `\set@mark@netw@`, `\set@marktw@`, `\set@markthr@@`, `\get@mark@one`, `\get@mark@tw@`, `\get@mark@thr@@`, `\get@mark@four@`.

`\@themark` FIXME: is it safer to eliminate `\@themark` in favor of a message that evaluates `\@@botmark?`

Note: these definitions expose the data structure of mark components.

```
194 \def\@themark{}{}{}{}%
195 \def\nul@mark{}{}{}{}{\@nul}%
```

`\set@mark@netw@` These procedures insert the new value of a particular mark component into the given argument. They expose the data structure of mark components.

`\set@marktw@`

`\set@markthr@@`

```
196 \def\set@mark@netw@#1#2#3#4#5#6#7{\gdef#1{#6}{#7}{#4}{#5}}\do@mark}%
197 \def\set@marktw@#1#2#3#4#5#6{\gdef#1{#2}{#6}{#4}{#5}}\do@mark}%
198 \def\set@markthr@@#1#2#3#4#5#6{\gdef#1{#2}{#3}{#6}{#5}}\do@mark}%
```

`\get@mark@one` These procedures retrieve the value of a particular mark component. They expose the data structure of mark components.

`\get@mark@tw@`

`\get@mark@thr@@`

`\get@mark@four@`

```
199 \def\get@mark@one#1#2#3#4#5\@nul{#1}%
200 \def\get@mark@tw@#1#2#3#4#5\@nul{#2}%
201 \def\get@mark@thr@@#1#2#3#4#5\@nul{#3}%
202 \def\get@mark@four@#1#2#3#4#5\@nul{#4}%
```

8.4.2 Procedures that do not expose the component data structure

`\mark@netw@` These procedures insert the new value of a particular mark component into `\@themark`, then execute `\do@mark`. They constitute the implementation layer

`\marktw@` for mark components one, two, and three. An analogous procedure for component

`\markthr@@` four could be defined; call it `\markfour@`.

```
203 \def\mark@netw@{\expandafter\set@mark@netw@\expandafter\@themark\@themark}%
204 \def\marktw@{\expandafter\set@marktw@\expandafter\@themark\@themark}%
205 \def\markthr@@{\expandafter\set@markthr@@\expandafter\@themark\@themark}%
```

`\do@mark` Access procedures `\mark`(AKA `\@@mark`). The `\do@mark` procedure is used when

`\do@@mark` a mark is being put down into the MVL; `\do@@mark` when this happens in the output routine.

```
206 \def\do@mark{\do@@mark\@themark\nobreak@mark}%
207 \def\do@@mark#1{%
208   \begin@group
209   \let@mark
210   \@@mark{#1}%
211   \end@group
212 }%
```

`\let@mark` The procedure that makes `\csnames` robust within a mark. Use `\appdef` and `\nobreak@mark` `\robust@` to extend the list.

```

213 \def\let@mark{%
214 \let\protect\@unexpandable@protect
215 \let\label\relax
216 \let\index\relax
217 \let\glossary\relax
218 }%
219 \def\nobreak@mark{%
220 \@if@sw@if@nobreak\fi{\@ifvmode{\nobreak}{}}{}}%
221 }%

```

8.4.3 Using mark components

These procedures use the component mark mechanism to implement a mark component that remembers the current environment (used in page makeup) and the the two mark components left over from the original L^AT_EX. The fourth component is presently unused.

`\mark@envir` The third mark component's access procedures. The `\mark@envir` and `\bot@envir` commands are a good model of how to write access procedures for a new mark component.

```

222 \def\mark@envir{\markthr@}%
223 \def\bot@envir{%
224 \expandafter\expandafter
225 \expandafter\get@mark@thr@
226 \expandafter\@botmark
227 \nul@mark
228 }%

```

`\markboth` Set procedures for legacy components.

`\markright` 229 `\def\markboth{\mark@netw@}%`

`\leftmark` 230 `\def\markright{\marktw@}%`

`\rightmark` Retrieval procedures for legacy mark components. The procedure for retrieving the first component from `\botmark` and the second component from `\firstmark` have names in L^AT_EX; they are called, respectively, `\leftmark` and `\rightmark`.

It is possible to retrieve the components of `\topmark` as well: use `\saved@topmark`.

```

231 \def\leftmark{%
232 \expandafter\expandafter
233 \expandafter\get@mark@ne
234 \expandafter\saved@botmark
235 \nul@mark
236 }%
237 \def\rightmark{%
238 \expandafter\expandafter
239 \expandafter\get@mark@tw@
240 \expandafter\saved@firstmark

```

```
241          \nul@mark
242 }%
```

8.5 Output Super-routine

We want to change L^AT_EX's output routine, but do not wish to remain vulnerable to interference from such "required" packages as `multicol` (authored by Frank Mittelbach) and `longtable` (authored by David P. Carlisle), which swap in their own output routines when the respective package is active.

The better mechanism, used here, is due to William Baxter (web at super-script.com), who has allowed his several ideas to be used in this package.

In what follows, we effectively wrap up the old L^AT_EX output routine inside a new, more flexible "super routine". When the output routine is called, the "super routine" acts as a dispatcher. If the old routine is needed, it is called.

If a package attempts to substitute in their own output routine, they will effectively be modifying a token register by the name of `\output`. The primitive `\output` is now known by a different name, which should no longer be necessary to use.

Usage note: to make a visit to the output routine employing the dispatcher, enter with a value of `\outputpenalty` that corresponds to a macro. Defining as follows:

```
%\@namedef{output@10000}{your code here}%
%
```

by convention, your output routine should void out `\box\@cc1v`.

In rewriting L^AT_EX's output dispatcher in a much simpler form, we also avoid the sin of multiple `\shipouts` within a single visit to the output routine.

Conceptually, we divide visits to the output routine into two classes. The first involves natural page breaks (at a `\newpage` or when `\pagetotal > \pagegoal`) and usually resulting in `\box\@cc1v` either being shipped out or salted away (e.g., each column in a multicolumn layout). We might call this class the "natural output routines"; the `\outputpenalty` will never be less than -10000 . Furthermore, we ensure that `\holdinginserts` is cleared when calling such routines.

The other class involves a forced visit to the output routine via a large negative penalty (< -10000). They do not generally result in a `\shipout` of `\box\@cc1v`: they may be dead cycles. We provide a mechanism (call it a "one-off" output routine) that allows us to specify certain processing to be done when T_EX reaches the current position on the page.

One-off output routines themselves fall into two divisions, ones that process `\box\@cc1v`, and ones that work on the main vertical list (MVL). The former are typified by changes to the page grid, perhaps even column balancing. The latter involve the insertion of penalties or glue and the processing of floats.

The natural output routine is a single procedure. We have not introduced multiple natural output routines based on the `\outputpenalty` because T_EX does not support such a thing: T_EX sometimes lays down a penalty whose value is

the sum of other penalties. Because of this, we cannot depend on the value of `\outputpenalty` in such areas.

We do introduce flexibility in the form of a mechanism for patching into the natural output routine. Three hooks are offered, allowing a procedure to prepare for the upcoming visit to the output routine, access to `\box\ccclv`, and after doing `\shipout` (or otherwise committing the material to the page).

Environments, commands, and even packages can install their own procedures into these hooks. For instance, if the `longtable` package is loaded, it will install its procedures, but those procedures will punt if the page break being processed does not actually fall within a `longtable` environment.

`\primitive@output` Here we remember the \TeX primitive `\output` and its value, and then proceed to take over the `\csname` of `\output`, making it a `\toks` register and installing the old value of the output routine.

```
243 \let\primitive@output\output
```

`\output@latex` Grab the tokens in `\the\output` (but without the extra set of braces). The value of `\toks@` must remain untouched until loaded into the appropriate token register; this is done a few lines below.

`\output`

```
244 \long\def\@tempa#1\@nil{#1}%
245         \toks@
246 \expandafter\expandafter
247 \expandafter{%
248 \expandafter \@tempa
249         \the\primitive@output
250         \@nil
251     }%
252 \newtoks\output@latex
253 \output@latex\expandafter{\the\toks@}%
254 \let\output\output@latex
```

A comment on compatibility with other packages that co-opt the output routine.

Somewhere on the LaTeX-L list, David Kastrup has urged macro writers to take over the output routine in such a way that others can do likewise. How is this to be accomplished?

Consider what the `lineno` package does when it loads.

1. It does `\let cmdtempa \output`. This has the effect of identifying `\@tempa` with the `\toks` register we created above to hold the old output routine of \LaTeX . Let us say that was `\toks14`.
2. `lineno` itself effectively does `\newtoks \@LN@output`, which assigns that `\csname` to `\toks15`.
3. It loads `\@LN@output` with the contents of `\@tempa` (that is, `\toks14`, our copy of \LaTeX 's output routine).

4. Then it loads `\@tempa` with its own desired procedure, to be executed at `\output` time, thereby taking over what it thinks is the output routine, but which is in reality the procedure `REVTEX` executes when it wants to pass control to `LATEX`'s original output routine.
5. It then does `\let \output \@LN@output`, which now identifies `\output` with `\toks15`, the output routine of `lineno`.
6. When the `\output` routine is triggered, the primitive output routine `\primitive@output` is executed, and if appropriate, control is passed to `\output@latex`, which `REVTEX` had loaded with the old `LATEX` output routine, but which is presently loaded with that of `lineno`.
7. The output routine of `lineno` is executed, and if appropriate control is passed to `\LN@output`, the old output routine of `LATEX`.
8. Furthermore, the `\csname \output` now points to `\LN@output` (`\toks15`). This means that someone coming in after `lineno` to take over the output routine will actually get executed after that of `lineno`, but before `LATEX`.

As you can see, the process of taking over the output routine may continue until all of the `\toks` registers have been allocated. If, say, `newpackage` would itself like to take over the output routine, and if it uses the above set of steps, then when the output routine is triggered, the order of execution is `REVTEX`, then `lineno`, then `newpackage`, then `LATEX`. Each new package inserts itself on front of `LATEX`.

`\dispatch@output` We now install our own output routine in place of the original output routine of `LATEX`, which is still available as `\the \output`.

The output routine is simply the procedure `\dispatch@output`. It either dispatches to a procedure based on a particular value of `\outputpenalty` or it executes `\the\output@latex` tokens.

```
255 \primitive@output{\dispatch@output}%
256 \def\dispatch@output{%
257 \let\par\@par
```

Try to interpret `\outputpenalty` as a dispatcher to a message handler, its value is, e.g., `\do@startpage@pen`.

```
258 \expandafter\let\expandafter\output@procedure\csname output@\the\outputpenalty\endcsname
```

If we have failed to find a dispatcher, then settle for `\output@latex`.

```
259 \@ifnotrelax\output@procedure}{}%
260 \expandafter\def\expandafter\output@procedure\expandafter{\the\output@latex}%
261 }%
```

Now test if the dispatcher is the special case of `\execute@message@pen`, in which case execute the `\@message@saved`.

```
262 \expandafter\ifx\expandafter\csname output@-\the\execute@message@pen\endcsname\output@proced
263 \let\output@procedure\@message@saved
264 }{}%
265 \outputdebug@sw{\output@debug}{}%
```

```

266 \output@procedure
267 }%

268 \def\set@output@procedure#1#2{%
269 \count@outputpenalty\advance\count@-#2%
270 \expandafter\let\expandafter#1\csname output@the\count@\endcsname
271 }%

272 \def\output@debug{%
273 \saythe\inputlineno
274 \saythe\holdinginserts
275 \saythe\outputpenalty
276 \saythe\interlinepenalty
277 \saythe\brokenpenalty
278 \saythe\clubpenalty
279 \saythe\widowpenalty
280 \saythe\displaywidowpenalty
281 \saythe\predisplaypenalty
282 \saythe\interdisplaylinepenalty
283 \saythe\postdisplaypenalty
284 \say\output@procedure
285 \saythe\badness
286 \say\thepagegrid
287 \saythe\pagegrid@col
288 \saythe\pagegrid@cur
289 %\say\bot@envir
290 \saythe\insertpenalties
291 %\say\@@topmark
292 %\say\saved@@topmark
293 %\say\@@firstmark
294 %\say\saved@@firstmark
295 \say\@@botmark
296 %\say\saved@@botmark
297 \saythe\pagegoal
298 \saythe\pagetotal
299 \saythe{\badness\@cclv}%
300 \say\@toplist
301 \say\@botlist
302 \say\@dbltoplist
303 \say\@deferlist
304 \trace@scroll{%
305 \showbox\@cclv
306 \showbox\@cclv@savd
307 \showbox\pagesofar
308 \showbox\footsofar
309 \showbox\footins@savd
310 \showbox\footins
311 \showlists
312 }%
313 }%
314 \@ifxundefined{\outputdebug@sw}{%

```

```

315 \@booleanfalse\outputdebug@sw
316 }{}%
317 \def\trace@scroll#1{\begingroup\showboxbreadth\maxdimen\showboxdepth\maxdimen\scrollmode#1\endg
318 \def\trace@box#1{\trace@scroll{\showbox#1}}%

```

`\@outputpage` The procedure `\@outputpage` of standard L^AT_EX is the sole place where a
`\@outputpage@head` `\shipout` is carried out. The procedures that build `\@outputbox` just before
`\@outputpage@tail` a page is shipped out by `\@outputpage` are: `\@makecolumn`, `\@combinepage`, and
`\@combinedblfloats`.

We need to head- and tailpatch this procedure, so we perform here the only
 modifications to that procedure that are essential. Elsewhere, we will build up the
 meanings of `\@outputpage@head` and `\@outputpage@tail`.

```

319 \prepdef\@outputpage{\@outputpage@head}%
320 \let\@outputpage@head\@empty
321 \appdef\@outputpage{\@outputpage@tail}%
322 \let\@outputpage@tail\@empty

```

`\show@box@size` Procedure `\show@box@size` is a diagnostic for the sizes of boxes; the boolean
`\show@text@box@size` `\show@box@size@sw` turns it on and off.

```

\show@pagesofar@size 323 \def\show@box@size#1#2{%
\show@box@size@sw 324 \show@box@size@sw{%
\total@text 325 \begingroup
326 \setbox\z@\vbox{\unvcopy#2\hrule}%
327 \class@info{Show box size: #1^^J%
328 (\the\ht\z@\space X \the\wd\z@)
329 \the\c@page\space\space\the\pagegrid@cur\space\the\pagegrid@col
330 }%
331 \endgroup
332 }{}%
333 }%

```

Procedure `\show@text@box@size` tallies the size of the indicated column. If `\box`
`\pagesofar` is a factor, then its height has been memorized in the depth of the
 tally box.

```

334 \def\show@text@box@size{%
335 \show@box@size{Text column}\@outputbox
336 \tally@box@size@sw{%
337 \@ifdim{\wd\@outputbox>\z@}{%
338 \dimen@ht\@outputbox\divide\dimen@\twopowerfourteen
339 \advance\dimen@-\dp\csname box@size@\the\pagegrid@col\endcsname
340 \@ifdim{\dimen@>\z@}{%
341 \advance\dimen@ \ht\csname box@size@\the\pagegrid@col\endcsname
342 \global\ht\csname box@size@\the\pagegrid@col\endcsname\dimen@
343 \show@box@size@sw{%
344 \class@info{Column: \the\dimen@}%
345 }{}%
346 }{}%
347 }{}%
348 \global\dp\csname box@size@\the\pagegrid@col\endcsname\z@

```

```

349 }{}%
350 }%

Take the height of \box \pagesofar into account.
351 \def\show@pagesofar@size{%
352 \show@box@size{Page so far}\pagesofar
353 \dimen@ht\pagesofar\divide\dimen@ \@twopowerfourteen
354 \global\dp\csname box@size@1\endcsname\dimen@
355 \show@box@size@sw{%
356 \class@info{Pagesofar: \the\dimen@}%
357 }{}%
358 }%
359 \@booleanfalse\tally@box@size@sw
360 \@booleanfalse\show@box@size@sw
361 \expandafter\newbox\csname box@size@1\endcsname
362 \expandafter\setbox\csname box@size@1\endcsname\hbox{}}%
363 \expandafter\newbox\csname box@size@2\endcsname
364 \expandafter\setbox\csname box@size@2\endcsname\hbox{}}%
365 \def\total@text{%
366 \@tempdima\the\ht\csname box@size@2\endcsname\divide\@tempdima\@twopowertwo\@tempcnta\@tempdim
367 \@tempdimb\the\ht\csname box@size@1\endcsname\divide\@tempdimb\@twopowertwo\@tempcntb\@tempdim
368 \class@info{Total text: Column(\the\@tempcnta pt), Page(\the\@tempcntb pt)}%
369 }%

```

8.6 Further thoughts about inserts

The only safe way to deal with inserts is to either set `\holdininserts` or to commit to using whatever insert comes your way: you cannot change your mind once you see a non-void `\box\footins`, say.

Therefore all output routine processing must proceed with `\holdinginserts` set until you are sure of the material to be committed to the page. At that point, you can clear `\holdinginserts`, spew `\box\@cc1v`, put down the appropriate penalty, and exit, with the knowledge that \TeX will re-find the same pagebreak, this time visiting the output routine with everything, including inserts, in their proper place. This technique applies to split elements (screens, longtable, index) as well as to manufactured pages (float pages and clearpage pages).

Therefore, the output routine must not make assumptions about whether `\holdinginserts` should be cleared; instead this must be left to the one-off output routines or the natural output routine.

If we are manufacturing pages (“float page processing”), and if `\pagegoal` is not equal to `\vsize`, then inserts are at hand, and our criterion should take into account the insert material, even though we cannot measure its height based on the size of `\box\footins` (because `\holdinginserts` is set, you see).

It would be better to take the complement of `\floatpagefraction` and use that as a standard for the looseness of the page. Since `\pagegoal` reflects the inserted material, the criterion becomes the difference of the aggregate height of the floats and the `\pagegoal` versus this “page looseness” standard.

As a check, consider what happens if we bail out: `\@deferlist` has never been touched, so it requires no attention. Also, `\holdinginserts` has never been cleared, so inserts require no attention. So we only have to ensure that marks are preserved, which is already taken care of by the message handler mechanism.

If we are doing ordinary page cutting, then the scheme would be to detect whether we are within a screen (or longtable as may be), do the adjustment to the page height, and return, but this time with `\holdinginserts` cleared. Upon reentering the output routine, we may or may not be within the screen environment, but we are now sure to have a final page break, and we can commit this material (by shipping out or by saving it out as a full column).

In the above, the first of the two visits to the output routine is a dead cycle and requires propagation of marks, but nothing else.

8.7 The difference between inserts and floats

While revisiting this package in 2008, I needed to clarify under what circumstances inserts would be added to the `\pagesofar`. My conclusion is that I had been treating them exactly the same as floats, but that was a mistake.

Floats can be committed at the top of a column, in the middle, or at the bottom. Footnotes (the only `\insert` that is used in L^AT_EX) may only be committed at the bottom of a column. So, it was necessary to provide two versions of `\@combinepage`, one that committed `\inserts`, and the other that did not, the former used only when a `\shipout` was certain to be performed.

8.8 The natural output routine

Here is the portion of the output routine that fields cases not handled by the dispatcher.

The default is to ship out a page and then look around for more material that might constitute a “float page”. However, because `\holdinginserts` is normally set, this output routine must first have a dead cycle and come back again with `\holdinginserts` cleared. Then, after shipping out, it puts down a message that will manufacture zero or more float pages, finally terminating with a procedure that commits floats to a new unfinished page.

To accommodate special processing, we execute hooks whose name is based on the value of the “envir” mark component. The default is “document”, ensured by an initial mark of that value; the associated procedures are all nil. Any unknown envir value will “`\relax` out”.

The test made by `\toggle@insert` tells whether we are on our first visit to the output routine (with `\holdinginserts` still positive), or our second (with `\holdinginserts` zeroed). The output routine will toggle the setting.

The commands `\hold@insertions` and `\move@insertions` respectively clear and set `\holdinginserts`, so this procedure effectively clears `\holdinginserts` just long enough to pick up the insertions. Important: any output routine that clears `\holdinginserts` must guarantee that it is restored on the subsequent visit to the output routine. Or, to put it another way, if an output routine detects that

`\holdinginserts` is cleared, it should take it upon itself to restore it to a positive value before exiting.

The branch with `\holdinginserts` set is executed first; the other branch follows on practically immediately thereafter. In the first branch, we simply execute the appropriate hook and then execute a dead cycle.

In the branch with `\holdinginserts` cleared, the procedure builds up the current column, which is now complete, with `\@makecolumn`, then dispatches to the shipout routine associated with the current page grid, `\output@column@`. At the end, it triggers the execution of an output routine to prepare the next column (or page).

8.9 Natural output routine

`\natural@output` Here is the output routine that handles natural pagebreaks: we now have page
`\output` that needs to be shipped out or a portion of a page that is ready to be committed to the page grid. Processing is of necessity divided into phases, `\output@holding` is executed upon first encountering the natural page-breaking point, while inserts are being held. The second phase, `\output@moving`, is set in motion by the first: here the same material (in most cases) will be processed with `\holdinginserts` cleared, and the insertions (e.g., footnotes) are split off into their assigned box registers.

```
370 \def\natural@output{\toggle@insert{\output@holding}{\output@moving}}%
371 \output@latex{\natural@output}%
```

In accordance with the scheme suggested by David Kastrup for allowing another output routine to slip itself into ours, we use a token register called `\output`. However, we reserve the ability to restore things if we so desire. This we must do in the case of the `ltxgrid.dtxlineno.sty` package, because its functionality is best served by being integrated into our own dispatcher-based output routine.

To restore our own output routine, we can repeat the above assignment,

```
%\output@latex{\natural@output}%
%
```

some time before the document begins.

`\output@holding` The procedure `\output@holding` is our first cycle through the output routine;
`\@if@exceed@pagegoal` `\holdinginserts` is still set. We give the current environment a heads up (it is through this means that `longtable` sets its running header and footer), then we execute a dead cycle, which should propagate marks.

One corner case that can crop up is the presence of a single unbreakable chunk whose size is larger than `\vsize`. Doing a dead cycle under such circumstances will not find the same breakpoint as this time (remember we threw in a `\mark` node). Instead, we attempt to remove the excess height of the material, so we can continue to propagate marks.

The corner case is at hand if the natural size of `\box\@cclv` exceeds `\pagegoal` and the contents cannot be shrunk to fit.

```

372 \def\output@holding{%
373 \csname output@init@bot@envir\endcsname
374 \@if@exceed@pagegoal{\unvcopy\@cclv}{%
375 \setbox\z@\vbox{\unvcopy\@cclv}%
376 \outputdebug@sw{\trace@box\z@}{}%
377 \dimen@ht\@cclv\advance\dimen@-ht\z@
378 \dead@cycle@repair\dimen@
379 }{%
380 \dead@cycle
381 }%
382 }%
383 \def\@if@exceed@pagegoal#1{%
384 \begingroup
385 \setbox\z@\vbox{#1}%
386 \dimen@ht\z@\advance\dimen@dp\z@
387 \outputdebug@sw{\saythe\dimen@}{}%
388 \@ifdim{\dimen@>\pagegoal}{%
389 \setbox\z@\vbox{\@mark}\unvbox\z@}%
390 \splittopskip\topskip
391 \splitmaxdepth\maxdepth
392 \vbadness\M
393 \vfuzz\maxdimen
394 \setbox\tw@\vsplit\z@ to\pagegoal
395 \outputdebug@sw{\trace@scroll{\showbox\tw@\showbox\z@}}{%
396 \setbox\tw@\vbox{\unvbox\tw@}%
397 \@ifdim{ht\tw@=\z@}{%
398 \ltxgrid@info{Found overly large chunk while preparing to move insertions. Attempting repair}
399 \aftergroup\true@sw
400 }{%
401 \aftergroup>false@sw
402 }%
403 }{%
404 \aftergroup>false@sw
405 }%
406 \endgroup
407 }%

```

`\output@moving` The procedure `\output@moving` is our second cycle through the output routine; `\cclv@nontrivial@sw` `\holdinginserts` is now cleared, and `\inserts` will have been split off into their respective box registers, like `\footins`.

1. Set the values of `\topmark` and `\firstmark`.
2. If we got here because of a `\clearpage` command, remove the protection box that this mechanism has left on the MVL.
3. If the contents of `\box\@cclv` are non-trivial, commit it to the current page (as a column) or ship it out, as the case may call for.
4. If not, discard it (we are at the end of `\clearpage` processing).

5. Set various values, including the available space for setting type on the next column (`\@colroom`).

The processing for a non-trivial `\box\@cclv` are:

1. Execute the head procedure for the current environment.
2. Make up a column and ship it out (or commit it to the current page) via a procedure keyed to the current page grid.
3. Put down an interrupt for `\do@startcolumn@pen`: this will force a visit to the output routine for the purpose of committing floats to the next column.
4. Possibly put down an interrupt to continue `\clearpage` processing.
5. Execute the tail procedure for the current environment.

The processing for a trivial `\box\@cclv` are:

1. Void out `\box\@cclv` and give appropriate warning messages and diagnostics.
2. Put down the same interrupts as for the non-trivial case above.

This instance of `\@makecolumn` is followed by `\output@column@`, that is, it builds a column for `\shipout` rather than for adding to `\pagesofar`.

We need to handle cases where the `\output@pre@`, `\output@column@`, or `\output@post@` dispatchers come up `\relaxed` out: the default is to execute the corresponding procedures from the `docuemnt` environment and the one-column grid respectively.

One such case comes up with frequency: at the end of the document, where the `\botmark` is now empty.

```

408 \def\output@moving{%
409   \set@top@firstmark
410   \ifnum\outputpenalty=\do@newpage@pen}{%
411     \setbox\@cclv\vbox{%
412       \unvbox\@cclv
413       \remove@lastbox
414       \@ifdim{\ht\z@=\ht\@protection@box}{\box\lastbox}{\unskip}%
415     }%
416   }{}%
417   \@cclv@nontrivial@sw{%
418     \expandafter\let\expandafter\output@prep@\csname output@prep@\bot@envir \endcsname
419     \outputdebug@sw{\say\output@prep@}{}%
420     \@ifx{\output@prep@\relax}{\output@prep@document}{\output@prep@}%
421     \@makecolumn>true@sw
422     \expandafter\let\expandafter\output@column@\csname output@column@\thepagegrid\endcsname
423     \outputdebug@sw{\say\output@column@}{}%
424     \@ifx{\output@column@\relax}{\output@column@one}{\output@column@}%
425     \protect@penalty\do@startcolumn@pen

```

```

426 \clearpage@sw{%
427   \protect@penalty\do@endpage@pen
428 }{%
429 \expandafter\let\expandafter\output@post@\csname output@post@\bot@envir \endcsname
430 \outputdebug@sw{\say\output@post@}{}%
431 \@ifx{\output@post@\relax}{\output@post@document}{\output@post@}%
432 }{%
433 \void@cclv
434 }%
435 \set@colht
436 \global\@mparbottom\z@
437 \global\@textfloatsheight\z@
438 }%
439 \def\void@cclv{\begingroup\setbox\z@\box@cclv\endgroup}%
440 \def\remove@lastbox{\setbox\z@\lastbox}%

```

The procedure `\cclv@nontrivial@sw` determines if this visit to `\output@moving` is a trivial one, which happens at the end of `\clearpage` processing and under some pathological circumstances. It emits a Boolean, so it is syntactically like `\true@sw`, albeit does not execute solely via expansion.

Note: the case where `\box@cclv` is void comes up at the very beginning of the job, when typesetting a (full-page-width) title block in a two-column layout.

Note: the code that removes the last box and skip from the output is intended to detect the case where the output has whatsit nodes followed by topskip and a protection box. This is what happens under normal circumstances at the end of `\clearpage` processing.

```

441 \def\@cclv@nontrivial@sw{%
442   \@ifx@empty@toplist{%
443     \@ifx@empty@botlist{%
444       \@ifvoid\footins{%
445         \@ifvoid@cclv{%
446           \false@sw
447         }{%
448           \setbox\z@\vbox{\unvcopy@cclv}%
449           \@ifdim{\ht\z@=\topskip}{%
450             \setbox\z@\vbox\bgroup
451               \unvbox\z@
452               \remove@lastbox
453               \dimen@lastskip\unskip
454               \@ifdim{\ht\z@=\ht@protection@box}{%
455                 \advance\dimen@\ht\z@
456                 \@ifdim{\dimen@=\topskip}{%
457                   \aftergroup\true@sw
458                 }{%
459                   \aftergroup\false@sw
460                 }%
461               }{%
462                 \aftergroup\false@sw
463               }%

```

End of \box\z@.

464 \egroup

465 {%

Normal for

```

466     \false@sw
467   }{%
468     \true@sw
469   }%
470 }{%
471   \@ifdim{\ht\z@=\z@}{%
472     \ltxgrid@info{Found trivial column. Discarding it}%
473     \outputdebug@sw{\trace@box\@cc1v}{}%
474     \false@sw
475   }{%
476     \true@sw
477   }%
478 }%
479 }%
480 }{%
481   \true@sw
482 }%
483 }{%
484   \true@sw
485 }%
486 }{%
487   \true@sw
488 }%
489 }%

```

`\protect@penalty` The procedure `\protect@penalty` is the utility procedure for invoking a one-off output routine. Such a routine can expect to find the protection box above it in `\box\@cc1v`: it should remove that box.

Note that `\execute@message` does the same thing as `\protect@penalty`, but in a slightly different way.

We create a specially formulated box that will be universally used when a protection box is needed. In this way, we can always recognize when `\box\@cc1v` is trivial: it will consist of `\whatsits` followed by `\topskip` glue and the `\@protection@box`.

```

490 \def\protect@penalty#1{\@protection@box\penalty-#1\relax}%
491 \newbox\@protection@box
492 \setbox\@protection@box\ vbox to1986sp{\vfil}%
493 \def\@protection@box{\nointerlineskip\copy\@protection@box}%

```

`\dead@cycle` The procedure `\dead@cycle` is defined separately as a utility which can be used by any output processing routine to emulate what takes place in the standard output routine.

Here, we have entered the output routine with `\holdinginserts` enabled, which means that we are not yet ready to ship out material, because the `\insert` registers are being held. We want to clear `\holdinginserts` and come back here with the same page break as before, whereupon we may properly proceed with page makeup.

To do this, we propagate marks, then spew the contents of `\box\@cc1v` followed

by the original output penalty that landed us here (but only if it is not 10000, the flag value for a pagebreak not at a penalty).

However, the natural output routine should do this only if `\box\@cclv` is nontrivial. A pathological case exists wherein a box of height greater than `\textheight` would cause an infinite loop involving the output routine. The procedure `\dead@cycle@repair`, attempts to catch this case and avoid the loop.

The test of the height of `\box\@cclv` is not the correct one, because this test will run afoul in the case where `\box\@cclv` contains nothing but an `\insert` node. What to do?

It is possible that the pathological case can be detected by looking at `\pagetotal`. If that quantity is zero, then `\box\@cclv` really is trivial.

In the procedure `\dead@cycle@repair`, if `\box\@cclv` is nontrivial, we execute `\dead@cycle`, otherwise it contains nothing but a mark, so we dispense with propagating marks and we simply spew out `\box\@cclv` without an accompanying mark. This has the effect of failing to propagate marks, but this problem is preferable to the infinite loop, which in principle could crash even a robust operating system by filling up the file system.

If a document has such a large chunk, it should be fixed, so we give a message in the log.

You ask, “In what way does this infinite loop come about?” Good question!

The setup is a chunk in the MVL that is taller than `\textheight`. (Yes, it’s that simple.) As soon as the previous page ships out, the MVL will contain a mark (propagated from the previous page) followed by that large chunk (call it the ‘big bad box’, albeit does not need to be a single box). The next visit to the output routine will be a natural page break, but `TEX` will select the juncture between the mark and the big bad box as the least-cost page break. Unless the test in `\dead@cycle` is done, the cycle is perpetuated when the macro reinserts the mark.

The crux matter is achieving, in a robust way, the goal of going from a `\holdinginserts` state to one where the insertions are moving.

```

494 \def\dead@cycle@repair#1{%
495   \expandafter\do@@mark
496   \expandafter{%
497     \@@botmark
498   }%
499   \unvbox\@cclv
500   \nointerlineskip
501   \vbox to#1{\vss}%
502   \@ifnum{\outputpenalty<\@M}{\penalty\outputpenalty}{}%
503 }%
504 \def\dead@cycle@repair@protected#1{%
505   \expandafter\do@@mark
506   \expandafter{%
507     \@@botmark
508   }%
509   \begingroup
510   \unvbox\@cclv

```


Remove the protection box

```
511 \remove@lastbox
512 \nointerlineskip
513 \advance#1-\ht\@protection@box
514 \vbox to#1{\vss}%
515 \protection@box % Reinsert protection box
516 \@ifnum{\outputpenalty<\@M}{\penalty\outputpenalty}{}%
517 \endgroup
518 }%
519 \def\dead@cycle{%
520 \expandafter\do@@mark
521 \expandafter{%
522     \@botmark
523 }%
524 \unvbox\@cclv
525 \@ifnum{\outputpenalty<\@M}{\penalty\outputpenalty}{}%
526 }%
```

`\output@init@document` The default processing simply provides for insertion of held-over footnotes. At a natural page break, we are either at the bottom of a column or at the bottom of a page. In either case, the `\output@init@` processing adjusts for the height of the held-over footnotes and bails out. Upon our return, at `\output@prep@` time, the page break will accomodate the material; it is now actually inserted by concatenating it with the contents of `\footins`. The default processing for `\output@post@` is nil.

```
527 \def\output@init@document{%
528 \set@adj@footins\vsizelocal\vsizelocal
529 }%
```

QUERY: the following procedure is very like `\combine@foot@inserts`. Should it be the same?

```
530 \def\output@prep@document{%
531 \@ifvoid\footsofar{}{%
532 \setbox\footins\vbox\bgroup
533 \unvbox\footsofar
534 \@ifvoid\footins{}{%
535 \marry@baselines
536 \unvbox\footins
537 }%
538 \egroup
539 }%
540 }%
541 \def\output@post@document{}%
```

`\@opcol` The standard L^AT_EX procedure `\@opcol` is now completely obsolete.

```
542 \let\@opcol\undefined
```

`\@makecolumn` The procedure `\@makecolumn` packages up a page along with all its insertions and floats. Therefore it is essential that it be executed with `\holdininserts` cleared.

Note that there is a corner case when in a multi-column grid, where the change back to one-column grid occurs just after a complete page ships out. We want to detect when `\@cclv` contains nothing but a `\mark`, but this is a \TeX impossibility.

Note on `\@kludgeins`: we have removed this mechanism from \LaTeX , because the implementation of `\enlargethispage` no longer requires it. Here, for consistency sake, we remove `\@makespecialcolbox`.

The argument of `\@makecolumn` is a Boolean and determines if we combine the footnote material into the present column. If the procedure is building a column for shipping out, then we will combine the footnote material, if not, we return with the `\footins` box unchanged.

I changed the behavior of this procedure in the case where the argument is `\false@sw`: send the unused footnote material to `\footsofar`.

```

543 \def\@makecolumn#1{%
544   \setbox\@outputbox\vbox{%
545     \boxmaxdepth\@maxdepth
546     \@tempdima\dp\@cclv
547     \unvbox\@cclv
548     \vskip-\@tempdima
549   }%
550   \xdef\@freelist{\@freelist\@midlist}\global\let\@midlist\@empty
551   \show@text@box@size
552   \@combinefloats
553   #1{%
554     \@combineinserts\@outputbox\footins
555   }{%
556     \combine@foot@inserts
557   }%
558   \set@adj@colht\dimen@
559   \count@\vbadness
560   \vbadness\@M
561   \setbox\@outputbox\vbox to\dimen@{%
562     \@texttop
563     \dimen@\dp\@outputbox
564     \unvbox\@outputbox
565     \vskip-\dimen@
566     \@textbottom
567   }%
568   \vbadness\count@
569   \global\maxdepth\@maxdepth
570 }%
571 \let\@makespecialcolbox\@undefined

```

`\@combineinserts` We split out the procedure to add the `\footins` insertions to the packaged-up page. Any other non-trivial insertions should also be dealt with at this time.

```

572 \def\@combineinserts#1#2{%
573   \setbox#1\vbox{%
574     \unvbox#1%
575     \vbox{%

```

```

576 \@ifvoid#2{ }{%
577 \show@box@size{Combining inserts}#2%
578 \vskip\skip#2%
579 \color@begingroup
580 \normalcolor
581 \footnoterule
582 \nointerlineskip
583 \box#2%
584 \color@endgroup
585 }{%
586 }%
587 }%
588 }%

```

`\@floatplacement` In standard L^AT_EX, someone (DPC?) makes the assumption that `\@fpmin` can be assigned locally. This is no longer true now that we ship no more than one page per visit to the output routine. We apply a bandaid.

```

589 \appdef\@floatplacement{%
590 \global\@fpmin\@fpmin
591 }%

```

`\pagebreak@pen` While we are in the way of registering certain penalty values, let us register the smallest one that will force a visit to the output routine. However, this penalty will not have an associated macro: we wish to execute the natural output routine instead.

Note that this penalty is invoked by `\clearpage` and `\newpage`.

```

592 \mathchardef\pagebreak@pen=\@M
593 \expandafter\let\csname output@-\the\pagebreak@pen\endcsname\relax

```

8.10 Float placement

`\do@startcolumn@pen` The procedure `\do@startcolumn@pen` is executed as a one-off output routine just after a page is shipped out (or, in a multicolumn page grid, a column is salted away).

Its job is to either generate a “float page” (in reality a column) for shipping out, or to commit deferred floats to the fresh column, concluding with a dead cycle. In the former case, we accomodate split footnotes and other insertions (by comparing `\vsize` and `\pagegoal`): the floats are spewed onto the page, whereupon L^AT_EX’s output routine will place the footnotes and ship out, iterating the process once again.

Note that when this procedure is invoked, `\box\@cc1v` still has within it the protection box, so we start by removing it. Note also that if there was a split insertion held over from the previous page, the insert node will be present in `\box\@cc1v`, *prior to* the protection box. For this reason, we cannot just throw away that box, as we might be tempted to do.

FIXME: where else do we possibly inappropriately discard `\box\@cc1v`?

Note that, because a column or page had previously just been completed, we can assume that there is nothing of importance on the page, and because no message is being passed, we can preserve marks in a simple way.

A Note on terminology: In a single-column page grid, you might expect that we would execute the procedure `\do@startpage`. But this is not so. \LaTeX has a confusion of long standing, in which the procedures that handle full-page width floats in a two-column page grid all have in their names the string ‘dbl’, which erroneously suggests having something to do with “double”. It does not: when you see ‘dbl’, think “full page width”.

```

594 \mathchardef\do@startcolumn@pen=10005
595 \@namedef{output@-\the\do@startcolumn@pen}{\do@startcolumn}%
596 \def\do@startcolumn{%
597   \setbox\@cclv\vbox{\unvbox\@cclv\remove@lastbox\unskip}%
598   \clearpage@sw{\@clearfloatplacement}{\@floatplacement}%
599   \set@colht
600   \@booleanfalse\pfloat@avail@sw
601   \begingroup
602     \colht\@colroom
603     \@booleanfalse\float@avail@sw
604     \@tryfloat\test@colfloat
605     \float@avail@sw{\aftergroup\@booleantrue\aftergroup\pfloat@avail@sw}{}%
606   \endgroup
607   \fcolmade@sw{%
608     \setbox\@cclv\vbox{\unvbox\@outputbox\unvbox\@cclv}%
        Now ask for a return visit, this time with insertions and all.
609     \outputpenalty-\pagebreak@pen
610     \dead@cycle
611   }{%
612     \begingroup
613       \let\@elt\@scolelt
614       \let\reserved@b\@deferlist\global\let\@deferlist\@empty\reserved@b
615     \endgroup
616     \clearpage@sw{%
617       \outputpenalty\@M
618     }{%
619       \outputpenalty\do@newpage@pen
620     }%
621     \dead@cycle
622   }%
623   \check@deferlist@stuck\do@startcolumn
624   \set@vsize
625 }%
626 \def\@scolelt#1{\def\@currbox{#1}\@addtonextcol}%
627 \def\test@colfloat#1{%
628   \csname @floatselect@sw@\thepagegrid\endcsname#1}{\@testtrue}%
629   \@if@sw@if@test\fi}{\aftergroup\@booleantrue\aftergroup\float@avail@sw}%
630 }%

```

`\@addtonextcol` We must adjust `\@addtonextcol` to take held-over inserts into account. Now

that all deferred floats are queued up together (in order), we must have a way of differentiating them; this is done by the page grid-dependent procedure `\@floatselect@sw@`.

```

631 \def\@addtonextcol{%
632 \begingroup
633 \insertfalse
634 \@setfloattypecounts
635 \csname @floatselect@sw@thepagegrid\endcsname\@currbox{%
636 \ifnum{\@fpstype=8 }{}{%
637 \ifnum{\@fpstype=24 }{}{%
638 \flsettextmin
639 \reqcolroom \ht\@currbox
640 \advance \reqcolroom \textmin
641 \advance \reqcolroom \vsize % take into account split insertions
642 \advance \reqcolroom -\pagegoal
643 \ifdim{\@colroom>\reqcolroom}{%
644 \flsetnum \@colnum
645 \ifnum{\@colnum>\z@}{%
646 \bitor\@currtype\@deferlist
647 \if@sw\if@test\fi}{%
648 \addtotoporbot
649 }%
650 }{}%
651 }{}%
652 }%
653 }%
654 }{}%
655 \if@sw\if@insert\fi}{%
656 \cons\@deferlist\@currbox
657 }%
658 \endgroup
659 }%

```

`\do@startpage@pen` Similar to `\do@startcolumn`, the procedure `\do@startpage` starts up a new page (not column) in a multi-column page grid. It is invoked after a page is shipped out in a multi-column page grid, and it commits full-page-width floats to the fresh page, possibly resulting in a float page. In implementation, it is similar to `\do@startcolumn`, except that it commits effectively via `\@addtodblcol` instead of `\@addtonextcol`. Note that this procedure will inevitably be followed by `\do@startcolumn`.

Some details of the procedure:

We begin by removing the protection box from `\box\@cclv`, then setting the values of the float placement parameters appropriately, and resetting `\@colht`, `\@colroom`, and `\vsize` to base values.

Next we attempt to compose a float page, a page consisting entirely of floats. If successful, we ship out the float page and lay down an interrupt that will send us back here for another try.

If no float page is formed, we attempt to commit full-page-width floats to the

text page, and return with a dead cycle. We are now ready to compose columns of text.

Note that all floats (both column floats and full-page-width floats) move through a single queue. To differentiate between the two, the width of the float is compared to `\textwidth`. This comparison is encapsulated in the macro `\@ifnotdblfloat`, which should be used whenever such a determination must be made. This procedure returns a Boolean.

```

660 \mathchardef\do@startpage@pen=10006
661 \@namedef{output@-\the\do@startpage@pen}{\do@startpage}%
662 \def\do@startpage{%
663   \setbox\cclv\vbox{\unvbox\cclv\remove@lastbox\unskip}%
664   \clearpage@sw{\clearfloatplacement}{\dblfloatplacement}%
665   \set@colht
666   \@booleanfalse\pfloat@avail@sw
667   \begingroup
668     \@booleanfalse\float@avail@sw
669     \@tryfcolumn\testdblfloat
670     \float@avail@sw{\aftergroup\@booleantrue\aftergroup\pfloat@avail@sw}{}%
671   \endgroup
672   \fcolmade@sw{%
673     \global\setbox\pagesofar\vbox{\unvbox\pagesofar\unvbox\@outputbox}%
674     \@output@combined@page
675   }{%
676     \begingroup
677       \@booleanfalse\float@avail@sw
678       \let\@elt\@sdblcolelt
679       \let\reserved@b\@deferlist\global\let\@deferlist\@empty\reserved@b
680     \endgroup
681     \@ifdim{\@colht=\textheight}{% No luck...
682       \pfloat@avail@sw{% ...but a float *was* available!
683         \forcefloats@sw{%
684           \ltxgrid@warn{Forced dequeuing of floats stalled}%
685         }{%
686           \ltxgrid@warn{Dequeuing of floats stalled}%
687         }%
688       }{%
689     }{%
690     \outputpenalty\@M
691     \dead@cycle
692   }%
693   \check@deferlist@stuck\do@startpage
694   \set@colht
695 }%

```

Procedure `\@output@combined@page` is a utility that ships out a page consisting of the result of `\@combinepage` and `\@combinedblfloats`, after which it prepares for the process to repeat.

It is coincidentally identical to what needs to happen with a float page that has been built by `\@tryfcolumn`, in the multi-column page grid, and also handles

the case where a page needs to be shipped out when in multicolumn mode.

```

696 \def\@output@combined@page{%
697 \@combinepage\true@sw
698 \@combinedblfloats
699 \@outputpage
700 \global\pagegrid@cur\@ne
701 \protect@penalty\do@startpage@pen
702 }%
703 \def\@sdblcolelt#1{\def\@currbox{#1}\@addtodblcol}%
704 \def\test@dblfloat#1{%
705 \@ifnotdblfloat{#1}{\@testtrue}{}%
706 \@ifsw@if@test\fi}{\aftergroup\@booleantrue\aftergroup\float@avail@sw}%
707 }%
708 \def\@ifnotdblfloat#1{\@ifdim{\wd#1<\textwidth}}%
709 \@booleanfalse\forcefloats@sw

```

`\@addtodblcol` The procedure `\@addtodblcol` is called into play at the beginning of each fresh page and operates on each deferred float, in the hopes of placing one or more such floats at the top of the current page.

We alter the procedure of standard L^AT_EX by putting failed floats into `\@deferlist` instead of `\@dbldeferlist`. Having done so, we must have a means of differentiating full-page-width floats from column-width floats. We assume that the latter will always be narrower than `\textwidth`.

In aid of detecting a stalled float flushing process, we set a Boolean if we encounter a qualified full-page-width float here. Any that qualify but fail the rest of the tests might still pass when reconsidered on an otherwise blank page.

```

710 \def\@addtodblcol{%
711 \begingroup
712 \@ifnotdblfloat{\@currbox}{%
713 \false@sw
714 }{%
715 \@setfloattypecounts
716 \@getfpsbit \tw@
717 \@bitor \@currtype \@deferlist
718 \@ifsw@if@test\fi{%
719 \false@sw
720 }{%
721 \@ifodd\@tempcnta{%
722 \aftergroup\@booleantrue\aftergroup\float@avail@sw
723 \@flsetnum \@dbltopnum
724 \@ifnum{\@dbltopnum>\z@}{%
725 \@ifdim{\@dbltoproom>\ht\@currbox}{%
726 \true@sw
727 }{%
728 \@ifnum{\@fpstype<\sist@n}{%
729 \begingroup
730 \advance \@dbltoproom \@textmin
731 \@ifdim{\@dbltoproom>\ht\@currbox}{%
732 \endgroup\true@sw

```

```

733     }{%
734     \endgroup\false@sw
735     }%
736   }{%
737     \false@sw
738     }%
739   }%
740 }{%
741   \false@sw
742   }%
743 }{%
744   \false@sw
745   }%
746 }%
747 }%
748 {%
749   \@tempdima -\ht\@currbox
750   \advance\@tempdima
751   -\@ifx{\@dbltoplist\@empty}{\dbltextfloatsep}{\dblfloatsep}%
752   \global \advance \dbltoproom \@tempdima
753   \global \advance \@colht \@tempdima
754   \global \advance \@dbltopnum \m@ne
755   \@cons \@dbltoplist \@currbox
756 }{%
757   \@cons \@deferlist \@currbox
758 }%
759 \endgroup
760 }%

```

`\@tryfcolumn` Whenever a page is shipped out, L^AT_EX automatically tries out a float column: a page containing nothing but floats (and, as we have added here, split footnotes).

`\@wtryfc` The following four procedures employ certain macros to communicate between

`\@xtryfc` each other:

`\@ztryfc`

- `\fcolmade@sw`, a boolean, says whether we were successful in making a float column.
- `\if@test`, a `\newif` switch, says a float has failed some test.
- `\@deferlist`, is the input to the process, a list, of deferred floats.
- `\@trylist`, a list, stores the deferred floats to be tried out on the float column.
- `\@failedlist`, a list of floats that have failed the selection for the float column.
- `\@flfail`, a list of floats that have failed the second selection for the float column.
- `\@flsucceed`, a list, the floats that have been successfully placed on the float column.
- `\@freelist`, a list, receives any freed floats.
- `\@colht`, a dimen, the available space for the column, including column floats and insertions (footnotes).
- `\@fpm`, a dimen, the required minimum height for the float column.
- `\@outputbox`, a box, the output of the process.

`\@fptop`, `\@fpsep`, `\@fpbot`, glue, placed above, between, and below floats on the float column.

`\@currtype`, a count, used temporarily for the float's bits.

`\@tempcnta`, a count, used temporarily for the float's bits.

In `\@tryfcolumn`, we alter the criterion for a float page, because if footnotes are present at this point (presumably due to a split insertion) then `\@fpminis` no longer the right threshold to apply.

Note that we have changed `\@tryfcolumn`, `\@xtryfc`, and `\@ztryfc` syntactically so that the procedure to test for the float's being a column float versus a full-page-width float is passed in as an argument.

```

761 \def\@tryfcolumn#1{%
762   \global\@booleanfalse\fcoldmade@sw
763   \@ifx@empty\@deferlist{}{%
764     \global\let\@trylist\@deferlist
765     \global\let\@failedlist\@empty
766     \begingroup
767       \dimen@vsize\advance\dimen@-\pagegoal\@ifdim{\dimen@>\z@}{%
768         \advance\@fpmin-\dimen@
769       }{%
770       \def\@elt{\@xtryfc#1}\@trylist
771     \endgroup
772     \fcoldmade@sw{%
773       \global\setbox\@outputbox\vbox{\vskip \@fptop}%
774       \let \@elt \@wtryfc \@flsucceed
775       \global\setbox\@outputbox\vbox{\unvbox\@outputbox
776         \unskip \vskip \@fpbot
777       }%
778       \let \@elt \relax
779       \xdef\@deferlist{\@failedlist\@flfail}%
780       \xdef\@freelist{\@freelist\@flsucceed}%
781     }{%
782   }%
783 }%
784 \def\@wtryfc #1{%
785   \global\setbox\@outputbox\vbox{\unvbox\@outputbox
786     \box #1\vskip\@fpsep
787   }%
788 }%
789 \def\@xtryfc#1#2{%
790   \@nextreserved@a\@trylist{}{}% trim \@trylist. Ugly!
791   \@currtype \count #2%
792   \divide\@currtype\@xxxii\multiply\@currtype\@xxxii
793   \@bitor \@currtype \@failedlist
794   \@testfp #2%
795   #1#2%
796   \@ifdim{\ht #2>\@colht }{\@testtrue}{}%
797   \@ifsw\if@test\fi{%
798     \@cons\@failedlist #2%
799   }{%

```

```

800 \begingroup
801 \gdef\@flsucceed{\@elt #2}%
802 \global\let\@flfail\@empty
803 \@tempdima\ht #2%
804 \def \@elt {\@ztryfc#1}\@trylist
805 \@ifdim{\@tempdima >\@fpmin}{%
806 \global\@booleantrue\@fcolmade@sw
807 }{%
808 \@cons\@failedlist #2%
809 }%
810 \endgroup
811 \@fcolmade@sw{%
812 \let \@elt \@gobble
813 }{%
814 }%
815 }%
816 \def \@ztryfc #1#2{%
817 \@tempcnta \count#2%
818 \divide\@tempcnta\@xxxii\multiply\@tempcnta\@xxxii
819 \@bitor \@tempcnta {\@failedlist \@flfail}%
820 \@testfp #2%
821 #1#2%
822 \@tempdimb\@tempdima
823 \advance\@tempdimb \ht#2\advance\@tempdimb\@fpsep
824 \@ifdim{\@tempdimb >\@colht}{%
825 \@testtrue
826 }{%
827 \@if@sw\if@test\fi{%
828 \@cons\@flfail #2%
829 }{%
830 \@cons\@flsucceed #2%
831 \@tempdima\@tempdimb
832 }%
833 }%

```

8.11 Clearing pages

Clearing the page is an elaboration of ending the page: it entails flushing all floats.

This package might make number of float flushing algorithms available, a very simple one that does not try to produce excellent pages, another that tries to make the best use of space, and a more complex one that tries to balance columns.

At the beginning of the page-clearing process, by definition all of the paragraph text involved is on the MVL and all floats have been encountered. There may be material in `\pagesofar`, and (in a multi-column page grid) any number of columns of the page have been composed. Also, there might be footnote material saved up in `\footsofar`.

Because we did not want to perform multiple `\shipouts` per visit to the output routine, our multi-column page makeup will not compose multiple columns

per visit. This implementation detail may not require alteration, but it is not a limitation that is truly necessary: it is only multiple `\shipouts` per visit that must be avoided.

The crux matter is how to continue with flushing floats even after the material in the MVL is exhausted. At that point, we must, upon completion of the output routine, insert into the MVL an interrupt that triggers the next step in the processing.

Therefore, after processing a `\do@startcolumn` interrupt, we must somehow force the completion of that column. This could be done by inserting a `\do@newpage@pen` interrupt.

And after processing a `\do@startpage@pen` interrupt, that results in `\@dbltopinserts`, we must ensure that the multiple columns on the page get completed, so that the page itself finally gets shipped out. This part will proceed automatically given that `\do@startcolumn` processing completes successfully.

The process will not be complete until all deferred floats have been placed and shipped out, and all saved-up footnotes have been inserted.

Full-page-width floats can get out of order of column floats. This problem can be remedied by holding them all in the same list. We therefore stop using `\@dbldeferlist` entirely, and all of the procedures that formerly used it have been rewritten to use `\@deferlist` instead. When traversing the list, we apply a selector on the given box that determines whether it is a column-width or page-width float. This selector is different depending on the page grid.

When the `\@deferlist` is processed (by any means), we have to take care of the case where a float of one category is passed over but we are looking for a float of the other category. Here, we must terminate processing, to avoid disordering the floats. This we do by the usual means.

The system has a Boolean that says we are clearing pages: `\clearpage@sw`; if it is true, then at the tail of `\do@startcolumn` processing, we should put down a (`\vfil?`) `\do@newpage@pen` interrupt. This is because the MVL is now empty, so we have to force the columns to complete.

One potential very pathological case would be where there is one or more deferred floats that never successfully get placed: placing floats has stalled, and we will ship out blank pages indefinitely. How to detect this case?

First, `\do@startpage` will evidently be stalled if the following are all true: a) `\@tryfcolumn` and `\@sdblcolelt` both fail, b) there are deferred floats available for page placement, and c) the `\@colht=\textheight`, that is, the full page height is available for placement of column floats.

Second, `\do@startcolumn` will evidently be stalled if the following are all true: a) `tryfcolumn` fails, b) there are deferred floats available for column placement, and a) the `\@colroom=\textheight`, that is, the full page height is available for placement of column floats.

<code>\cleardoublepage</code>	The function of <code>\clearpage</code> is to end the current page with <code>\newpage</code> and then
<code>\clearpage</code>	ship out additional pages until <code>()</code> inserts and <code>(deferred)</code> floats are exhausted.
<code>\newpage</code>	The method involves setting the float placement parameters to completely per-
<code>\newpage@prep</code>	missive values and kicking out the current page (using a non-discardable penalty).

A possibly short page will be shipped out, followed by any number of float pages. However these float pages, because using permissive float placement, will exhaust all inserts and deferred floats.

Bug Note: in the code for `\clearpage`, the first penalty we output is an unprotected `\pagebreak@pen`. I tried using a protected `\do@newpage@pen`, but that gave rise to a corner case where a blank page was output.

At present, the `\clearpage` procedure does the same as `\newpage`, except that `\clearpage@sw` is turned on, and the (discardable) `\newpage` is inevitably followed by the same procedures that are executed if a page is shipped out.

FIXME: it seems that better than `\pagebreak@pen` would be an unprotected penalty of a special value that would entail output routine processing consisting of the following steps: 3) `\unvbox\@cclv`, 1) set `\clearpage@sw` to `\true@sw`, 2) put down a protected `\do@startcolumn@pen`, 4) take a dead cycle.

The effect would be to liberalize float placement options for the current column as well as further columns that may be output as part of `\clearpage` processing. Of course, it would still be necessary to set `\clearpage@sw` again via an interrupt.

An optimization might be to clear `\clearpage@sw` as part of the same interrupt, but that would actually not work properly, because it is necessary for `\do@endpage` to possibly invoke further visits to the output routine before `clearpage` processing ceases.

```

834 \def\newpage@prep{%
835   \if@noskipsec
836     \ifx \@nodocument\relax
837       \leavevmode
838       \global \@noskipsecfalse
839     \fi
840   \fi
841   \if@inlabel
842     \leavevmode
843     \global \@inlabelfalse
844   \fi
845   \if@nobreak \@nobreakfalse \everypar{}\fi
846   \par
847 }%
848 \def \newpage {%
849   \newpage@prep
850   \do@output@MVL{%
851     \vfil
852     \penalty-\pagebreak@pen
853   }%
854 }%
855 \def\clearpage{%
856   \newpage@prep
857   \do@output@MVL{%
858     \vfil
859     \penalty-\pagebreak@pen
860     \global\@booleantrue\clearpage@sw
861     \protect@penalty\do@startcolumn@pen

```

```

862 \protect@penalty\do@endpage@pen
863 }%
864 \do@output@MVL{%
865 \global\@booleanfalse\clearpage@sw
866 }%
867 }%
868 \def\cleardoublepage{%
869 \clearpage
870 \@if@sw@if@twoside\fi{%
871 \@ifodd\c@page}{}%
872 \null\clearpage
873 }%
874 }{}%
875 }%
876 \@booleanfalse\clearpage@sw

```

`\do@endpage@pen` The penalty `\do@endpage@pen` simply dispatches to the page grid procedure that forces an end page. That procedure should test whether there is anything to ship out (say committed floats), then act accordingly. Note that as part of this work, it should `\unvbox\@cclv`, which has been left boxed up so it can be measured.

```

877 \mathchardef\do@endpage@pen=10007
878 \@namedef{output@-\the\do@endpage@pen}{\csname end@column@\thepagegrid\endcsname}%

```

`\do@newpage@pen` The penalty `\do@newpage@pen` allows a “non-discardable `\newpage`” command: a `\newpage` command that will not disappear at a pagebreak. This visit to the output routine will not be dispatched to an interrupt, rather the natural output routine will be executed, where it will remove the protection box.

Call this routine by executing `\protect@penalty\do@newpage@pen`.

```

879 \mathchardef\do@newpage@pen=10001
880 \expandafter\let\csname output@-\the\do@newpage@pen\endcsname\relax

```

`\@clearfloatplacement` The procedure `\@clearfloatplacement` sets all of the float placement parameters to completely permissive values. The standard values appear as comments.

```

881 \def\@clearfloatplacement{%
882 \global\@topnum \maxdimen % \c@topnumber
883 \global\@toproom \maxdimen % \topfraction\@colht
884 \global\@botnum \maxdimen % \c@bottomnumber
885 \global\@botroom \maxdimen % \bottomfraction\@colht
886 \global\@colnum \maxdimen % \c@totalnumber
887 %\global\@fpmin \z@ % \floatpagefraction\@colht
888 \global\@dbltopnum \maxdimen % \c@dbltopnumber
889 \global\@dbltoproom \maxdimen % \dbltopfraction\@colht
890 \global\@textmin \z@ % \@colht\advance \@textmin -\@dbltoproom
891 \global\@fpmin \z@ % \dblfloatpagefraction\textheight
892 \let\@testfp@gobble
893 \apptdef\@setfloattypecounts{\@fpstype16\advance\@fpstype\m@ne}%
894 }%

```

`\doclearpage` The `\doclearpage` procedure is now obsolete, as is `\makefcolumn`, which it invoked.

```
895 \let\doclearpage\undefined
896 \let\makefcolumn\undefined
```

`\clr@top@firstmark` We want accurate values of `\topmark` and `\firstmark`, but we must deal with the
`\set@top@firstmark` fact that there are many different ways of contributing material to the page. Only
`\@outputpage@tail` upon the first contribution to the page is the value of `\topmark` accurate. However,
with `\firstmark` we must potentially examine each contribution because the first
mark on the page may happen to fall in the last piece of material contributed.

To begin, we define the procedure that initializes the macros to appropriate flag values.

```
897 \def\clr@top@firstmark{%
898   \global\let\saved@@topmark\undefined
899   \global\let\saved@@firstmark\@empty
900   \global\let\saved@@botmark\@empty
901 }%
902 \clr@top@firstmark
```

Note that the flag value for `\saved@@topmark` is `\undefined`, just as one would expect. But that for `\saved@@firstmark` and `\saved@@botmark` is `\@empty`.

Next, we define procedure `\set@top@firstmark`; it will be exercised everywhere material is contributed, capturing the mark values if appropriate.

```
903 \def\set@top@firstmark{%
904   \ifundefined\saved@@topmark{\expandafter\gdef\expandafter\saved@@topmark\expandafter{\@topmark}}%
905   \ifempty\saved@@firstmark{\expandafter\gdef\expandafter\saved@@firstmark\expandafter{\@firstmark}}%
906   \ifempty\@botmark{\expandafter\gdef\expandafter\saved@@botmark\expandafter{\@botmark}}%
907 }%
```

When should `\set@top@firstmark` be called? A good candidate for a universal procedure for handling contributed material is the natural output routine; are any other calls needed?

Yes, in `\save@column` we must execute `\set@top@firstmark` because we are about to save away `\box\@cc1v`, and we will never see its marks again (unless it is unboxed into the MVL), because \TeX lets one access a box's marks only within an output routine that has put that box into `\box\@cc1v`.

As soon as a page is shipped out, we initialize the two macros that hold the values of `\topmark` and `\firstmark`, respectively.

```
908 \appdef\@outputpage@tail{%
909   \clr@top@firstmark
910 }%
```

8.12 Other interfaces to \LaTeX

`\float` The \LaTeX kernel procedures `\float` and `\dblfloat` are treated on an equal
`\dblfloat` footing. Each now takes environment-specific float placement defaults. If none
`\@yfloat` are defined for the calling environment, we apply a default.

```
\fps@
\fpsd@
```

A parameter is passed that will set the width of text within the float, normally `\columnwidth`, and in the "dbl" version, `\textwidth`. However, an environment such as `turnpage` may change the meanings of these macros to allow `turnpage` floats.

Note on `\@xfloat`: the optional argument must come to it fully expanded, because the macro does a weird procedure on this argument, involving `\@onelevel@sanitize`, which I do not understand, and which does not work if not so expanded.

```

911 \def\@float#1{%
912 \ifnextchar[{%
    }]{Brace-matching klotch
913 \@yfloat\width@float{#1}%
914 }{%
915 \ifxundefinedcs{fps@#1}{\expandafter\let\expandafter\fps@\csname fps@#1\endcsname}%
916 \expandafter\@argswap\expandafter{\expandafter[\fps@]}{\@yfloat\width@float{#1}}%
917 }%
918 }%
919 \def\@dblfloat#1{%
920 \@ifnum{\pagegrid@col=\@ne}{%
921 \@float{#1}%
922 }{%
923 \ifnextchar[{%
    }]{Brace-matching klotch
924 \@yfloat\widthd@float{#1}%
925 }{%
926 \@ifxundefinedcs{fpsd@#1}{\expandafter\let\expandafter\fpsd@\csname fpsd@#1\endcsname}%
927 \expandafter\@argswap\expandafter{\expandafter[\fpsd@]}{\@yfloat\widthd@float{#1}}%
928 }%
929 }%
930 }%

931 \def\@yfloat#1#2[#3]{%
932 \@xfloat{#2}[#3]%
933 \hsize#1\linewidth\hsize
934 \let\set@footnotewidth\@empty
935 \minipagefootnote@init
936 }%
937 \def\fps@{tbp}%
938 \def\fpsd@{tp}%
939 \def\width@float{\columnwidth}%
940 \def\widthd@float{\textwidth}%

```

`\end@float` L^AT_EX kernel procedures `\end@float` and `\end@dblfloat` have been changed to work alike; in particular, floats of both classes are deferred into the same queue.

`\end@dblfloat` This measure ensures that they will be placed in their original order, an aspect in which L^AT_EX is broken.

`\end@@float` This measure ensures that they will be placed in their original order, an aspect in which L^AT_EX is broken.

`\check@currbox@count` Note: when retrieving floats from the queues, we can differentiate those of the two categories by the width of the box.

`\minipagefootnote@init`

`\minipagefootnote@here`

Floats are processed via an output routine message, and are checked for sanity in re the float placement options. In the case of full-page-width floats, we ensure that the h and b float placement options are never asserted, because they make no sense.

Note that if we get to the end of the float box and still have pending footnotes, we put them out.

LaTeX Bug note: if a user types `\begintable*[h]`, the float will never succeed in being placed! we try to catch such cases.

Note that the macro `\check@currbox@count` tries to catch cases where the float placement options are such that the float can never be placed.

The calls to `\@iffpsbit` are part of a procedure to deny certain of the float placement parameters: “h” and “b” are not possible, the former because the `\marginpar` mechanism cannot place a full-page-width float within a multicolumn page grid, the latter because nobody has yet written the code to do so (pretty bad reason, I know).

```

941 \def\end@float{%
942 \end@@float{%
943 \check@currbox@count
944 }%
945 }%
946 \def\end@dblfloat{%
947 \@ifnum{\pagegrid@col=\@ne}{%
948 \end@float
949 }{%
950 \end@@float{%
951 \@iffpsbit\@ne{\global\advance\count\@currbox\@m@ne}{}%
952 \@iffpsbit\@four{\global\advance\count\@currbox-4\relax}{}%
953 \global\wd\@currbox\textwidth % Klootch
954 \check@currbox@count
955 }%
956 }%
957 }%
958 \def\end@@float#1{%
959 \minipagefootnote@here
960 \@endfloatbox
961 #1%
962 \@ifnum{\@floatpenalty <\z@}{%
963 \@largefloatcheck
964 \@cons\@currlist\@currbox
965 \@ifnum{\@floatpenalty <-\@Mii}{%
966 \do@output@cclv{\@add@float}%
967 }{%
968 \vadjust{\do@output@cclv{\@add@float}}%
969 \@Esphack
970 }%
971 }{}}%
972 }%

```

The float package of Anselm Lingnau fails when used under `ltxgrid`, but we

can fix things. We also repair a bug in that package.

```
973 \newcommand\float@end@float{%
974 \endfloatbox
975 \global\setbox\@currbox\float@makebox\columnwidth
976 \let\endfloatbox\relax
977 \end@float
978 }%
979 \newcommand\float@end@ltx{%
980 \end@float{%
981 \global\setbox\@currbox\float@makebox\columnwidth
982 \check@currbox@count
983 }%
984 }%
985 \newcommand\newfloat@float[3]{%
986 \@namedef{ext@#1}{#3} %!
987 \let\float@do=\relax
988 \xdef\@tempa{\noexpand\float@exts{\the\float@exts \float@do{#3}}}%
989 \@tempa
990 \floatplacement{#1}{#2}%
991 \@ifundefined{fname@#1}{\floatname{#1}{#1}}{} %!
992 \expandafter\edef\csname ftype@#1\endcsname{\value{float@type}}%
993 \addtocounter{float@type}{\value{float@type}} %!
994 \restylefloat{#1}%
995 \expandafter\edef\csname fnum@#1\endcsname{%
996 \expandafter\noexpand\csname fname@#1\endcsname} %!
997 \expandafter\noexpand\csname the#1\endcsname
998 }
999 \@ifnextchar [%]
1000 {%
1001 \float@newx{#1}%
1002 }-%
1003 \@ifundefined{c@#1}{\newcounter{#1}\@namedef{the#1}{\arabic{#1}}}{}%
1004 }%
1005 }%
1006 \newcommand\newfloat@ltx[3]{%
1007 \@namedef{ext@#1}{#3}%
1008 \let\float@do=\relax
1009 \xdef\@tempa{\noexpand\float@exts{\the\float@exts \float@do{#3}}}%
1010 \@tempa
1011 \floatplacement{#1}{#2}%
1012 \@ifundefined{fname@#1}{\floatname{#1}{#1}}{}%
1013 \expandafter\edef\csname ftype@#1\endcsname\expandafter{\the\c@float@type}%
1014 \addtocounter{float@type}{\value{float@type}}%
1015 \restylefloat{#1}%
1016 \expandafter\edef\csname fnum@#1\endcsname{%
1017 \expandafter\noexpand\csname fname@#1\endcsname}%
1018 \expandafter\noexpand\csname the#1\endcsname
1019 }
1020 \@ifnextchar [%]
```

```

1021  {%
1022  \float@newx{#1}%
1023  }{%
1024  \@ifundefined{c@#1}{\newcounter{#1}\@namedef{the#1}{\arabic{#1}}}{}%
1025  }%
1026 }%
1027 \appdef\document@inithook{%
1028 \@ifxundefined\newfloat{}{%
1029 \ifx{\float@end\float@end@float}{%
1030 \ifx{\newfloat\newfloat@float}{\true@sw}{\false@sw}%
1031 }{\false@sw}%
1032 {%
1033 \class@warn{Repair the float package}%
1034 \let\float@end\float@end@ltx
1035 \let\newfloat\newfloat@ltx
1036 }{%
1037 \class@warn{Failed to patch the float package}%
1038 }%
1039 }%
1040 }%

```

Boolean procedure `\@iffpsbit` is similar to the `\@getfpsbit` of L^AT_EX, except that we do not expose the scratch count register or even change its value.

```

1041 \def\@iffpsbit#1{%
1042 \begingroup
1043 \@tempcnta\count\@currbox
1044 \divide\@tempcnta#1\relax
1045 \@ifodd\@tempcnta{\aftergroup\true@sw}{\aftergroup\false@sw}%
1046 \endgroup
1047 }%

```

In procedure `\check@currbox@count`, we calculate the net float placement directive (encoded into `\count\@currbox`'s least significant four bits). If zero, issue a warning.

```

1048 \def\check@currbox@count{%
1049 \@ifnum{\count\@currbox>\z@}{%
1050 \count@\count\@currbox\divide\count@\sist@n\multiply\count@\sist@n
1051 \@tempcnta\count\@currbox\advance\@tempcnta-\count@
1052 \@ifnum{\@tempcnta=\z@}{%
1053 \ltxgrid@warn{Float cannot be placed}%
1054 }{}%
1055 \expandafter\tally@float\expandafter{\@captive}%
1056 }{}%

```

In this case, the float is a `\marginpar`.

```

1057 }%
1058 }%
1059 \providecommand\minipagefootnote@init{}%
1060 \providecommand\minipagefootnote@here{}%
1061 \providecommand\tally@float[1]{}%

```

`\@specialoutput` The `\@add@float` procedure used to reside in standard L^AT_EX's `\@specialoutput`, which is no more.

Historical Note: `\@specialoutput` and Lamport's method of an output routine dispatcher is the genesis of our more powerful and refined way of using T_EX's output routine to safely accomplish page makeup tasks. To it and to him we owe acknowledgement and thanks.

```
1062 \let\@specialoutput\@undefined
```

`\@add@float` In the following, we do not need to execute `\@reinserts`, which was wrong anyway, as you cannot reliably recover insertions when they split (unless you have a way of reinserting the captured insertion ahead of the split-off part).

Now that full-page-width floats are being processed the same as column floats, we have to nip in here and cause them always to be deferred.

At the very end, the `\vsize` is adjusted for any newly committed float.

```
1063 \def\@add@float{%
1064   \@pageht\ht\@cclv\@pagedp\dp\@cclv
1065   \unvbox\@cclv
1066   \@next\@currbox\@currlist{%
1067     \csname @floatselect@sw@thepagegrid\endcsname\@currbox{%
1068       \@ifnum{\count\@currbox>\z@}{%
1069         \advance \@pageht \@pagedp
```

Do not assume `\holdinginsertsis` cleared:

```
1070   \advance \@pageht \vsize \advance \@pageht -\pagegoal
```

Commit an 'h' float:

```
1071   \@addtocurcol
1072   }{%
1073   \@addmarginpar
1074   }%
1075   }{%
1076   \@resethfps
1077   \@cons\@deferlist\@currbox
1078   }%
1079   }{\@latexbug}%
1080   \@ifnum{\outputpenalty<\z@}{%
1081     \@ifsw@if@nobreak\fi{%
1082       \nobreak
1083     }{%
1084       \addpenalty \interlinepenalty
1085     }%
1086   }{}%
1087   \set@vsize
1088   }%
```

`\@reinserts` The `\@reinserts` procedure of standard L^AT_EX is now obsoleted (it had been erroneous anyway).

```
1089 \let\@reinserts\@undefined
```

`\@addtocurcol` We modify the `\@addtocurcol` procedure of standard L^AT_EX so that a float placed “here” may break over pages.

```

1090 \def \@addtocurcol {%
1091   \@insertfalse
1092   \@setfloatypecounts
1093   \ifnum \@fpstype=8
1094   \else
1095     \ifnum \@fpstype=24
1096     \else
1097       \@flsettextmin
1098       \advance \@textmin \@textfloatsheight
1099       \@reqcolroom \@pageht
1100       \ifdim \@textmin>\@reqcolroom
1101         \@reqcolroom \@textmin
1102       \fi
1103       \advance \@reqcolroom \ht\@currbox
1104       \ifdim \@colroom>\@reqcolroom
1105         \@flsetnum \@colnum
1106         \ifnum \@colnum>\z@
1107           \@bitor\@currtype\@deferlist
1108           \if@test
1109             \else
1110               \@bitor\@currtype\@botlist
1111               \if@test
1112                 \@addtobot
1113             \else
1114               \ifodd \count\@currbox
1115                 \advance \@reqcolroom \intextsep
1116                 \ifdim \@colroom>\@reqcolroom
1117                   \global \advance \@colnum \m@ne
1118                   \global \advance \@textfloatsheight \ht\@currbox
1119                   \global \advance \@textfloatsheight 2\intextsep
1120                   \@cons \@midlist \@currbox
1121                   \if@nobreak
1122                     \nobreak
1123                     \@nobreakfalse
1124                   \everypar{}%
1125                 \else
1126                   \addpenalty \interlinepenalty
1127                 \fi
1128                 \vskip \intextsep
1129                 \unvbox\@currbox %A0
1130                 \penalty\interlinepenalty
1131                 \vskip\intextsep
1132                 \ifnum\outputpenalty <-\@Mii \vskip -\parskip\fi
1133                 \outputpenalty \z@
1134               \@inserttrue
1135             \fi
1136           \fi

```

```

1137             \if@insert
1138             \else
1139                 \@addtotoporbob
1140             \fi
1141         \fi
1142     \fi
1143 \fi
1144 \fi
1145 \fi
1146 \fi
1147 \if@insert
1148 \else
1149     \@resethfps
1150     \@cons\@deferlist\@currbox
1151 \fi
1152 }%

```

`\if@twocolumn` The `\newif` switch `\if@twocolumn` is entirely unused. However its access words are invoked by L^AT_EX's `\document` procedure, so we de-fang it.

```

1153 \@twocolumnfalse
1154 \let\@twocolumntrue\@twocolumnfalse

```

`\@addmarginpar` The procedure `\@addmarginpar` used to access `\if@twocolumn`, but that switch is not reliable; the better way is to use `\thepagegrid`. We establish a convention for a page-grid-oriented procedure, e.g., `\@addmarginpar@one`, that emits a boolean, telling this procedure whether to set the marginpar on the left or right.

```

1155 \def\@addmarginpar{%
1156   \@next\@marbox\@currlist{%
1157     \@cons\@freelist\@marbox\@cons\@freelist\@currbox
1158   }\@latexbug
1159   \setbox\@marbox\hb@xt@\columnwidth{%
1160     \csname @addmarginpar@\thepagegrid\endcsname{%
1161       \hskip-\marginparsep\hskip-\marginparwidth
1162       \box\@currbox
1163     }{%
1164       \hskip\columnwidth\hskip\marginparsep
1165       \box\@marbox
1166     }%
1167   \hss
1168 }%
1169 \setbox\z@\box\@currbox
1170   \@tempdima\@mparbottom
1171   \advance\@tempdima -\@pageht
1172   \advance\@tempdima\ht\@marbox
1173 \@ifdim\@tempdima >\z@}{%
1174   \@latex@warning@no@line {Marginpar on page \thepage\space moved}%
1175 }{%
1176   \@tempdima\z@
1177 }%

```

```

1178 \global\@mparbottom\@pageht
1179 \global\advance\@mparbottom\@tempdima
1180 \global\advance\@mparbottom\dp\@marbox
1181 \global\advance\@mparbottom\marginparpush
1182 \advance\@tempdima -\ht\@marbox
1183 \global\setbox \@marbox
1184         \vbox {\vskip \@tempdima
1185                 \box \@marbox}%
1186 \global \ht\@marbox \z@
1187 \global \dp\@marbox \z@
1188 \kern -\@pagedp
1189 \nointerlineskip
1190 \box\@marbox
1191 \nointerlineskip
1192 \hbox{\vrule \@height\z@ \@width\z@ \@depth\@pagedp}%
1193 }%

```

turnpage Any float (viz., figure or table) within the scope of this environment will be a turnpage float: It will be assumed to occupy an entire page (constitute a float page), the width will be `\textheight`, the height `\textwidth`, and the entire float will be presented rotated 90 degrees.

The implementation requires the services of the `\rotatebox` command, so we supply a dummy definition that explains things to the user.

```

1194 \newenvironment{turnpage}{%
1195 \def\width@float{\textheight}%
1196 \def\widthd@float{\textheight}%
1197 \appdef\@endfloatbox{%
1198 \ifxundefined\@currbox{%
1199 \ltxgrid@warn{Cannot rotate! Not a float}%
1200 }{%
1201 \setbox\@currbox\vbox to\textwidth{\vfil\unvbox\@currbox\vfil}%
1202 \global\setbox\@currbox\vbox{\rotatebox{90}{\box\@currbox}}%
1203 }%
1204 }%
1205 }{%
1206 }%
1207 \def\rotatebox@dummy#1#2{%
1208 \ltxgrid@warn{You must load the graphics or graphicx package in order to use the turnpage envi.
1209 #2%
1210 }%
1211 \appdef\document@inithook{%
1212 \ifxundefined\rotatebox{\let\rotatebox\rotatebox@dummy}{}%
1213 }%

```

8.13 One-off output routines

These procedures are executed in lieu of `\the\output` when the output penalty has the associated flag value.

`output@-1073741824` The first one-off output routine handles the end of the job, wherein L^AT_EX executes `\@@end`, and breaks to the output with a penalty of "4000000 = $2^{32}/4 = 1073741824$. We simply discard `\box\@cclv` and leave. This means that L^AT_EX is obligated to do `\clearpage` as part of its `\end{document}` processing, otherwise material will be lost.

```
1214 \@namedef{output@-1073741824}{%
1215 \deadcycles\z@
```

```
    %\showbox\@cclv
    %
```

```
1216 \void@cclv
1217 }%
```

`\save@column@pen` The one-off output routine associated with `\penalty\save@column@pen` will be called within a sequence of three such routines by `\execute@message@or` its companion routine `\execute@message@insert`. This procedure must save away any the current page and preserve marks.

```
1218 \mathchardef\save@column@pen=10016
1219 \@namedef{output@-\the\save@column@pen}{\save@column}%
```

`\@cclv@saved` We take over the `\@holdpg` box register. Hereafter, we no longer use the `\@holdpg` box register, so let the world know. This should decisively break packages that assume standard L^AT_EX. Breaking decisively is preferred to quietly proceeding erroneously.

```
1220 \let \@cclv@saved \@holdpg
1221 \let \@holdpg \@undefined
```

`\save@column` The procedure `\save@column` does the actual work of saving away the material on the page. It is invoked both by `\save@column@pen` and by `\save@column@insert@pen`. We save `\box\@cclv` and the primitive `\@@topmark`.

```
1222 \def\save@column{%
1223 \@ifvoid\@cclv@saved{%
1224 \set@top@firstmark
1225 \global\@topmark@saved\expandafter{\@@topmark}%
1226 }-}%
1227 \global\setbox\@cclv@saved\vbox{%
1228 \@ifvoid\@cclv@saved{}-%
1229 \unvbox\@cclv@saved
1230 \marry@baselines
1231 }%
1232 \unvbox\@cclv
1233 \lose@breaks
1234 \remove@lastbox
1235 }%
1236 }%
1237 \newtoks\@topmark@saved
```

`\prep@cclv` The procedure `\prep@cclv` is used by message handlers to set up their environment to ape that of the usual output routine, with the boxed-up page in `\box@cclv`. Here, we retrieve the material from `\cclv@saved`, where it was saved away by the one-off output routine associated with `\save@column@pen`.

```
1238 \def\prep@cclv{%
1239 \void@cclv
1240 \setbox@cclv\box@cclv@saved
1241 \vbadness@M
1242 }%
```

`\save@column@insert@pen` The one-off output routine associated with `\penalty\save@column@insert@pen` is similar to that of `\save@column@pen` augmented with the processing of insertions. It is called by `\execute@message@insert` (i.e., at a grid change) and saves away the current page and preserves marks. In addition, it saves away any insertions that fall on the current page. As with the regular output routine, it executes in two phases, first with `\holdinginserts` set, then with it cleared.

```
1243 \mathchardef\save@column@insert@pen=10017
```

```
1244 \@namedef{output@-\the\save@column@insert@pen}{\toggle@insert{\savecolumn@holding}{\savecolumn@
```

The procedure `\savecolumn@holding` is the first phase of saving a column with its inserts. This phase must detect and remedy the one circumstance that will confound our efforts to propagate marks. It is similar to `\output@holding`, except that we have to deal with the protection box, which must remain, because the messaging mechanism is being used.

If it appears that we have the pathological “Big Bad Box” case at hand, we use the `\dead@cycle@repair@protected` procedure instead of `\dead@cycle` to do our dead cycle.

```
1245 \def\savecolumn@holding{%
1246 \if@exceed@pagegoal{\unvcopy@cclv\remove@lastbox}{%
1247 \setbox@z\vbox{\unvcopy@cclv\remove@lastbox}%
1248 \outputdebug@sw{\trace@box@z}{-%
1249 \dimen@ht@cclv\advance\dimen@-ht@z@
1250 \dead@cycle@repair@protected\dimen@
1251 }{%
1252 \dead@cycle
1253 }%
1254 }%
```

The procedure `\save@column@moving` is the second phase of saving a column with its inserts. Now that `\holdinginserts` is cleared, we can look in the various `\insert` registers for our inserts (at present there is only one, `\footins`). if anything is there, we save it away and ask for another cycle (because it may have split).

Note that the message that is about to be executed had better deal properly with the contents of the `\footins@saved` box.

```
1255 \def\savecolumn@moving{%
1256 \cclv@nontrivial@sw{%
1257 \save@column
```



```

1258 }{%
1259 \void@ccclv
1260 }%
1261 \@ifvoid\footins{}{%
1262 \outputdebug@sw{\trace@box\footins}{}%
1263 \global\setbox\footins@saved\ vbox{\unvbox\footins@saved\marry@baselines\unvbox\footins}%
1264 \protect@penalty\save@column@insert@pen
1265 }%
1266 }%
1267 \newbox\footins@saved

```

`\save@message@pen` The one-off output routine associated with `\penalty\save@message@pen` saves away the message that has been passed. This procedure is penultimate in a sequence of one-off output routine calls; earlier ones have saved away the MVL and preserved marks, the last executes the message.

Note that we are passing tokens to TeX's primitive `\mark` mechanism, so we must ensure that they are not inappropriately expanded. We use the same mechanism for all such cases, namely `\let@mark`.

Note: we expect that `\box\ccclv`'s contents are well known: `\topskip`, protection box, and a `\mark`, the latter containing the message. But if we came here via `\penalty10017`, there might be an `\insert` node present as well, because a footnote may have split. Because this procedure simply voids out `\box\ccclv`, such material would be lost. Perhaps we can repair things by manipulating the `\insert` mechanism temporarily.

```

1268 \mathchardef\save@message@pen=10018
1269 \@namedef{output@-\the\save@message@pen}{\save@message}%
1270 \def\save@message{%
1271 \void@ccclv
    FIXME: what if \box\ccclvis not empty?
1272 \toks@\expandafter{\@@firstmark}%
1273 \expandafter\gdef\expandafter\@message@saved\expandafter{\the\toks@}%
1274 \expandafter\do@@mark\expandafter{\the\@topmark@saved}%
1275 }%
1276 \gdef\@message@saved{}%

```

`\execute@message@pen` The one-off output routine associated with `\execute@message@pen` simply executes the given message. It is last in a sequence of one-off output routine calls; earlier ones have saved all that require saving.

```

1277 \mathchardef\execute@message@pen=10019
1278 \@namedef{output@-\the\execute@message@pen}{\@message@saved}%

```

8.14 Output messages

Message handlers are procedures that execute output messages, tokens that are passed to the output routine for execution in an environment appropriate to page makeup.

How it works. We put down three large negative penalties, each of which will be handled by the output dispatcher (*not* the natural output routine), each penalty being protected by a removable, non-discardable item (i.e., a box). Either three or four invocations of one-off output routines are involved per message.

We make the last of the three protection boxes have a depth equal to the value of `\prevdepth` that was current when the procedure is called. This effectively restores `\prevdepth`.

In each case, the one-off output routine will remove the extraneous box we have inserted. And the second and third one-off routines will simply void `\box\@cclv`, because its contents are entirely artificial.

FIXME: not so! If `\holdinginserts` is cleared, that box may have an insert node; it must be preserved, too.

The first routine saves away the current column contents and remembers the `\topmark` for later use. There is a variant routine that first clears `\holdinginserts`, so that the message can handle any inserts present in the boxed-up page; this of course entails yet another visit to the output routine.

The penultimate routine saves away the tokens transmitted in via the `\@@mark:` the argument of the macro. These tokens are of course the very thing we wish to execute within the safety of the output routine. It also puts down a mark containing the `\topmark` tokens saved by the first routine. By this means, the mark, which we have clobbered, is restored.

The last routine simply executes the given tokens. In the course of doing this, it must take care of `\box\@cclv`, either by shipping it out, or by `\unvboxing` it onto the MVL.

`\execute@message` The procedure `\execute@message` simply calls the utility procedure `\@execute@message` with a penalty value for the standard treatment.

```
1279 \def\execute@message{%
1280 \@execute@message\save@column@pen
    Implicit second argument
1281 }%
```

`\execute@message@insert` The procedure `\execute@message@insert` is like `\execute@message` in all respects except that the penalty value is `\save@column@insert@pen`, which arranges for the message handler involved to deal with the page's insertions. At the same time, we prepare the `\footins` box so that these insertions can be dealt with.

Note: If more insertions are added to L^AT_EX (presumably via `\newinsert`), then they must be dealt with in a way entirely analogous to `\footins`.

```
1282 \def\execute@message@insert#1{%
1283 \@execute@message\save@column@insert@pen{\setbox \footins \box \footins@saved#1}%
1284 }%
```

`\@execute@message` The utility procedure `\@execute@message` is called by `\execute@message` and `\execute@message@insert`. We prepare by creating a `\vbox` containing all the needed nodes and proceed by simply `\unvboxing` that box onto the MVL. We

ensure that `\box\@cclv` is properly set up for the output message handler by always inserting `\prep@cclv` in advance of the argument.

Note that each one-off output routine is invoked effectively the same as `\protect@penalty`, except that the second invocation involves an additional `\mark` node, and the third a specially prepared protection box.

Note also that T_EX's primitive `\mark` is called here without any expansion protection. This is the only place where it is called that way, but it's OK because those tokens have been pre-expanded by procedures that call `\execute@message`. **FIXME:** all procedures calling `\execute@message` must pre-expand their tokens!

```

1285 \long\def\execute@message#1#2{%
1286 \begingroup
1287 \dimen@prevdepth\@ifdim{\dimen@<z@}{\dimen@z@}{}%
1288 \setboxz@\vbox{%
1289 \protect@penalty#1%
1290 \protection@box
1291 \toks@{\prep@cclv#2}%
1292 \@mark{\the\toks@}%
1293 \penalty-\save@message@pen

% \hbox{\vrule\@heightz@\@widthz@\@depth\dimen@}%
%

1294 \setboxz@\null\dpz@\dimen@\htz@-\dimen@
1295 \nointerlineskip\boxz@
1296 \penalty-\execute@message@pen
1297 }\unvboxz@
1298 \endgroup
1299 }%

```

`\do@output@cclv` The procedure `\do@output@cclv` provides access to message handlers at their simplest. The message will execute in the usual environment of the output routine, with the boxed-up page in `\box\@cclv`, and we assume that `\holdinginserts` remains set. This procedure must be invoked within main vertical mode; it is the obligation of the macro writer to ensure that this is the case.

```

1300 \def\do@output@cclv{\execute@message}%

```

`\do@output@MVL` The procedure `\do@output@MVL`, like `\do@output@cclv`, is an interface for messages, but provides two additional services: the command may also be invoked in horizontal mode, and the message handler will execute with the MVL unboxed.

```

1301 \def\do@output@MVL#1{%
1302 \@ifvmode{%
1303 \begingroup\execute@message{\unvbox\@cclv#1}\endgroup
1304 }{%
1305 \@ifhmode{%
1306 \adjust{\execute@message{\unvbox\@cclv#1}}%
1307 }{%
1308 \@latexerr{\string\do@output@MVL\space cannot be executed in this mode!}\@eha
1309 }%

```

```
1310 }%
1311 }%
```

`\lose@breaks` The purpose of this procedure is to get rid of all the extraneous `\penalty\@M` nodes that tend to build up in the MVL.

```
1312 \def\lose@breaks{%
1313 \loopwhile{%
1314 \count@\lastpenalty
1315 \@ifnum{\count@=\@M}{%
    Note: 10000 is a TeX magic number!
1316 \unpenalty\true@sw
1317 }{%
1318 \false@sw
1319 }%
1320 }%
1321 }%
```

`\removestuff` `\removestuff` is a document-level command that removes the bottom skip glue item from the MVL.

```
1322 \def\removestuff{\do@output@MVL{\unskip\unpenalty}}%
```

`\removephantombox` The procedure `\removephantombox` is a special-purpose message handler exclusively for preventing incorrect spacing above display math. It must be issued in horizontal mode within the phantom paragraph generated when display math starts up in vertical mode.

```
1323 \def\removephantombox{%
1324 \vadjust{%
1325 \execute@message{%
1326 \unvbox\@cclv
1327 \remove@lastbox
1328 \unskip
1329 \unskip
1330 \unpenalty
1331 \penalty\predisplayskip
1332 \vskip\abovedisplayskip
1333 }%
1334 }%
1335 }%
```

`\addstuff` `\addstuff` is a document-level command that adds penalty, glue, or both to the MVL. The penalty and glue items are rearranged so that all penalties nodes precede all the glue nodes, which is the canonical arrangement.

```
1336 \def\addstuff#1#2{\edef\@tempa{\noexpand\do@output@MVL{\noexpand\@addstuff{#1}{#2}}}\@tempa}%
1337 \def\@addstuff#1#2{%
1338 \skip@\lastskip\unskip
1339 \count@\lastpenalty\unpenalty
1340 \@ifempty{#1}{\penalty#1\relax}%
1341 \@ifnum{\count@=\z@}{\penalty\count@}%

```

```

1342 \vskip\skip@
1343 \@ifempty{#2}{}\vskip#2\relax}%
1344 }%

```

`\replacestuff` `\replacestuff` is a document-level command similar to `\addstuff`; but it replaces penalty, glue, or both in the MVL. The penalty and glue items are rearranged so that all penalties nodes precede all the glue nodes, which is the canonical arrangement.

```

1345 \def\replacestuff#1#2{\edef\tempa{\noexpand\do@output@MVL{\noexpand\@replacestuff{#1}{#2}}}\@t
1346 \def\@replacestuff#1#2{%
1347 \skip@\lastskip\unskip
1348 \count@\lastpenalty\unpenalty
1349 \@ifempty{#1}{}\@%
1350 \@ifnum{\count@>\@M}{}\@%
1351 \@ifnum{\count@=\z@}{\count@=#1\relax}{\@%
1352 \@ifnum{\count@<#1\relax}{}\@%
1353 \count@=#1\relax
1354 }%
1355 }%
1356 }%
1357 }%
1358 \@ifnum{\count@=\z@}{\penalty\count@}%
1359 \@ifempty{#2}{}\@%
1360 \@tempskipa#2\relax
1361 \@ifdim{\z@>\@tempskipa}{\@%
1362 \advance\skip@-\@tempskipa
1363 }{\@%
1364 \@ifdim{\skip@>\@tempskipa}{}\@%
1365 \skip@\@tempskipa
1366 }%
1367 }%
1368 }%
1369 \vskip\skip@
1370 }%

```

`\move@insertions` In order to avoid bolluxing up `\insert` registers by our one-off output routines,
`\hold@insertions` we set `\holdinginserts` to zero by default and only clear it (briefly) while we handle cases where we want inserts to show up.

```

1371 \def\move@insertions{\global\holdinginserts\z@}%
1372 \def\hold@insertions{\global\holdinginserts\@ne}%
1373 \hold@insertions
1374 \def\toggle@insert#1#2{%
1375 \@ifnum{\holdinginserts>\z@}{\move@insertions#1}{\hold@insertions#2}%
1376 }%

```

8.15 Messages to alter the page grid

Here is the implementation of the grid-switching procedures. We perform two checks when changing the page grid; first to ensure that the target page grid is

known (defensive programming), second to ensure that the switch is a non-trivial one. The latter check must be performed within the safety of the output routine, so requires using an output message. Thus, a grid change requires two messages, for a total of six visits to the output routine.

`\do@columngrid` Utility procedure `\do@columngrid` changes the page grid. Note that this command forces an end to the current paragraph. This is necessary, because a page grid change makes no sense unless we can alter the `\hsize` before commencing to typeset the following paragraph. So the command should never be executed in horizontal mode anyway.

```

1377 \def\do@columngrid#1#2{%
1378   \par
1379   \expandafter\let\expandafter\@tempa\csname open@column@#1\endcsname
1380   \@ifx{\relax\@tempa}{%
1381     \ltxgrid@warn{Unknown page grid #1. No action taken}%
1382   }{%
1383     \do@output@MVL{\start@column{#1}{#2}}%
1384   }%
1385 }%

```

`\start@column` Procedure `\start@column` lays down the interrupts to switch the page grid. If the change to the page grid would have been trivial, it bails out. It seems a reasonable tradeoff of processing versus security: once we commit to changing the page grid, we clear `\holdinginserts`, so there is no turning back.

Note that the second argument to the macro allows us to pass an argument to the page grid that is starting up. This can be handy, because a single procedure can handle multiple page grids, differing only by the value of a parameter.

FIXME: this means that you cannot switch between mlt page grids in a single step. But do we want to do this, at all, at all?

```

1386 \def\start@column#1#2{%
1387   \def\@tempa{#1}\@ifx{\@tempa\thepagegrid}{%
1388     \ltxgrid@info{Already in page grid \thepagegrid. No action taken}%
1389   }{%
1390     \expandafter\execute@message@insert
1391     \expandafter{%
1392       \csname shut@column@\thepagegrid\expandafter\endcsname
1393       \csname open@column@#1\endcsname{#2}%
1394       \set@vsize
1395     }%
1396   }%
1397 }%

```

`\thepagegrid` The macro `\thepagegrid` tracks what kind of page grid we are in.

Note: Access `\thepagegrid` only within the safety of the output routine.

Warning: The page grid should be changed only within the safety of the output routine. People who write multicol page grid mechanisms appear not to understand the matter, so they should particularly heed this warning. Think about

it: obviously L^AT_EX did so, which is why his `\twocolumn` command forced a pagebreak, which is limiting, but safe.

```
1398 \def\thepagegrid{one}%
```

8.16 Application Note: implementing a page grid

If you want to create a new page grid for L^AT_EX, you must define five procedures with specific names: `\open@column@name`, `\shut@column@name`, `\end@column@name`, `\output@column@name`, and `\@addmarginpar@name`, where “name” is the name of your page grid.

The procedure `\open@column@name` starts the new page grid. It should define `\thepagegrid`, deal with `\box\pagesofar` and `\box\footsofar` (perhaps by leaving them alone), and it should set the values of L^AT_EX’s page layout parameters for the column size and height.

The procedure `\shut@column@name` should expect to be called with `\holdinginserts` cleared (it can assume that `\holdinginserts` will automatically be restored). It should properly deal with insertions (like footnotes); calling `\@makecolumn` with an argument of `\false@sw` will do this. It should know that the page grid is being terminated in the middle of a page, so it should make arrangements to carry the footnotes down to the bottom of the column or page, and it should possibly salt away the material for later incorporation into the page. The box registers `\footsofar` and `\pagesofar` are customarily used for this purpose.

The procedure `\end@column@name` should kick out a possibly short page containing all the floats committed to the page. It will be invoked during `\clearpage` processing. After that, it should `\unvbox\@cclv`.

The procedure `\output@column@name` should ship out or commit the current `\@outputbox`. In a one-column layout, you ship out; in a multicolumn layout, you commit the box as the contents of a particular column, and if that column is the last, you ship out.

The procedure `\@addmarginpar@name` should return a boolean (either `\true@sw` or `\false@sw` or an equivalent) to tell the marginpar mechanism to place the marginal material to the right or left, respectively.

You can use the existing page grids “one” and “mlt” as a point of departure for creating others. The former can be the basis for, say, a single-column page grid with a side column.

```
\pagesofar
\footsofar 1399 \newbox\pagesofar
            1400 \newbox\footsofar
            1401 \def\combine@foot@inserts{%
            1402 \@ifvoid\footins{%
            1403 }{%
            1404 \@ifvoid\footsofar{%
            1405 \global\setbox\footsofar\box\footins
            1406 }{%
            1407 \global\setbox\footsofar\vbox\bgroup
```

```

1408   \unvbox\footsofar
1409   \marry@baselines
1410   \unvbox\footins
1411   \egroup
1412 }%
1413 }%
1414 }%

```

8.16.1 One-column page grid

`\onecolumngrid` Here are all the procedures necessary for the standard page grid named “one”: a single column layout. It is, of course, L^AT_EX’s familiar `\onecolumn` layout. We begin with the procedure exposed to the style writer. This is, however, not a L^AT_EX command; users should not change the page grid.

```

\end@column@one 1415 \newcommand\onecolumngrid{\do@columngrid{one}{\@ne}}%
\output@column@one
\@addmarginpar@one

```

Note that a document class that issues the command `\onecolumn` will break. This includes L^AT_EX’s standard classes.dtx-based classes: if your class descends from one of these, you must expunge it of all such commands.

```
1416 \let\onecolumn\undefined
```

The procedure `\open@column@one` takes advantage of the special nature of the one-column page grid to deal with `\box\pagesofar`, therefore it must also reset `\@colroom`.

```

1417 \def\open@column@one#1{%
1418   \unvbox\pagesofar
1419   \gdef\thepagegrid{one}%
1420   \global\pagegrid@col#1%
1421   \global\pagegrid@cur\@ne
1422   \set@colht
1423   \set@column@hsize\pagegrid@col
1424 }%

```

The procedure `\shut@column@one` saves away the one-column material into the box register `\pagesofar`. Because it is called from a message handler, we are assured that marks are properly taken care of.

This instance of `\@makecolumn` is building a column for saving into `\pagesofar`.

```

1425 \def\shut@column@one{%
1426   \@makecolumn>false@sw
1427   \global\setbox\pagesofar\vbox{\unvbox\@outputbox\recover@footins}%
1428   \combine@foot@inserts
1429   \set@colht
1430 }%

```

The procedure `\float@column@one` takes care of a float column that has been built by `\@tryfcolumn`, in the single-column page grid.

This instance of `\@makecolumn` is followed by `\@outputpage`: it is building a column for `\shipout`, rather than for saving into `\pagesofar`.

```
1431 \def\float@column@one{%
```



```

1432 \@makecolumn>true@sw
1433 \@outputpage
1434 }%

```

The procedure `\end@column@one` is executed at the end of `\clearpage` processing, if we were in a one-column page grid, once all permissive float pages have been shipped out. At this point, one could perhaps assume that nothing more need be done, but let us anyway test for committed floats and force a shipout.

FIXME: this procedure does the same as `\end@column@mlt` (except for the test of `\@ifx@empty\@dbltoplist`): the two could almost be the same procedure.

I have changed this procedure to avoid the testing it once did: it simply puts down interrupts, upon which it relies to correctly do what `\clearpage` requires.

```

1435 \def\end@column@one{%
1436 \unvbox\@cclv\remove@lastbox
1437 \protect@penalty\do@newpage@pen
1438 }%

```

The procedure `\output@column@one` is dispatched from the output routine when we have completed a page (that is, a column in a one-column page grid); it ships out the page using the `\@outputpage`. It will be followed up with an output routine message to prepare a new column.

Query: by what mechanism do the footnotes get placed onto such a page?

```

1439 \def\output@column@one{%
1440 \@outputpage
1441 }%

```

The following procedure determines which side of the page a marginpar will appear. It reproduces the behavior of standard L^AT_EX.

```

1442 \def\@addmarginpar@one{%
1443 \@if@sw@if@mparswitch\fi{%
1444 \@ifodd\c@page{\false@sw}{\true@sw}%
1445 }{\false@sw}{%
1446 \@if@sw@if@reversemargin\fi{\false@sw}{\true@sw}%
1447 }{%
1448 \@if@sw@if@reversemargin\fi{\true@sw}{\false@sw}%
1449 }%
1450 }%

```

The following procedure yields a Boolean value; it determines whether a float in the deferred queue is appropriate for placing. In the one-column grid, all floats are so.

```

1451 \def\@floatselect@sw@one#1{\true@sw}%
1452 \def\onecolumngrid@push{%
1453 \do@output@MVL{%
1454 \@ifnum{\pagegrid@col}=\@one}{%
1455 \global\let\restorecolumngrid\@empty
1456 }{%
1457 \xdef\restorecolumngrid{%
1458 \noexpand\start@column{\thepagegrid}{\the\pagegrid@col}%
1459 }%

```

```

1460   \start@column{one}{\@ne}%
1461   }%
1462 }%
1463 }%
1464 \def\onecolumngrid@pop{%
1465   \do@output@MVL{\restorecolumngrid}%
1466 }%

```

8.16.2 Two-column page grid

`\twocolumngrid` Here are all the procedures necessary for the standard page grid named “mlt”:
`\open@column@mlt` the multi-column page grid. With an argument of “2”, it is, of course, L^AT_EX’s
`\shut@column@mlt` familiar `\twocolumn` layout.
`\end@column@mlt` We start with the procedure to switch to the two-column page grid.

```

\output@column@mlt 1467 \newcommand\twocolumngrid{\do@columngrid{mlt}{\tw@}}%

```

`\@addmarginpar@mlt` The corresponding command of L^AT_EX is obsolete.

```

1468 \let\twocolumn\@undefined

```

Of course, `\@topnewpage` is also obsolete. Just do

```

\clearpage\onecolumngrid;vertical mode material;\twocolumngrid.

```

```

1469 \let\@topnewpage\@undefined

```

If your document class descends from one of L^AT_EX’s standard classes.dtx-derived classes, it will break. You must expunge from it all such commands.

```

1470 \def\open@column@mlt#1{%
1471   \gdef\thepagegrid{mlt}%
1472   \global\pagegrid@col#1%
1473   \global\pagegrid@cur\@ne
1474   \set@column@hsize\pagegrid@col
1475   \set@colht
1476 }%

```

The procedure `\shut@column@mlt` ends the current column, balances the columns, and salts away all in `\pagesofar`. Because it is called in a message handler, we are assured that marks are handled properly. Attention: because this procedure balances columns, all footnotes are held aside in `\footsofar` for placement at the bottom of the page.

Bug note: the last macro executed by this procedure is `\set@colht`, but had been erroneously `\set@colroom`. I now believe that the latter should be changed pretty much everywhere to the former.

This instance of `\@makecolumn` is building material for `\pagesofar`, rather than for `\shipout`.

```

1477 \def\shut@column@mlt{%
1478   \cc@lv@nontrivial@sw{%
1479     \@makecolumn>false@sw
1480     \@ifnum{\pagegrid@cur<\pagegrid@col}{%
1481       \expandafter\global\expandafter\setbox\csname col@the\pagegrid@cur\endcsname\box\@outputbox

```

```

1482   \global\advance\pagegrid@cur\@ne
1483   }{}%
1484 }{}%
1485   \void@cclv
1486 }%
1487 \@ifnum{\pagegrid@cur>\@ne}{%
1488   \csname balance@the\pagegrid@col\endcsname
1489   \grid@column{}%
1490   \@combinepage>false@sw
1491   \@combinedblfloats
1492   \global\setbox\pagesofar\box\@outputbox
1493   \show@pagesofar@size
1494 }{}%
1495   \set@colht
1496 }%

```

The procedure `\float@column@mlt` takes care of a float page that has been built by `\@tryfcolumn`, in the multi-column page grid. It is coincidentally identical to what happens in `\do@startpage` when a page needs to be shipped out.

```

1497 \def\float@column@mlt{%
1498   \@output@combined@page
1499 }%

```

The procedure `\end@column@mlt` is executed at the end of `\clearpage` processing, if we were in a multi-column page grid, once all permissive float pages have been shipped out. If no floats are committed and if no columns are yet filled, we have nothing to do. Otherwise, we kick out a column and try again.

Note that in our code to kick out a column, we must deal properly with the case where the column is trivial: it will have nothing but `\topskip` glue plus a protection box. We substitute an ordinary `\null` for the protection box.

```

1500 \def\end@column@mlt{%
1501   \@ifx@empty\@toplist{%
1502     \@ifx@empty\@botlist{%
1503       \@ifx@empty\@dbltoplist{%
1504         \@ifx@empty\@deferlist{%
1505           \@ifnum{\pagegrid@cur=\@ne}{%
1506             \false@sw
1507           }{}%
1508             \true@sw
1509           }%
1510         }{}%
1511         \true@sw
1512       }%
1513     }{}%
1514     \true@sw
1515   }%
1516 }{}%
1517   \true@sw
1518 }%
1519 }{}%

```

```

1520 \true@sw
1521 }%
1522 % true = kick out a column and try again
1523 {%
1524 \@cclv@nontrivial@sw{%
1525 \unvbox\@cclv\remove@lastbox
1526 }{%
1527 \unvbox\@cclv\remove@lastbox\unskip\null
1528 }%
1529 \protect@penalty\do@newpage@pen
1530 \protect@penalty\do@endpage@pen
1531 }{%
1532 \unvbox\@cclv\remove@lastbox
1533 }%
1534 }%

```

The procedure `\output@column@mlt` (cf. `\output@column@one`) is dispatched from the output routine when we have completed a column in a multi-column page grid). (It replaces the `\@outputdblcol` of standard L^AT_EX.) If a complete set of columns is at hand, it ships out the page and lays down an interrupt for `\do@startpage@pen`, which will commit the full-page-width floats to the next page. Like `\output@column@mlt`, this is followed by an output routine message to prepare a new column.

If a page needs to be shipped out, it uses the same mechanism as `\do@startpage`.

```

1535 \def\output@column@mlt{%
1536 \@ifnum{\pagegrid@cur<\pagegrid@col}{%
1537 \expandafter\global\expandafter\setbox\c@name col@the\pagegrid@cur\endcsname\box\@outputbox
1538 \global\advance\pagegrid@cur@one
1539 }{%
1540 \set@adj@colht\dimen@
1541 \grid@column{}%
1542 \@output@combined@page
1543 }%
1544 }%

```

The procedure `\output@column@mlt` obsoletes L^AT_EX's `\@outputdblcol`

```

1545 \let\@outputdblcol\@undefined

```

The following procedure yields a Boolean value; it determines whether a float in the deferred queue is appropriate for placement in the column. In the multi-column grid, only those narrower than `\textwidth` are so.

```

1546 \def\@floatselect@sw@mlt#1{\@ifnotdblfloat{#1}}%

```

The following procedure determines which side of the page a marginpar will appear. It reproduces the behavior of standard L^AT_EX.

```

1547 \def\@addmarginpar@mlt{% emits a boolean
1548 \@ifnum{\pagegrid@cur=\@ne}%
1549 }%

```

8.16.3 Page grid utility procedures

`\pagegrid@cur` We take over L^AT_EX's `\col@number` and `\@leftcolumn`, which are obsolete. We
`\pagegrid@col` create two counters to hold the columns in the page grid and the current column
`\col@` within. We also create the first of a set of box registers to hold the committed
`\pagegrid@init` columns.

```
1550 \let\pagegrid@cur\col@number
1551 \let\col@number\@undefined
1552 \newcount\pagegrid@col
1553 \pagegrid@cur\@ne
1554 \expandafter\let\csname col@\the\pagegrid@cur\endcsname\@leftcolumn
1555 \let\@leftcolumn\@undefined
```

The default is for maximum two columns. If your class will require more columns, assign that number to `\pagegrid@col` before `\begin{document}` time.

```
1556 \pagegrid@col\two
```

The procedure `\pagegrid@init` exercises `\newbox` sufficiently to create the boxes for holding the columns in the page grid.

```
1557 \def\pagegrid@init{%
1558   \advance\pagegrid@cur\@ne
1559   \@ifnum{\pagegrid@cur<\pagegrid@col}{%
1560     \csname newbox\expandafter\endcsname\csname col@\the\pagegrid@cur\endcsname
1561     \pagegrid@init
1562   }{%
1563   }%
1564 }%
1565 \appdef\class@documenthook{%
1566   \pagegrid@init
1567 }%
```

`\grid@column` The procedure `\grid@column` knows how to lay up the columns in a multi-column page grid. It uses utility procedures `\append@column` and `\box@column`.

```
1568 \def\grid@column#1{%
1569   \global\setbox\@outputbox\vbox{%
1570     \hb@xt@\textwidth{%
1571       \vrule\@height\z@\@width\z@\@ifempty{#1}{}\{\@depth#1}%
1572       \pagegrid@cur\@ne
1573       \append@column
1574       \box@column\@outputbox
1575     }%
1576     \vskip\z@skip % FIXME: page depth!
1577   }%
1578 }%
```

`\append@column` The procedure `\append@column` appends columns for `\grid@column`, `\box@column`
`\box@column` builds the columns for `\append@column`, and `\marry@baselines` pastes vertical
`\marry@baselines` things back together.

Note that `\box@column` makes an attempt to prevent excessive `\topskip` or `\baselineskip` glue from being applied by T_EX when `\@outputbox` is contributed

to the MVL. If this is not done, it is possible to get into an infinite loop in the corner case, wherein the page grid is changed to one column and the balanced-up columns are already sufficient to fill the page.

Note (AO 0920): I have changed the dimension involved with `\box@column` from `\vsize` to `\textheight`, because the former is certainly not the correct value to use: it will change if floats have been placed in the last column of the page. I believe `\textheight` is the correct parameter to use here.

A REVTeX4 user, Sergey Strelkov (strelkov@maik.rssi.ru), wants the option of ragged-bottom columns. Implementing this feature properly means reboxing the columns to their natural height only if `\raggedcolumn@sw` is true. Otherwise, they get reboxed to their common height (`\@colht?`).

Note that the default has hereby changed from ragged to flush. It's not clear that anyone but Sergey will notice.

The macro `\marry@skip` addresses (in a limited way) the fact that neither the value of `\baselineskip` nor that of `\topskip` can be relied upon for the purpose of marrying the baselines of two split columns. (Because there might have been a local change to their values at the point where the output routine got triggered.)

For best results, your document class should call for grid changes only when in basal text settings. The `\marry@baselines` procedure will use the values appropriate to that point when attempting to put the columns back together.

In any case, we are not attempting to solve the more general problem of how to marry baselines where the leading can change arbitrarily within the galley or where glue could have been trimmed at a page top.

```

1579 \def\append@column{%
1580   \@ifnum{\pagegrid@cur<\pagegrid@col}{%
1581     \expandafter\box@column\csname col@the\pagegrid@cur\endcsname
1582     \hfil
1583     \vrule \@width\columnseprule
1584     \hfil
1585     \advance\pagegrid@cur\@ne
1586     \append@column
1587   }{%
1588   }%
1589 }%
1590 \def\box@column#1{%
1591   \raise\topskip
1592   \hb@xt@\columnwidth{%
1593     \dimen@ht#1\@ifdim{\dimen@>\@colht}{\dimen@\@colht}{}%
1594     \advance\dimen@-\topskip
1595     \count@\vbadness\vbadness\M
1596     \dimen@ii\vfuzz\vfuzz\maxdimen
1597     \outputdebug@sw{\saythe\@colht\saythe\dimen@}{}%
1598     \vtop to\dimen@
1599     \@ifdim{\ht#1>\textheight}{to\textheight}{}%
1600     {\hrule\@height\z@
1601      \unvbox#1%
1602      \raggedcolumn@skip

```

```

1603 }%
1604 \vfuzz\dimen@ii
1605 \vbadness\count@
1606 \hss
1607 }%
1608 }%
1609 \def\marry@baselines{%
1610 \vskip\marry@skip\relax
1611 }%
1612 \gdef\marry@skip{\z@skip}%
1613 \def\set@marry@skip{%
1614 \begingroup
1615 \skip@baselineskip\advance\skip@-\topskip
1616 \@ifdim{\skip@>\z@}-{%
1617 \xdef\marry@skip{\the\skip@}%
1618 }{%
1619 \endgroup
1620 }%

1621 \appdef\document@inithook{%
1622 \@ifxundefined\raggedcolumn@sw{\@booleanfalse\raggedcolumn@sw}{}%
1623 }%
1624 \def\raggedcolumn@skip{%
1625 \vskip\z@\raggedcolumn@sw{\@plus.0001fil\@minus.0001fil}{}\relax
1626 }%

```

`\@combinepage` The procedure `\@combinepage` prepends the stored page (`\pagesofar`) to `\@outputbox` and employs `\@combineinserts` to lay down the footnotes. The next event will usually be shipping out the made-up page, but not always. Therefore the argument of `\@combinepage`, which must be a Boolean, determines if the footnotes are to be combined into this page.

QUERY: In the following, if `\box\footins` is not void, its contents are lost. Can this ever happen?

```

1627 \def\@combinepage#1{%
1628 \@ifvoid\pagesofar}{%
1629 \setbox\@outputbox\vbox{%
1630 \unvbox\pagesofar
1631 \marry@baselines
1632 \unvbox\@outputbox
1633 }%
1634 }%
1635 #1{%
1636 \@ifvoid\footsofar}{%
1637 \show@box@size{Combining page footnotes}\footsofar
1638 \setbox\footins\box\footsofar
1639 \@combineinserts\@outputbox\footins
1640 }%
1641 }%

```

QUERY: The following line was removed, probably to fix a bug. When was this

done?

```
% \global\setbox\footins\box\footsofar  
%
```

```
1642 }%
```

```
1643 }%
```

`\@cflt` We modify L^AT_EX's `\@cflt` and `\@cflb` to remove the unwanted glue with `\unskip`.

```
\@cflb 1644 \def \@cflt{%  
1645 \let \@elt \@comflelt  
1646 \setbox\@tempboxa \vbox{}%  
1647 \@toplist  
1648 \setbox\@outputbox \vbox{%  
1649 \boxmaxdepth \maxdepth  
1650 \unvbox\@tempboxa\unskip  
1651 \topfigrule\vskip \textfloatsep  
1652 \unvbox\@outputbox  
1653 }%  
1654 \let\@elt\relax  
1655 \xdef\@freelist{\@freelist\@toplist}%  
1656 \global\let\@toplist\@empty  
1657 }%  
1658 \def \@cflb {%  
1659 \let\@elt\@comflelt  
1660 \setbox\@tempboxa \vbox{}%  
1661 \@botlist  
1662 \setbox\@outputbox \vbox{%  
1663 \unvbox\@outputbox  
1664 \vskip \textfloatsep\botfigrule  
1665 \unvbox\@tempboxa\unskip  
1666 }%  
1667 \let\@elt\relax  
1668 \xdef\@freelist{\@freelist\@botlist}%  
1669 \global\let\@botlist\@empty  
1670 }%
```

`\@combinedblfloats` We modify L^AT_EX's `\@combinedblfloats` to be more appropriate for incremental page building: we `\unvbox` the `\@outputbox`.

```
1671 \def\@combinedblfloats{%  
1672 \@ifx@empty\@dbltoplist{}{}%  
1673 \setbox\@tempboxa\vbox{}%  
1674 \let\@elt\@comdblfilelt\@dbltoplist  
1675 \let\@elt\relax\xdef\@freelist{\@freelist\@dbltoplist}%  
1676 \global\let\@dbltoplist\@empty  
1677 \setbox\@outputbox\vbox{%  
1678 %\boxmaxdepth\maxdepth %% probably not needed, CAR  
1679 \unvbox\@tempboxa\unskip  
1680 \@ifnum{\@dbltopnum>\m@ne}{\dblfigrule}{}%FIXME: how is \@dbltopnum maintained?  
1681 \vskip\dbltextfloatsep
```



```

1682   \unvbox\@outputbox
1683 }%
1684 }%
1685 }%

```

`\set@column@hsize` The procedure `\set@column@hsize` takes care of setting up the horizontal dimensions for the current page grid. The present routine will certainly not be adequate for more complex page layouts (e.g., with a side column), but works for the common ones.

```

1686 \def\set@column@hsize#1{%
1687   \pagegrid@col#1%
1688   \global\columnwidth\textwidth
1689   \global\advance\columnwidth\columnsep
1690   \global\divide\columnwidth\pagegrid@col
1691   \global\advance\columnwidth-\columnsep
1692   \global\hsize\columnwidth
1693   \global\linewidth\columnwidth
1694   \skip@\baselineskip\advance\skip@-\topskip
1695   \ifnum{\pagegrid@col>\@ne}{\set@marry@skip}{}%
1696 }%

```

`\set@colht` The story of `\textheight`, `\@colht`, `\@colroom`, and `\vsize`.
`\set@colroom` `\textheight`—height of the text column. Not a running parameter, however,
`\set@vsize` each time a page is shipped out, the `\textheight` could in principle be altered.
`\set@adj@colht` This must be done before

`\@colht`—`\textheight` minus the height of any full-page-width floats. The latter are committed only just after shipping out, and only if we are in a multicolumn page grid. Therefore, `\@colht` should be set after a `\shipout` (by `\@outputpage`) and will be adjusted when full-page-width floats are committed to the fresh page by `\do@startpage`.

`\@colroom`—`\@colht` (adjusted by `\pagesofar`) minus the height of any column-width floats. The latter are committed anywhere on the page, at which point `\@colroom` must be adjusted. Therefore, `\@colroom` should be set (by `\set@colroom`) whenever a column is prepared (by). FIXME: committed (by `\output@column@`) and will be adjusted (by `\@add@float` or `\do@startcolumn`) whenever a float is committed to the column.

`\vsize`—`\@colroom`. Therefore, `\vsize` should be set (by `\set@vsize`) whenever the `\@colroom` is set (by `\set@colroom`) or adjusted (by `\@add@float` or `\do@startcolumn`) FIXME: or when the `\pagesofar` box is changed (after invoking `\open@column@`).

Question: what if there are committed floats? Footnotes? Answer: full-page-width floats are only committed at top, and they are already reckoned with in `\@colht`. Column-width committed floats are incorporated by `\@makecolumn`; footnotes need help.

Note: FIXME: adjusting for `\pagesofar` is done at not quite the right time. I need to reexamine `\set@colht`, because `\@dbltoplist` and `\pagesofar` really should be on the same footing. Perhaps `\@colht` and `\@colroom` should both deal with their respective “lists” in the same way?

These concerns will be particularly germane if we ever extend this package to deal with full-page-width floats placed at the bottom of the page, or committed on the same page as called out.

It occurs to me that we should ditch `\set@colroom` and only ever execute `\set@colht`, which sets `\@colroom` as a side effect. If so, we can make `\@colht` take `\pagesofar` into account, as it should. Then `\@colht` will return to its original significance as the value that `\@colroom` is set to after a column is committed.

On the other hand, why not simply forget all this caching and (re-)calculate `\vsize` as late as possible? Particularly, `\@colht` is an artifact of the old way of doing things, where once it was set, it would never change.

```

1697 \def\set@colht{%
1698   \set@adj@textheight\@colht
1699   \global\let\enlarge@colroom\@empty
1700   \set@colroom
1701 }%
1702 \def\set@adj@textheight#1{%
1703   #1\textheight
1704   \def\@elt{\adj@page#1}%
1705   \@booleantrue\firsttime@sw\@dbltoplist
1706   \let\@elt\relax
1707   \global#1#1\relax
1708   \outputdebug@sw{\saythe#1}{}%
1709 }%
1710 \def\set@colroom{%
1711   \set@adj@colht\@colroom
1712   \@ifempty\enlarge@colroom{}{%
1713     \global\advance\@colroom\enlarge@colroom\relax
1714     \outputdebug@sw{\saythe\@colroom}{}%
1715   }%
1716   \@ifdim{\@colroom>\topskip}{}{%
1717     \ltxgrid@info{Not enough room: \string\@colroom=\the\@colroom; increasing to \the\topskip}%
1718     \@colroom\topskip
1719   }%
1720   \global\@colroom\@colroom
1721   \set@vsize
1722 }%
1723 %
1724 \def\set@vsize{%
1725   \global\vsize\@colroom
1726   \outputdebug@sw{\saythe\vsize}{}%
1727 }%

1728 \def\set@adj@colht#1{%
1729   #1\@colht
1730   \outputdebug@sw{\saythe#1}{}%
1731   \@ifvoid\pagesofar{}{%
1732     \advance#1-\ht\pagesofar\advance#1-\dp\pagesofar
1733     \outputdebug@sw{\saythe#1}{}%
1734   }%

```

```

1735 \set@adj@footins#1%
1736 \def\@elt{\adj@column#1}%
1737 \@booleantrue\firsttime@sw\@toplist
1738 \@booleantrue\firsttime@sw\@botlist
1739 \let\@elt\relax
1740 }%
1741 \def\adj@column#1#2{%
1742 \advance#1-\ht#2%
1743 \advance#1-\firsttime@sw{\textfloatsep\@booleanfalse\firsttime@sw}{\floatsep}%
1744 \outputdebug@sw{\saythe#1}{}%
1745 }%
1746 \def\adj@page#1#2{%
1747 \advance#1-\ht#2%
1748 \advance#1-\firsttime@sw{\dbltextfloatsep\@booleanfalse\firsttime@sw}{\dblfloatsep}%
1749 \outputdebug@sw{\saythe#1}{}%
1750 }%

```

Bug note (AO): I had used `\dimen@` as a scratch register within this procedure, because I had thought that doing, e.g., `\advance\dimen@\skip\footins` would not work in \TeX . Not only was I wrong about that, but I then proceeded to execute `\set@adj@footins\dimen@`, which understandably did not work. Ah! The treachery of the human mind.

```

1751 \def\set@adj@footins#1{%
1752 \@booleanfalse\temp@sw
1753 \set@adj@box#1\footins
1754 \set@adj@box#1\footins@saved
1755 \set@adj@box#1\footsofar
1756 \temp@sw{\advance#1-\skip\footins}{}%
1757 }%
1758 \def\set@adj@box#1#2{%
1759 \@ifvoid#2}{%
1760 \advance#1-\ht#2\advance#1-\dp#2%
1761 \@booleantrue\temp@sw
1762 \outputdebug@sw{\saythe#1}{}%
1763 }%
1764 }%

```

`\@outputpage@tail` In `\@outputpage@tail`, we set `\@colht` and the float placement parameters (this is the one point where it is appropriate to set `\@colht`). At `\do@startpage` time, we adjust `\@colht`'s value to reflect committed full-page-width floats.

Note: with a correctly written output routine, a call to `\@outputpage` will inevitably be followed by a call to `\do@startpage`, so these procedure calls would be unneeded.

```

1765 \appdef\@outputpage@tail{%
1766 \set@colht % FIXME: needed?
1767 \@floatplacement % FIXME: needed?
1768 \@dblfloatplacement % FIXME: needed?
1769 }%

```

`balance@2` We define procedures for balancing columns in a multicolumn layout. For now, we define only one: a procedure for the two-column grid. All others will simply `\relax` out.

The following code defines `\balance@2` without all the clunky `\csname` commands in the replacement part, which appears on the right-hand side of the assignment to `\toks@`.

The method is straightforward: balance the two columns of text, and balance the footnotes. Later on, `\@combineinserts` will be called to place the footnotes after the now-balanced columns.

It was necessary to deal with the case where `\box\footsofar` was not empty upon execution of this balancing code. We store it away in `\box\footins` and add it back in afterwards.

```

1770 \begingroup
1771 \catcode'\1=\cat@letter
1772 \catcode'\2=\cat@letter
    \toks@ contains the replacement part for an effective \def\balance@2.
1773 \toks@{%
1774 \setbox\footins\box\footsofar
1775 \balance@two\col@1\@outputbox

    % \global\setbox\col@1\box\col@1
    %

1776 \combine@foot@inserts
1777 \@ifvoid\footsofar{}{%
1778 \global
1779 \setbox\footsofar\ vbox\bgroup
1780 \setbox\z@\box\@tempboxa
1781 \let\recover@footins\relax
1782 \balance@two\footsofar\@tempboxa
1783 \hb@xt@\textwidth{\box\footsofar\hfil\box\@tempboxa}%
1784 \egroup
1785 }%
1786 }%
1787 \aftergroup\def\aftergroup\balance@2\expandafter
1788 \endgroup\expandafter{\the\toks@}%

```

`\balance@two` The procedure `\balance@two` takes two columns and balances them; in the process it removes any footnotes that may be present to a place of safety, for later placement at the foot of the shipped-out page. The box register `\box\@ne` is the aggregate of all columns. The box register `\box\z@` is the last column. The box register `\box\tw@` is the first column. The `\dimen` register `\dimen@` is the trial value to balance to, initially half the height of `\box\@ne`. The `\dimen` register `\dimen@i` is the increment for the next trial; its initial value is equal to the initial value of `\dimen@`. The `\dimen` register `\dimen@ii` is the difference of the heights of the two columns.

The procedure uses a binary search for that value of `\dimen@` which is stable to within `.5\p@` and which makes the last column be shorter than the others.

This procedure can be extended to multiple columns simply by changing it to execute `\vsplit` multiple times (one less than the total number of columns in the page layout) and to calculating `\dimen@ii` to be the difference of the heights of last column and the `\dimen@`. Upon termination of the search, one would execute the `\vsplits` once again, this time using the actual `\col@` box registers to store the balanced columns, thereby clobbering their former contents.

Bug Note: as originally written, this macro had a bug, which is well worth avoiding under similar circumstances anywhere. So, learn from the mistakes of others, as they say. In trying to remove the depth of the boxes created via `\vsplit` within the `\loopwhile` control, I originally coded `\unvbox\z@\setbox\z@\lastbox \dimen@dp\z@ \box\z@ \vskip-\dimen@`. The error here is that the (horizontal) shift of the last box in the vertical list will be lost in the process. Simply put, `\setbox\z@\lastbox` fails to retain the shift of the box node in the vertical list, and when it is put down again via `\box\z@`, it will no longer have the correct shift.

This bug affected things placed in the MVL with `\moveleft`, `\moveright`, `\parshape`, and `\hangindent`, as well as things shifted by \TeX 's primitive mechanisms.

A superior strategy for removing the depth of the last line of the list is more expensive, but safer: make a separate copy of the list, measure the depth of the last box as above, but then discard the list, retaining only the value of the dimension.

Note that this procedure will not work if the material within is excessively chunky. A particular failure mode exists where none of the material is allocated to the last (right) column. We detect this case and revert to unbalanced columns.

Another failure mode is where a large chunk occurs at the beginning of the composite box. In this case, the left column may fill up even when `\dimen@` is very small. If this configuration leaves the left column longer than the right, then we are done, but `\dimen@` by no means represents the height of either finished box.

Therefore the last step in the process is to rebox the two columns to a common height determined independently of the balancing process.

The dimension involved is checked against the current `\@colroom` to guard against the case where excessive material happens to fall in either column.

```

1789 \def\balance@two#1#2{%
1790 \outputdebug@sw{\trace@scroll{\showbox#1\showbox#2}}{}}%
1791 \setbox\@ne\vbox{%
1792 \@ifvoid#1}{}%
1793 \unvcopy#1\recover@footins
1794 \@ifvoid#2}{\marry@baselines}%
1795 }%
1796 \@ifvoid#2}{}%
1797 \unvcopy#2\recover@footins
1798 }%
1799 }%
1800 \dimen@ht\@ne\divide\dimen@\tw@
1801 \dimen@i\dimen@
1802 \vbadness\@M
1803 \vfuzz\maxdimen

```

```

1804 \loopwhile{%
1805   \dimen@i=.5\dimen@i
1806   \outputdebug@sw{\saythe\dimen@\saythe\dimen@i\saythe\dimen@ii}{}%
1807   \setbox\z@\copy\@ne\setbox\tw@\vsplit\z@ to\dimen@
1808   \setbox\z@ \vbox{%
1809     \unvcopy\z@
1810     \setbox\z@\vbox{\unvbox\z@ \remove@lastbox\aftergroup\vskip\aftergroup-\expandafter}\the\dp\
1811   }%
1812   \setbox\tw@\vbox{%
1813     \unvcopy\tw@
1814     \setbox\z@\vbox{\unvbox\tw@\remove@lastbox\aftergroup\vskip\aftergroup-\expandafter}\the\dp\
1815   }%
1816   \dimen@ii\ht\tw@\advance\dimen@ii-\ht\z@
1817   \@ifdim{\dimen@i>.5\p@}{%
1818     \advance\dimen@\@ifdim{\dimen@ii<\z@}{-}\dimen@i
1819     \true@sw
1820   }{%
1821     \@ifdim{\dimen@ii<\z@}{%
1822       \advance\dimen@\tw@\dimen@i
1823       \true@sw
1824     }{%
1825       \false@sw
1826     }%
1827   }%
1828 }%
1829 \outputdebug@sw{\saythe\dimen@\saythe\dimen@i\saythe\dimen@ii}{}%
1830 \@ifdim{\ht\z@=\z@}{%
1831   \@ifdim{\ht\tw@=\z@}{%
1832     \true@sw
1833   }{%
1834     \false@sw
1835   }%
1836 }{%
1837   \true@sw
1838 }%
1839 {%
1840 }{%
1841   \ltxgrid@info{Unsatifactorily balanced columns: giving up}%
1842   \setbox\tw@\box#1%
1843   \setbox\z@ \box#2%
1844 }%
1845 \setbox\tw@\vbox{\unvbox\tw@\vskip\z@skip}%
1846 \setbox\z@ \vbox{\unvbox\z@ \vskip\z@skip}%
1847 \set@colht
1848 \dimen@\ht\z@\@ifdim{\dimen@<\ht\tw@}{\dimen@\ht\tw@}{}%
1849 \@ifdim{\dimen@>\@colroom}{\dimen@\@colroom}{}%
1850 \outputdebug@sw{\saythe{\ht\z@}\saythe{\ht\tw@}\saythe\@colroom\saythe\dimen@}{}%
1851 \setbox#1\vbox to\dimen@{\unvbox\tw@\unskip\raggedcolumn@skip}%
1852 \setbox#2\vbox to\dimen@{\unvbox\z@ \unskip\raggedcolumn@skip}%
1853 \outputdebug@sw{\trace@scroll{\showbox#1\showbox#2}}{ }%

```

1854 }%

`\recover@footins` The procedure `\recover@footins` is the utility procedure for recovering the footnotes from the bottom of a column. It is used when the page grid is changed, so that footnotes can be set at the bottom of the shipped out page.

```
1855 \def\recover@footins{%
1856   \skip\z@ \lastskip\unskip
1857   \skip\@ne\lastskip\unskip
1858   \setbox\z@\lastbox
1859   \@ifvbox\z@{%
1860     \setbox\z@\vbox{%
1861       \unvbox\z@
1862       \setbox\z@\lastbox
1863       \@ifvoid\z@{}-%
1864       \global\setbox\footsofar\vbox{%
1865         \unvbox\footsofar
1866         \@ifvbox\z@{%
1867           \unvbox\z@
1868         }-%
1869         \box\z@
1870       }%
1871     }%
1872   }%
1873 }%
1874 }-%
1875 \outputdebug@sw{\trace@box\footsofar}{}%
1876 }%
```

`\@begindocumenthook` Initialization: we initialize to the page grid named “one”. If the class decides to initially set type in a different grid, it should execute these same commands, but changing the first to the appropriate procedure.

Note that the point where this sequence is executed would be an excellent place to arrange for floats to be committed to the first page of a document. That is, we execute `\do@startpage`, which triggers `\do@startcolumn`.

FIXME: it should be the job of the page grid to determine the procedure to execute at the start of the job. Make this a hook.

```
1877 \prepdef\@begindocumenthook{%
1878   \open@column@one\@ne
1879   \set@colht
1880   \@floatplacement
1881   \@dblfloatplacement
1882 }%
```

Comment: our technique of balancing columns is severely limited, because it cannot properly work with `longtable`, which places material at the bottom and top of the column break.

The proper way to handle a grid change in the middle of the page is to accumulate all the material for an entire article (or chapter) and then assemble finished

pages therefrom. This approach is fundamentally superior for complex layouts: it corresponds to real-world workflows. Such a scheme is an excellent subject for another L^AT_EX package.

8.17 Patches for the longtable package

L^AT_EX’s “required” package `longtable` (written by David P. Carlisle), which is part of `/latex/required/tools`, is incompatible with both L^AT_EX’s “required” package `multicol` and with L^AT_EX’s native `\twocolumn` capability. There is no essential reason for this incompatibility, aside from implementation details, and the `ltxgrid` package gives us the ability to lift them.

Only four of `longtable`’s procedures require rewriting: `\longtable`, `\endlongtable`, `\LT@start`, and `\LT@end@hd@ft`. The procedure `\switch@longtable` checks against their expected meanings and, if all is as expected, applies the patches. In the process, we simplify things considerably and also make them more secure.

Why does `longtable` need to access the output routine, anyway? What it comes down to, is what happens when a pagebreak falls within a long table. If this happens, we would like to append a row at the bottom of the broken table and add a row at the top of the next page.

These things can be accommodated easily by the `ltxgrid` output routine hooks.

```

\longtable
1883 \def\longtable@longtable{%
1884 \par
1885 \ifx\multicols\@undefined\else\ifnum\col@number>\@ne\@twocolumntrue\fi\fi
1886 \if@twocolumn\LT@err{longtable not in 1-column mode}\@ehc\fi
1887 \begingroup
1888 \@ifnextchar[\LT@array{\LT@array[x]}%
1889 }%
1890 \def\longtable@new{%
1891 \par
1892 \@ifnextchar[\LT@array{\LT@array[x]}%
1893 }%

\endlongtable
1894 \def\endlongtable@longtable{%
1895 \crrc
1896 \noalign{%
1897 \let\LT@entry\LT@entry@chop
1898 \xdef\LT@save@row{\LT@save@row}}%
1899 \LT@echunk
1900 \LT@start
1901 \unvbox\z@
1902 \LT@get@widths
1903 \if@filesw
1904 {\let\LT@entry\LT@entry@write\immediate\write\@auxout{%
1905 \gdef\expandafter\noexpand
1906 \csname LT@\romannumeral\c@LT@tables\endcsname

```



```

1907         {\LT@save@row}}}%
1908     \fi
1909     \ifx\LT@save@row\LT@@save@row
1910     \else
1911         \LT@warn{Column \@width s have changed\MessageBreak
1912             in table \thetable}%
1913         \LT@final@warn
1914     \fi
1915     \endgraf\penalty -\LT@end@open
1916     \endgroup
1917     \global\@mparbottom\z@
1918     \pagegoal\vsizel
1919     \endgraf\penalty\z@\addvspace\LTpost
1920     \ifvoid\footins\else\insert\footins{}\fi
1921 }%

1922 \def\endlongtable@new{%
1923     \crr
1924     \noalign{%
1925         \let\LT@entry\LT@entry@chop
1926         \xdef\LT@save@row{\LT@save@row}%
1927     }%
1928     \LT@echunk
1929     \LT@start
1930     \unvbox\z@
1931     \LT@get@widths
1932     \@ifsw@if@filesw\fi{%
1933         {%
1934             \let\LT@entry\LT@entry@write
1935             \immediate\write\@auxout{%
1936                 \gdef\expandafter\noexpand\csname LT@\romannumeral\c@LT@tables\endcsname
1937                 {\LT@save@row}%
1938             }%
1939         }%
1940     }{}%
1941     \@ifx{\LT@save@row\LT@@save@row}{-}{%
1942         \LT@warn{%
1943             Column \@width s have changed\MessageBreak in table \thetable
1944         }\LT@final@warn
1945     }%
1946     \endgraf
1947     \nobreak
1948     \box\@ifvoid\LT@lastfoot{\LT@foot}{\LT@lastfoot}%
1949     \global\@mparbottom\z@
1950     \endgraf
1951     \LT@post
1952 }%

```

\LT@start

```

1953 \def\LT@start@longtable{%

```

```

1954 \let\LT@start\endgraf
1955 \endgraf\penalty\z@\vskip\LTpre
1956 \dimen@ \pagetotal
1957 \advance\dimen@ \ht\ifvoid\LT@firsthead\LT@head\else\LT@firsthead\fi
1958 \advance\dimen@ \dp\ifvoid\LT@firsthead\LT@head\else\LT@firsthead\fi
1959 \advance\dimen@ \ht\LT@foot
1960 \dimen@ii\vfuzz
1961 \vfuzz\maxdimen
1962 \setbox\tw@\copy\z@
1963 \setbox\tw@\vsplit\tw@ to \ht\@arstrutbox
1964 \setbox\tw@\vbox{\unvbox\tw@}%
1965 \vfuzz\dimen@ii
1966 \advance\dimen@ \ht
1967 \ifdim\ht\@arstrutbox>\ht\tw@\@arstrutbox\else\tw@\fi
1968 \advance\dimen@\dp
1969 \ifdim\dp\@arstrutbox>\dp\tw@\@arstrutbox\else\tw@\fi
1970 \advance\dimen@ -\pagegoal
1971 \ifdim \dimen@>\z@\vfil\break\fi
1972 \global\@colroom\@colht
1973 \ifvoid\LT@foot\else
1974 \advance\vsizel-\ht\LT@foot
1975 \global\advance\@colroom-\ht\LT@foot
1976 \dimen@\pagegoal\advance\dimen@-\ht\LT@foot\pagegoal\dimen@
1977 \maxdepth\z@
1978 \fi
1979 \ifvoid\LT@firsthead\copy\LT@head\else\box\LT@firsthead\fi

```

At some point before version 4.11, the `\nobreak` was added.

```

1980 \nobreak
1981 \output{\LT@output}%
1982 }%
1983 \def\LT@start@new{%
1984 \let\LT@start\endgraf
1985 \endgraf
1986 \markthr@@}%
1987 \LT@pre
1988 \@ifvoid\LT@firsthead{\LT@top}{\box\LT@firsthead\nobreak}%
1989 \mark@envir{longtable}%
1990 }%

```

`\LT@end`

```

1991 \def\LT@end@hd@ft@longtable#1{%
1992 \LT@echunk
1993 \ifx\LT@start\endgraf
1994 \LT@err{Longtable head or foot not at start of table}{Increase LTchunksize}%
1995 \fi
1996 \setbox#1\box\z@
1997 \LT@get@widths\LT@bchunk
1998 }%
1999 \def\LT@end@hd@ft@new#1{%

```

```

2000 \LT@echunk
2001 \@ifx{\LT@start\endgraf}{%
2002 \LT@err{Longtable head or foot not at start of table}{Increase LTchunksize}%
2003 }%
2004 \global\setbox#1\box\z@
2005 \LT@get@widths
2006 \LT@bchunk
2007 }%

```

\LT@array

```

2008 \def\LT@array@longtable[#1]#2{%
2009 \refstepcounter{table}\stepcounter{LT@tables}%
2010 \if l#1%
2011 \LTleft\z@ \LTright\fill
2012 \else\if r#1%
2013 \LTleft\fill \LTright\z@
2014 \else\if c#1%
2015 \LTleft\fill \LTright\fill
2016 \fi\fi\fi
2017 \let\LT@mcol\multicolumn
2018 \let\LT@@tabarray\@tabarray
2019 \let\LT@hline\hline
2020 \def\@tabarray{%
2021 \let\hline\LT@hline
2022 \LT@@tabarray}%
2023 \let\LT@tabularcr\let\@tabularnewline\%
2024 \def\newpage{\noalign{\break}}%
2025 \def\pagebreak{\noalign{\ifnum' }=0\fi\@testopt{\LT@no@pgbk-}4}%
2026 \def\nopagebreak{\noalign{\ifnum' }=0\fi\@testopt{\LT@no@pgbk4}}%
2027 \let\hline\LT@hline \let\kill\LT@kill\let\caption\LT@caption
2028 \@tempdima\ht\strutbox
2029 \let\@endpbox\LT@endpbox
2030 \ifx\extrarowheight\@undefined
2031 \let\@acol\@tabacol
2032 \let\@classz\@tabclassz \let\@classiv\@tabclassiv
2033 \def\@startpbox{\vtop\LT@startpbox}%
2034 \let\@@startpbox\@startpbox
2035 \let\@@endpbox\@endpbox
2036 \let\LT@LL@FM@cr\@tabularcr
2037 \else
2038 \advance\@tempdima\extrarowheight
2039 \col@sep\@tabcolsep
2040 \let\@startpbox\LT@startpbox\let\LT@LL@FM@cr\@arraycr
2041 \fi
2042 \setbox\@arstrutbox\hbox{\vrule
2043 \@height \arraystretch \@tempdima
2044 \@depth \arraystretch \dp \strutbox
2045 \@width \z@}%
2046 \let\@sharp#\let\protect\relax
2047 \begingroup

```

```

2048 \mkpream{#2}%
2049 \xdef\LT@bchunk{%
2050   \global\advance\c@LT@chunks\@ne
2051   \global\LT@rows\z@\setbox\z@\vbox\bgroup
2052   \LT@setprevdepth

```

At some point before version 4.11, the `\noexpand` was added. We need not change our own version, because we did it right, back in 1998 (using `\appdef`).

```

2053   \tabskip\LTleft \noexpand\halign to\hsize\bgroup
2054   \tabskip\z@ \@arstrut \@preamble \tabskip\LTRight \cr}%
2055 \endgroup
2056 \expandafter\LT@nofcols\LT@bchunk&\LT@nofcols
2057 \LT@make@row
2058 \m@th\let\par\@empty
2059 \everycr{}\lineskip\z@\baselineskip\z@
2060 \LT@bchunk}%
2061 \def\LT@LR@l{\LTleft\z@ \LTRight\fill}%
2062 \def\LT@LR@r{\LTleft\fill \LTRight\z@ }%
2063 \def\LT@LR@c{\LTleft\fill \LTRight\fill}%
2064 \def\LT@array@new[#1]#2{%
2065   \refstepcounter{table}\stepcounter{LT@tables}%
2066   \table@hook
2067   \LTleft\fill \LTRight\fill
2068   \csname LT@LR@#1\endcsname
2069   \let\LT@mcol\multicolumn
2070   \let\LT@@hl\hline
2071   \prepdef\@tabarray{\let\hline\LT@@hl}%
2072   \let\\\LT@tabularcr
2073   \let\tabularnewline\\%
2074   \def\newpage{\noalign{\break}}%
2075   \def\pagebreak{\noalign{\ifnum' =0\fi\@testopt{\LT@no@pgbk-}4}%
2076   \def\nopagebreak{\noalign{\ifnum' =0\fi\@testopt{\LT@no@pgbk4}%
2077   \let\hline\LT@hline
2078   \let\kill\LT@kill
2079   \let\caption\LT@caption
2080   \@tempdima\ht\strutbox
2081   \let\@endpbox\LT@endpbox
2082   \@ifundefined\extrarowheight{%
2083     \let\@acol\@tabacol
2084     \let\@classz\@tabclassz
2085     \let\@classiv\@tabclassiv
2086     \def\@startpbox{\vtop\LT@startpbox}%
2087     \let\@@startpbox\@startpbox
2088     \let\@@endpbox\@endpbox

```

Because `ltxutil` patches L^AT_EX's `\@tabularcrand` `\@xtabularcr`, we must restore these procedures in the scope of `longtable`. Ironically, the patches in `ltxutil` were for the purpose of extending the `tabular` environment to prevent pagebreaks with the `*`-form of `\\`, just the same as is being done here. But the two mechanisms conflict.

```

2089 \let\LT@LL@FM@cr\@tabularcr@LaTeX
2090 \let\@xtabularcr\@xtabularcr@LaTeX
2091 }{%
2092 \advance\@tempdima\extrarowheight
2093 \col@sep\@tabcolsep
2094 \let\@startpbox\LT@startpbox

2095 \let\LT@LL@FM@cr\@arraycr@array
2096 }%
2097 %
2098 \let\@acoll\@tabacoll
2099 \let\@acolr\@tabacolr
2100 \let\@acol\@tabacol
2101 %
2102 \setbox\@arstrutbox\hbox{%
2103 \vrule
2104 \@height \arraystretch \@tempdima
2105 \@depth \arraystretch \dp \strutbox
2106 \@width \z@
2107 }%
2108 \let\@sharp##%
2109 \let\protect\relax
2110 \begingroup
2111 \@mkpream{#2}%
2112 \@mkpream@relax
2113 \edef\@preamble{\@preamble}%
2114 \prepdef\@preamble{%
2115 \global\advance\c@LT@chunks\@ne
2116 \global\LT@rows\z@
2117 \setbox\z@\vbox\bgroup
2118 \LT@setprevdepth
2119 \tabskip\LTleft
2120 \halign to\hsize\bgroup
2121 \tabskip\z@
2122 \@arstrut
2123 }%
2124 \appdef\@preamble{%
2125 \tabskip\LTright
2126 \cr
2127 }%
2128 \global\let\LT@bchunk\@preamble
2129 \endgroup
2130 \expandafter\LT@nofcols\LT@bchunk&\LT@nofcols
2131 \LT@make@row
2132 \m@th
2133 \let\par\@empty
2134 \everycr{}%
2135 \lineskip\z@
2136 \baselineskip\z@
2137 \LT@bchunk

```

```

2138 }%
2139 \appdef\table@hook{}%

```

`\switch@longtable` Here is the switch from standard `longtable` to the new, `ltxgrid`-compatible values.

At this point, we extend `longtable` with a `longtable*` form, which signifies that we want to use the full page width for setting the table. You can think this way: `longtable*` is to `longtable` as `table*` is to `table`.

```

2140 \def\switch@longtable{%
2141   \@ifpackageloaded{longtable}{%
2142     \@ifx{\longtable\longtable@longtable}{%
2143       \@ifx{\endlongtable\endlongtable@longtable}{%
2144         \@ifx{\LT@start\LT@start@longtable}{%
2145           \@ifx{\LT@end@hd@ft\LT@end@hd@ft@longtable}{%
2146             \@ifx{\LT@array\LT@array@longtable}{%
2147               \true@sw
2148             }{\false@sw}%
2149           }{\false@sw}%
2150         }{\false@sw}%
2151       }{\false@sw}%
2152     }{\false@sw}%
2153   }%
2154   \class@info{Patching longtable package}%
2155 }{%
2156   \class@info{Patching unrecognized longtable package. (Proceeding with fingers crossed)}%
2157 }%
2158 \let\longtable\longtable@new
2159 \let\endlongtable\endlongtable@new
2160 \let\LT@start\LT@start@new
2161 \let\LT@end@hd@ft\LT@end@hd@ft@new
2162 \let\LT@array\LT@array@new
2163 \newenvironment{longtable*}{%
2164   \onecolumngrid@push
2165   \longtable
2166 }{%
2167   \endlongtable
2168   \onecolumngrid@pop
2169 }%

```

Removed obsolete code.

```

2170 }{}%
2171 }%

```

`\LT@pre` Note that at the end of the `longtable` environment, we reestablish the `\mark@envir`
`\LT@bot` of the containing environment. We have left `\curr@envir` alone, so this will work.

```

\LT@top 2172 \def\LT@pre{\penalty\z@\vskip\LTpre}%
\LT@post 2173 \def\LT@bot{\nobreak\copy\LT@foot\vfil}%
\LT@adj 2174 \def\LT@top{\copy\LT@head\nobreak}%
2175 \def\LT@post{\penalty\z@\addvspace\LTpost\mark@envir{\curr@envir}}%

```

```

2176 \def\LT@adj{%
2177   \setbox\z@\vbox{\null}\dimen@-\ht\z@
2178   \setbox\z@\vbox{\unvbox\z@\LT@bot}\advance\dimen@\ht\z@
2179   \global\advance\vsizel-\dimen@
2180 }%

```

```

output@init
output@prep 2181 \def\output@init@longtable{\LT@adj}%
output@post 2182 \def\output@prep@longtable{\setbox\@cclv\vbox{\unvbox\@cclv\LT@bot}}%
2183 \def\output@post@longtable{\LT@top}%

```

8.18 Patches for index processing

Another feature that uses the output routine hooks occurs within an index, where one wishes to apply a “continue head” when a column breaks within a primary index entry. Some book designs call for the continue head to only be applied at a turnpage break.

In any case, it is easy enough for `\output@post@theindex` to do this in conjunction with component marks. Only the bare outlines are shown here.

```

\output@init
\output@prep 2184 \let\output@init@theindex\@empty
\output@post 2185 \let\output@prep@theindex\@empty
2186 \def\output@post@theindex{%
2187   \@ifodd\c@page{}-%
2188   \@ifnum{\pagegrid@cur=\@ne}-%

```

We have the leftmost column of a verso page: Insert the current top-level continued head.

```

2189 }%
2190 }%
2191 }%

```

8.19 Checking the auxiliary file

We relegate the checking of the auxiliary file to the output routine. This task must wait until the last page is shipped out, because otherwise the stream might get closed before the last page is shipped out. Obviously, we must use `\do@output@MVL` for the job.

```

\check@aux
2192 \def\check@aux{\do@output@MVL{\do@check@aux}}%

```

8.20 Dealing with stuck floats and stalled float dequeuing

L^AT_EX’s float placement mechanism is fundamentally flawed, as evidenced by its warning message “too many unprocessed floats”, which users understandably find

frustrating. The `ltxgrid` package provides tools for ameliorating the situation somewhat.

Two cases require detection and rectification:

1. A float is “stuck” in the `\@deferlist`: for whatever reason, the float fails to be committed, even at the start of a fresh page. Once this condition prevails, following floats can never be committed, subsequently all of L^AT_EX’s float registers are used up.

If this condition is detected, we reconsider float dequeuing under permissive (`\clearpage`-style) processing.

2. The `\@freelist` is exhausted: a large concentration of floats, say, uses up all of L^AT_EX’s float registers all at once. This condition commonly occurs when the user collects floats at the end of the document, for some reason.

When a float is encountered, L^AT_EX uses a float register (allocated from a pool of free registers) to contain it until it can be placed. However, no further action is taken until the pagebuilder is visited, so floats can accumulate. Also, even after the pagebuilder is visited, deferred floats can accumulate, and these are not committed until a column (or page) of text is completed.

Once the last free float register is used, action should be taken that will commit some of the deferred floats, even if this might require ending the page right where we are (resulting in a short page).

Perhaps, committed floats should be stored using some mechanism other than a list, as is currently done. A feasible alternative storage method would be to use a `\box` register in place of `\@toplist`, `\@botlist`, and `\@dbltoplist`. This is probably just fine, since such committed floats are not reconsidered (I think).

The emergency processing implemented here immediately ends the current page and begins to output float pages under (`\clearpage`-style) rules. It proceeds until all deferred floats have been flushed.

Users should expect non-optimal page makeup under these circumstances.

Note that there is a weakness in our approach that we have not attempted to repair: if floats are being added as part of a paragraph, we will not be able to take these remedial steps until the paragraph ends. This means that the approach implemented here cannot fix all L^AT_EX documents. Users can still construct documents that exhaust L^AT_EX’s pool of float registers!

`\check@deferlist@stuck`
`\@outputpage@tail`

We detect the case where, at the start of a fresh page, there are deferred floats, but none are committed. We memorize the `\@deferlist` at `\shipout` time, then examine it at the point where our efforts to commit floats to the new page are complete. If it has not changed, the first float must be stuck, and we attempt to fix things via `\force@deferlist@stuck`.

This simple approach is completely effective in for typical documents.

Note that we try to avoid an infinite loop by examining the value of `\clearpage@sw`: if we come here with that boolean true, we are in a loop.


```

2193 \def\check@deferlist@stuck#1{%
2194   \@ifx{\@deferlist@postshipout\@empty}{-}{%
2195     \@ifx{\@deferlist@postshipout\@deferlist}{%
2196       \fltstk
2197       \clearpage@sw{%
2198         \ltxgrid@warn{Deferred float stuck during \string\clearpage\space processing}%
2199       }{%
2200         \force@deferlist@stuck#1%
2201       }%
2202     }{%

```

We have successfully committed float(s)

```

2203   }%
2204   \global\let\@deferlist@postshipout\@empty
2205 }%
2206 }%
2207 \def\@fltstk{%
2208   \@latex@warning{A float is stuck (cannot be placed without \string\clearpage)}%
2209 }%
2210 \appdef\@outputpage@tail{%
2211   \global\let\@deferlist@postshipout\@deferlist
2212 }%

```

`\@next` We rewrite the L^AT_EX kernel macros that dequeue float registers from, e.g., `\@xnext` `\@deferlist`, providing a test for the condition where the pool of free registers is about to underflow.

In this case, we attempt to fix things via `\force@deferlist@empty`.

```

2213 \def\@next#1#2{%
2214   \@ifx{#2\@empty}{\false@sw}{%
2215     \expandafter\@xnext#2\@@#1#2%
2216     \true@sw
2217   }%
2218 }%
2219 \def\@xnext\@elt#1#2\@@#3#4{%
2220   \def#3{#1}%
2221   \gdef#4{#2}%
2222   \def\@tempa{#4}\def\@tempb{\@freelist}%
2223   \@ifx{\@tempa\@tempb}{%
2224     \@ifx{#4\@empty}{%
2225       \force@deferlist@empty%{Float register pool exhausted}%
2226     }{%
2227   }{%
2228 }%

```

`\force@deferlist@stuck` The procedure `\force@deferlist@empty` is an attempt to rectify a situation where L^AT_EX's float placement mechanism may fail ("too many unprocessed floats").

`\force@deferlist@empty` We put down interrupts that call for the float placement to be redone, but under permissive conditions, just the same as if `\clearpage` had been invoked.

Note that the attempt to rectify the error is contingent on the setting of `\force@deferlist@sw`, default false. A document class using this package that wishes to enable this error recovery mechanism should set this boolean to true.

The interrupt `\do@forcecolumn@pen`, which invokes the procedure `\do@forcecolumn`, does the same as `\do@startcolumn`, except under permissive conditions: we are trying to empty out the float registers completely.

In order to properly with the case where there is material in `\box\@cclv`, `\@toplist`, `\@botlist`, `\@dbltoplist`, etc, we do what amounts to `\newpage` to get things rolling.

In `\force@deferlist@stuck`, we take advantage of already being in the output routine: simply reinvoke `\do@startcolumn` under permissive conditions.

```

2229 \def\force@deferlist@stuck#1{%
2230 \force@deferlist@sw{%
2231 \booleantrue\clearpage@sw
2232 \booleantrue\forcefloats@sw
2233 #1%
2234 }{%
2235 }%
2236 }%
2237 \def\force@deferlist@empty{%
2238 \force@deferlist@sw{%
2239
2240 % \ltxgrid@info{#1, attempting rectification}%
2241 %
2242
2243 \penalty-\pagebreak@pen
2244 \protect@penalty\do@forcecolumn@pen
2245 }{%
2246
2247 % \ltxgrid@info{#1}%
2248 %
2249
2250 }%
2251 }%
2252
2253 \booleanfalse\force@deferlist@sw
2254 \mathchardef\do@forcecolumn@pen=10009
2255 \@namedef{output@-\the\do@forcecolumn@pen}{\do@forcecolumn}%
2256 \def\do@forcecolumn{%
2257 \booleantrue\clearpage@sw
2258 \booleantrue\forcefloats@sw
2259
2260 %\unvbox\@cclv
2261 %\vfil
2262 %\penalty-\pagebreak@pen
2263 %
2264
2265 \do@startcolumn
2266 }%

```

A more thorough revision of L^AT_EX's float placement mechanism would involve substituting a single `\box` register for the `\@deferlist`. This way, L^AT_EX's ability to have latent floats would be limited by box memory alone.

Because only the `\box` and `\count` components of the float box register are actually used by L^AT_EX, our scheme can be accomplished if we can find a way to encode the information held in the `\count` component.

A first-in, first-out mechanism exists, wherein a box-penalty pair is dequeued by `\lastbox\lastpenalty\unpenalty` and enqueued by `\setbox\foo=\hbox\bgroup\penalty\floatpenalty`.

Note that this scheme is made possible by our change to L^AT_EX's float placement mechanism, wherein we consolidated the two `\@deferlists` into one.

9 Support for legacy L^AT_EX commands

We provide support for the `\enlargethispage` command.

Note: using a command of this sort does not automatically enlarge both pages of a spread, which would be the convention in page composition.

Timing Note: In a multicolumn page grid, the user should issue the `\enlargethispage` command while the first column of the page is being typeset. We provide a helpful message if the timing is wrong.

This code can serve as a model for introducing commands that need to execute within the safety of the output routine. We ensure that the arguments are fully expanded, then execute `\do@output@MVL` to cause an output procedure, `\@@enlargethispage`, to execute. When it does execute, the MVL will be exposed.

The `\@@enlargethispage` procedure simply adjusts the vertical dimensions of the page. The adjustment will persist until the column is committed, at which point the page dimension will revert to its standard value.

```

2252 \def\enlargethispage{%
2253   \@ifstar{%
2254     \@enlargethispage{}%
2255   }{%
2256     \@enlargethispage{}%
2257   }%
2258 }%
2259 \def\@enlargethispage#1#2{%
2260   \begingroup
2261   \dimen@#2\relax
2262   \edef\@tempa{#1}%
2263   \edef\@tempa{\noexpand\@enlargethispage{\@tempa}{\the\dimen@}}%
2264   \expandafter\do@output@MVL\expandafter{\@tempa}%
2265   \endgroup
2266 }%
2267 \def\@@enlargethispage#1#2{%
2268   \def\@tempa{one}%
2269   \@ifx{\thepagegrid\@tempa}{%
2270     \true@sw

```

```

2271 }{%
2272 \def\@tempa{mlt}%
2273 \@ifx{\thepagegrid\@tempa}{%
2274 \@ifnum{\pagegrid@cur=\@ne}{%
    OK to adjust this page
2275 \gdef\enlarge@colroom{#2}%
2276 \true@sw
2277 }{%
    Can only adjust this column; give up
2278 \ltxgrid@warn{Too late to enlarge this page; move the command to the first column.}%
2279 \false@sw
2280 }%
2281 }{%
    Unknown page grid
2282 \ltxgrid@warn{Unable to enlarge a page of this kind.}%
2283 \false@sw
2284 }%
2285 }%
2286 {%
2287 \class@info{Enlarging page \thepage\space by #2}%
2288 \global\advance\@colroom#2\relax
2289 \set@vsize
2290 }{%
    Could not adjust this page
2291 }%
2292 }%
2293 \let\enlarge@colroom\@empty
    The \@kludgeins insert register is now unneeded. Ensure that packages using
    this mechanism break (preferable to subtle bugs).
2294 \let\@kludgeins\@undefined

```

9.0.1 Building the page for shipout

\@outputpage@head We set \@outputpage@head to make the \@outputbox be of fixed height.

```

2295 \@booleantrue\textheight@sw
2296 \prepdef\@outputpage@head{%
2297 \textheight@sw{%
2298 \count@\vbadness\vbadness\@M
2299 \dimen@\vfuzz\vfuzz\maxdimen
2300 \setbox\@outputbox\vbox to\textheight{\unvbox\@outputbox}%
2301 \vfuzz\dimen@
2302 \vbadness\count@
2303 }{}}%
2304 }%

```

9.0.2 Warning message

`\ltxgrid@info` Something has happened that the user might be interested in. Print a message to
`\ltxgrid@warn` the log, but only if the user selected the verbose option.

```
2305 \def\ltxgrid@info{%
2306 \ltxgrid@info@sw{\class@info}{\@gobble}%
2307 }%
2308 \@booleanfalse\ltxgrid@info@sw
2309 \def\ltxgrid@warn{%
2310 \ltxgrid@warn@sw{\class@warn}{\@gobble}%
2311 }%
2312 \@booleantrue\ltxgrid@warn@sw
```

10 Line-wise processing

Sometimes we wish to process each line of type that will be placed into the galley, for example, applying line numbering to a document. To accomplish the task, we have to force a visit to the output routine after each such line, whereupon we can process it accordingly (in the case of line numbering, we could do as `ltxgrid.dtxlineno.sty` and append an appropriately formed box to the MVL).

In implementing such a scheme, we will have to instantiate interrupts for the following cases:

\interlinepenalty and friends These include `\clubpenalty`, `\widowpenalty`, `\displaywidowpenalty`, and `\brokenpenalty`.

Display math penalties Includes `\predisdisplaypenalty`, `\postdisplaypenalty`, and `\interdisplaylinepenalty`.

\par The penalty following the last line of the paragraph.

\vadjust A trap for any `\vadjust` command that falls in the paragraph.

`\def@next@handler` Utility procedures `\def@next@handler` and `\def@line@handler` help in the cre-
`\def@line@handler` ation of interrupt handlers.

`\def@next@handler` increments the scratch count register (argument 1), using this value to `\mathchardef` its second argument as the negative of the flag value to be used as a penalty for exciting the interrupt (argument 3). As a byproduct, it leaves the given scratch counter incremented.

```
2313 \def\def@next@handler#1#2#3{%
2314 \advance#1@ne\mathchardef#2\the#1%
2315 \expandafter\def\csname output@-\the#1\endcsname{#3}%
```

The following line is for diagnostic purposes.

```
% \typeout{\string#2(\expandafter\string\csname output@\the#1\endcsname:\expandafter\meaning\csname
%
```

```
2316 }%
```

`\def@line@handler` uses `\int@parpenalty` as a base. The interrupt is the sum of that base with the first argument, and the handler is the second argument.

```
2317 \def\def@line@handler#1#2{%
2318 \begingroup
2319 \@tempcnta\int@parpenalty
2320 \advance\@tempcnta-#1%
```

The following line is for diagnostic purposes.

```
% \typeout{Defining: \expandafter\string\csname output@the\linenopenalty\endcsname}%
%
```

```
2321 \aftergroup\def
2322 \expandafter\aftergroup\csname output@-the\@tempcnta\endcsname
2323 \endgroup{#2}%
2324 }%
```

`\int@parpenalty` We first set `\int@parpenalty` to our chosen base value ≤ -11012 . We then define `\@handle@line@ltx` all the handlers for lines within a paragraph, of which there are 12 different cases.

```
\@handle@line@ltx 2325 \mathchardef\int@parpenalty11012
2326 \def@line@handler\z@{\@handle@line@ltx}{-}{-}}%
2327 \def@line@handler\one{\@handle@line@ltx}{-}{\brokenpenalty@ltx}}%
2328 \def@line@handler\tw@{\@handle@line@ltx}{-}{\clubpenalty@ltx}{-}}%
2329 \def@line@handler\thr@{\@handle@line@ltx}{-}{\brokenpenalty@ltx}}%
2330 \def@line@handler\four@{\@handle@line@ltx}{\widowpenalty@ltx}{-}}%
2331 \def@line@handler{5}{\@handle@line@ltx}{\widowpenalty@ltx}{-}{\brokenpenalty@ltx}}%
2332 \def@line@handler{6}{\@handle@line@ltx}{\widowpenalty@ltx}{-}{\clubpenalty@ltx}{-}}%
2333 \def@line@handler{7}{\@handle@line@ltx}{\widowpenalty@ltx}{-}{\clubpenalty@ltx}{-}{\brokenpenalty@ltx}}%
2334 \def@line@handler{8}{\@handle@line@ltx}{\displaywidowpenalty@ltx}{-}}%
2335 \def@line@handler{9}{\@handle@line@ltx}{\displaywidowpenalty@ltx}{-}{\brokenpenalty@ltx}}%
2336 \def@line@handler{10}{\@handle@line@ltx}{\displaywidowpenalty@ltx}{-}{\clubpenalty@ltx}{-}}%
2337 \def@line@handler{11}{\@handle@line@ltx}{\displaywidowpenalty@ltx}{-}{\clubpenalty@ltx}{-}{\brokenpena
```

The default handler for lines within a paragraph simply restores the value of the `\penalty` to the normal value. If something more useful needs to be done, we can change the definition of `\@handle@line@ltx`.

```
2338 \def\@handle@line@ltx#1#2#3{%
2339 \@handle@line@ltx
2340 \@tempcntb\lastpenalty
2341 \@tempcntb\interlinepenalty@ltx\relax
2342 \@ifempty{#1}{-}{\advance\@tempcntb#1\relax}%
2343 \@ifempty{#2}{-}{\advance\@tempcntb#2\relax}%
2344 \@ifempty{#3}{-}{\advance\@tempcntb#3\relax}%
2345 \penalty\@ifnum{\@tempcnta<\@tempcntb}{\@tempcntb}{\@tempcnta}%
2346 }%
2347 \let\@handle@line@ltx\empty
```

`\int@postparpenalty` We herewith define all the handlers for cases relating to display math: last line before a display math, last line of a display math, and a line within a display math.
`\int@vadjustpenalty`
`\int@whatsitpenalty` We also handle the last line of a paragraph, a whatsit node, and a `\vadjust`.

```
\int@predisplaypenalty
\int@interdisplaylinepenalty
\int@postdisplaypenalty
\@handle@display@ltx
\@handle@display@ltx
\handle@par@ltx
```

```

2348 \@tempcnta\int@parpenalty
2349 \def@next@handler\@tempcnta\int@postparpenalty{\reset@queues@ltx\handle@par@ltx}%
2350 \def@next@handler\@tempcnta\int@vadjustpenalty{\handle@vadjust@ltx}%
2351 \def@next@handler\@tempcnta\int@whatsitpenalty{\handle@whatsit@ltx}%
2352 \def@next@handler\@tempcnta\int@predisplaypenalty{\reset@queues@ltx\@handle@display@ltx{\predis
2353 \def@next@handler\@tempcnta\int@interdisplaylinepenalty{\@handle@display@ltx{\interdisplaylinep
2354 \def@next@handler\@tempcnta\int@postdisplaypenalty{\@handle@display@ltx{\postdisplaypenalty@ltx

```

The default handler for display math lines simply restores the value of the `\penalty` to the normal value. If something more useful needs to be done, we can change the definition of `\@handle@display@ltx`.

```

2355 \def\@handle@display@ltx#1{%
2356 \@handle@display@ltx
2357 \@tempcnta\lastpenalty
2358 \@tempcntb#1%
2359 \penalty\@ifnum{\@tempcnta<\@tempcntb}{\@tempcntb}{\@tempcnta}%
2360 }%
2361 \let\@handle@display@ltx\empty

```

We provide stub definitions for the handlers for the last line of a paragraph, a `\vadjust`, and a `whatsit` node (e.g., `\write`, `\special`). There is no canonical penalty for such cases.

```

2362 \def\handle@par@ltx{}%

```

Note that a `whatsit` needs to be handled differently from a `\vadjust`: a `whatsit` node does not affect the (crucial) depth of `\box\@cclv`, while the more general `\vadjust` may cause any kind of vertical mode material to be interposed just below the line we are trying to trap, in particular `\vskips` and `\penalty`s.

`\set@linepenalties` Now we define utility procedures that set up for a paragraph to be broken into lines, restoring the penalties afterwards.

`\set@displaypenalties` Utility procedure `\set@linepenalties` systematically sets the penalties of paragraph breaking to flag values, meanwhile storing away the normal values for access by the output routine.

```

2363 \def\set@linepenalties{%
2364 \expandafter\def\expandafter\interlinepenalty@ltx\expandafter{\the\interlinepenalty}%
2365 \interlinepenalty-\int@parpenalty
2366 \expandafter\def\expandafter\brokenpenalty@ltx\expandafter{\the\brokenpenalty}%
2367 \brokenpenalty\@ne
2368 \expandafter\def\expandafter\clubpenalty@ltx\expandafter{\the\clubpenalty}%
2369 \clubpenalty\tw@
2370 \expandafter\def\expandafter\widowpenalty@ltx\expandafter{\the\widowpenalty}%
2371 \widowpenalty\f@ur
2372 \expandafter\def\expandafter\displaywidowpenalty@ltx\expandafter{\the\displaywidowpenalty}%
2373 \displaywidowpenalty8\relax
2374 }%

```

Utility procedure `\restore@linepenalties` restores the values of the penalty parameters that were modified by `\set@linepenalties`.

```

2375 \def\restore@linepenalties{%

```

```

2376 \interlinepenalty\interlinepenalty@ltx
2377 \brokenpenalty\brokenpenalty@ltx
2378 \clubpenalty\clubpenalty@ltx
2379 \widowpenalty\widowpenalty@ltx
2380 \displaywidowpenalty\displaywidowpenalty@ltx
2381 \relax
2382 }%

```

In the following, the first argument should be a boolean (either `\true@sw` or `\false@sw`).

```

2383 \def\set@displaypenalties#1{%
2384 \expandafter\def\expandafter\predisplaypenalty@ltx\expandafter{\the\predisplaypenalty}%
2385 \expandafter\def\expandafter\interdisplaylinepenalty@ltx\expandafter{\the\interdisplaylinepenalty}%
2386 \expandafter\def\expandafter\postdisplaypenalty@ltx\expandafter{\the\postdisplaypenalty}%
2387 \@ifhmode{\predisplaypenalty-\int@predisplaypenalty\relax}{}%
2388 #1{\interdisplaylinepenalty-\int@interdisplaylinepenalty\relax}{}%
2389 #1{\postdisplaypenalty-\int@postdisplaypenalty\relax}{}%
2390 }%

```

We provide no procedure to restore the respective penalties, because they are altered within a group: T_EX's context stack will automatically restore things.

```

\enqueue@whatsit@ltx Here is a facility for dealing with whatsit nodes while we are trapping paragraph
\handle@whatsit@ltx lines. We simply enqueue a macro that will create the desired whatsit node,
\do@whatsit dequeuing it in the output routine.

```

```

\g@pop@ltx 2391 \def\enqueue@whatsit@ltx#1{%
2392 \gappdef\g@whatsit@queue{{#1}}%
2393 \vadjust{\penalty-\int@whatsitpenalty}%
2394 }%
2395 \def\handle@whatsit@ltx{%
2396 \unvbox\@cclv
2397 \g@pop@ltx\g@whatsit@queue\@tempa
2398 \expandafter\do@whatsit\expandafter{\@tempa}%
2399 }%
2400 \def\do@whatsit#1{%
2401 \def\g@pop@ltx#1#2{%
2402 \expandafter\g@pop@ltx#1{}{\@#1#2}%
2403 }%
2404 \def\g@pop@ltx#1#2\@#3#4{%
2405 \gdef#3{#2}%
2406 \def#4{#1}%
2407 }%

```

```

\vspace We wish to prevent ltxgrid.dtxlineno.sty from patching \vspace and \pagebreak,
\pagebreak because that package does it through global assignments, which is prone to failure.
\nopagebreak We also wish to prevent that package from patching \@arrayparboxrestore,
\@arrayparboxrestore because it prevents us from \unvboxing vertical mode material into the MVL and
numbering those lines.

```

We start by retaining the original definitions of these commands, so we can restore them if ltxgrid.dtxlineno.sty does get loaded.


```

2408 \let\vspace@ltx\vspace
2409 \let\pagebreak@ltx\pagebreak
2410 \let\nopagebreak@ltx\nopagebreak
2411 \let\endline@ltx\
2412 \let\@arrayparboxrestore@ltx\@arrayparboxrestore

```

Next, we provide for line-wise processing by patching the procedures associated with these same three commands.

There are exactly four core L^AT_EX procedures that use `\vadjust` to insert vertical mode material into the main vertical list: `\vspace`, `\pagebreak`, `\nopagebreak`, and `\`. Other commands may use `\vadjust`, but they are inserting an interrupt (via a penalty < 10000), and such a thing does not mask the depth of `\box\@cc1v`, hence is permissible.

In each case, we replace the core L^AT_EX procedure with one that itself replaces `\vadjust` with `\ex@vadjust@ltx`. The meaning of this procedure can be left as `\vadjust`, or it can be changed to one that accomplishes the equivalent without masking the depth of `\box\@cc1v`.

The first procedure is `\@vspace`, here shown in original form and in the patched alternative form. This procedure and `\@vspacer` implement the `\vspace` command.

```

2413 \def\@vspace@org #1{%
2414   \ifvmode
2415     \vskip #1
2416     \vskip\z@skip
2417   \else
2418     \@bsphack
2419     \vadjust{\@restorepar
2420               \vskip #1
2421               \vskip\z@skip
2422             }%
2423     \@esphack
2424   \fi
2425 }%
2426 \def\@vspace@ltx#1{%
2427   \@ifvmode{%
2428     \vskip#1\vskip\z@skip
2429   }{%
2430     \@bsphack
2431     \ex@vadjust@ltx{%
2432       \@restorepar
2433       \nobreak
2434       \vskip#1\vskip\z@skip
2435     }%
2436     \@esphack
2437   }%
2438 }%

```

The second procedure is `\@vspacer`.

```

2439 \def\@vspacer@org#1{%

```

```

2440 \ifvmode
2441   \dimen@\prevdepth
2442   \hrule \@height\z@
2443   \nobreak
2444   \vskip #1
2445   \vskip\z@skip
2446   \prevdepth\dimen@
2447 \else
2448   \@bsphack
2449   \vadjust{\@restorepar
2450             \hrule \@height\z@
2451             \nobreak
2452             \vskip #1
2453             \vskip\z@skip}%
2454   \@esphack
2455 \fi
2456 }%
2457 \def\@vspacer@ltx#1{%
2458 \@ifvmode{%
2459   \dimen@\prevdepth
2460   \hrule \@height\z@
2461   \nobreak
2462   \vskip#1\vskip\z@skip
2463   \prevdepth\dimen@
2464 }{%
2465   \@bsphack
2466   \ex@vadjust@ltx{%
2467     \@restorepar
2468     \hrule \@height\z@
2469     \nobreak
2470     \vskip#1\vskip\z@skip
2471   }%
2472   \@esphack
2473 }%
2474 }%

```

The procedure \@no@pgbk implements both \pagebreak and \nopagebreak.

```

2475 \def\@no@pgbk@org #1[#2]{%
2476 \ifvmode
2477   \penalty #1\@getpen{#2}%
2478 \else
2479   \@bsphack
2480   \vadjust{\penalty #1\@getpen{#2}}%
2481   \@esphack
2482 \fi
2483 }%
2484 \def\@no@pgbk@ltx#1[#2]{%
2485 \@ifvmode{%
2486   \penalty#1\@getpen{#2}%
2487 }{%

```

```

2488 \@bsphack
2489 \ex@vadjust@ltx{%
2490 \penalty#1\@getpen{#2}%
2491 }%
2492 \@esphack
2493 }%
2494 }%

```

The command to end a line of type, `\@`, is defined via `\DeclareRobustCommand`, so we must proceed carefully: A procedure is defined whose `\long\csname` is constructed via the incantation: `\csname\expandafter\@gobble\string\@end\endcsname`. Note the non-trivial space character after the `\@`: it is incorporated into the `\csname`.

Here is the original core \LaTeX definition for the procedure involved, along with our revised version.

```

2495 \long\def\end@line@org{%
2496 \let\reserved@e\relax
2497 \let\reserved@f\relax
2498 \@ifstar{%
2499 \let\reserved@e\vadjust
2500 \let\reserved@f\nobreak
2501 \@xnewline
2502 }%
2503 \@xnewline
2504 }%
2505 \long\def\end@line@ltx{%
2506 \let\reserved@e\relax
2507 \let\reserved@f\relax
2508 \@ifstar{%
2509 \let\reserved@e\ex@vadjust@ltx
2510 \let\reserved@f\nobreak
2511 \@xnewline
2512 }{%
2513 \@xnewline
2514 }%
2515 }%

```

An additional procedure requiring patching has the following original core \LaTeX definition; we modify it correspondingly.

```

2516 \def\@newline@org[#1]{%
2517 \let\reserved@e\vadjust
2518 \@gnewline{\vskip#1}%
2519 }%
2520 \def\@newline@ltx[#1]{%
2521 \let\reserved@e\ex@vadjust@ltx
2522 \@gnewline{\vskip#1}%
2523 }%

```

We now install our patches. If some package overrides these macros, we will detect and complain.

```

2524 \@ifx{\@vspace\@vspace@org}{%
2525   \@ifx{\@vspacer\@vspacer@org}{%
2526     \@ifx{\@no@pgbk\@no@pgbk@org}{%
2527       \@ifx{\@newline\@newline@org}{%
2528         \expandafter\@ifx\expandafter{\csname\expandafter\@gobble\string\ \endcsname\end@line@org
2529         \true@sw
2530         }{\false@sw}%
2531         }{\false@sw}%
2532         }{\false@sw}%
2533         }{\false@sw}%
2534       }{\false@sw}%
2535     }%
2536     \class@info{0overriding \string\@vspace, \string\@vspacer, \string\@no@pgbk, \string\@newline,
2537     \let\@normalcr\end@line@ltx
2538     \expandafter\let\csname\expandafter\@gobble\string\ \endcsname\@normalcr
2539     \let\@newline\@newline@ltx
2540     \let\@vspace\@vspace@ltx
2541     \let\@vspacer\@vspacer@ltx
2542     \let\@no@pgbk\@no@pgbk@ltx
2543   }{%
2544     \class@warn{%
2545       Failed to recognize \string\@vspace, \string\@vspacer, \string\@no@pgbk, \string\@newline, a
2546       no patches applied. Please get a more up-to-date class,
2547     }%
2548   }%

```

Note that we have assigned the same meaning to `\@normalcr`, which is necessary to L^AT_EX.

`\ex@vadjust@ltx` Here we give the default definition for `\ex@vadjust@ltx` along with the definitions `\enqueue@vadjust@ltx` for the alternative version and its the associated handler.

```

\handle@vadjust@ltx 2549 \let\ex@vadjust@ltx\vadjust
\g@vadjust@line 2550 \def\enqueue@vadjust@ltx#1{%
\reset@queues@ltx 2551 \gappdef\g@vadjust@queue{{#1}}%
2552 \vadjust{\penalty-\int@vadjustpenalty}%
2553 }%
2554 \def\handle@vadjust@ltx{%
2555 \unvbox\@cclv
2556 \g@pop@ltx\g@vadjust@queue\@tempa
2557 \expandafter\gappdef\expandafter\g@vadjust@line\expandafter{\@tempa}%
2558 }%
2559 \let\g@vadjust@line\@empty

```

Procedure `\reset@queues@ltx` resets the `whatsit` queue and the `\vadjust` queues to their empty state. This should be done whenever we leave horizontal mode and complete the processing of these queues: upon executing, effectively, primitive `\par` or interrupting a paragraph with display math.

```

2560 \def\reset@queues@ltx{%
2561 \global\let\g@whatsit@queue\@empty
2562 \global\let\g@vadjust@queue\@empty

```

2563 }%

11 Patching the `lineno.sty` package

`ltxgrid.dtxlineno.sty` is a L^AT_EX package that applies line numbering to a document. The basic method is to give `\interlinepenalty` and like penalties such a value as to force a visit to the output routine, where the line of type is given its number. In order to properly measure the depth of `\box\@cclv`, it defers `\vadjust` commands that may insert `\vskip` or `\penalty` nodes.

The implementation of that package, however, manipulates `\holdinginserts` in a dangerous way: outside the safety of the output routine. It also alters the meaning of `\vadjust` using global assignments. We patch its code to avoid these problems. The `ltxgrid.dtxltxgrid` package already has the needed mechanisms in place to do these jobs correctly.

The methods we use can accomodate any values of penalties like `\clubpenalty`, etc: we do not make assumptions about the range of values these penalty parameters could take.

`\linenomathWithnumbers` Here are the definitions of procedures in `ltxgrid.dtxlineno.sty` that alter `\holdinginserts`.
`\linenomathNonumbers` They are current as of version v4.41, 2005/11/02. We patch them to avoid doing
`\endlinenomath` this: in `ltxgrid`-based classes like REVT_EX, the output routine properly manages
`\linenumberpar` `\holdinginserts`, so packages should not attempt to do so. Also, we will want
`\linenumberpar` to set `\interlinepenalty` to dispatch to `\MakeLineNo`.

```
2564 \newcommand\linenomathWithnumbers@LN{%
2565   \ifLineNumbers
2566     \ifnum\interlinepenalty>-\linenopenaltypar
2567       \global\holdinginserts\thr@@
2568       \advance\interlinepenalty \linenopenalty
2569       \ifhmode
2570         \advance\predisplaypenalty \linenopenalty
2571       \fi
2572       \advance\postdisplaypenalty \linenopenalty
2573       \advance\interdisplaylinepenalty \linenopenalty
2574     \fi
2575   \fi
2576   \ignorespaces
2577 }%
2578 \newcommand\linenomathNonumbers@LN{%
2579   \ifLineNumbers
2580     \ifnum\interlinepenalty>-\linenopenaltypar
2581       \global\holdinginserts\thr@@
2582       \advance\interlinepenalty \linenopenalty
2583     \ifhmode
2584       \advance\predisplaypenalty \linenopenalty
2585     \fi
2586   \fi
2587 \fi
```

```

2588 \ignorespaces
2589 }%
2590 \def\endlinenomath@LN{%
2591   \ifLineNumbers
2592     \global\holdinginserts\@LN@outer@holdins
2593   \fi
2594   \global\@ignoretrue
2595 }
2596 \def\linenumberpar@LN{%
2597   \ifvmode \@@@par \else
2598     \ifinner \@@@par \else
2599       \xdef\@LN@outer@holdins{\the\holdinginserts}%
2600       \advance \interlinepenalty \linenopenalty
2601       \linenoprevgraf \prevgraf
2602       \global \holdinginserts \thr@@
2603       \@@@par
2604       \ifnum\prevgraf>\linenoprevgraf
2605         \penalty-\linenopenaltypar
2606       \fi
2607       \@LN@parpgbrk
2608       \global\holdinginserts\@LN@outer@holdins
2609       \advance\interlinepenalty -\linenopenalty
2610     \fi
2611   \fi
2612 }%

```

`\class@documenthook` We patch only if we recognize the definitions of all the procedures we are to patch.

```

2613 \appdef\class@documenthook{%
2614   \ifpackageloaded{lineno}{%
2615     \ifx{\linenomathWithnumbers\linenomathWithnumbers@LN}{%
2616       \ifx{\linenomathNonumbers\linenomathNonumbers@LN}{%
2617         \ifx{\endlinenomath\endlinenomath@LN}{%
2618           \ifx{\linenumberpar\linenumberpar@LN}{%
2619             \true@sw
2620           }\false@sw}%
2621         }\false@sw}%
2622       }\false@sw}%
2623     }\false@sw}%
2624   }%
2625   \class@info{Overriding lineo.sty, restoring output routine,}%

```

We commence overriding the procedures of `ltxgrid.dtxlineno.sty`.

```

2626 \let\linenumberpar\linenumberpar@ltx
2627 \let\endlinenomath\endlinenomath@ltx
2628 \expandafter\let\csname endlinenomath*\endcsname\endlinenomath@ltx
2629 \let\linenomathWithnumbers\linenomathWithnumbers@ltx
2630 \let\linenomathNonumbers\linenomathNonumbers@ltx

```

Override `ltxgrid.dtxlineno.sty`'s equipment for `\vadjust` and `\lineatop`: we have existing interrupts and handlers for these purposes.

```

2631 \let\ex@vadjust@ltx\ex@vadjust@line
2632 \let\@LN@postlabel\enqueue@whatsit@ltx
2633 \let\do@whatsit\write@linelabel

```

Redirect handlers to those provided by ltxgrid.dtxlineno.sty, and give an appropriate meaning to the respective headpatch within the handlers.

```

2634 \let\handle@par@ltx\handle@par@LN
2635 \let\@@handle@line@ltx\Make@LineNo@ltx
2636 \let\@@handle@display@ltx\Make@LineNo@ltx

```

Next, we undo the action taken by ltxgrid.dtxlineno.sty wherein it took over the output routine. Instead, we service ltxgrid.dtxlineno.sty existing equipment of ltxgrid.dtxltxgrid. We also revert the core L^AT_EX definitions of `\vspace`, `\pagebreak`, `\nopagebreak`, and `\`, which that package takes over (we have our own ways of doing these things).

```

2637 \output@latex{\natural@output}%
2638 \let\vspace\vspace@ltx
2639 \let\pagebreak\pagebreak@ltx
2640 \let\nopagebreak\nopagebreak@ltx
2641 \let\@arrayparboxrestore\@arrayparboxrestore@ltx
2642 \let\\\endline@ltx

```

When line numbering is in effect, we must avoid any attempt to number the lines of a footnote.

```

2643 \appdef\set@footnotefont{%
2644 \let\par\@@par
2645 \let\@@par\@@@par
2646 }%

```

At last, we detect if the `\linenumbers` command has already been given; if so, we do its assignments again, because we have changed the meaning of `\linenumberpar`.

```

2647 \@if@sw@ifLineNumbers\fi{%
2648 \class@info{Reinvoke \string\linenumbers}%
2649 \let\@par\linenumberpar
2650 \@ifx{\@par\linenumberpar@LN}{\let\@par\linenumberpar}{}%
2651 \@ifx{\par\linenumberpar@LN}{\let\par\linenumberpar}{}%
2652 }{%
2653 \class@info{Line numbering not turned on yet}%
2654 }%

```

Here ends the “true branch” of the patch code.

```

2655 }{%

```

If the ltxgrid.dtxlineno.sty package is loaded, but we fail to patch it, notify the user.

```

2656 \class@warn{Failed to recognize lineno.sty procedures; no patches applied. Please get a more
2657 }%
2658 }{%

```

ltxgrid.dtxlineno.sty is not loaded, so no patches are needed.

```

2659 }%

```

```

2660 }%

\linenumberpar Procedure \linenumberpar takes the place of \par when line numbering is in
\@linenumberpar effect; It executes the \par primitive if we are in vertical mode. Otherwise we are
in horizontal mode in the MVL and wish to end the current paragraph, or we have
\unvboxed material onto the MVL.

2661 \def\linenumberpar@ltx{\@ifvmode{\@@@par}{\@linenumberpar}}%
    Procedure \@linenumberpar
2662 \def\@linenumberpar{%
    Prepare for our trip into the output routine by saving away the current value of
    \prevgraf.
2663   \linenoprevgraf\prevgraf
    The following will be used in the output routine dispatcher to sense that we came
    from here.
2664   \set@linepenalties
    Finally, call primitive \par with the signal value of \interlinepenalty and
    friends.
2665   \@@@par
    We are now in vertical mode. If lines of type were contributed to the MVL (non-
    trivial paragraph), we must force another trip into the output routine to apply
    line numbering to the last line of the paragraph.
2666   \@ifnum{\prevgraf>\linenoprevgraf}{
2667     \penalty-\int@postparpenalty
2668   }{}%
    Execute procedure \@LN@parpgbrk, which has been set up in the output routine
    for us to invoke here.
2669   \@LN@parpgbrk
    To wrap things up, we restore the original value of \interlinepenalty and
    friends.
    Query: why not employ TEX's context stack to do the restore? Would there
    be something wrong with executing primitive \par within a group?
2670   \restore@linepenalties
2671 }%

\linenomathWithnumbers Here are the patched definitions for the commands enabling line numbering in
\linenomathNonnumbers display math.
2672 \newcommand\linenomathWithnumbers@ltx{\@linenomathnumbers@ltx>true@sw}%
2673 \newcommand\linenomathNonnumbers@ltx{\@linenomathnumbers@ltx>false@sw}%

\@linenomathnumbers We have just begun a display math, and any paragraph we are setting will now
\endlinenomath end. We set all relevant penalties to interrupt values; in the visit to the output
routine, we will replace the penalty with its normal value.
2674 \def\@linenomathnumbers@ltx#1{%

```



```

2675 \if@sw\ifLineNumbers\fi{%
2676   \set@linepenalties
2677   \set@displaypenalties#1%
2678 }{%
2679 \ignorespaces
2680 }%
2681 \def\endlinenomath@ltx{%
2682 \global\@ignoretrue
2683 }%

```

We provide a handler for the last line of a paragraph.

```

2684 \def\handle@par@LN{%
2685 \Make@LineNo@ltx

```

After setting the line number, we arrange for an appropriate penalty to be laid down after this visit to the output routine ends.

Query: why not contribute the penalty right here in the visit to the output routine?

```

2686 \@tempcnta\lastpenalty
2687 \ifnum{\@tempcnta=\z@}{-}{%
2688 \expandafter\gdef
2689 \expandafter\@LN@parpgbrk
2690 \expandafter{%
2691 \expandafter\penalty
2692 \the\@tempcnta

```

When `\@LN@parpgbrk` is executed, it resets itself to the default value, `\@LN@screenoff@pen`.

Query: `\@LN@screenoff@pen` appears to try to restore the depth of the last box: why is this being done outside the safety of the output routine?

```

2693 \global\let\@LN@parpgbrk\@LN@screenoff@pen
2694 }%
2695 }%
2696 }%

```

`\Make@LineNo` The procedure `\Make@LineNo` sets the box containing the line number itself.

```

2697 \def\Make@LineNo@ltx{%
2698 \@LN@maybe@normalLineNumber

```

We measure the depth of `\box\@cclv` and unbox it. At this point, it is crucial that that box have the same depth as that of the last box within it.

In the simple case, `\box\@cclv` is a `\vbox` containing as its last box the `\hbox` of the paragraph we are processing.

Query: under what circumstances will this *not* be the case?

```

2699 \boxmaxdepth\maxdimen\setbox\z@\vbox{\unvbox\@cclv}%
2700 \@tempdima\dp\z@
2701 \unvbox\z@

```

Then we create the box with the line number, setting its height to zero.

```

2702 \sbox\@tempboxa{\hb@xt@\z@\makeLineNumber}}%
2703 \ht\@tempboxa\z@

```

With these preparations, we invoke `\@LN@depthbox`, which lays that box down (with its depth appropriately set): this procedure depends on our having set `\@tempdima` and `\@tempboxa` (kinda kludgy way of passing arguments, really).

```
2704 \@LN@depthbox
```

Now increment the line number. I have relocated this token past `\@LN@depthbox`: this may induce a bug, but I am going to hereby force the issue: why split up the procedure that lays down boxes with a procedure that sets a register value?

```
2705 \stepLineNumber
```

Finally, execute the `\vadjusts` that fell within the line that we just handled.

Note that `\enqueue@vadjust@ltx` had queued up all the `\vadjust` commands for the paragraph into `\g@vadjust@queue`, laying down an `(\int@vadjustpenalty)` interrupt in each ones' place. The interrupts associated with this line of the paragraph have now moved the tokens to `\g@vadjust@line`, which we now expand and execute.

```
2706 \g@vadjust@line
```

```
2707 \global\let\g@vadjust@line\empty
```

```
2708 }%
```

```
2709 \def\write@linelabel#1{%
```

```
2710 \protected@write\@auxout{}{%
```

```
2711 \string\newlabel{#1}{{\theLineNumber}{\thepage}{}}{}}%
```

```
2712 }%
```

```
2713 }%
```

```
2714 \def\ex@vadjust@line{%
```

```
2715 \@if@sw@ifLineNumbers\fi{\enqueue@vadjust@ltx}{\vadjust}}%
```

```
2716 }%
```

Note that the `\linelabel` commands use a mechanism different from `\vadjust`, embodied in the procedure `\enqueue@vadjust@ltx`, wherein the `\write` primitives are enqueued while the paragraph is being processed, each replaced with an interrupt, then dequeued and executed by the interrupt handler, leaving a `\write` node in place of the interrupt (just where the `\vadjust`'s vertical mode material would had been) just below the box containing the line of type. This `\write`, like all whatsits, does not affect the depth of `\box@cc@lv`, unlike the case of general vertical mode material, which could have interfered.

12 End of the ltxgrid DOCSTRIP module

Here ends the module.

```
2717 %</kernel>
```

Here ends the programmer's documentation.

Index

Symbols	
\$TEXMF/	3
.dtx	5
.ins	5
\@@	2215, 2219, 2402, 2404
\@@@par	2597, 2598, 2603, 2644, 2645, 2661, 2665
\@@botmark	16, 17
\@@botmark	188, 226, 295, 497, 507, 522, 906
\@@end	55
\@@endpbox	2035, 2088
\@@enlargethispage	91
\@@enlargethispage	2263, 2267
\@@firstmark	16
\@@firstmark	188, 293, 905, 1272
\@@handle@display@ltx	95
\@@handle@display@ltx	2348, 2636
\@@handle@line@ltx	94
\@@handle@line@ltx	2325, 2635
\@@mark	16, 17, 58
\@@mark	188, 210, 389, 1292
\@@nil	244, 250
\@@nul	195, 199–202
\@@par	257, 2645, 2649
\@s@splitbotmark	188
\@s@splitfirstmark	188
\@startpbox	2034, 2087
\@topmark	16, 55
\@topmark	188, 291, 904, 1225
\@Esphack	969
\@LN@depthbox	106
\@LN@depthbox	2704
\@LN@maybe@normalLineNumber	2698
\@LN@outer@holdins	2592, 2599, 2608
\@LN@output	20, 21
\@LN@parpgbrk	104, 105
\@LN@parpgbrk	2607, 2669, 2689, 2693
\@LN@postlabel	2632
\@LN@screenoff@open	105
\@LN@screenoff@open	2693
\@M	60
\@Mii	965, 1132
\@acol	2031, 2083, 2100
\@acoll	2098
\@acolr	2099
\@add@float	51, 73
\@add@float	966, 968, 1063
\@addmarginpar	53
\@addmarginpar	1073, 1155
\@addmarginpar@	63
\@addmarginpar@mkt	1467
\@addmarginpar@one	53
\@addmarginpar@one	1415
\@addstuff	1336, 1337
\@addtobot	1112
\@addtocurcol	52
\@addtocurcol	1071, 1090
\@addtodblcol	37, 39
\@addtodblcol	703, 710
\@addtonextcol	36, 37
\@addtonextcol	626, 631
\@addtotoporbob	648, 1139
\@argswap	916, 927
\@arraycr	2040
\@arraycr@array	2095
\@arrayparboxrestore	96
\@arrayparboxrestore	2408, 2641
\@arrayparboxrestore@ltx	2412, 2641
\@arstrut	2054, 2122
\@arstrutbox	1963, 1967, 1969, 2042, 2102
\@auxout	1904, 1935, 2710
\@begindocumenthook	1877
\@bitor	646, 717, 793, 819, 1107, 1110
\@booleanfalse	315, 359, 360, 600, 603, 666, 668, 677, 709, 762, 865, 876, 1622, 1743, 1748, 1752, 2244, 2308

<code>\@booleantrue</code>	605, 629, 670, 706, 722, 806, 860, 1705, 1737, 1738, 1761, 2231, 2232, 2248, 2249, 2295, 2312	<code>\@colroom</code>	602, 643, 1104, 1116, 1711, 1713, 1714, 1716–1718, 1720, 1725, 1849, 1850, 1972, 1975, 2288
<code>\@botlist</code>	88, 90	<code>\@combinedbfloats</code> . .	23, 38, 72
<code>\@botlist</code> .	301, 443, 1110, 1502, 1661, 1668, 1669, 1738	<code>\@combinedbfloats</code> .	698, 1491, <u>1671</u>
<code>\@botnum</code>	884	<code>\@combinefloats</code>	552
<code>\@botroom</code>	885	<code>\@combineinserts</code>	71, 76
<code>\@bsphack</code>	2418, 2430, 2448, 2465, 2479, 2488	<code>\@combineinserts</code>	554, <u>572</u> , 1639
<code>\@capyte</code>	1055	<code>\@combinepage</code>	23, 25, 38, 71
<code>\@cclv</code> . .	19, 20, 24, 26–29, 31, 32, 34, 35, 37, 44–46, 55–59, 63, 90, 95, 97, 101, 105, 106	<code>\@combinepage</code> . .	697, 1490, <u>1627</u>
<code>\@cclv</code>	299, 305, 374, 375, 377, 411, 412, 439, 445, 448, 473, 499, 510, 524, 546, 547, 597, 608, 663, 1064, 1065, 1232, 1240, 1246, 1247, 1249, 1303, 1306, 1326, 1436, 1525, 1527, 1532, 2182, 2396, 2555, 2699	<code>\@comdblfilelt</code>	1674
<code>\@cclv@nontrivial@sw</code>	29	<code>\@comflelt</code>	1645, 1659
<code>\@cclv@nontrivial@sw</code>	<u>408</u> , 1256, 1478, 1524	<code>\@cons</code>	656, 755, 757, 798, 808, 828, 830, 964, 1077, 1120, 1150, 1157
<code>\@cclv@saved</code>	56	<code>\@currbox</code>	50
<code>\@cclv@saved</code>	306, <u>1220</u> , 1223, 1227–1229, 1240	<code>\@currbox</code>	626, 635, 639, 656, 703, 712, 725, 731, 749, 755, 757, 951–953, 964, 975, 981, 1043, 1049–1051, 1066–1068, 1077, 1103, 1114, 1118, 1120, 1129, 1150, 1157, 1162, 1169, 1198, 1201, 1202
<code>\@cflb</code>	72	<code>\@currlist</code>	964, 1066, 1156
<code>\@cflb</code>	<u>1644</u>	<code>\@currtype</code>	41
<code>\@cflt</code>	72	<code>\@currtype</code>	646, 717, 791–793, 1107, 1110
<code>\@cflt</code>	<u>1644</u>	<code>\@dbldeferlist</code>	39, 43
<code>\@classiv</code>	2032, 2085	<code>\@dblfloat</code>	46
<code>\@classz</code>	2032, 2084	<code>\@dblfloat</code>	<u>911</u>
<code>\@clearfloatplacement</code>	45	<code>\@dblfloatplacement</code>	664, 1768, 1881
<code>\@clearfloatplacement</code>	598, 664, <u>881</u>	<code>\@dbltopinsert</code>	43
<code>\@colht</code>	37, 40, 43, 70, 73–75	<code>\@dbltoplist</code>	65, 73, 88, 90
<code>\@colht</code>	602, 681, 753, 796, 824, 883, 885, 887, 889, 890, 1593, 1597, 1698, 1729, 1972	<code>\@dbltoplist</code>	302, 751, 755, 1503, 1672, 1674–1676, 1705
<code>\@colnum</code>	644, 645, 886, 1105, 1106, 1117	<code>\@dbltopnum</code>	723, 724, 754, 888, 1680
<code>\@colroom</code>	28, 37, 43, 64, 73, 74, 77	<code>\@dbltoproom</code>	725, 730, 731, 752, 889, 890
		<code>\@deferlist</code>	25, 39, 40, 43, 88, 89, 91
		<code>\@deferlist</code>	303, 614, 646, 656, 679, 717, 757, 763,

764, 779, 1077, 1107, 1150, 1504, 2195, 2211	\@fpsep 41
\@deferlist@postshipout . 2194, 2195, 2204, 2211	\@fpsep 786, 823
\@depth . . 1192, 1571, 2044, 2105	\@fpstype . . . 636, 637, 728, 893, 1093, 1095
\@docclearpage 46	\@fptop 41
\@docclearpage 895	\@fptop 773
\@eha 1308	\@freelist 40, 88
\@ehc 1886	\@freelist 550, 780, 1157, 1655, 1668, 1675, 2222
\@elt 613, 678, 770, 774, 778, 801, 804, 812, 1645, 1654, 1659, 1667, 1674, 1675, 1704, 1706, 1736, 1739, 2219	\@g@pop@ltx 2391
\@empty 46	\@getfpsbit 50
\@endfloatbox 960, 974, 976, 1197	\@getfpsbit 716
\@endpbox 2029, 2035, 2081, 2088	\@getpen . 2477, 2480, 2486, 2490
\@enlargethispage . 2254, 2256, 2259	\@gnewline 2518, 2522
\@esphack 2423, 2436, 2454, 2472, 2481, 2492	\@gobble 99
\@execute@message 58	\@handle@display@ltx . . . 2348
\@execute@message . 1280, 1283, 1285	\@handle@line@ltx 2325
\@failedlist 40	\@height . 1192, 1571, 1600, 2043, 2104, 2442, 2450, 2460, 2468
\@failedlist 765, 779, 793, 798, 808, 819	\@holdpg 55
\@flfail 40	\@holdpg 1220, 1221
\@flfail 779, 802, 819, 828	\@if@empty 905, 906, 1340, 1343, 1349, 1359, 1571, 1712, 2342–2344
\@float 46	\@if@exceed@pagegoal 372, 1246
\@float 911	\@if@notdblfloat 38
\@floatpenalty 962, 965	\@if@notdblfloat 660, 712, 1546
\@floatplacement 589, 598, 1767, 1880	\@if@sw 220, 629, 647, 655, 706, 718, 797, 827, 870, 1081, 1443, 1446, 1448, 1932, 2647, 2675, 2715
\@floatselect@sw@ 37	\@ifdim . . 337, 340, 388, 397, 414, 449, 454, 456, 471, 643, 681, 708, 725, 731, 767, 796, 805, 824, 1173, 1287, 1361, 1364, 1593, 1599, 1616, 1716, 1817, 1818, 1821, 1830, 1831, 1848, 1849
\@floatselect@sw@m1t . . . 1546	\@iffpsbit 48, 50
\@floatselect@sw@one . . . 1451	\@iffpsbit 951, 952, 1041
\@flsetnum 644, 723, 1105	\@ifhmode 1305, 2387
\@flsettextmin 638, 1097	\@ifnextchar 912, 923, 999, 1020, 1888, 1892
\@flsucceed 40	\@ifnotrelax 259
\@flsucceed . . 774, 780, 801, 830	\@ifnum 180, 186, 410, 502, 516, 525, 636, 637,
\@fltstk 2196, 2207	
\@fpbot 41	
\@fpbot 776	
\@fpmin 35, 40, 41	
\@fpmin . . 590, 768, 805, 887, 891	

645, 724, 728, 920, 947, 962, 965, 1049, 1052, 1068, 1080, 1315, 1341, 1350–1352, 1358, 1375, 1454, 1480, 1487, 1505, 1536, 1548, 1559, 1580, 1680, 1695, 2188, 2274, 2345, 2359, 2666, 2687	<code>\@linenomathnumbers@ltx</code> 2672– 2674
<code>\@ifodd</code> 721, 871, 1045, 1444, 2187	<code>\@linenumberpar</code> 104
<code>\@ifpackageloaded</code> .. 2141, 2614	<code>\@linenumberpar</code> 2661
<code>\@ifstar</code> 2253, 2498, 2508	<code>\@makecolumn</code> . 12, 23, 26, 28, 33, 34, 63, 64, 66, 73
<code>\@ifundefined</code> .. 991, 1003, 1012, 1024	<code>\@makecolumn</code> 421, 543, 1426, 1432, 1479
<code>\@ifvbox</code> 1859, 1866	<code>\@makefcolumn</code> 46
<code>\@ifvmode</code> . 220, 1302, 2427, 2458, 2485, 2661	<code>\@makefcolumn</code> 896
<code>\@ifvoid</code> 444, 445, 531, 534, 576, 1223, 1228, 1261, 1402, 1404, 1628, 1636, 1731, 1759, 1777, 1792, 1794, 1796, 1863, 1948, 1988	<code>\@makespecialcolbox</code> 34
<code>\@ifx</code> 262, 420, 424, 431, 751, 1029, 1030, 1380, 1387, 1941, 2001, 2142–2146, 2194, 2195, 2214, 2223, 2224, 2269, 2273, 2524–2528, 2615–2618, 2650, 2651	<code>\@makespecialcolbox</code> 571
<code>\@ifx@empty</code> 65	<code>\@marbox</code> . 1156, 1157, 1159, 1165, 1172, 1180, 1182, 1183, 1185– 1187, 1190
<code>\@ifx@empty</code> 442, 443, 763, 1501–1504, 1672	<code>\@maxdepth</code> 545, 569
<code>\@ifxundefined</code> .. 314, 904, 1028, 1198, 1212, 1622, 2082	<code>\@message@saved</code> 21
<code>\@ifxundefined@cs</code> 915, 926	<code>\@message@saved</code> 263, 1273, 1276, 1278
<code>\@inlabelfalse</code> 843	<code>\@midlist</code> 550, 1120
<code>\@insertfalse</code> 633, 1091	<code>\@mkpream</code> 2048, 2111
<code>\@inserttrue</code> 1134	<code>\@mkpream@relax</code> 2112
<code>\@kludgeins</code> 34, 92	<code>\@mparbottom</code> 436, 1170, 1178–1181, 1917, 1949
<code>\@kludgeins</code> 2294	<code>\@myadjust</code> 11, 12
<code>\@largefloatcheck</code> 963	<code>\@namedef</code> 595, 661, 878, 986, 1003, 1007, 1024, 1214, 1219, 1244, 1269, 1278, 2246
<code>\@latex@warning</code> 2208	<code>\@ne</code> 76
<code>\@latex@warning@no@line</code> . 1174	<code>\@newline</code> 2527, 2536, 2539, 2545
<code>\@latexbug</code> 1079, 1158	<code>\@newline@ltx</code> 2520, 2539
<code>\@latexerr</code> 1308	<code>\@newline@org</code> 2516, 2527
<code>\@leftcolumn</code> 69	<code>\@next</code> 790, 1066, 1156, 2213
<code>\@leftcolumn</code> 1554, 1555	<code>\@no@pgbk</code> 98
<code>\@linenomathnumbers</code> 2674	<code>\@no@pgbk</code> 2526, 2536, 2542, 2545
	<code>\@no@pgbk@ltx</code> 2484, 2542
	<code>\@no@pgbk@org</code> 2475, 2526
	<code>\@nbreakfalse</code> 845, 1123
	<code>\@nodocument</code> 836
	<code>\@normalcr</code> 100
	<code>\@normalcr</code> 2537, 2538
	<code>\@noskipsecfalse</code> 838
	<code>\@onelevel@sanitize</code> 47
	<code>\@opcol</code> 33
	<code>\@opcol</code> 542
	<code>\@output@combined@page</code> 38

<code>\@output@combined@page</code>	...	660 , 1498, 1542	<code>\@startpbox</code>	..	2033, 2034, 2040, 2086, 2087, 2094
<code>\@outputbox</code>	23 , 40 , 63 , 69 , 71 , 72 , 92		<code>\@tabacol</code>	2031, 2083, 2100
<code>\@outputbox</code>	335, 337, 338, 544, 554, 561, 563, 564, 608, 673, 773, 775, 785, 1427, 1481, 1492, 1537, 1569, 1574, 1629, 1632, 1639, 1648, 1652, 1662, 1663, 1677, 1682, 1775, 2300	<code>\@tabacoll</code>	2098
<code>\@outputdblcol</code>	12 , 68	<code>\@tabacolr</code>	2099
<code>\@outputdblcol</code>	1545	<code>\@tabarray</code>	2018, 2020, 2071
<code>\@outputpage</code>	.	23 , 64 , 65 , 73 , 75	<code>\@tabclassiv</code>	2032, 2085
<code>\@outputpage</code>	319 , 699, 1433, 1440		<code>\@tabclassz</code>	2032, 2084
<code>\@outputpage@head</code>	23 , 92	<code>\@tabularcr</code>	84
<code>\@outputpage@head</code>	...	319 , 2295	<code>\@tabularcr</code>	2036
<code>\@outputpage@tail</code>	23 , 75	<code>\@tabularcr@LaTeX</code>	2089
<code>\@outputpage@tail</code>	...	319 , 897 , 1765 , 2193	<code>\@tempa</code>	20 , 21
<code>\@pagedp</code>	.	1064, 1069, 1188, 1192	<code>\@tempa</code>	..	180, 181, 244, 248, 988, 989, 1009, 1010, 1336, 1345, 1379, 1380, 1387, 2222, 2223, 2262–2264, 2268, 2269, 2272, 2273, 2397, 2398, 2556, 2557
<code>\@pageht</code>	.	1064, 1069, 1070, 1099, 1171, 1178	<code>\@tempb</code>	2222, 2223
<code>\@par</code>	2650	<code>\@tempboxa</code>	106
<code>\@preamble</code>	2054, 2113, 2114, 2124, 2128		<code>\@tempcnta</code>	41
<code>\@protection@box</code>	31	<code>\@tempcntb</code>	.	367, 368, 2341–2345, 2358, 2359
<code>\@protection@box</code>	414, 454, 491–493, 513	<code>\@tempdima</code>	106
<code>\@reinserts</code>	51	<code>\@tempskipa</code>	..	1360–1362, 1364, 1365
<code>\@reinserts</code>	1089	<code>\@testfp</code>	794, 820, 892
<code>\@replacestuff</code>	1345, 1346	<code>\@testopt</code>		2025, 2026, 2075, 2076
<code>\@reqcolroom</code>	639–643, 1099–1101, 1103, 1104, 1115, 1116		<code>\@testtrue</code>	...	628, 705, 796, 825
<code>\@resethfps</code>	1076, 1149	<code>\@textbottom</code>	566
<code>\@restorepar</code>	.	2419, 2432, 2449, 2467	<code>\@textfloatsheight</code>	.	437, 1098, 1118, 1119
<code>\@scolet</code>	613, 626	<code>\@textmin</code>	..	640, 730, 890, 1098, 1100, 1101
<code>\@sdblclet</code>	43	<code>\@texttop</code>	562
<code>\@sdblclet</code>	660	<code>\@themark</code>	16 , 17
<code>\@setfloattypecounts</code>	634, 715, 893, 1092		<code>\@themark</code>	194 , 203–206
<code>\@sharp</code>	2046, 2108	<code>\@toplist</code>	88 , 90
<code>\@specialoutput</code>	51	<code>\@toplist</code>	.	300, 442, 1501, 1647, 1655, 1656, 1737
<code>\@specialoutput</code>	1062	<code>\@topmark@saved</code>		1225, 1237, 1274
<code>\@startcolumn</code>	12	<code>\@topnewpage</code>	66
			<code>\@topnewpage</code>	1469
			<code>\@topnum</code>	882
			<code>\@toproom</code>	883
			<code>\@tryfcolumn</code>	.	38 , 41 , 43 , 64 , 67
			<code>\@tryfcolumn</code>	604, 669, 761

`\@trylist` 40
`\@trylist` 764, 770, 790, 804
`\@twocolumnfalse` ... 1153, 1154
`\@twocolumntrue` 1154, 1885
`\@twopowerfourteen` ... 338, 353
`\@twopowertwo` 366, 367
`\@undefined` 46
`\@undefined` 15, 542, 571, 895, 896,
898, 1062, 1089, 1221, 1416,
1468, 1469, 1545, 1551, 1555,
1885, 2030, 2294
`\@unexpandable@protect` ... 214
`\@vspace` 97
`\@vspace` . 2524, 2536, 2540, 2545
`\@vspace@ltx` 2426, 2540
`\@vspace@org` 2413, 2524
`\@vspacer` 97
`\@vspacer` 2525, 2536, 2541, 2545
`\@vspacer@ltx` 2457, 2541
`\@vspacer@org` 2439, 2525
`\@width` .. 1192, 1571, 1583, 1911,
1943, 2045, 2106
`\@wtryfc` 761
`\@xfloat` 47
`\@xfloat` 932
`\@xnewline` 2501, 2503, 2511, 2513
`\@xnext` 2213
`\@xtabularcr` 84
`\@xtabularcr` 2090
`\@xtabularcr@LaTeX` 2090
`\@xtryfc` 41
`\@xtryfc` 761
`\@xxxii` 792, 818
`\@yfloat` 911
`\@ztryfc` 41
`\@ztryfc` 761
`\@` 84, 97, 99, 103
`\@` 2408
`\{` 120
`\}` 120
`_` 40
`00readme` 3, 5

Numbers

`\1` 1771
`\2` 1772

`_` 29, 35, 54, 57, 58, 62, 82, 88, 93,
97, 101, 102

A

`\abovedisplayskip` 1332
`\addstuff` 11, 60, 61
`\addstuff` 1336
`\addtocounter` 993, 1014
`\adj@column` 1736, 1741
`\adj@page` 1704, 1746
`\advance` 75
`\aftergroup` 399, 401, 404,
457, 459, 462, 605, 629, 670,
706, 722, 1045, 1787, 1810,
1814, 2321, 2322
`\appdef` 18, 84
`\appdef` 321, 589, 893,
908, 1027, 1197, 1211, 1565,
1621, 1765, 2124, 2139, 2210,
2613, 2643
`\append@column` 69
`\append@column` 1573, 1579
`\arabic` 1003, 1024
argument
glue 11
penalty 11
`\arraystretch` . 2043, 2044, 2104,
2105
`\author` 38

B

`\badness` 285, 299
`\balance@` 1787
`\balance@2` 1770
`\balance@two` 76
`\balance@two` .. 1775, 1782, 1789
`\baselineskip` 69, 70
`\baselineskip` . 1615, 1694, 2059,
2136
`\begin` 48
`\bgroup` 91
`bk10.clo` 12
`\bot@envir` 18
`\bot@envir` 223, 289, 373, 418, 429
`\botfigrule` 1664

`\botmark` 16, 18, 28
`\botmark` 191
`\bottomfraction` 885
`\box` 19, 20, 23, 24, 26–32, 35, 37,
46, 55–59, 63, 64, 71, 76, 77,
88, 90, 91, 95, 97, 101, 105,
106
`\box@column` 69, 70
`\box@column` 1574, 1579
`\boxmaxdepth` . . 545, 1649, 1678,
2699
`\break` 1971, 2024, 2074
`\brokenpenalty` 93
`\brokenpenalty` . 277, 2366, 2367,
2377
`\brokenpenalty@ltx`
. . 2327, 2329, 2331, 2333,
2335, 2337, 2366, 2377

C

`\c@bottomnumber` 884
`\c@dbltopnumber` 888
`\c@float@type` 1013
`\c@linecount` 186
`\c@LT@chunks` 2050, 2115
`\c@LT@tables` 1906, 1936
`\c@page` 329, 871, 1444, 2187
`\c@topnumber` 882
`\c@totalnumber` 886
`\caption` 2027, 2079
`\cat@letter` 1771, 1772
`\catcode` 1771, 1772
`\changes` 125–155
`\check@aux` 2192
`\check@currbox@count` . . 48, 50
`\check@currbox@count` 941
`\check@deferlist@stuck` . . 623,
693, 2193
`\class@documenthook` 1565, 2613
`\class@info` 327, 344, 356,
368, 2154, 2156, 2287, 2306,
2536, 2625, 2648, 2653
`\class@warn` . . 1033, 1037, 2310,
2544, 2656
(*class customization commands*) place-
holder 9

`\classname` 49, 56, 109, 111, 113,
155
`\cleardoublepage` 834
`\clearpage` 27–29, 35, 43, 44, 55,
63, 65–67, 88, 89
`\clearpage` 834, 2198, 2208
`\clearpage@sw` 43, 44, 88
`\clearpage@sw` 426,
598, 616, 664, 860, 865, 876,
2197, 2231, 2248
`\clr@top@firstmark` 897
`\clubpenalty` 93, 101
`\clubpenalty` . . 278, 2368, 2369,
2378
`\clubpenalty@ltx`
. . 2328, 2329, 2332, 2333,
2336, 2337, 2368, 2378
`\cmd` 120
`\col@` 77
`\col@` 1550, 1775
`\col@number` 69
`\col@number` . . . 1550, 1551, 1885
`\col@sep` 2039, 2093
`\color@begingroup` 579
`\color@endgroup` 584
`\columnsep` 1689, 1691
`\columnseprule` 1583
`\columnwidth` 47
`\columnwidth` 939, 975, 981, 1159,
1164, 1592, 1688–1693
`\combine@foot@inserts` 33
`\combine@foot@inserts` 556,
1401, 1428, 1776
`\copy` 493, 1807, 1962, 1979, 2173,
2174
`\copyright` 35
`\count` 50, 91
`\count` . . 791, 817, 951, 952, 1043,
1049–1051, 1068, 1114
`\count@` 269, 270, 559,
568, 1050, 1051, 1314, 1315,
1339, 1341, 1348, 1350–1353,
1358, 1595, 1605, 2298, 2302
`\crcr` 1895, 1923
`\cs` . 125–132, 136–146, 148, 149,
152–154

<code>\csname</code>	18, 20, 21, 76, 99		1822, 1829, 1848–1852, 1956–
<code>\csname</code>	20, 258, 262,		1959, 1966, 1968, 1970, 1971,
	270, 339, 341, 342, 348, 354,		1976, 2177–2179, 2261, 2263,
	361–364, 366, 367, 373, 418,		2299, 2301, 2441, 2446, 2459,
	422, 429, 593, 628, 635, 878,		2463
	880, 915, 926, 992, 995–	<code>\dimen@i</code>	76
	997, 1013, 1016–1018, 1067,	<code>\dimen@i</code> .	1801, 1805, 1806, 1817,
	1160, 1379, 1392, 1393, 1481,		1818, 1822, 1829
	1488, 1537, 1554, 1560, 1581,	<code>\dimen@ii</code>	76, 77
	1906, 1936, 2068, 2315, 2322,	<code>\dimen@ii</code>	1596, 1604, 1806, 1816,
	2528, 2538, 2628		1818, 1821, 1829, 1960, 1965
<code>\curr@envir</code>	86	<code>\dispatch@output</code>	21
<code>\curr@envir</code>	2175	<code>\dispatch@output</code>	255
		<code>\displaywidowpenalty</code>	93
		<code>\displaywidowpenalty</code>	280, 2372,
			2373, 2380
		<code>\displaywidowpenalty@ltx</code>	2334–
			2337, 2372, 2380
		<code>\do@mark</code>	17
		<code>\do@mark</code>	206, 495, 505, 520, 1274
		<code>\do@check@aux</code>	2192
		<code>\do@columngrid</code>	62
		<code>\do@columngrid</code> .	1377, 1415, 1467
		<code>\do@endpage</code>	44
		<code>\do@endpage@pen</code>	45
		<code>\do@endpage@pen</code> . .	427, 862, 877,
			1530
		<code>\do@forcecolumn</code>	90
		<code>\do@forcecolumn</code>	2229
		<code>\do@forcecolumn@pen</code>	90
		<code>\do@forcecolumn@pen</code>	2229
		<code>\do@main@vlist</code>	11
		<code>\do@mark</code>	17
		<code>\do@mark</code>	196–198, 206
		<code>\do@newpage@pen</code>	43–45
		<code>\do@newpage@pen</code> . .	410, 619, 879,
			1437, 1529
		<code>\do@output@ccclv</code>	59
		<code>\do@output@ccclv</code> .	966, 968, 1300
		<code>\do@output@MVL</code>	59, 87, 91
		<code>\do@output@MVL</code>	850, 857,
			864, 1301, 1322, 1336, 1345,
			1383, 1453, 1465, 2192, 2264
		<code>\do@startcolumn</code> 37, 43, 73, 79, 90	
		<code>\do@startcolumn</code> . .	595, 596, 623,
			2250

D

<code>\dblfigrule</code>	1680
<code>\dblfloatpagefraction</code>	891
<code>\dblfloatsep</code>	751, 1748
<code>\dbltextfloatsep</code> 751, 1681, 1748	
<code>\dbltopfraction</code>	889
<code>\dead@cycle</code>	31, 32, 56
<code>\dead@cycle</code> .	380, 494, 610, 621,
	691, 1252
<code>\dead@cycle@repair</code>	32
<code>\dead@cycle@repair</code> . . .	378, 494
<code>\dead@cycle@repair@protected</code>	
	56
<code>\dead@cycle@repair@protected</code>	
	504, 1250
<code>\deadcycles</code>	1215
<code>\DeclareRobustCommand</code>	99
<code>\def</code>	76
<code>\def@line@handler</code>	93, 94
<code>\def@line@handler</code>	2313,
	2326–2337
<code>\def@next@handler</code>	93
<code>\def@next@handler</code>	2313,
	2349–2354
<code>\dimen</code>	76
<code>\dimen@</code>	75–77
<code>\dimen@</code>	338–342, 344, 353, 354, 356,
	377, 378, 386–388, 453, 455,
	456, 558, 561, 563, 565, 767,
	768, 1249, 1250, 1287, 1294,
	1540, 1593, 1594, 1597, 1598,
	1800, 1801, 1806, 1807, 1818,

`\do@startcolumn@open` . 28, 35, 44
`\do@startcolumn@open` . 425, 594, 861
`\do@startpage` . 36, 37, 43, 67, 68, 73, 75, 79
`\do@startpage` 661, 662, 693
`\do@startpage@open` . . . 21, 43, 68
`\do@startpage@open` 660
`\do@whatsit` 2391, 2633
`doc` 5
`doc/` 3
`\DocInput` 9
`docuemnt environment` 28
`\document` 53
document class
 `float` 48, 129
 `ftnright` 12–14
 `lineno` 20, 21
 `longtable` . 10, 12, 13, 19, 26, 80, 86
 `ltxdoc` 5, 9
 `ltxgrid` . 1, 2, 12–15, 48, 80, 86, 88
 `ltxgrid.dtx` 3
 `ltxgrid.pdf` 3
 `ltxgrid.sty` 3
 `ltxkrnext` 15
 `ltxutil` 10, 84
 `multicol` . . 10, 12, 13, 19, 80
 `newpackage` 21
document environment . 5, 55, 69
`\document@inithook` 1027, 1211, 1621
`\dp` 77
`\dp` 339, 348, 354, 386, 546, 563, 1064, 1180, 1187, 1294, 1732, 1760, 1810, 1814, 1958, 1968, 1969, 2044, 2105, 2700

E

`\edef` 180, 181, 992, 995, 1013, 1016, 1336, 1345, 2113, 2262, 2263
`\egroup` 91
`\end@float` 941
`\end@column@` 63
`\end@column@milt` 65, 67
`\end@column@milt` 1467
`\end@column@one` 65
`\end@column@one` 1415
`\end@dblfloat` 47
`\end@dblfloat` 941
`\end@float` 47
`\end@float` 941
`\end@line@ltx` 2505, 2537
`\end@line@org` 2495, 2528
`\endcsname` 99
`\endcsname` 20, 258, 262, 270, 339, 341, 342, 348, 354, 361–364, 366, 367, 373, 418, 422, 429, 593, 628, 635, 878, 880, 915, 926, 992, 995–997, 1013, 1016–1018, 1067, 1160, 1379, 1392, 1393, 1481, 1488, 1537, 1554, 1560, 1581, 1906, 1936, 2068, 2315, 2322, 2528, 2538, 2628
`\endgraf` . 1915, 1919, 1946, 1950, 1954, 1955, 1984, 1985, 1993, 2001
`\endline@ltx` 2411, 2642
`\endlinenomath` 2564, 2617, 2627, 2674
`\endlinenomath@LN` . . 2590, 2617
`\endlinenomath@ltx` 2627, 2628, 2681
`\endlongtable` 80
`\endlongtable` . 1894, 2143, 2159, 2167
`\endlongtable@longtable` . 1894, 2143
`\endlongtable@new` . . 1922, 2159
`\enlarge@colroom` . . 1699, 1712, 1713, 2275, 2293
`\enlargethispage` 34, 91
`\enlargethispage` 2252
`\enqueue@vadjust@ltx` 106
`\enqueue@vadjust@ltx` 2549, 2715
`\enqueue@whatsit@ltx` 2391, 2632
environment
 `docuemnt` 28
 `document` 5, 55, 69

figure	54	00readme	3, 5
longtable	13, 79, 84, 86	bk10.clo	12
longtable*	86	doc	5
table	54, 86	doc/	3
table*	86	latex/	3
tabular	84	ltxgrid	2, 3, 15, 106
turnpage	47	ltxgrid.dtx	3
environments:		ltxgrid.sty	3
turnpage	1194	makeindex	3
\ex@vadjust@line	2631, 2714	revtex/	3
\ex@vadjust@ltx	97, 100	source/	3
\ex@vadjust@ltx	2431, 2466, 2489, 2509, 2521, 2549, 2631	src/ltxgrid.pdf	1
\execute@message	31, 55, 58, 59	tex/	3
\execute@message	1279, 1300, 1303, 1306, 1325	texmf-local/	3
\execute@message@insert	55, 56, 58	TEXMF/	3
\execute@message@insert	1282, 1390	texmf/tex/macros/latex/revtex/.	1
\execute@message@pen	21, 57		
\execute@message@pen	262, 1277, 1296	\file	69, 71, 77, 78, 87, 93, 94, 97, 99, 101, 109, 111, 113, 115, 117
\expandafter	99	\fill	2011, 2013, 2015, 2061–2063, 2067
\extrarowheight	2030, 2038, 2082, 2092	\firstmark	16, 18, 27, 46
		\firstmark	190
		\firsttime@sw	1705, 1737, 1738, 1743, 1748
		float document class	48, 129
		\float@avail@sw	603, 605, 629, 668, 670, 677, 706, 722
		\float@column@mlt	67
		\float@column@mlt	1497
		\float@column@one	64
		\float@column@one	1415
		\float@do	987, 988, 1008, 1009
		\float@end	1029, 1034
		\float@end@float	973, 1029
		\float@end@ltx	979, 1034
		\float@exts	988, 1009
		\float@makebox	975, 981
		\float@newx	1001, 1022
		\floatbox	91
		\floatname	991, 1012
		\floatpagefraction	24
		\floatpagefraction	887
		\floatpenalty	91
		\floatplacement	990, 1011
F			
\f@ur	952, 2330, 2371		
\false@sw	34, 63, 96		
\false@sw	401, 404, 446, 459, 462, 466, 474, 713, 719, 734, 737, 741, 744, 1030, 1031, 1045, 1318, 1426, 1444–1446, 1448, 1479, 1490, 1506, 1825, 1834, 2148– 2152, 2214, 2279, 2283, 2530– 2534, 2620–2623, 2673		
\fcolmade@sw	40		
\fcolmade@sw	607, 672, 762, 772, 806, 811		
figure environment	54		
file			
\$TEXMF/	3		
.dtx	5		
.ins	5		

[\floatsep](#) 1743
[\foo](#) 91
[\footins](#) [24](#), [27](#), [33](#), [34](#), [56](#), [58](#), [71](#),
[75](#), [76](#)
[\footins](#) . 310, 444, 532, 534, 536,
554, 1261–1263, 1283, 1402,
1405, 1410, 1638, 1639, 1753,
1756, 1774, 1920
[\footins@saved](#) 56
[\footins@saved](#) . 309, 1263, 1267,
1283, 1754
[\footnoterule](#) 581
[\footnotesize](#) [12](#), [13](#)
[\footsofar](#) ... [34](#), [42](#), [63](#), [66](#), [76](#)
[\footsofar](#) 308, 531,
533, [1399](#), 1636–1638, 1755,
1774, 1777, 1779, 1782, 1783,
1864, 1865, 1875
[\force@deferlist@empty](#) 89
[\force@deferlist@empty](#) .. 2225,
[2229](#)
[\force@deferlist@stuck](#) . 88, 90
[\force@deferlist@stuck](#) .. 2200,
[2229](#)
[\force@deferlist@sw](#) 90
[\force@deferlist@sw](#) [2229](#)
[\forcefloats@sw](#) [660](#), 2232, 2249
[\fps@](#) [911](#)
[\fpsd@](#) [911](#)
ftnright document class . [12–14](#)

G

[\g@pop@ltx](#) 2397, 2401, 2556
[\g@vadjust@line](#) [106](#)
[\g@vadjust@line](#) [2549](#), 2706, 2707
[\g@vadjust@queue](#) [106](#)
[\g@vadjust@queue](#) .. 2551, 2556,
2562
[\g@whatsit@queue](#) .. 2392, 2397,
2561
[\gappdef](#) 2392, 2551, 2557
[\gdef](#) [16](#)
[\get@mark@@ne](#) [17](#)
[\get@mark@@ne](#) [199](#), 233
[\get@mark@f@ur](#) [17](#)
[\get@mark@f@ur](#) [199](#)

[\get@mark@thr@@](#) [17](#)
[\get@mark@thr@@](#) [199](#), 225
[\get@mark@tw@](#) [17](#)
[\get@mark@tw@](#) [199](#), 239
[\GetFileInfo](#) 26
[\glossary](#) 217
glue, argument [11](#)
[\grid@column](#) [69](#)
[\grid@column](#) .. 1489, 1541, [1568](#)

H

[\handle@par@LN](#) 2634, 2684
[\handle@par@ltx](#) [2348](#), 2634
[\handle@vadjust@ltx](#) 2350, [2549](#)
[\handle@whatsit@ltx](#) 2351, [2391](#)
[\hangindent](#) [77](#)
[\hb@xt@](#) ... 178, 1159, 1570, 1592,
1783, 2702
[\hbox](#) [91](#), [105](#)
[\hline](#) ... 2019, 2021, 2027, 2070,
2071, 2077
[\hold@insertions](#) [25](#)
[\hold@insertions](#) [1371](#)
[\holdinginserts](#) [19](#), [24–27](#), [31](#), [32](#),
[51](#), [56](#), [58](#), [59](#), [61–63](#), [101](#)
[\holdinginserts](#)
... 274, 1371, 1372, 1375,
2567, 2581, 2592, 2599, 2602,
2608
[\holdininserts](#) [24](#), [33](#)
[\hsize](#) [62](#)
[\ht](#) 328, 338, 341, 342,
353, 366, 367, 377, 386, 397,
414, 449, 454, 455, 471, 513,
639, 725, 731, 749, 796, 803,
823, 1064, 1103, 1118, 1172,
1182, 1186, 1249, 1294, 1593,
1599, 1732, 1742, 1747, 1760,
1800, 1816, 1830, 1831, 1848,
1850, 1957, 1959, 1963, 1966,
1967, 1974–1976, 2028, 2080,
2177, 2178, 2703

I

[\if](#) 2010, 2012, 2014
[\if@filesw](#) 1903, 1932

<code>\if@inlabel</code>	841	<code>\interlinepenalty</code>	
<code>\if@insert</code>	655, 1137, 1147	...	276, 1084, 1126, 1130,
<code>\if@mparswitch</code>	1443		2364, 2365, 2376, 2566, 2568,
<code>\if@nobreak</code> 220, 845, 1081, 1121			2580, 2582, 2600, 2609
<code>\if@noskipsec</code>	835	<code>\interlinepenalty@ltx</code> ...	2341,
<code>\if@reversemargin</code> ..	1446, 1448		2364, 2376
<code>\if@test</code>	40	<code>\interlinepenalty</code>	101
<code>\if@test</code> .	629, 647, 706, 718, 797,	<code>\intextsep</code> 1115, 1119, 1128, 1131	
	827, 1108, 1111	<code>\item</code> ..	91, 96, 104, 108, 110, 112,
<code>\if@twocolumn</code>	13 , 53		116, 118
<code>\if@twocolumn</code>	1153 , 1886	K	
<code>\if@twoside</code>	870	<code>\kill</code>	2027, 2078
<code>\iffalse</code>	42	L	
<code>\ifinner</code>	2598	<code>\label</code>	215
<code>\ifLineNumbers</code> 2565, 2579, 2591,	2647, 2675, 2715	<code>\lastbox</code>	77 , 91
<code>\ifodd</code>	1114	<code>\lastbox</code> ...	414, 440, 1858, 1862
<code>\ifvoid</code> ..	1920, 1957, 1958, 1973,	<code>\lastpenalty</code>	91
	1979	<code>\lastpenalty</code> .	1314, 1339, 1348,
<code>\ignorespaces</code> ..	2576, 2588, 2679		2340, 2357, 2686
<code>\immediate</code>	1904, 1935	<code>\lastskip</code> .	453, 1338, 1347, 1856,
<code>\index</code>	216		1857
<code>\inputlineno</code>	273	<code>\LaTeX</code> .	29, 57, 58, 67, 79, 82, 97,
<code>\insert</code> .	25 , 27 , 31 , 32 , 56 , 57 , 61		101, 102
<code>\insert</code>	1920	<code>latex/</code>	3
<code>\insertpenalties</code>	290	<code>\LaTeXe</code>	62, 88, 166
<code>\int@interdisplaylinepenalty</code>		<code>\leaders</code>	182
.....	2348 , 2388	<code>\leftmark</code>	18
<code>\int@parpenalty</code>	94	<code>\leftmark</code>	229
<code>\int@parpenalty</code> 2319, 2325 , 2348,		<code>\let</code>	20 , 21
2365		<code>\let@mark</code>	57
<code>\int@postdisplaypenalty</code> .	2348 ,	<code>\let@mark</code>	209, 213
2389		<code>\linelabel</code>	102 , 106
<code>\int@postparpenalty</code> 2348 , 2667		<code>\lineloop</code>	16
<code>\int@predisplaypenalty</code> ..	2348 ,	<code>\lineloop</code>	175
2387		<code>lineno</code> document class ...	20 , 21
<code>\int@vadjustpenalty</code>	106	<code>\linenomathNonumbers</code> ...	2564 ,
<code>\int@vadjustpenalty</code> 2348 , 2552			2616, 2630, 2672
<code>\int@whatsitpenalty</code> 2348 , 2393		<code>\linenomathNonumbers@LN</code> .	2578,
<code>\interdisplaylinepenalty</code> ..	93		2616
<code>\interdisplaylinepenalty</code> .	282,	<code>\linenomathNonumbers@ltx</code>	2630,
2385, 2388, 2573			2673
<code>\interdisplaylinepenalty@ltx</code>		<code>\linenomathWithnumbers</code> ..	2564 ,
.....	2353, 2385		2615, 2629, 2672
<code>\interlinepenalty</code> .	93 , 101 , 104		

<code>\linenomathWithnumbers@LN</code>	2564, 2615	<code>\LT@caption</code> 2027, 2079
<code>\linenomathWithnumbers@ltx</code>	..	<code>\LT@echunk</code>	1899, 1928, 1992, 2000
.....	2629, 2672	<code>\LT@end</code> 1991
<code>\linenopenalty</code>	<code>\LT@end@hd@ft</code> 80
..	2568, 2570, 2572, 2573, 2582, 2584, 2600, 2609	<code>\LT@end@hd@ft</code> 2145, 2161
<code>\linenopenaltypar</code>	. 2566, 2580, 2605	<code>\LT@end@hd@ft@longtable</code>	. 1991, 2145
<code>\linenoprevgraf</code>	2601, 2604, 2663, 2666	<code>\LT@end@hd@ft@new</code>	.. 1999, 2161
<code>\linenumberpar</code>	... 101 , 103 , 104	<code>\LT@end@pen</code> 1915
<code>\linenumberpar</code>	2564 , 2618, 2626, 2649–2651, 2661	<code>\LT@endpbox</code> 2029, 2081
<code>\linenumberpar@LN</code>	. 2596, 2618, 2650, 2651	<code>\LT@entry</code>	1897, 1904, 1925, 1934
<code>\linenumberpar@ltx</code>	. 2626, 2661	<code>\LT@entry@chop</code> 1897, 1925
<code>\linenumbers</code> 103	<code>\LT@entry@write</code> 1904, 1934
<code>\linenumbers</code> 2648	<code>\LT@err</code> 1886, 1994, 2002
<code>\lineskip</code> 2059, 2135	<code>\LT@final@warn</code> 1913, 1944
<code>\linewidth</code> 933, 1693	<code>\LT@firsthead</code>	. 1957, 1958, 1979, 1988
<code>\long</code> 99	<code>\LT@foot</code>	. 1948, 1959, 1973–1976, 2173
<code>\longtable</code> 80	<code>\LT@get@widths</code>	1902, 1931, 1997, 2005
<code>\longtable</code>	1883 , 2142, 2158, 2165	<code>\LT@head</code>	. 1957, 1958, 1979, 2174
longtable document class	10 , 12 , 13 , 19 , 26 , 80 , 86	<code>\LT@hline</code> 2027, 2077
longtable environment	13 , 79 , 84 , 86	<code>\LT@kill</code> 2027, 2078
longtable* environment 86	<code>\LT@lastfoot</code> 1948
<code>\longtable@longtable</code>	1883, 2142	<code>\LT@LL@FM@cr</code>	. 2036, 2040, 2089, 2095
<code>\longtable@new</code> 1890, 2158	<code>\LT@LR@c</code> 2063
<code>\loop@line</code> 176, 186	<code>\LT@LR@l</code> 2061
<code>\loopwhile</code> 77	<code>\LT@LR@r</code> 2062
<code>\loopwhile</code> 186, 1313, 1804	<code>\LT@make@row</code> 2057, 2131
<code>\lose@breaks</code> 1233, 1312	<code>\LT@mcol</code> 2017, 2069
<code>\LT@chl</code>	.. 2019, 2021, 2070, 2071	<code>\LT@no@pgbk</code>	.. 2025, 2026, 2075, 2076
<code>\LT@save@row</code> 1909, 1941	<code>\LT@nofcols</code> 2056, 2130
<code>\LT@tabarray</code> 2018, 2022	<code>\LT@output</code> 1981
<code>\LT@adj</code> 2172 , 2181	<code>\LT@post</code> 1951, 2172
<code>\LT@array</code>	1888, 1892, 2008 , 2146, 2162	<code>\LT@pre</code> 1987, 2172
<code>\LT@array@longtable</code>	2008, 2146	<code>\LT@rows</code> 2051, 2116
<code>\LT@array@new</code> 2064, 2162	<code>\LT@save@row</code>	. 1898, 1907, 1909, 1926, 1937, 1941
<code>\LT@bchunk</code>	1997, 2006, 2049, 2056, 2060, 2128, 2130, 2137	<code>\LT@setprevdepth</code>	... 2052, 2118
<code>\LT@bot</code> 2172 , 2182	<code>\LT@start</code> 80
		<code>\LT@start</code>	1900, 1929, 1953 , 1993, 2001, 2144, 2160
		<code>\LT@start@longtable</code>	1953, 2144

<code>\LT@start@new</code>	1983, 2160	<code>\mark@envir</code> . . .	222 , 1989, 2175
<code>\LT@startpbox</code> .	2033, 2040, 2086, 2094	<code>\mark@netw@</code>	203 , 229
<code>\LT@tabularcr</code>	2023, 2072	<code>\markboth</code>	229
<code>\LT@top</code>	1988, 2172 , 2183	<code>\markf@ur</code>	17
<code>\LT@warn</code>	1911, 1942	<code>\markright</code>	229
<code>\LTleft</code> . .	2011, 2013, 2015, 2053, 2061–2063, 2067, 2119	<code>\markthr@@</code>	203 , 222, 1986
<code>\LTpost</code>	1919, 2175	<code>\marktw@</code>	203 , 230
<code>\LTpre</code>	1955, 2172	<code>\marry@baselines</code>	69 , 70
<code>\LTRight</code> .	2011, 2013, 2015, 2054, 2061–2063, 2067, 2125	<code>\marry@baselines</code>	535, 1230, 1263, 1409, 1579 , 1631, 1794
<code>ltxdoc</code> document class	5 , 9	<code>\marry@skip</code>	70
<code>ltxgrid</code>	2 , 3 , 15 , 106	<code>\marry@skip</code> . . .	1610, 1612, 1617
<code>ltxgrid</code> document class . . .	1 , 2 , 12–15 , 48 , 80 , 86 , 88	<code>\mathchardef</code>	93
<code>ltxgrid.dtx</code>	3	<code>\mathchardef</code>	592, 594, 660, 877, 879, 1218, 1243, 1268, 1277, 2245, 2314, 2325
<code>ltxgrid.dtx</code> document class . . .	3	<code>\maxdepth</code> .	391, 569, 1649, 1678, 1977
<code>ltxgrid.pdf</code> document class . . .	3	<code>\maxdimen</code> . . .	317, 393, 882–886, 888, 889, 1596, 1803, 1961, 2299, 2699
<code>ltxgrid.sty</code>	3	<code><meddle with the MVL></code> placeholder	11
<code>ltxgrid.sty</code> document class . . .	3	<code>\MessageBreak</code>	1911, 1943
<code>\ltxgrid@info</code> . . .	398, 472, 1388, 1717, 1841, 2305	<code>\minipagefootnote@here</code> . . .	941
<code>\ltxgrid@info@sw</code> . . .	2306, 2308	<code>\minipagefootnote@init</code>	935, 941
<code>\ltxgrid@warn</code>	684, 686, 1053, 1199, 1208, 1381, 2198, 2278, 2282, 2305	<code>\move@insertions</code>	25
<code>\ltxgrid@warn@sw</code> . . .	2310, 2312	<code>\move@insertions</code>	1371
<code>ltxkrnext</code> document class . . .	15	<code>\moveleft</code>	77
<code>ltxutil</code> document class . .	10 , 84	<code>\moveright</code>	77
M			
<code>\Make@LineNo</code>	105	<code>multicol</code> document class .	10 , 12 , 13 , 19 , 80
<code>\Make@LineNo</code>	2697	<code>\multicols</code>	1885
<code>\Make@LineNo@ltx</code> . .	2635, 2636, 2685, 2697	<code>\multicolumn</code>	2017, 2069
<code>makeindex</code>	3	<code>\multiply</code>	792, 818, 1050
<code>\MakeLineNo</code>	101	<code>\myadjust</code>	11
<code>\makeLineNumber</code>	2702	N	
<code>\maketitle</code>	47	<code>\natural@output</code>	370 , 2637
<code>\marginpar</code>	48 , 50	<code>\newbox</code>	69
<code>\marginparpush</code>	1181	<code>\newbox</code>	361, 363, 491, 1267, 1399, 1400
<code>\marginparsep</code>	1161, 1164	<code>\newcount</code>	1552
<code>\marginparwidth</code>	1161	<code>\newfloat</code>	1028, 1030, 1035
<code>\mark</code>	17 , 26 , 34 , 57 , 59	<code>\newfloat@float</code>	985, 1030
<code>\mark@envir</code>	18 , 86	<code>\newfloat@ltx</code>	1006, 1035

<code>\newif</code>	40, 53	<code>\output@column@mlt</code>	68
<code>\newinsert</code>	58	<code>\output@column@mlt</code>	1467
<code>\newlabel</code>	2711	<code>\output@column@one</code>	65, 68
<code>\newpackage</code> document class ...	21	<code>\output@column@one</code> ..	424, 1415
<code>\newpage</code>	19, 35, 43–45, 90	<code>\output@debug</code>	265, 272
<code>\newpage</code>	834	<code>\output@holding</code>	26, 56
<code>\newpage@prep</code>	834	<code>\output@holding</code>	370, 372
<code>\newtoks</code>	20	<code>\output@init</code>	2181, 2184
<code>\newtoks</code>	252, 1237	<code>\output@init@</code>	33
<code>\noalign</code>	11	<code>\output@init@document</code>	527
<code>\noalign</code> .	1896, 1924, 2024–2026, 2074–2076	<code>\output@init@longtable</code> ..	2181
<code>\nobreak</code>	11, 82	<code>\output@init@theindex</code> ...	2184
<code>\nobreak@mark</code>	206, 213	<code>\output@latex</code>	21
<code>\noexpand</code>	84	<code>\output@latex</code> 244, 260, 371, 2637	
<code>\noexpand</code>	988, 996, 997, 1009, 1017, 1018, 1336, 1345, 1458, 1905, 1936, 2053, 2263	<code>\output@moving</code>	26, 27, 29
<code>\nointerlineskip</code> .	493, 500, 512, 582, 1189, 1191, 1295	<code>\output@moving</code>	370, 408
<code>\nopagebreak</code>	97, 98, 103	<code>\output@post</code>	2181, 2184
<code>\nopagebreak</code> .	2026, 2076, 2408, 2640	<code>\output@post@</code>	28, 33
<code>\nopagebreak@ltx</code> ...	2410, 2640	<code>\output@post@</code>	429–431
<code>\normalcolor</code>	580	<code>\output@post@document</code> .	431, 527
<code>\nul@mark</code>	17	<code>\output@post@longtable</code> ..	2183
<code>\nul@mark</code> ...	195, 227, 235, 241	<code>\output@post@theindex</code>	87
<code>\null</code>	67	<code>\output@post@theindex</code> ...	2186
O			
<code>\onecolumn</code>	10, 13, 64	<code>\output@pre@</code>	28
<code>\onecolumn</code>	1416	<code>\output@prep</code>	2181, 2184
<code>\onecolumngrid</code>	10, 66	<code>\output@prep@</code>	33
<code>\onecolumngrid</code>	1415	<code>\output@prep@</code>	418–420
<code>\onecolumngrid@pop</code> .	1464, 2168	<code>\output@prep@document</code> .	420, 527
<code>\onecolumngrid@push</code>	1452, 2164	<code>\output@prep@longtable</code> ..	2182
<code>\open@column@</code>	63, 73	<code>\output@prep@theindex</code> ...	2185
<code>\open@column@mlt</code>	1467	<code>\output@procedure</code> 258–260, 262, 263, 266, 284	
<code>\open@column@one</code>	64	<code>\outputdebug@sw</code> ..	265, 314, 315, 376, 387, 395, 419, 423, 430, 473, 1248, 1262, 1597, 1708, 1714, 1726, 1730, 1733, 1744, 1749, 1762, 1790, 1806, 1829, 1850, 1853, 1875
<code>\open@column@one</code> ...	1415, 1878	<code>\outputpenalty</code>	19–21
<code>\output</code>	13, 19–21, 26, 54	<code>\outputpenalty</code> ...	258, 269, 275, 410, 502, 516, 525, 609, 617, 619, 690, 1080, 1132, 1133
<code>\output</code>	243, 244, 370, 1981	P	
<code>\output@-1073741824</code>	1214	<code>\p@</code>	76
<code>\output@column@</code> ..	26, 28, 63, 73	<code>\package@font</code>	15
<code>\output@column@</code>	422–424		

<code>\package@name</code>	164, 165	<code>\postdisplaypenalty</code>	283, 2386, 2389, 2572
<code>\PackageInfo</code>	165	<code>\postdisplaypenalty@ltx</code>	2354, 2386
<code>\pagebreak</code>	96–98, 103	<code>\predisplaypenalty</code>	93
<code>\pagebreak</code>	2025, 2075, 2408, 2639	<code>\predisplaypenalty</code>	281, 1331, 2384, 2387, 2570, 2584
<code>\pagebreak@ltx</code>	2409, 2639	<code>\predisplaypenalty@ltx</code>	2352, 2384
<code>\pagebreak@pen</code>	44	<code>\prep@cclv</code>	56, 59
<code>\pagebreak@pen</code>	592, 609, 852, 859, 2239	<code>\prep@cclv</code>	1238, 1291
<code>\pagegoal</code>	19, 24, 26, 35	<code>\prepdef</code>	319, 1877, 2071, 2114, 2296
<code>\pagegoal</code>	297, 388, 394, 642, 767, 1070, 1918, 1970, 1976	<code>\prevdepth</code>	58
<code>\pagegrid@col</code>	69	<code>\prevdepth</code>	1287, 2441, 2446, 2459, 2463
<code>\pagegrid@col</code>	287, 329, 339, 341, 342, 348, 920, 947, 1420, 1423, 1454, 1458, 1472, 1474, 1480, 1488, 1536, 1550, 1580, 1687, 1690, 1695	<code>\prevgraf</code>	104
<code>\pagegrid@cur</code>	288, 329, 700, 1421, 1473, 1480–1482, 1487, 1505, 1536–1538, 1548, 1550, 1572, 1580, 1581, 1585, 2188, 2274	<code>\prevgraf</code>	2601, 2604, 2663, 2666
<code>\pagegrid@init</code>	69	<code>\primitive@output</code>	21
<code>\pagegrid@init</code>	1550	<code>\primitive@output</code>	243, 249, 255
<code>\pagesofar</code>	10, 23–25, 28, 42, 63, 64, 66, 71, 73, 74	<code>\protect@penalty</code>	31, 45, 59
<code>\pagesofar</code>	307, 352, 353, 673, 1399, 1418, 1427, 1492, 1628, 1630, 1731, 1732	<code>\protect@penalty</code>	425, 427, 490, 701, 861, 862, 1264, 1289, 1437, 1529, 1530, 2240
<code>\pagetotal</code>	19, 32	<code>\protected@write</code>	2710
<code>\pagetotal</code>	298, 1956	<code>\protection@box</code>	490, 493, 515, 1290
<code>\par</code>	100, 104	<code>\providecommand</code>	1059–1061
<code>\parshape</code>	77	<code>\ProvidesFile</code>	4, 6
<code>\parskip</code>	1132	R	
<code>\penalty</code>	55–57, 60, 91, 94, 95, 101	<code>\raggedcolumn@skip</code>	1602, 1624, 1851, 1852
<code>penalty, argument</code>	11	<code>\raggedcolumn@sw</code>	70
<code>\pfloat@avail@sw</code>	600, 605, 666, 670, 682	<code>\raggedcolumn@sw</code>	1622, 1625
placeholder		<code>\raise</code>	1591
(<i>class customization commands</i>)		<code>\RecordChanges</code>	24
..... 9		<code>\recover@footins</code>	79
(<i>meddle with the MVL</i>)	11	<code>\recover@footins</code>	1427, 1781, 1793, 1797, 1855
(<i>your code here</i>)	19	<code>\relax</code>	25, 28, 76
(<i>your document here</i>)	9	<code>\remove@lastbox</code>	413, 440, 452, 511, 597, 663, 1234, 1246, 1247, 1327, 1436, 1525, 1527, 1532, 1810, 1814
<code>\postdisplaypenalty</code>	93		

<code>\removephantombox</code>	60	<code>\savecolumn@holding</code>	1244, 1245
<code>\removephantombox</code>	<u>1323</u>	<code>\savecolumn@moving</code> .	1244, 1255
<code>\removestuff</code>	60	<code>\saved@botmark</code>	46
<code>\removestuff</code>	<u>1322</u>	<code>\saved@botmark</code> . .	234, 296, 900, 906
<code>\replacestuff</code>	11, 61	<code>\saved@firstmark</code>	46
<code>\replacestuff</code>	<u>1345</u>	<code>\saved@firstmark</code> .	240, 294, 899, 905
<code>\RequirePackage</code> .	13, 14, 17, 169	<code>\saved@topmark</code>	18, 46
<code>\reserved@a</code>	790	<code>\saved@topmark</code> . .	292, 898, 904
<code>\reserved@b</code>	614, 679	<code>\say</code>	284, 286, 289, 291–296, 300–303, 419, 423, 430
<code>\reserved@e</code> . .	2496, 2499, 2506, 2509, 2517, 2521	<code>\saythe</code>	273–283, 285, 287, 288, 290, 297–299, 387, 1597, 1708, 1714, 1726, 1730, 1733, 1744, 1749, 1762, 1806, 1829, 1850
<code>\reserved@f</code> . .	2497, 2500, 2507, 2510	<code>\sbox</code>	2702
<code>\reset@queues@ltx</code>	100	<code>\sc</code>	79
<code>\reset@queues@ltx</code> .	2349, 2352, <u>2549</u>	<code>\scrollmode</code>	317
<code>\restore@linepenalties</code>	95	<code>\section</code>	75
<code>\restore@linepenalties</code> . .	<u>2363</u> , 2670	<code>\set@adj@box</code> . .	1753–1755, 1758
<code>\restorecolumngrid</code>	1455, 1457, 1465	<code>\set@adj@colht</code> .	558, 1540, <u>1697</u>
<code>\restylefloat</code>	994, 1015	<code>\set@adj@footins</code>	75
<code>\revtex</code>	159	<code>\set@adj@footins</code>	528, 1735, 1751
<code>revtex/</code>	3	<code>\set@adj@textheight</code>	1698, 1702
<code>\rightmark</code>	18	<code>\set@colht</code>	66, 73, 74
<code>\rightmark</code>	<u>229</u>	<code>\set@colht</code>	435, 599, 665, 694, 1422, 1429, 1475, 1495, <u>1697</u> , 1766, 1847, 1879
<code>\robust@</code>	18	<code>\set@colroom</code>	66, 73, 74
<code>\romannumeral</code>	1906, 1936	<code>\set@colroom</code>	<u>1697</u>
<code>\rotatebox</code>	54	<code>\set@column@hsize</code>	73
<code>\rotatebox</code>	1202, 1212	<code>\set@column@hsize</code> .	1423, 1474, <u>1686</u>
<code>\rotatebox@dumy</code> . . .	1207, 1212	<code>\set@displaypenalties</code> . . .	<u>2363</u> , 2677
S			
<code>\save@column</code>	46, 55	<code>\set@footnotefont</code>	2643
<code>\save@column</code> . .	1219, <u>1222</u> , 1257	<code>\set@footnotewidth</code>	934
<code>\save@column@insert@pen</code>	55, 56, 58	<code>\set@linepenalties</code>	95
<code>\save@column@insert@pen</code> .	<u>1243</u> , 1283	<code>\set@linepenalties</code>	<u>2363</u> , 2664, 2676
<code>\save@column@moving</code>	56	<code>\set@mark@netw@</code>	17
<code>\save@column@pen</code>	55, 56	<code>\set@mark@netw@</code>	<u>196</u> , 203
<code>\save@column@pen</code> . . .	<u>1218</u> , 1280	<code>\set@markthr@@</code>	17
<code>\save@message</code>	1269, 1270	<code>\set@markthr@@</code>	<u>196</u> , 205
<code>\save@message@pen</code>	57		
<code>\save@message@pen</code> . .	<u>1268</u> , 1293		
<code>\savecolumn@holding</code>	56		

<code>\set@marktw@</code>	17	<code>\stepLineNumber</code>	2705
<code>\set@marktw@</code>	196, 204	<code>\StopEventually</code>	6
<code>\set@marry@skip</code>	1613, 1695	<code>\string</code>	99
<code>\set@output@procedure</code>	268	<code>\string</code> ..	1308, 1717, 2198, 2208, 2528, 2536, 2538, 2545, 2648, 2711
<code>\set@top@firstmark</code>	46	<code>\strutbox</code>	2028, 2044, 2080, 2105
<code>\set@top@firstmark</code> ..	409, 897, 1224	<code>\subsection</code>	85, 124
<code>\set@vsize</code>	73	<code>\switch@longtable</code>	80
<code>\set@vsize</code> 624, 1087, 1394, 1697, 2289		<code>\switch@longtable</code>	2140
<code>\setbox</code>	77, 91	T	
<code>\shipout</code> 19, 20, 23, 25, 28, 42, 43, 64, 66, 73, 88		<code>\tabcolsep</code>	2039, 2093
<code>\show@box@size</code>	23	table environment	54, 86
<code>\show@box@size</code> ..	323, 577, 1637	table* environment	86
<code>\show@box@size@sw</code>	23	<code>\table@hook</code>	2066, 2139
<code>\show@box@size@sw</code>	323	<code>\tableofcontents</code>	73
<code>\show@pagesofar@size</code> 323, 1493		<code>\tabskip</code> .	2053, 2054, 2119, 2121, 2125
<code>\show@text@box@size</code>	23	tabular environment	84
<code>\show@text@box@size</code> ..	323, 551	<code>\tabularnewline</code>	2023, 2073
<code>\showbox</code> 305–310, 318, 395, 1790, 1853		<code>\tally@box@size@sw</code> ...	336, 359
<code>\showboxbreadth</code>	317	<code>\tally@float</code>	1055, 1061
<code>\showboxdepth</code>	317	<code>\temp@sw</code>	1752, 1756, 1761
<code>\showlists</code>	311	<code>\test@colfloat</code>	604, 627
<code>\shut@column@</code>	63	<code>\test@dblfloat</code>	660
<code>\shut@column@mlt</code>	66	<code>\TeX</code>	54, 62, 93, 159
<code>\shut@column@mlt</code>	1467	tex/	3
<code>\shut@column@one</code>	64	texmf-local/	3
<code>\shut@column@one</code>	1415	TEXMF/	3
<code>\sixt@on</code>	728, 1050	texmf/tex/macros/latex/revtex/.	1
<code>\skip</code>	75	<code>\textfloatsep</code> ..	1651, 1664, 1743
<code>\skip@</code> 1338, 1342, 1347, 1362, 1364, 1365, 1369, 1615–1617, 1694		<code>\textheight</code> ..	32, 43, 54, 70, 73
source/	3	<code>\textheight</code>	681, 891, 1195, 1196, 1599, 1703, 2300
<code>\special</code>	95	<code>\textheight@sw</code>	2295, 2297
<code>\special</code>	181	<code>\texttt</code> 40, 106, 115, 117, 120, 121	
<code>\splitbotmark</code>	193	<code>\textwidth</code> ...	38, 39, 47, 54, 68
<code>\splitfirstmark</code>	192	<code>\textwidth</code> .	708, 940, 953, 1201, 1570, 1688, 1783
<code>\splitmaxdepth</code>	391	<code>\thanks</code>	30, 34, 40
<code>\splittopskip</code>	390	<code>\the</code>	20, 21, 54
src/ltxgrid.pdf	1	<code>\theLineNumber</code>	2711
<code>\start@column</code>	62	<code>\thepage</code>	1174, 2287, 2711
<code>\start@column</code> .	1383, 1386, 1458, 1460	<code>\thepagegrid</code>	53, 62, 63
<code>\stepcounter</code>	2009, 2065		

	1846, 1851, 1852, 1860, 1864, 1964, 2051, 2117, 2177, 2178, 2182, 2300, 2699	<code>\z@skip</code> . . 1576, 1612, 1845, 1846, 2416, 2421, 2428, 2434, 2445, 2453, 2462, 2470
<code>\vfil</code>		43
<code>\vfuzz</code> 393, 1596, 1604, 1803, 1960, 1961, 1965, 2299, 2301		
<code>\void@cclv</code> 433, 439, 1216, 1239, 1259, 1271, 1485		
<code>\vrule</code> 182, 1192, 1571, 1583, 2042, 2103		
<code>\vsize</code> 24, 26, 35, 37, 51, 70, 73, 74		
<code>\vsize</code> . 528, 641, 767, 1070, 1725, 1726, 1918, 1974, 2179		
<code>\vskip</code>		77, 95, 101
<code>\vspace</code>		96, 97, 103
<code>\vspace</code>		2408
<code>\vspace@ltx</code>	2408, 2638	
<code>\vsplit</code>		77
<code>\vsplit</code>	394, 1807, 1963	
<code>\vss</code>	501, 514	
<code>\vtop</code>	1598, 2033, 2086	
W		
<code>\widowpenalty</code>		93
<code>\widowpenalty</code> . . 279, 2370, 2371, 2379		
<code>\widowpenalty@ltx</code> . . 2330–2333, 2370, 2379		
<code>\width@float</code> 913, 916, 939, 1195		
<code>\widthd@float</code> 924, 927, 940, 1196		
<code>\write</code>		95, 106
<code>\write</code>	1904, 1935	
<code>\write@linelabel</code> . . . 2633, 2709		
X		
<code>\xdef</code>	550, 779, 780, 988, 1009, 1457, 1617, 1655, 1668, 1675, 1898, 1926, 2049, 2599	
Y		
<code><your code here></code> placeholder . .		19
<code><your document here></code> placeholder		9
Z		
<code>\z@</code>		30, 76, 77

Change History

4.0a	General: <code>\@yfloat:</code> de-fang <code>\set@footnotewidth</code> (see <code>ltxutil.dtx</code>): we have already done its job. 4	<code>\@addmarginpar@one:</code> Change <code>\@makecol</code> to <code>\@makecolumn</code> with argument 64
	Introduce <code>\marry@height</code> 4	Change <code>\set@colroom</code> to <code>\set@colht</code> 64
	Introduce <code>\set@marry@height</code> 4	<code>\@cclv@nontrivial@sw:</code> Change <code>\@makecol</code> to <code>\@makecolumn</code> with argument 28
	<code>\fpsd@:</code> <code>\@yfloat:</code> de-fang <code>\set@footnotewidth</code> (see <code>ltxutil.dtx</code>): we have already done its job. 47	Change <code>\set@colroom</code> to <code>\set@colht</code> 28
	<code>\marry@baselines:</code> Introduce <code>\marry@height</code> 69	New procedure for showing a box contents, <code>\trace@box</code> 29
	<code>\set@column@hsize:</code> Introduce <code>\set@marry@height</code> 73	<code>\@cflb:</code> 72
4.1a	General: Change <code>\LT@array@new:</code> restore <code>\@tabularcr</code> and <code>\@xtabularcr</code> 4	<code>\@combinepage:</code> (AO, 452) Support length checking: show size of shipped out text. 71
	Change <code>\LT@array@new:</code> set <code>\LT@LL@FM@cr</code> to <code>\@arraycr@array</code> instead of <code>\@arraycr</code> 4	Change <code>\@combinepage</code> to <code>\@combinepage</code> with argument 71
	Repair error in <code>\endlongtable@new</code> involving <code>\@ifx:</code> argument not delimited. 4	<code>\@if@exceed@pagegoal:</code> New procedure for showing a box contents, <code>\trace@box</code> 26
	<code>\endlongtable:</code> Repair error in <code>\endlongtable@new</code> involving <code>\@ifx:</code> argument not delimited. 81	<code>\@if@notdblfloat:</code> Change <code>\@combinepage</code> to <code>\@combinepage</code> with argument 39
	<code>\LT@array:</code> Change <code>\LT@array@new:</code> restore <code>\@tabularcr</code> and <code>\@xtabularcr</code> 84	Change <code>\set@colroom</code> to <code>\set@colht</code> 38
	Change <code>\LT@array@new:</code> set <code>\LT@LL@FM@cr</code> to <code>\@arraycr@array</code> instead of <code>\@arraycr</code> 85	New procedure <code>\@output@combined@page</code> 39
4.1b	<code>\@addmarginpar@mlt:</code> Change <code>\@combinepage</code> to <code>\@combinepage</code> with argument 66	<code>\@makecolumn:</code> Change <code>\@makecol</code> to <code>\@makecolumn</code> with argument 33
	Change <code>\@makecol</code> to <code>\@makecolumn</code> with argument 66	<code>\@outputpage@head:</code> Procedure <code>\@outputpage@head</code> headpatches <code>\@outputpage</code> 92
	Change <code>\set@colroom</code> to <code>\set@colht</code> 66	<code>\@outputpage@tail:</code> Procedure <code>\@outputpage@head</code> headpatches <code>\@outputpage</code> 23
	New procedure <code>\@output@combined@page</code> 67, 68	Procedure <code>\@outputpage@tail</code> tailpatches <code>\@outputpage</code> 23, 46, 75, 88
		General: 4
		(AO, 452) Support length checking: show size of shipped out text. 4
		(AO, 456) Compatibility with other packages that override the

output routine, following suggestion by David Kastrup.	4	Turn off the <code>\set@footnotewidth</code> mechanism; a float ‘knows’ its proper typesetting context . . .	47
Box <code>\footbox</code> changed to <code>box</code>		<code>\marry@baselines</code> : Use <code>\document@inithook</code> instead of <code>\AtBeginDocument</code> .	71
<code>\footsofar</code>	4	<code>\minipagefootnote@here</code> : New procedure <code>\@iffpsbit</code> replaces <code>\@getfpsbit</code>	48
Change <code>\@combinepage</code> to <code>\@combinepage</code> with argument .	4	Tally the height of the float . .	50
Change <code>\@makecol</code> to <code>\@makecolumn</code> with argument .	4	<code>\output@post@document</code> : Box <code>\footbox</code> changed to <code>box</code> <code>\footsofar</code>	33
Change <code>\set@colroom</code> to <code>\set@colht</code>	4	Procedure <code>\set@adj@footins</code> to adjust for footnotes and other inserts	33
Get rid of the <code>\reserved@a</code> idiom	4	<code>\recover@footins</code> : New procedure for showing a box contents, <code>\trace@box</code>	79
New procedure <code>\@iffpsbit</code> replaces <code>\@getfpsbit</code>	4	<code>\save@column@insert@open</code> : New procedure for showing a box contents, <code>\trace@box</code>	56
New procedure <code>\@output@combined@page</code>	4	Use <code>\trace@box</code> instead of <code>\showbox</code>	56
New procedure for showing a box contents, <code>\trace@box</code>	4	<code>\set@adj@colht</code> : Procedure <code>\set@adj@footins</code> to adjust for footnotes and other inserts	74, 75
Procedure <code>\@outputpage@head</code> headpatches <code>\@outputpage</code> . . .	4	<code>\total@text</code> : (AO, 452) Support length checking: show size of shipped out text.	23
Procedure <code>\@outputpage@tail</code> tailpatches <code>\@outputpage</code>	4	<code>turnpage</code> : Use <code>\document@inithook</code> instead of <code>\AtBeginDocument</code> .	54
Procedure <code>\balance@2</code> defined more transparently	4	4.1f	
Procedure <code>\set@adj@footins</code> to adjust for footnotes and other inserts	4	<code>\@addmarginpar@one</code> : (AO, 519) Preserve footnotes that are in <code>\footsofar</code> across a page grid change	64
Tally the height of the float . . .	4	<code>\@makecolumn</code> : (AO, 519) <code>\footins</code> content must be preserved and reintegrated	34
Turn off the <code>\set@footnotewidth</code> mechanism; a float ‘knows’ its proper typesetting context	4	General: (AO, 515) Prevent line numbering within a footnote . .	4
Use <code>\document@inithook</code> instead of <code>\AtBeginDocument</code> . . .	4	(AO, 518) Tally register overflow when document is long	4
Use <code>\trace@box</code> instead of <code>\showbox</code>	4	(AO, 519) Do not use <code>\dimen@</code> register as a scratch register in <code>\set@adj@footins</code>	4
<code>balance@2</code> : Procedure <code>\balance@2</code> defined more transparently . .	75	(AO, 519) Preserve footnotes that are in <code>\footsofar</code> across a page grid change	4
<code>\balance@two</code> : Change <code>\set@colroom</code> to <code>\set@colht</code>	77		
<code>\dispatch@output</code> : (AO, 456) Compatibility with other packages that override the output routine, following suggestion by David Kastrup.	21		
<code>\do@startcolumn@open</code> : Change <code>\set@colroom</code> to <code>\set@colht</code> .	36		
<code>\fpsd@</code> : Get rid of the <code>\reserved@a</code> idiom	47		

(AO, 519) <code>\footins</code> content must be preserved and reintegrated	4		
<code>balance@2</code> : (AO, 519) <code>\footins</code> content must be preserved and reintegrated	76		
<code>\class@documenthook</code> : (AO, 515) Prevent line numbering within a footnote	103		
<code>\set@adj@colht</code> : (AO, 519) Do not use <code>\dimen@</code> regis-			
			ter as a scratch register in <code>\set@adj@footins</code>
			75
		<code>\total@text</code> : (AO, 518) Tally register overflow when locument is long	23
		4.1g	
		General: (AO, 531) Fix package <code>float</code>	4
		<code>\minipagefootnote@here</code> : (AO, 531) Fix package <code>float</code>	49